Autores: Araceli Ruiz Vallecillo, Fernando Jesús Fuentes Carrasco, Javier Jordán Luque y Teodoro Hidalgo Guerrero

Actividad 2: Planificación para un *rover* marciano

Parte 1. Instalación y prueba del entorno de desarrollo.

1.1. Indique qué acciones serían las potencialmente ejecutables en un primer paso por un planificador que opera mediante encadenamiento hacia delante, partiendo desde el estado inicial del problema.

Un planificador que utiliza **encadenamiento hacia adelante** sin aplicar heurísticas es esencialmente equivalente a una búsqueda *no informada* en el espacio de estados. En este enfoque, el proceso comienza desde el estado inicial, generando y evaluando todos los estados posibles que pueden alcanzarse mediante la aplicación de las acciones disponibles.

En este contexto, las **acciones potencialmente ejecutables** en el primer paso por parte del planificador son aquellas cuyas **precondiciones** están completamente satisfechas en el estado inicial.

Las acciones posibles son las siguientes:

- **`navigate`:** Se pueden ejecutar 3 posibles acciones de este tipo.
    - Parámetros:
        - `rover0 - Rover`
        - `waypoint3 - Waypoint`
        - `(waypoint0, waypoint1, waypoint2) waypointX- Waypoint`
    - Precondición:
        - `can_traverse rover0 waypoint3 waypointX`
        - `available rover0`
        - `at rover0 waypoint3`
    - Efecto:
        - `not (at rover0 waypoint3)`
        - `at rover0 waypointX`
- **`sample_soil`:** Se puede ejecutar 1 posible acción de este tipo.
    - Parámetros:
        - `rover0 - Rover`
        - `rover0store - Store`
        - `waypoint3 - Waypoint`
    - Precondición:
        - `at rover0 waypoint3`

        - **at_soil_sample** waypoint3
        - **equipped_for_soil_analysis** rover0
        - **store_of** rover0store rover0
        - **empty** rover0store
    - Efecto:
        - not (**empty** rover0store)
        - **full** rover0store
        - **have_soil_analysis** rover0 waypoint3
        - **not** (at_soil_sample waypoint3)

- **sample_rock:** Se puede ejecutar 1 posible acción de este tipo.
    - Parámetros:
        - rover0 - **Rover**
        - rover0store - **Store**
        - waypoint3 - **Waypoint**
    - Precondiciones:
        - **at** rover0 waypoint3
        - **at_rock_sample** waypoint3
        - **equipped_for_rock_analysis** rover0
        - **store_of** rover0store rover0
        - **empty** rover0store
    - Efecto:
        - not (**empty** rover0store)
        - **full** rover0store
        - **have_rock_analysis** rover0 waypoint3
        - not (**at_rock_sample** waypoint3)

- **calibrate:** Se puede ejecutar 1 posible acción de este tipo.
    - Parámetros:
        - rover0 - **Rover**
        - camera0 - **Camera**
        - objective0 - **Objective**
        - waypoint3 - **Waypoint**
    - Precondiciones:
        - **equipped_for_imaging** rover0
        - **calibration_target** camera0 objective1
        - **at** rover0 waypoint3
        - **visible_from** objective1 waypoint3
        - **on_board** camera0 rover0
    - Efecto:
        - **calibrated** camera0 objective1

1.2. Suponga que tenemos un planificador que opera mediante encadenamiento hacia atrás. Indique qué acciones serían las consideradas en un primer paso a partir del objetivo *(goal)* y cuáles serían los valores de los parámetros.

Un planificador que utiliza **encadenamiento hacia atrás** sin aplicar heurísticas es esencialmente equivalente a una búsqueda no informada que opera desde los estados objetivo hacia el estado inicial. En este enfoque, el proceso comienza desde las condiciones objetivo, identificando las acciones que podrían generar dichos objetivos como efectos, y retrocediendo paso a paso para determinar los estados y acciones necesarios para alcanzarlos.

En este contexto, las **acciones potencialmente seleccionables** en el primer paso por parte del planificador son aquellas cuyos **efectos** satisfacen directamente algunas de las condiciones del estado objetivo.

Con los 3 predicados que representan el objetivo, existen **9 posibles acciones** que alcanzan este estado:

- **`communicate_soil_data`**: Se pueden ejecutar 3 acciones de este tipo.
    - Parámetros:
        - `rover0` - **`Rover`**
        - `general` - **`Lander`**
        - `waypoint2` - **`Waypoint`**
        - `(waypoint1, waypoint2, waypoint3) waypointX` - **`Waypoint`**
        - `waypoint0` - **`Waypoint`**
- **`communicate_rock_data`**: Se pueden ejecutar 3 acciones de este tipo.
    - Parámetros:
        - `rover0` - **`Rover`**
        - `general` - **`Lander`**
        - `waypoint3` - **`Waypoint`**
        - `(waypoint1, waypoint2, waypoint3) waypointX` - **`Waypoint`**
        - `waypoint0` - **`Waypoint`**
- **`communicate_image_data`**: Se pueden ejecutar 3 acciones de este tipo.
    - Parámetros:
        - `rover0` - **`Rover`**
        - `general` - **`Lander`**
        - `objective1` - **`Objective`**
        - `high_res` - **`Mode`**
        - `(waypoint1, waypoint2, waypoint3) waypointX` - **`Waypoint`**
        - `waypoint0` - **`Waypoint`**

1.3. Ejecute el planificador para resolver el problema y documente el resultado.

El problema se ha ejecutado con el planificador **LAMA-first** (Landmark-Aware Multi-Heuristic Planner), conocido por su enfoque eficiente en la planificación basada en hitos (landmarks) y el uso combinado de múltiples heurísticas. **LAMA-first** utiliza una estrategia que prioriza los hitos críticos para alcanzar el objetivo, asegurando que el plan generado cumpla con las restricciones esenciales de forma incremental. Además, combina una búsqueda primero en mejor (Best-First Search) con heurísticas basadas en costos y estimaciones de progreso hacia los hitos. Esto permite reducir drásticamente el tiempo de búsqueda y enfocar los esfuerzos computacionales en las acciones más prometedoras. Este enfoque permite a **LAMA-first** lograr un equilibrio entre la calidad del plan y el tiempo de ejecución, siendo particularmente efectivo en problemas complejos con restricciones múltiples.

En esta ejecución, el planificador ha generado **94** nodos y ha expandido **16** para finalmente encontrar un plan con **10** acciones:

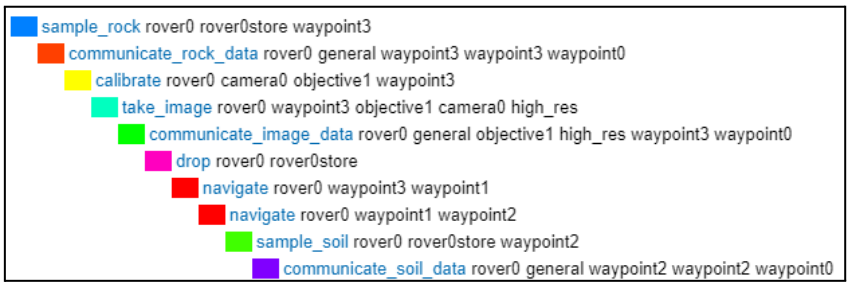

Figura 1. Ejecución del planificador LAMA-first para el caso original.

Parte 2. Modificación del estado inicial y objetivos

2.1. Aparece un nuevo objetivo (objetivo2) que debe ser fotografiado en alta resolución y en color. Este solo es visible desde una ubicación accesible por el rover0 desde waypoint1. Desde dicha ubicación no es visible la base tipo lander.

Se han añadido los siguientes objetos al problema:

- `objective2 - Objective`
- `colour_high_res - Mode`

Se han añadido los siguientes predicados al estado inicial:

- **`visible_from`** `objective2 waypoint2`

Se han eliminado los siguientes predicados del estado inicial:

- **`visible`** `waypoint2 waypoint0`

- **visible** waypoint0 waypoint2

Se han añadido el siguiente objetivo al problema:

- **communicated_image_data** objective2 colour_high_res

2.2. Está disponible un segundo rover (rover1) que no tiene capacidad para análisis de suelo y rocas, y sí para tomar imágenes. Este rover empieza en la ubicación waypoint2.

Se ha añadido los siguientes objetos al problema:

- rover1 - **Rover**
- camera1 - **Camera**

Se han añadido los siguientes predicados al estado inicial:

- **can_traverse** rover1 waypointX waipointY (misma movilidad que *rover0*)
- **equipped_for_imaging** rover1
- Se definen las mismas propiedades en *camera1* que en *camera0*.

Se han eliminado los siguientes predicados del estado inicial:

- **visible** waypoint1 waypoint3

Parte 3. Ejecución y evaluación del planificador

3.1. Documente la ejecución del planificador para el caso 2.1. Compare con el plan original obtenido en la prueba de la instalación. ¿El *rover* realiza algún movimiento innecesario? ¿Por qué cree que ocurre esto?

En esta ejecución, el planificador ha generado **235** nodos y ha expandido **96** para finalmente encontrar un plan con **15** acciones:

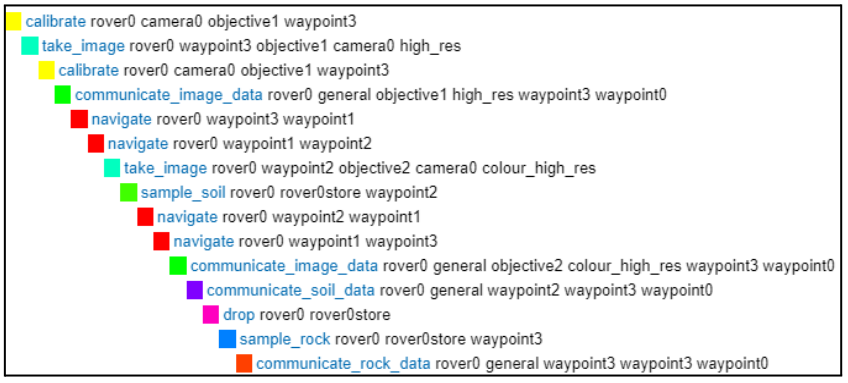

Figura 2. Ejecución del planificador LAMA-first para el caso 2.1.

Debido a que se ha añadido un nuevo objetivo, realizar una fotografía en color y alta resolución, el plan se ve afectado aumentando su coste de **10** a **15** acciones, además se ve como en el plan original el planificador genera un número considerablemente menor (94 nodos) en comparación al generado en este nuevo apartado (235 nodos).

En el **caso 2.1**, el planificador realiza **un movimiento innecesario** al navegar desde *waypoint1* a *waypoint3* para recoger la muestra de roca (`sample_rock rover0 rover0store waypoint`). Este paso podría haberse evitado si el *rover* hubiera recogido la muestra directamente al inicio desde *waypoint3*, para luego comunicarse con el *lander* (`communicate_rock_data rover0 general waypoint3 waypoint3 waypoint0`) y soltar la muestra (`drop rover0 rover0store`). Posteriormente, el rover podría continuar con el resto de la ejecución indicada en la Figura 2, **ahorrando el paso de navegación de *waypoint1* a *waypoint3*** (`navigate rover0 waypoint 1 waypoint3`). De este modo, sería posible comunicar los resultados de la imagen y del análisis del suelo directamente desde *waypoint1*, sin la necesidad de ir a *waypoint3* para recoger la muestra de roca, lo que habría optimizado el plan.

La generación de este plan **no óptimo** se puede justificar debido a que LAMA-first utiliza una estrategia heurística que prioriza cumplir los objetivos inmediatos sin buscar optimizar movimientos intermedios, calculando las heurísticas de forma secuencial, por lo que puede no considerar estos pasos alternativos más eficientes.

3.2. Documente la ejecución del planificador para el caso 2.2 y compare con el plan original. ¿El nuevo plan utiliza el nuevo *rover* (rover1) introducido en el problema? ¿Puede evaluar por qué se da este efecto? Haga ahora que el objetivo de objetivo1 no sea visible desde la posición inicial del rover0, ejecute el planificador y documente la ejecución de nuevo. ¿Hay cambios? Si los hay, comente las razones de dichos cambios.

En esta ejecución, el planificador ha generado **109** nodos y ha expandido **14** para finalmente encontrar un plan con **10** acciones:

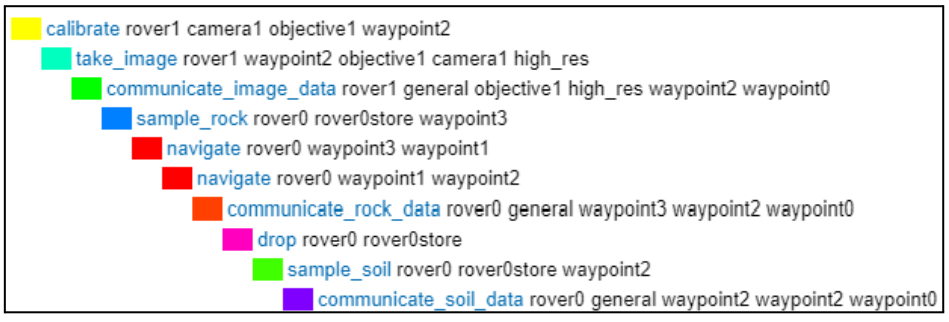

Figura 3. Ejecución del planificador LAMA-first para el caso 2.2.

La inclusión de un nuevo *rover* (*rover1*) aumenta las posibilidades de generar planes óptimos, y en este caso, lo encuentra. Usando este planificador, el *rover1* sí participa en el plan generado, aunque no implique que sea necesario para su completitud.

Al tratar con planificadores secuenciales, el tiempo de ejecución se mantiene constante (una acción ejecutada como máximo en cada instante). En cambio, si se habilitan ejecuciones paralelas, el tiempo total del plan podría reducirse hasta **7** (al poder realizar *rover0* sus acciones y a la vez *rover1* realizar la fotografía).

Al realizar el cambio ignorando la posibilidad de observar el *objective1* desde el *waypoint3* se obtiene el mismo plan de ejecución. En el caso de que se hubiese generado en un primer instante un plan sin hacer uso de *rover1*, se tendría tanto un plan que siga sin hacer uso de *rover1* al navegar *rover0* por otros waypoints que tengan visión sobre el objetivo, o algún plan en el que *rover1* si realice la fotografía.

Parte 4. Modificación del dominio

4.1. Describa brevemente en la memoria cómo se ha resuelto esta parte (significado de los elementos introducidos, funcionamiento de las nuevas acciones, etc.).

El concepto de la batería se ha introducido mediante el tipo *battery-number*, que representa los diferentes niveles de la batería. Los objetos de este tipo son números que indican dichos niveles. Para gestionar el consumo de la batería, se ha declarado el predicado (**`battery-predecessor`** `?b - battery-number ?c - battery-number`), el cual permite determinar si un nivel de batería tiene un predecesor. Si no lo tiene, significa que la batería está agotada. Además, para asignar un nivel de batería a cada *rover*, se ha creado otro predicado (**`has-battery`** `?r - rover ?b - battery-number`), que indica qué nivel de batería tiene un *rover* en específico.

Tal y como se especifica en el enunciado, la batería sólo se consumirá en acciones de movimiento. Por lo tanto, para involucrar las baterías en el dominio, se ha modificado la acción ***navigate*** añadiendo **dos precondiciones** y **dos nuevos efectos**:

- Para poder navegar, se debe verificar que exista un predecesor del nivel de la batería para poder reducirlo (**`battery-predecesso`**r `?c ?b -> ?c` precede a `?b`) y que nuestro rover tenga ese nivel de batería `?b` (**`has-battery`** `?x ?b`).
- Tras navegar, se hace uso de ese predecesor para asignarle ese nivel de batería al rover (**`has-battery`** `?x ?c`) y se negará que siga teniendo el nivel anterior (`not` (**`has-battery`** `?x ?b`)).

Para poder recuperar el nivel de batería, definimos la acción **charge**, que toma como parámetros el rover, el nivel de batería actual, el nivel de batería sucesor, el lander y un waypoint:

- Como precondiciones, se verifica que tanto el lander como el rover se encuentren en el mismo waypoint `?w` (`(`**`at_lander`** `?l ?w) y (`**`at`** `?x ?w)`) como que el rover tenga ese nivel de batería `?b` (**`has-battery`** `?x ?b`) y que exista un nivel de batería sucesor `?c` (**`has-battery`** `?x ?c`).

- Como efecto, se niega que tenga el nivel actual de batería (`not (**has-battery** ?x ?b)`) y se le asigna el nivel de batería sucesor (**`has-battery ?x ?c`**).

4.2. Ejecute el planificador y documente el resultado.

En esta ejecución, el planificador ha generado **96** nodos y ha expandido **18** para finalmente encontrar un plan con **13** acciones:

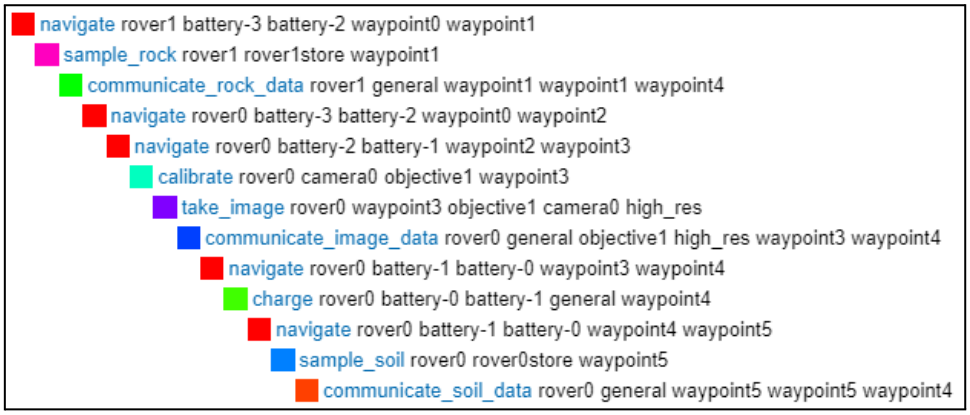

Figura 4. Ejecución del planificador LAMA-first para la parte 4.

Se puede observar como tanto el *rover1* como el *rover0* cumplen con las expectativas del problema, ya que:

- Ambos rovers parten desde el mismo punto (*waypoint0*).
- El rover1 navega hacia el *waypoint1*, coge la muestra y la comunica tal y como se esperaba.
- Por otro lado, el *rover0*, navega hasta el *waypoint3*, donde una vez llega, calibra la cámara, toma la imagen y la comunica. Sin embargo, al seguir navegando a su próximo destino ve necesario cargar (*waypoint4*), solo carga en uno ya que solo está a 1 de distancia de su objetivo final. Una vez cargado, navega hasta su objetivo, coge la muestra en cuestión y lo comunica.

Dificultades encontradas

Venimos del planeta Maude ;)