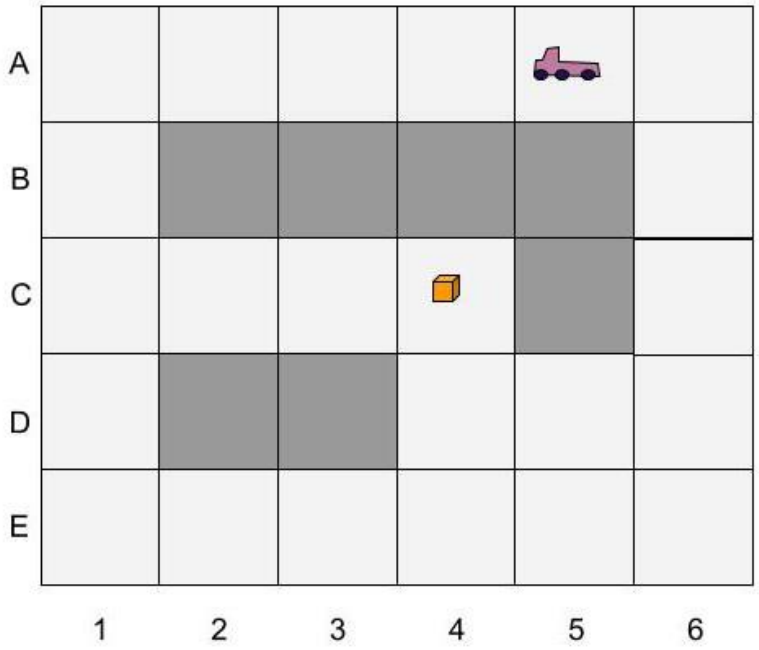
Razonamiento y planificación automática

Alejandro Cervantes

Resolución Actividad 1: Búsqueda Heurística

Planteamiento

Enunciado



5.a) Ejecutar Amplitud, Profundidad y A* con distancia a la meta como heurística (Manhattan y Euclídea) por separado

5.b) Un estado inicial en el que el algoritmo de búsqueda en profundidad obtenga la solución óptima expandiendo menos nodos que el resto

5.c) Una situación en la que el coste del movimiento varía de la siguiente forma: los movimientos hacia abajo, izquierda y derecha tienen un coste de 1, mientras que los movimientos hacia arriba tienen un coste de 5

Objetivo

Entender bien los algoritmos de búsqueda

- **Qué partes del algoritmo dependen del problema:** las funciones que generan los sucesores, la función que detecta la meta, la que calcula la heurística y la que calcula el coste.
- Un algoritmo se programa sin tener en cuenta un problema u otro. Por ejemplo, hay que definir un orden entre operadores que puede ser bueno o malo, pero sin heurística no lo sabemos.
- Entender cómo funciona la heurística y justificar si una heurística es admisible o no.

Objetivo

Comparar los algoritmos de búsqueda entre sí

- Distinguir entre **completo** (encuentra solución si la hay), **óptimo** (encuentra la solución con menor coste, o número de acciones si no hay coste) y **eficiente** (expande pocos nodos o tarda poco tiempo).
- Comparar numéricamente los valores relevantes

Respuestas

5.a. Algoritmos sobre situación inicial

- Lo básico es poder generar esta tabla

```
########
#     T·#
# ####·#
#    P#·#
# ##···#
#       #
########
```

Amp.
A*
(ambos)

```
########
#····T #
#·#### #
#···P# #
# ##    #
#       #
########
```

Prof.

	¿Solución?	Coste	Expandidos	Max. Abierta
Amplitud	Sí	7	17	4
Profundidad	Sí	9	10	3
A* Manhattan	Sí	7	11	5
A* Euclidea	Sí	7	11	5

Conclusiones iniciales

Todos los algoritmos encuentran solución. Esto no querría decir automáticamente que sean **completos**, pero sabemos **que todos ellos lo son** (porque evitan ciclos y mantienen la lista ABIERTA para hacer backtracking).

Todos los algoritmos encuentran solución de coste 7 menos Profundidad (9). Es decir hay tres que encuentran el óptimo. Pero esto no quiere decir que estos algoritmos "sean" óptimos siempre, porque esto no está garantizado. Sabemos por la teoría que en este caso 5.a estos tres lo son.

A* con cualquiera de las heurísticas expande menos nodos que Amplitud. Por lo tanto es más **eficiente**. El motivo es que descarta explorar el camino de la izquierda (h suma una cantidad creciente, y eso hace que sea peor ir en esa dirección que ir a la derecha).

5.a. Algoritmo Profundidad sobre situación inicial

1) El orden favorece explorar las acciones en este orden: (left,right,down,up)
2) El control de meta se hace solo cuando se expande

```
[Node <(5, 1)>] Chosen node: Node <(5, 1)> Expanded [Node <(5, 1)>] Successors: [[Node <(6, 1)>,
Node <(4, 1)>]]
[Node <(4, 1)>, Node <(6, 1)>] Chosen node: Node <(4, 1)> Expanded [Node <(4, 1)>] Successors:
[[Node <(5, 1)>, Node <(3, 1)>]]
[Node <(3, 1)>, Node <(6, 1)>] Chosen node: Node <(3, 1)> Expanded [Node <(3, 1)>] Successors:
[[Node <(4, 1)>, Node <(2, 1)>]]
[Node <(2, 1)>, Node <(6, 1)>] Chosen node: Node <(2, 1)> Expanded [Node <(2, 1)>] Successors:
[[Node <(3, 1)>, Node <(1, 1)>]]
[Node <(1, 1)>, Node <(6, 1)>] Chosen node: Node <(1, 1)> Expanded [Node <(1, 1)>] Successors:
[[Node <(1, 2)>, Node <(2, 1)>]]
[Node <(1, 2)>, Node <(6, 1)>] Chosen node: Node <(1, 2)> Expanded [Node <(1, 2)>] Successors:
[[Node <(1, 1)>, Node <(1, 3)>]]
[Node <(1, 3)>, Node <(6, 1)>] Chosen node: Node <(1, 3)> Expanded [Node <(1, 3)>] Successors:
[[Node <(1, 2)>, Node <(1, 4)>, Node <(2, 3)>]]
[Node <(2, 3)>, Node <(1, 4)>, Node <(6, 1)>] Chosen node: Node <(2, 3)> Expanded [Node <(2,
3)>] Successors: [[Node <(3, 3)>, Node <(1, 3)>]]
[Node <(3, 3)>, Node <(1, 4)>, Node <(6, 1)>] Chosen node: Node <(3, 3)> Expanded [Node <(3,
3)>] Successors: [[Node <(4, 3)>, Node <(2, 3)>]]
[Node <(4, 3)>, Node <(1, 4)>, Node <(6, 1)>] Chosen node: Node <(4, 3)>

Is goal! Path from initial to goal: [(None, (5, 1)), ('left', (4, 1)), ('left', (3, 1)),
('left', (2, 1)), ('left', (1, 1)), ('down', (1, 2)), ('down', (1, 3)), ('right', (2, 3)),
('right', (3, 3)), ('right', (4, 3))]
```

5.b Caso en que DFS sea más eficiente

Es decir, que encuentre la misma solución pero requiera menos nodos. Esto depende del orden en que DFS expanda los nodos. Para “favorecerlo” basta mover el inicio un poco a la izquierda, y el camino que encuentra pasa a ser el óptimo.

Podemos verlo depurando (WebViewer o ConsoleViewer).

```
########
#··T←  #
#·#### #
#···P# #
# ##   #
#      #
########
```

	Coste	Exp.	Max. ABIERTA
Amplitud	7	19	4
Profundidad	7	8	3
A* Manhattan	7	13	3
A* Euclidea	7	13	3

5.c Problema con coste

Ni Amplitud ni Profundidad tienen en cuenta el coste, así que si encuentran el óptimo sería por casualidad. La solución será la misma, pero ahora el coste del último movimiento se incrementa.

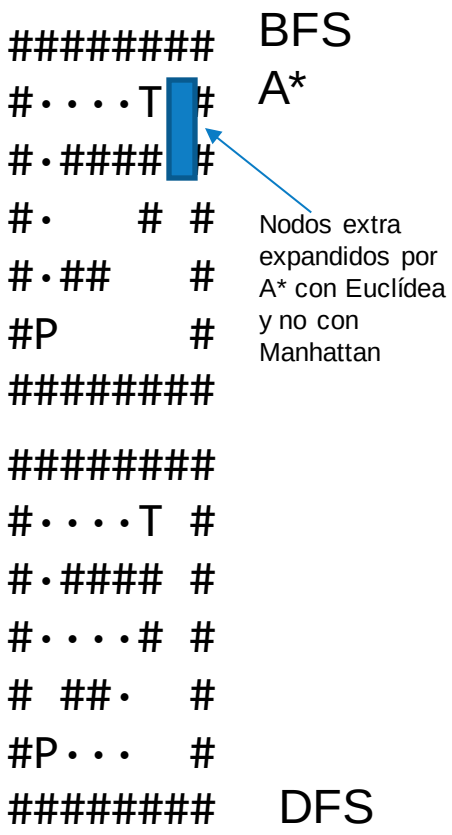
```
########
#     T·#
# ####·#
#    P#·#
# ##···#
#       #
########

########
#····T #
#·#### #
#···P# #
# ##    #
#       #
########
```

Aquí Amplitud ya no encuenta el óptimo. Coincide que ahora el camino de DFS es el óptimo. A* sí encuentra este camino porque la heurística es admisible (ambas lo son, pero hay que decirlo). Para conseguir esto, A* tiene que expandir ahora muchos más nodos (el camino de la izquierda no puede descartarse).

	¿Solución?	Coste	Expandidos	Max. Abierta
Amplitud	Sí	11	17	4
Profundidad	Sí	9	10	3
A* + Manh.	Sí	9	19	5
A* + Euclídea	Sí	9	19	5

Efecto de las heurísticas diferentes en A*

Poniendo el objetivo en otro punto algo más distante tenemos ya alguna diferencia entre usar como heurística la distancia de Manhattan al objetivo, y usar la distancia Euclídea al objetivo.



Aquí BFS encuentra el mismo camino que A*. DFS encuentra uno peor, casualmente aquí las posiciones no le son favorables.
A* con Manhattan es más eficiente que con Euclídea. Se puede demostrar que es más eficiente en general porque genera números iguales o mayores, por lo tanto más próximos al coste real (h*(n)). Manhattan es por tanto más informada que Euclídea.

	¿Solución?	Coste	Exp.	Max. Abierta
Amplitud	Sí	8	22	4
Profundidad	Sí	14	11	21
A* Manhattan	Sí	8	9	3
A* Euclidea	Sí	8	11	3

Admisibilidad de las heurísticas

Se puede generar una heurística admisible para un problema adoptando como heurística la solución óptima de un problema relajado: es decir, un problema igual pero donde las acciones tienen menos restricciones.

Reglas del problema inicial: se puede mover left,right,bottom o top si no hay pared en el destino (cuatro reglas, el resultado es la casilla destino)
Regla del problema relajado: se puede mover siempre left, right, bottom o top (cuatro reglas, el resultado es la casilla destino)

La solución óptima $h_{opt,relajado}$ del problema relajado es trivial:
a) se necesitan tantos movimientos como columnas separen de la columna objetivo para actualizar la coordenada de columna, ya que las reglas como mucho actualizan la X en una unidad
b) se necesitan además un número similar de movimientos de fila, ya que las reglas como mucho actualizan Y en una unidad

- Por tanto el número de movimientos necesarios es exactamente la distancia de Manhattan entre cada estado y el objetivo. Esta heurística es admisible.
- La distancia Euclídea, puede calcular "pasos" de 1.4142... Pero al mover, el agente consumirá dos movimientos, porque las reglas no permiten ese movimiento directamente. El valor al sumar todos los movimientos será inferior o igual al anterior, así que es también admisible. Si $M \leq h_{opt,inicial}$ y $E \leq M$, entonces $E \leq h_{opt,inicial}$

Conclusiones clave

- Amplitud, mencionar que encuentra el óptimo en cuanto a longitud
- Amplitud, expande muchos nodos
- Profundidad, mencionar el orden y por qué expande pocos nodos
- Profundidad, encontrar un caso en el que el objetivo esté en el camino (y expanda poco)
- A*, mencionar cómo optimiza el número de nodos expandidos
- Costes, mencionar que A* encuentra el óptimo
- Costes, mencionar que encontrar el óptimo le cuesta a A* muchas expansiones adicionales
- Heurísticas, mencionar y si se puede, demostrar, que son admisibles
- Heurísticas, ser capaces de compararlas entre sí

www.unir.net