FUNDAMENTOS DE LA INGENIERÍA ARTIFICIAL

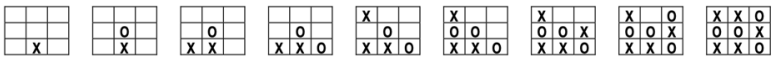
PRUEBA DE EVALUACIÓN CONTINUA 1 (PEC1)
Curso 2023-2024

BÚSQUEDA DE UN ESPACIO DE ESTADOS.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

EJERCICIO 1:

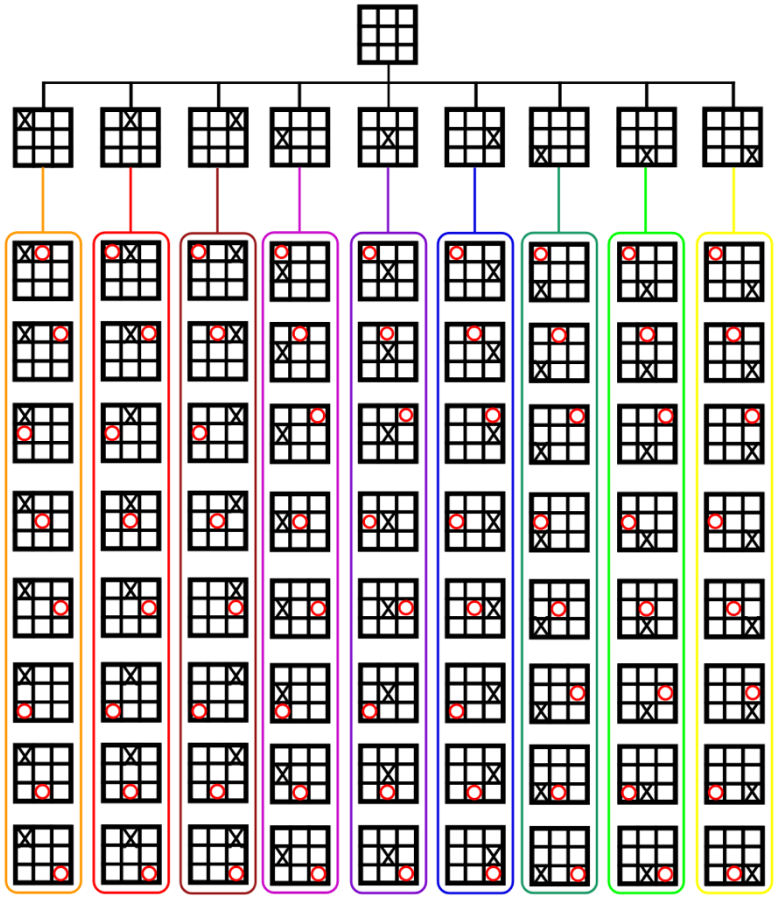
Dibuje mediante un grafo dirigido o describa detalladamente mediante una tabla el **espacio de estados** (o espacio de búsqueda) completo para el *problema del Tres en Raya* descrito más adelante. Para ello especifique: el conjunto de todos los estados posibles, el estado inicial, el o los estados meta, los operadores aplicables a cada estado y el coste asociado a cada operador. En el problema del Tres en Raya, dos jugadores llamados "X" y "O" se turnan para marcar los espacios en una cuadrícula de 3×3. El primer jugador que coloca tres marcas propias en una fila, columna o diagonal gana la partida. Por ejemplo, en la siguiente partida hay un empate final:



Suponga que en el estado inicial todos los espacios están sin marcar y que el primer jugador en marcar es "X".

NOTA:

Dado el amplio número de estados diferentes en este problema, resulta muy difícil representar en una tabla o en un grafo el conjunto total de estados del espacio de búsqueda. Por ello, se pide únicamente dibujar el subgrafo resultante de partir del estado inicial y generar aquellos estados cuya profundidad sea menor o igual que 2.



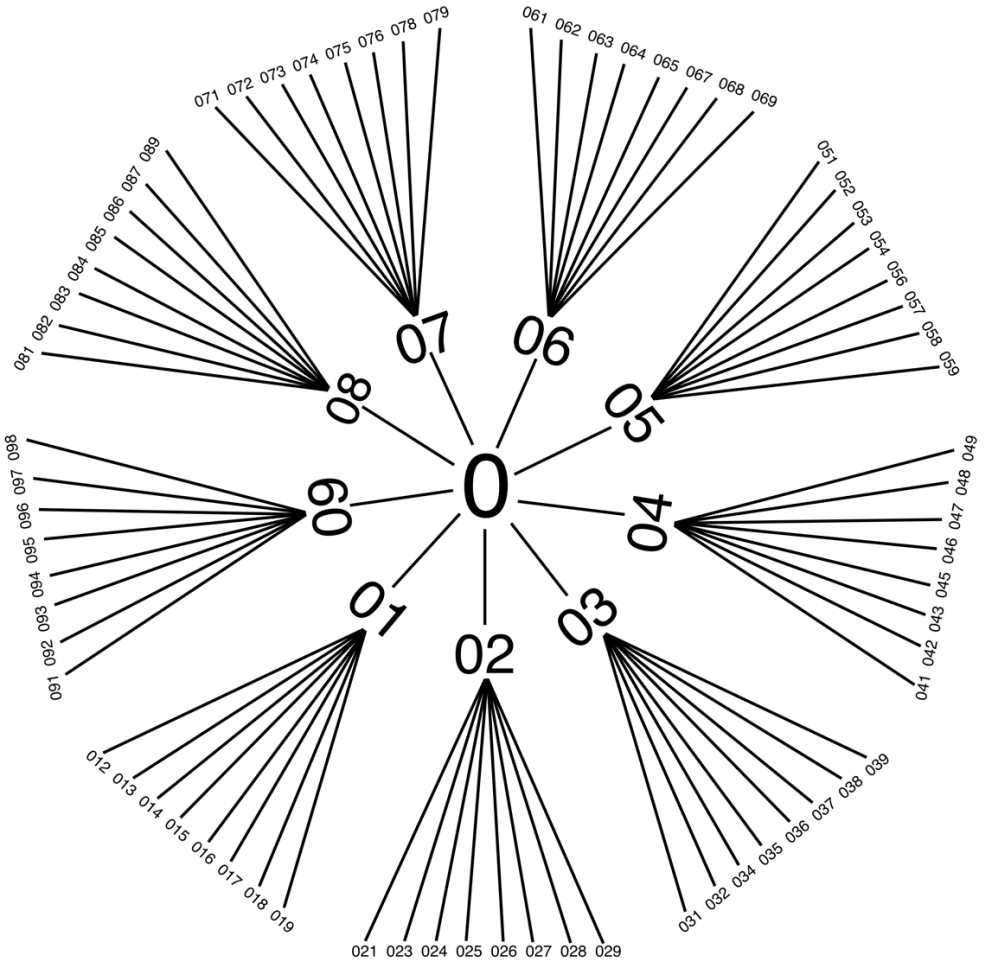
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

En la figura podemos ver representada las dos primeras posibles Jugadas en el juego del 3 en raya por parte de cada jugador teniendo en cuenta que el tablero está vacío, que inician las X y que el segundo movimiento es de O.

A de esta representación he elaborado un grafo: El centro del grafo es el tablero vacio. Representado por el numero 0.
El primer movimiento se ve en la primera profundidad. Lo representamos por 01-02-03-04-05-06-07-08-09. Esta representación viene dada asi ya que represento con la primera cifra la profundidad 0 que es el tablero vacio y la segunda cifra representa la primera profundidad y primer movimiento. Representado por los números de 1-9 siendo estas la posición en el tablero del siguiente modo:

1	2	3
4	5	6
7	8	9

El segundo movimiento se ve en la segunda profundidad. Lo representamos por 3 cifras numéricas, añadiendo a las jugada anterior el numero de posición del movimiento de la misma manera que antes.
Un ejemplo es 012 : Esto representa (Inicial tablero vacio, primer movimiento X en la casilla (1) y primer movimiento de O (tras el movimiento de x) en la posición del tableo 2 (2):

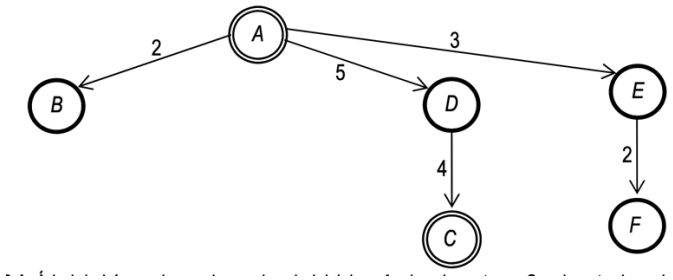


Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

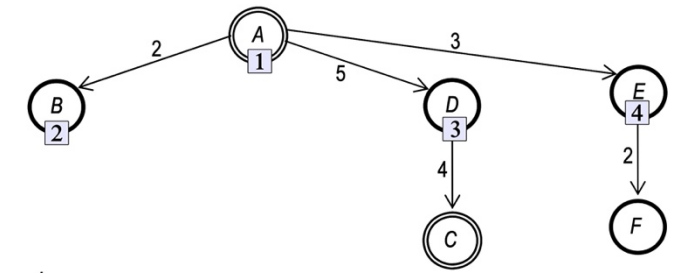
EJERCICIO 2:

Considere el espacio de búsqueda de la figura 2.1, que tiene forma de árbol, donde el nodo raíz del árbol es el nodo inicial, existe un único nodo meta y cada operador tiene asociado un coste. Explique razonadamente **en qué orden se expandirían** los nodos de dicho árbol de búsqueda a partir de cada uno de los métodos siguientes de búsqueda sin información del dominio:

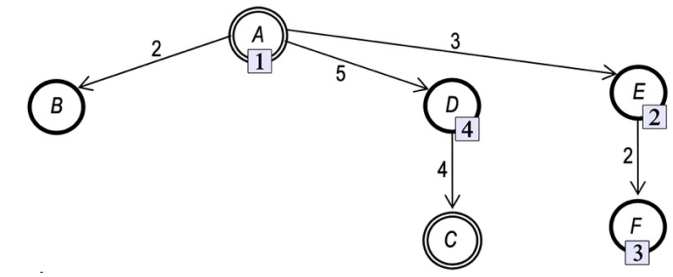
- (2a) Búsqueda Primero en Anchura (de izquierda a derecha)
- (2b) Búsqueda Primero en Profundidad (de derecha a izquierda)
- (2c) Búsqueda de Coste Uniforme
- (2d) Búsqueda en Anchura Iterativa (de derecha a izquierda)
- (2e) Búsqueda en Profundidad Iterativa (de izquierda a derecha)



(2a) Búsqueda Primero en Anchura (de izquierda a derecha) : Este algoritmo explora el árbol de búsqueda por niveles de profundidad, así que el orden de expansión de los nodos de izquierda a derecha sería el reflejado en la figura:



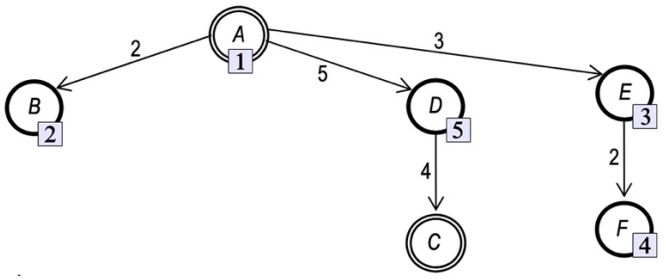
(2b) Búsqueda Primero en Profundidad (de derecha a izquierda) : Este algoritmo explora el árbol de búsqueda bajando de nivel siempre que sea posible. Si no es posible, se sube al nodo más cercano al nodo actual desde el que poder seguir bajando de nivel. El orden de expansión de los nodos de derecha a izquierda según este algoritmo se dibuja en la figura:



Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

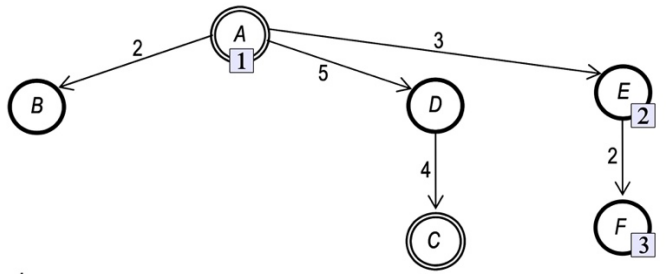
Según la búsqueda primero en profundidad (de derecha a izquierda). Observe que el nodo meta no es realmente expandido (es decir, sus hijos no son generados), ya que justo antes de su expansión se comprueba que es un nodo meta y, por tanto, el algoritmo termina en ese momento sin que se lleguen a generar sus hijos.

(2c) Búsqueda de Coste Uniforme: Este algoritmo explora el árbol de búsqueda expandiendo aquel nodo disponible cuyo coste al nodo inicial sea el menor.

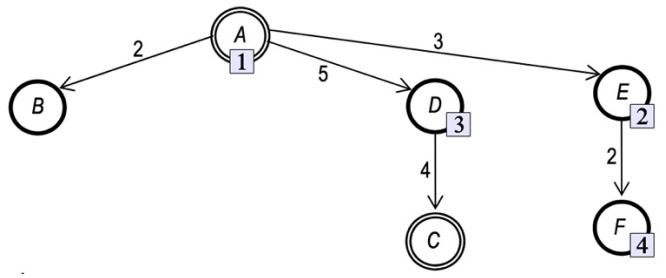


(2d) Búsqueda en Anchura Iterativa (de derecha a izquierda) : Este algoritmo ejecuta iterativamente varias búsquedas primero en anchura, de manera que entre iteración e iteración se incrementa en una unidad el número máximo de hijos que se generan en cada expansión de un nodo padre. Al principio (en la primera iteración), únicamente un hijo es generado en cada expansión

ITERACIÓN 1:



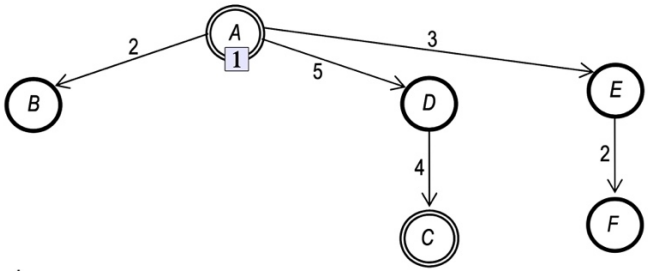
ITERACIÓN 2:



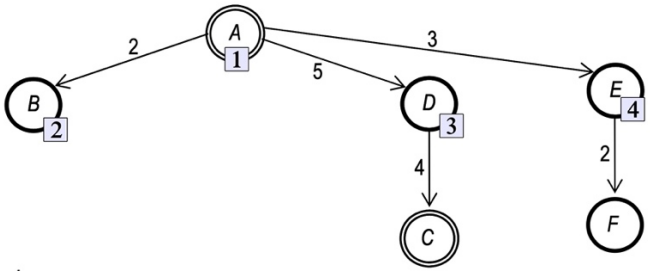
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

(2e) Búsqueda en Profundidad Iterativa (de izquierda a derecha) : Este algoritmo ejecuta iterativamente varias búsquedas primero en profundidad, de manera que entre iteración e iteración se incrementa en una unidad la profundidad límite. Al principio (en la primera iteración), la profundidad límite es igual a 1, así que sólo se podrá expandir el nodo inicial, cuya profundidad es igual a 0.

ITERACIÓN 1:



ITERACIÓN 2:

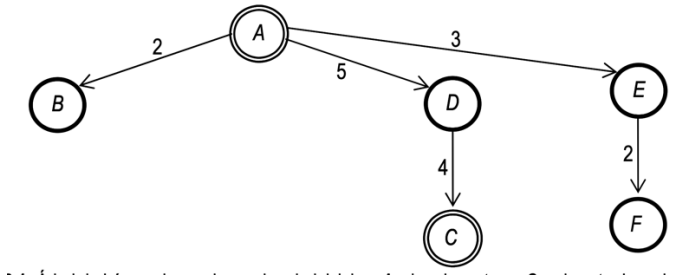


Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

EJERCICIO 3:

Considere el espacio de búsqueda de la figura 3.1, que tiene forma de árbol, donde el nodo raíz del árbol es el nodo inicial, existe un único nodo meta y cada operador tiene asociado un coste. Describa cuál es el contenido de **ABIERTA**, previamente a cada extracción de un nodo de la misma, a partir de cada uno de los métodos siguientes de búsqueda sin información del dominio:

- (3a) Búsqueda Primero en Anchura (de izquierda a derecha)
- (3b) Búsqueda Primero en Profundidad (de derecha a izquierda)
- (3c) Búsqueda de Coste Uniforme
- (3d) Búsqueda en Anchura Iterativa (de derecha a izquierda)
- (3e) Búsqueda en Profundidad Iterativa (de izquierda a derecha)



(3a). Búsqueda Primero en Anchura (de izquierda a derecha) : Este algoritmo usa ABIERTA como una cola, de manera que siempre se saca el primer nodo de la cola y se introducen sus hijos al final de la misma. El contenido de ABIERTA previamente a cada extracción de un nodo es el siguiente:

Antes de sacar *A*: {*A*}

Antes de sacar *B*: {*B, D, E*}

Antes de sacar *D*: {*D, E*}

Antes de sacar *E*: {*E, C,}*

Antes de sacar *C*: {*C, F*}

META ENCONTRADA

Observe que, tras cada expansión del nodo sacado de ABIERTA, sus nodos hijos más a la izquierda se introducen en ABIERTA antes que los situados más a la derecha.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

(3b) Búsqueda Primero en Profundidad (de derecha a izquierda) Este algoritmo usa ABIERTA como una pila, de manera que siempre se saca el primer nodo de la pila y se introducen sus hijos al principio de la misma. El contenido de ABIERTA previamente a cada extracción de un nodo es el siguiente:

Antes de sacar *A*: {*A*}

Antes de sacar *E*: {*E, D, B*}

Antes de sacar *F*: {*F, D, B*}

Antes de sacar *D*: {*D, B*}

Antes de sacar *D*: {*C, B*}

META ENCONTRADA

Observe que, tras cada expansión del nodo sacado de ABIERTA, sus nodos hijos más a la derecha se introducen en ABIERTA después que los situados más a la izquierda.

(3c) Búsqueda de Coste Uniforme :Este algoritmo mantiene ABIERTA ordenada según el coste del camino desde cada nodo al nodo inicial. En este ejercicio desharemos de forma arbitraria los posibles empates que surjan al determinar qué nodo se debe sacar de ABIERTA. El contenido de ABIERTA previamente a cada extracción de un nodo es el siguiente:

Antes de sacar *A*: {*A*(0)}

Antes de sacar *B*: {*B*(2), *E*(3), *D*(5)}

Antes de sacar *E*: {*E*(3), *D*(5)}

Antes de sacar *D*: {*D*(5) , *F*(5)}

Antes de sacar *F*: {*F*(5) , *C*(9)}

Antes de sacar *C*: {*C*(9)}

META ENCONTRADA

Observe que, tras cada expansión de un nodo, sus nodos hijos son introducidos en ABIERTA, donde todos los nodos quedan siempre ordenados por coste creciente. Para facilitar el seguimiento del algoritmo, entre paréntesis y al lado de cada nodo de ABIERTA se especifica el coste del camino para ir desde dicho nodo hasta el nodo inicial. En el caso de igualdad de costes, mantengo la prioridad de orden en el contenido de ABIERTA.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

(3d) Búsqueda en Anchura Iterativa (de derecha a izquierda) :Debido a que este algoritmo es una iteración de la búsqueda primero en anchura, los dos gestionan ABIERTA del mismo modo, es decir, como una cola. La diferencia reside en que en la búsqueda en anchura iterativa en cada iteración se fija un número máximo de hijos que pueden ser generados al expandir un nodo padre. Este número empieza valiendo 1 y es incrementado en una unidad al principio de cada nueva iteración.

Iteración 1 (cada expansión de un nodo padre genera como máximo un hijo):

Antes de sacar *A*: {*A*}

Antes de sacar *E*: {*E*}

Antes de sacar *F*: {*F*}

Iteración 2 (cada expansión de un nodo padre genera como máximo dos hijos):

Antes de sacar *A*: {*A*}

Antes de sacar *E*: {*E, D*}

Antes de sacar *D*: {*D, F*}

Antes de sacar *F*: {*F*,*C*}

Antes de sacar *C*: {*C*}

META ENCONTRADA

(3e) Búsqueda en Profundidad Iterativa (de izquierda a derecha) : Debido a que este algoritmo es una iteración de la búsqueda primero en profundidad, los dos gestionan ABIERTA del mismo modo, es decir, como una pila. La diferencia reside en que en la búsqueda en profundidad iterativa en cada iteración se fija una profundidad límite propia. Este número empieza valiendo 1, lo cual permite expandir únicamente el nodo inicial, y es incrementado en una unidad al principio de cada nueva iteración.

Iteración 1 (profundidad límite igual a 1):

Antes de sacar *A*: {*A*}

Antes de sacar *B*: {*B, D, E*} (Nótese que *B* no es expandido por coincidir su profundidad con la profundidad límite.)

Antes de sacar *D*: {*D, E*} (Nótese que *D* no es expandido por coincidir su profundidad con la profundidad límite.)

Antes de sacar *E*: {*E*} (Nótese que *E* no es expandido por coincidir su profundidad con la profundidad límite.)

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

Iteración 2 (profundidad límite igual a 2):

Antes de sacar *A*: {*A*}

Antes de sacar *B*: {*B, D, E*}

Antes de sacar *D*: {*D, E*}

Antes de sacar *C*: {*C, E*}

META ENCONTRADA

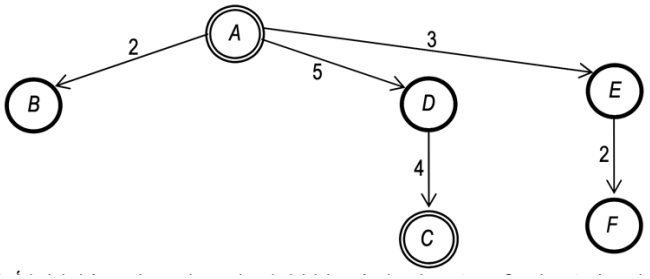
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

EJERCICIO 4:

Considere el espacio de búsqueda de la figura 4.1, que tiene forma de árbol, donde el nodo raíz del árbol es el nodo inicial, existe un único nodo meta y cada operador tiene asociado un coste. Describa cuál es el contenido de **TABLA_A**, posteriormente a cada expansión de un nodo, a partir de cada uno de los métodos siguientes de búsqueda sin información del dominio:

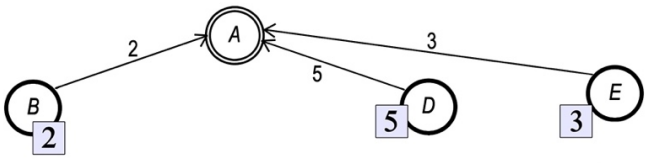
- (4a) Búsqueda Primero en Anchura (de izquierda a derecha)
- (4b) Búsqueda Primero en Profundidad (de derecha a izquierda)
- (4c) Búsqueda de Coste Uniforme
- (4d) Búsqueda en Anchura Iterativa (de derecha a izquierda)
- (4e) Búsqueda en Profundidad Iterativa (de izquierda a derecha)

Para cada nodo de TABLA_A incluya la siguiente información: su nodo padre y el coste al nodo inicial. (En este ejercicio no es necesario incluir en TABLA_A información sobre los hijos de cada nodo expandido, ya que sólo existe un camino desde cada nodo al nodo inicial y, por tanto, el mejor camino desde cada nodo al nodo inicial no cambia a lo largo del proceso de búsqueda.)



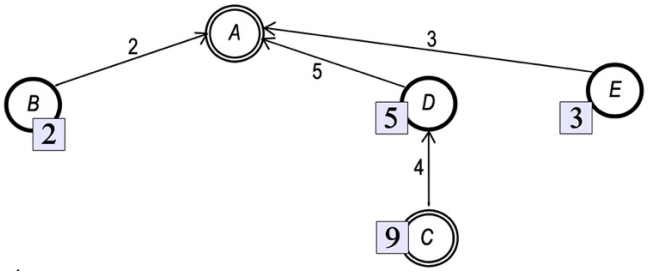
De cara a ilustrar la respuesta convenientemente, incluimos la información de TABLA_A gráficamente: por un lado, para cada nodo generado en una expansión trazamos un arco ascendente a su padre y, por otro lado, indicamos el coste al nodo inicial al lado de cada nodo generado.

(4a)Búsqueda Primero en Anchura (de izquierda a derecha) : Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación expandimos el nodo inicial, generando sus nodos hijos. Desde cada nodo hijo trazamos un nodo ascendente a su nodo padre. Al lado de cada nodo hijo indicamos el coste desde dicho nodo al nodo inicial.



- Expandimos *B* y TABLA_A no varía. A continuación expandimos *D*:

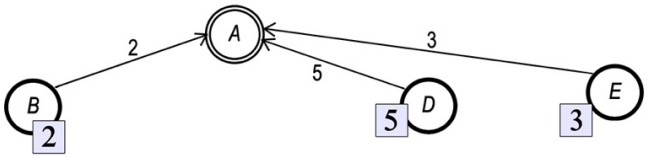
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W



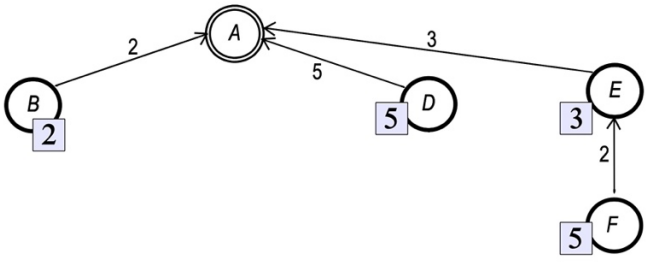
- Expandimos *E* y TABLA_A no varía. A continuación elegiríamos *C* para su expansión y llegaríamos a la meta. El camino hallado hasta la meta tiene coste 9.

(4b) Búsqueda Primero en Profundidad (de derecha a izquierda) :Cuando sea necesario, en este algoritmo aplicaremos la función LimpiarTABLA_A. Tras la extracción de un nodo de ABIERTA y su posible expansión, dicha función elimina aquellos nodos que ya no son necesarios en el proceso de búsqueda de la meta. Esto permite preservar la linealidad de la complejidad espacial de este algoritmo con respecto a la profundidad de la solución.

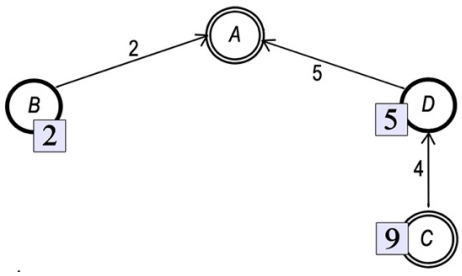
- Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial.



- Expandimos *E*. Seguidamente, aplicamos la función LimpiarTABLA_A a *E* por no tener hijos en ABIERTA y es sacado de TABLA_A. A continuación expandimos *D*:



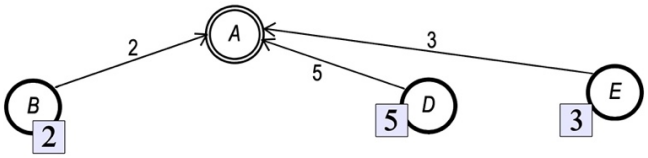
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W



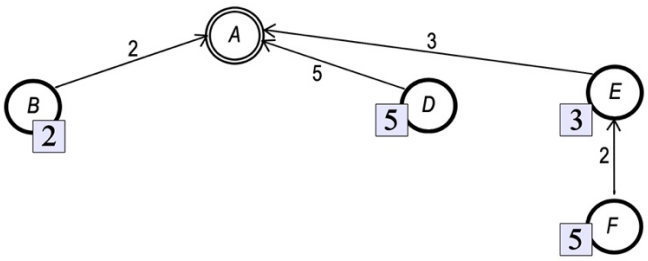
- A continuación elegiríamos *C* para su expansión y llegaríamos a la meta. El camino hallado hasta la meta tiene coste 9.

(4c) Búsqueda de Coste Uniforme

- Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial.

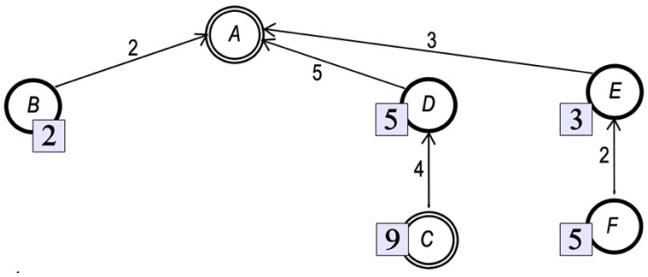


- Expandimos *B* y TABLA_A no varía. A continuación expandimos *E*:



Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

- Expandimos *D:*

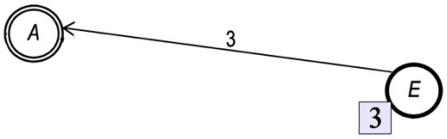


- Al intentar expandir *C*, alcanzamos la meta. El coste del camino solución hallado es 9.

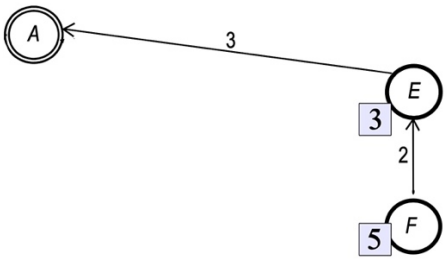
(4d) Búsqueda en Anchura Iterativa (de derecha a izquierda)

Iteración 1 (se genera como máximo 1 nodo hijo en cada expansión de un nodo padre):

- Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial:



- Expandimos *E*:

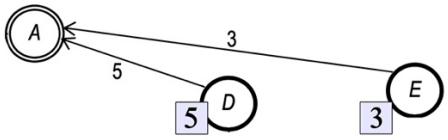


- Expandimos *F*, que no tiene hijos, con lo que TABLA_A no cambia y esta iteración termina.

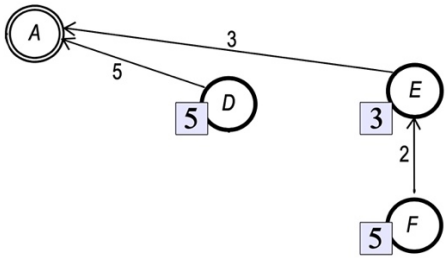
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

Iteración 2 (se generan como máximo 2 nodos hijo en cada expansión de un nodo padre):

Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial:

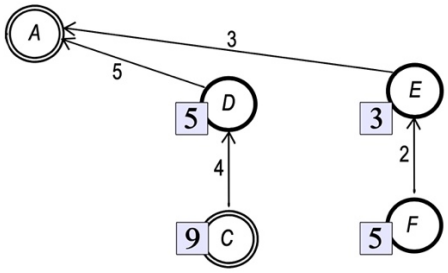


- Expandimos *E*:



- Expandimos *E*, que no tiene hijos, con lo que TABLA_A no cambia.

- Expandimos *D*:



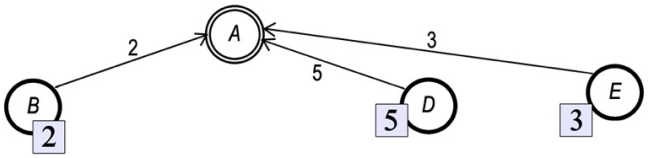
- Finalmente, al intentar expandir *C* alcanzamos la meta. El camino encontrado hasta el nodo inicial tiene coste 9.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

(4e) Búsqueda en Profundidad Iterativa (de izquierda a derecha)

Iteración 1 (profundidad límite igual a 1):

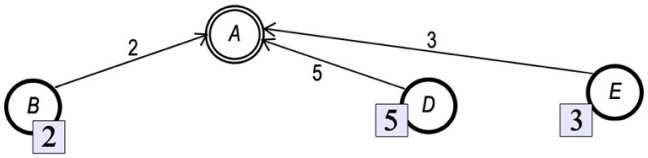
- Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial:



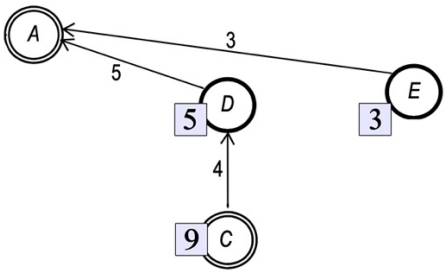
- Tras comprobar que *B* no es nodo meta, se le aplica la función LimpiarTABLA_A por estar en la profundidad límite y es sacado de TABLA_A. De igual manera, tras comprobar respectivamente que *D* y *E* no son nodo meta, se les aplica la función LimpiarTABLA_A por estar en la profundidad límite y son sacados de TABLA_A. Seguidamente, tras aplicarle la función LimpiarTABLA_A, *A* es sacado de TABLA_A por no tener hijos en ABIERTA. A continuación pasamos a la siguiente iteración.

Iteración 2 (profundidad límite igual a 2):

- Inicialmente, *A* sería incluido en TABLA_A. Gráficamente esto se correspondería con un único nodo *A*. A continuación, expandimos el nodo inicial:



- Expandimos *B*. Seguidamente, aplicamos la función LimpiarTABLA_A a *B* por no tener hijos en ABIERTA y es sacado de TABLA_A. A continuación, expandimos *D*:



- Tras elegir *C* para su expansión y comprobar que es nodo meta, finaliza el algoritmo habiendo hallado un camino entre el nodo inicial y el nodo meta de coste 9.

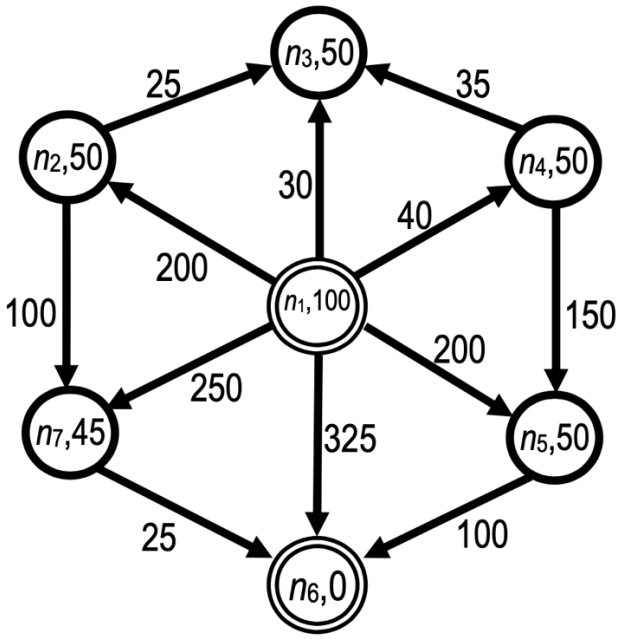
Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

EJERCICIO 5:

Considere el grafo de la figura 5.1, donde el nodo inicial es $n1$ y donde el nodo meta es $n6$. Cada arco u operador lleva asociado su coste y en cada nodo aparece la estimación de la menor distancia desde ese nodo a una meta. Aplique paso a paso el **algoritmo A*** al grafo dado, indicando de forma razonada la siguiente información en cada paso del algoritmo:

1. Qué nodo es expandido.
2. Cuál es el contenido de ABIERTA tras la expansión del nodo, indicando el valor de la función de evaluación heurística para cada nodo de ABIERTA.
3. Cuál es el contenido de TABLA_A tras la expansión del nodo. Para cada nodo de TABLA_A incluya la siguiente información:
    1. a) Su nodo padre que indique el camino de menor coste hasta el nodo inicial encontrado hasta el momento
    2. b) El coste del camino de menor coste hasta el nodo inicial encontrado hasta el momento
    3. c) Sus nodos hijos (si el nodo de TABLA_A actual ya ha sido expandido)

Por último, ¿cuál es el camino solución hallado y su coste?



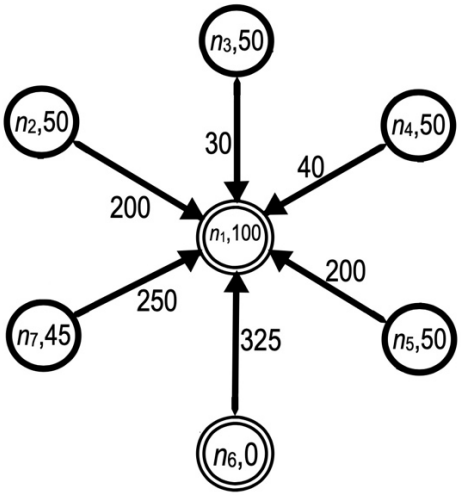
De cara a ilustrar la respuesta convenientemente, incluimos la información pedida sobre TABLA_A gráficamente. Para ello es necesario trazar, para cada nodo generado en una expansión, un arco ascendente a su padre expandido; además, hay que anotar para cada nodo su mejor padre encontrado hasta el momento (punto 3a del enunciado). De esta manera, siguiendo cada arco al mejor padre, se puede saber cuál es el mejor camino encontrado hasta el momento desde cada nodo al nodo inicial (punto 3b del enunciado). Además, los arcos ascendentes que llegan a un nodo ya expandido lo enlazan a sus nodos hijos (punto 3c del enunciado).

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

- PASO 0. El nodo inicial $n1$ es introducido en ABIERTA y en TABLA_A, con lo que tenemos la siguiente situación:

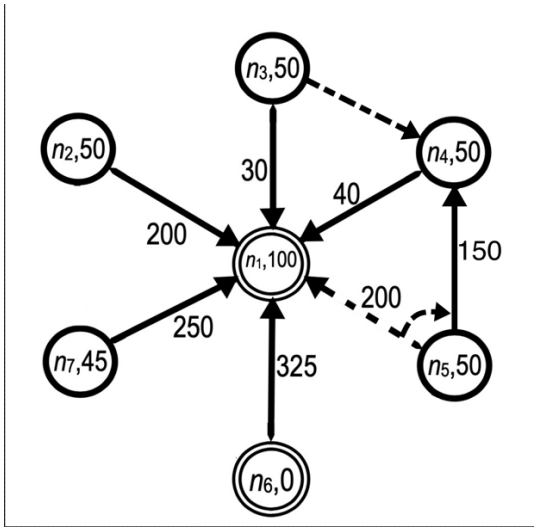
ABIERTA: { n_1 (0+100) }

- PASO 1. Expandimos el nodo $n1$ de ABIERTA. Tras la expansión, la situación es la siguiente:



ABIERTA: { n_3 (30+50) , n_4 (40+50), n_5 (200+50) , n_2 (200+50), n_7 (250+45) , n_6 (325+0) }

- PASO 2 y 3. Expandimos $n3$ por ser el nodo de ABIERTA con menor valor de la función de evaluación heurística, f=g+h (al ser g=30 y h=50), y nada cambia en TABLA_A. A continuación expandimos $n4$:



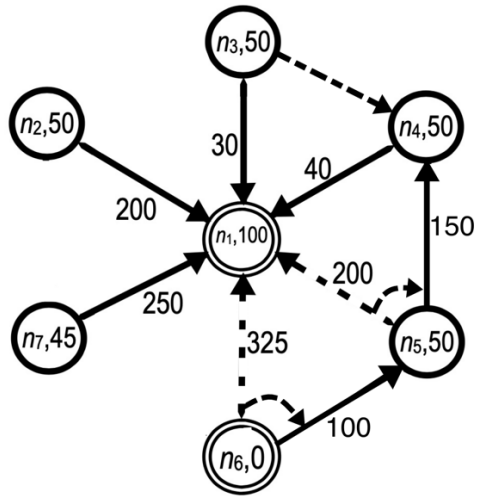
ABIERTA: { n_5 (190+50) , n_2 (200+50), n_7 (250+45) , n_6 (325+0) }

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

Usamos trazo continuo para formar el grafo hasta el momento y los discontinuos para los casos descartados por tener mayor coste.

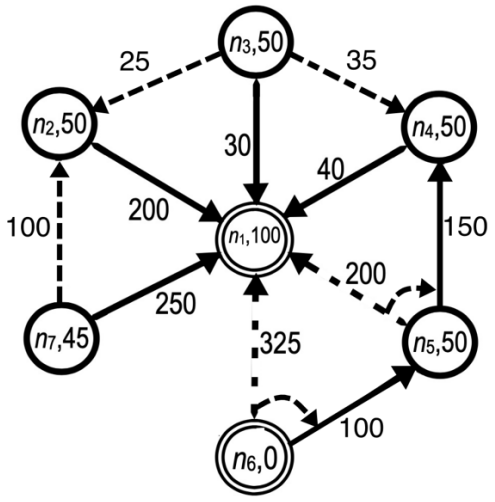
En la decisión de escoger el manor coste del camino hay reorientaciones.

- PASO 4. Expandimos *n*5 por ser el nodo de ABIERTA con menor valor de la función de evaluación heurística, $f=g+h$:



ABIERTA: { n_2 (200+50), n_6 (290+0) n_7 (250+45) }

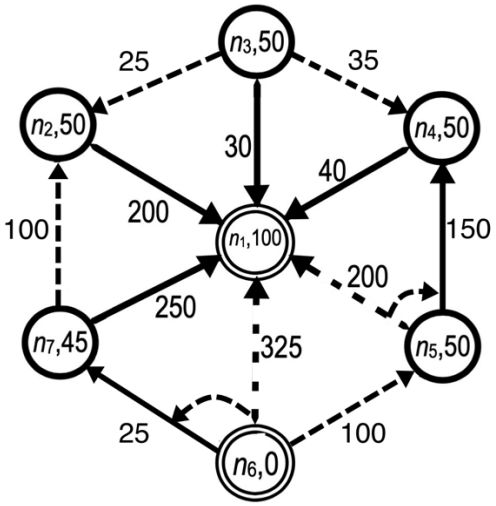
- PASO 5. Expandimos *n*2:



ABIERTA: { n_6 (290+0) n_7 (250+45) }

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

- PASO 6. Expandimos *n*6



:

Alcanzamos una meta, con lo que el algoritmo termina. El camino solución encontrado es: *n*1 → *n*7 → *n*6, cuyo coste es 250+25=275.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

EJERCICIO 6:

Considere el grafo de la figura 6.1, donde el nodo inicial es *D* y donde los nodos meta son desconocidos. Cada arco u operador lleva asociado su coste y en cada nodo aparece su valor de la función de evaluación heurística (que hay que minimizar). Aplique paso a paso el **algoritmo de escalada o máximo gradiente** al grafo dado. Para ello indique de forma razonada qué nodo se expande en cada paso y cuál es el nodo final devuelto por el algoritmo. Utilice como *criterio de selección* el de mejor vecino. Utilice como *criterio de terminación* el que no se hayan producido mejoras durante los tres últimos pasos del algoritmo.

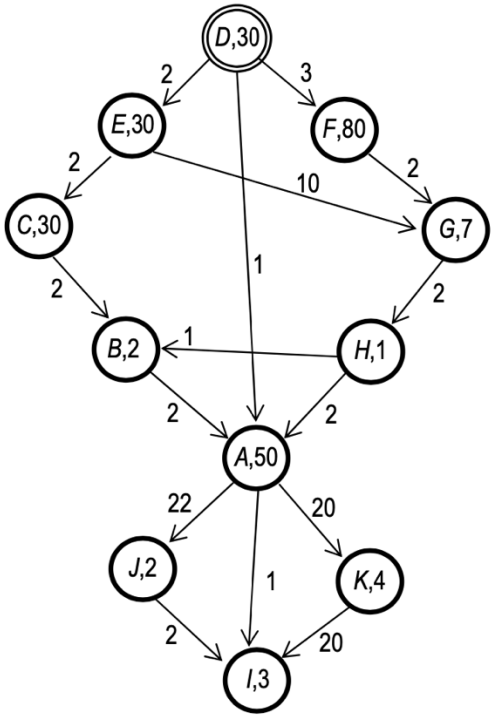


Figura 6.1: Grafo de búsqueda en el que el nodo inicial es *D*, los nodos meta son desconocidos, el coste de cada operador aparece al lado del arco que lo representa y al lado de cada nodo aparece el valor de su función de evaluación heurística (que hay que minimizar).

- PASO 1: Al principio expandimos el nodo inicial *D*, generando sus nodos hijos {*E*(30), *A*(50), *F*(80)}. Seleccionamos el nodo *E* por ser el mejor de los nodos hijos. A continuación aceptamos el nodo *E* como nuevo nodo actual en sustitución de *D*, debido a que el valor de la función de evaluación heurística de *E* es mejor o igual que el de *D* (30≤30).

- PASO 2: Expandimos el nodo actual *E* y generamos sus hijos: { *G*(7) *C*(30)}. Seleccionamos el nodo *G* por ser el mejor de los nodos hijos. Seguidamente aceptamos el nodo *G* como nuevo nodo actual en sustitución de *E*, debido a que el valor de la función de evaluación heurística de *G* es mejor o igual que el de *E* (7≤30). (Observe que la condición de terminación del algoritmo todavía no se cumple tras este paso, ya que sólo hemos completado dos pasos sin mejora, cuando la condición de terminación del enunciado nos indica que tienen que ser tres.)

- PASO 3: Expandimos el nodo actual *G* y generamos sus hijos: {*H*(1)}. Seleccionamos el nodo *H* por ser el mejor de los nodos hijos. A continuación aceptamos el nodo *H* como nuevo nodo actual en sustitución de *G*, debido a que el valor de la función de evaluación heurística de *H* es mejor o igual que el de *G* (se cumple que 1≤7).

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W

- **PASO 4:** Expandimos el nodo actual *H*, generando sus nodos hijos {*B*(2), *A*(50)}. Seleccionamos el nodo *B* por ser el mejor de los nodos hijos. A continuación rechazamos el nodo *B* como nuevo nodo actual en sustitución de *H*, debido a que el valor de la función de evaluación heurística de *B* no es mejor o igual que el de *H* (no se cumple que $2 \leq 1$).

La búsqueda terminaría en este punto. El algoritmo devolvería el nodo *H*(1) como mejor nodo encontrado.

Pablo Suárez Peña
psuarez142@alumno.uned.es
53647180W