

Algoritmo de búsqueda primero en profundidad

Stalin Muñoz Gutiérrez

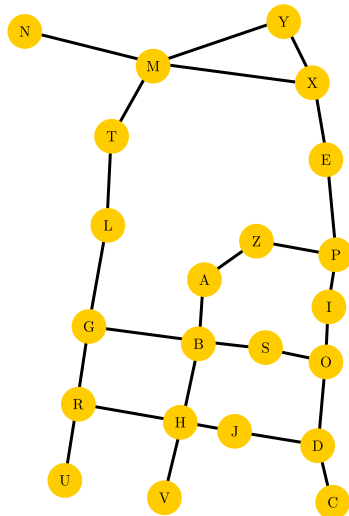
Centro de Ciencias de la Complejidad
Universidad Nacional Autónoma de México (UNAM)

Estamos listos para abordar el primer algoritmo de búsqueda no informada.

El algoritmo primero en profundidad.

- └ Algoritmo primero en profundidad (Depth-first search)

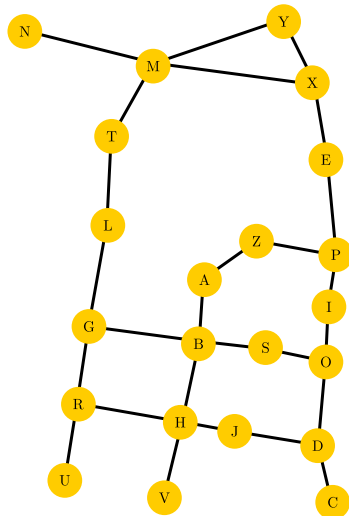
En adelante nos referiremos al algoritmo primero en profundidad como algoritmo *DFS* por sus siglas en inglés que significan *Depth-first search*.



Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:



Algoritmo de búsqueda primero en profundidad

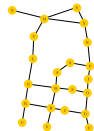
2018-09-13

└ Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search - DFS)

Usa dos estructuras de datos:



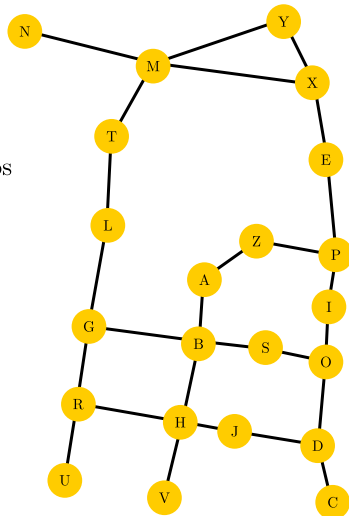
Para explorar el grafo de estados acciones el algoritmo utiliza dos estructuras de datos para almacenar los estados descubiertos y tomar decisiones sobre la dirección de la búsqueda.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.



La primera estructura de datos la denominaremos agenda. Otra forma de referirse a la agenda es como *estados abiertos*.

En la agenda almacenaremos los estados que se vayan descubriendo durante la búsqueda.

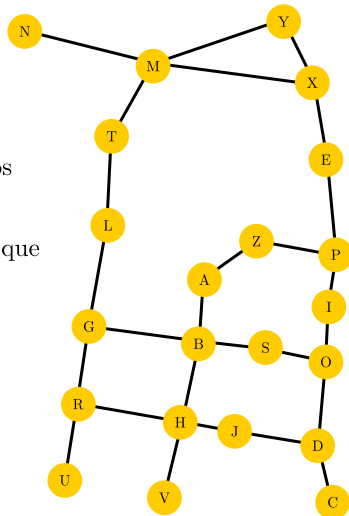
Vamos a sacar de la agenda los estados que se vayan a expandir.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.



La segunda estructura de datos es la del conjunto de estados expandidos, también denominada como *estados cerrados*.

La regla que utilizaremos en el algoritmo es que un estado no puede expandirse más de una vez.

Antes de expandir un estado verificaremos contra este conjunto si ya se expandió.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

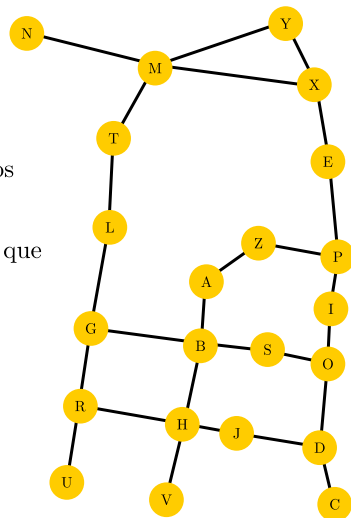
Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

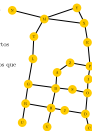
Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila
dos operaciones:
1. agregar (push)



Utilizaremos una pila o *stack* para almacenar los estados de la agenda.

Nos interesan dos operaciones sobre esta estructura de datos.

La primera es la operación *agregar*, en inglés *push*.

Partimos de una agenda vacía, es decir que no tiene elementos.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

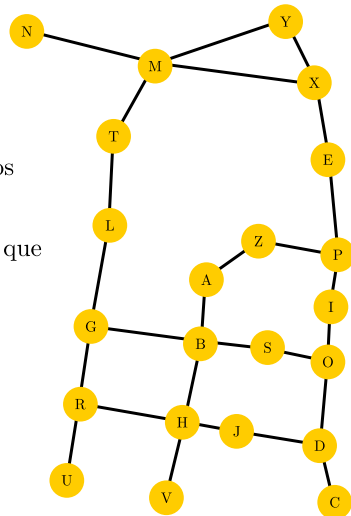
La agenda es una pila

dos operaciones:

1. agregar (push)



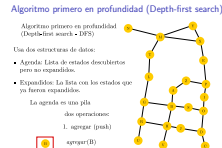
agregar(B)



Algoritmo de búsqueda primero en profundidad

2018-09-13

Algoritmo primero en profundidad (Depth-first search)



Podemos agregar el estado *B*.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

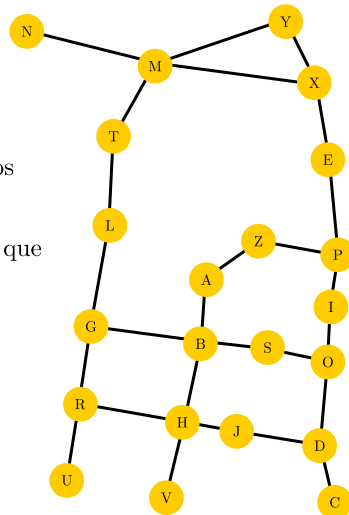
- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)

agregar(A)



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Luego agregar el estado A.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

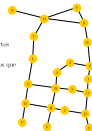
Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)
2. agregar (A)



Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

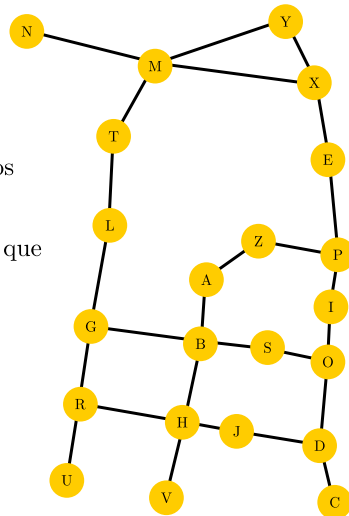
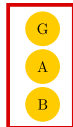
- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)

agregar(G)



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Luego G .

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

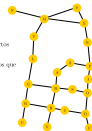
- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)

agregar(G)



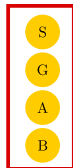
Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

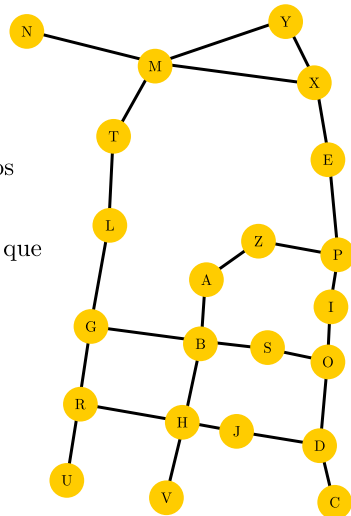
La agenda es una pila



dos operaciones:

1. agregar (push)

agregar(S)



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Agregar *S*.

Algoritmo primero en profundidad (Depth-first search)

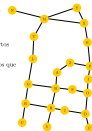
Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:
1. agregar (push)
agregar(S)



Algoritmo primero en profundidad (Depth-first search)

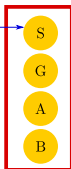
Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

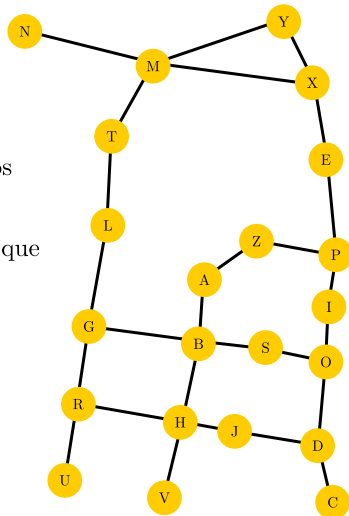
La agenda es una pila

tope →



dos operaciones:

1. agregar (push)
2. sacar (pop)



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

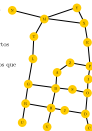
Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

- dos operaciones:
1. agregar (push)
 2. sacar (pop)



El último elemento en entrar a una pila es denominado el *tope* de la pila.

Este elemento será importante para la segunda operación que nos interesa realizar en la agenda.

La operación *sacar*, en inglés *pop*.

Algoritmo primero en profundidad (Depth-first search)

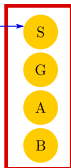
Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

tope →

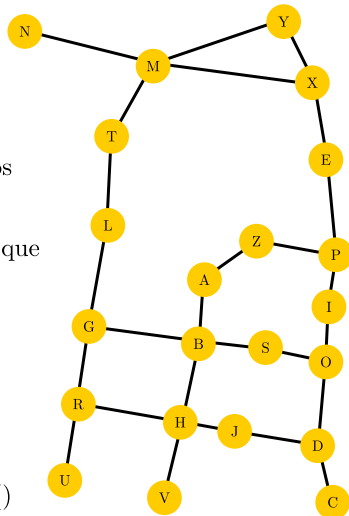


dos operaciones:

1. agregar (push)

2. sacar (pop)

sacar()



Algoritmo de búsqueda primero en profundidad

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

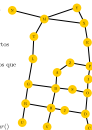
Algoritmo primero en profundidad (Depth-first search)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

- dos operaciones:
1. agregar (push)
 2. sacar (pop) *sacar()*



La operación *sacar* removera el tope de la pila y nos lo entregará como resultado.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

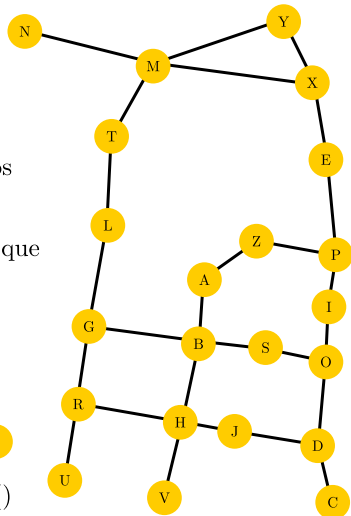
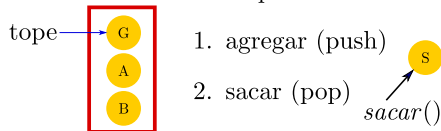
La agenda es una pila

dos operaciones:

1. agregar (push)

2. sacar (pop)

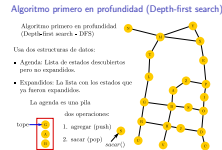
sacar()



Algoritmo de búsqueda primero en profundidad

2018-09-13

Algoritmo primero en profundidad (Depth-first search)



Aquí vemos que *sacar* nos entrega el elemento *S*.

El tope de la pila se ajusta al siguiente elemento, en este caso el penúltimo que entro, el nodo *G*.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Usa dos estructuras de datos:

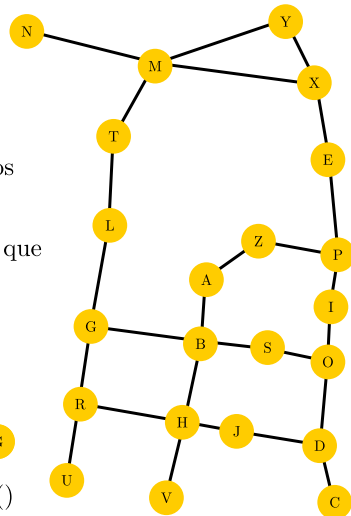
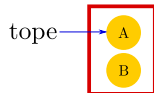
- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La agenda es una pila

dos operaciones:

1. agregar (push)

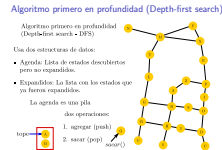
2. sacar (pop)



Algoritmo de búsqueda primero en profundidad

2018-09-13

Algoritmo primero en profundidad (Depth-first search)



Si sacamos nuevamente, obtenemos G y el nuevo tope es A .

El algoritmo agrega y saca estados de la agenda conforme avanza la exploración.

Los estados almacenados en la agenda se denominan también como la *frontera de búsqueda*.

El agente no sabe que estados va a encontrar más allá de la frontera.

Estos estados constituyen el horizonte de lo que conoce.

Algoritmo primero en profundidad (Depth-first search)

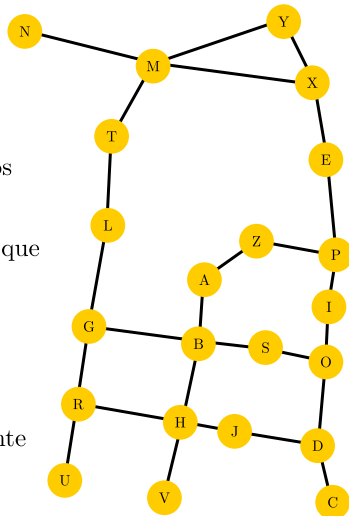
Algoritmo primero en profundidad (Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La lista de expandidos es un conjunto basado en una tabla de dispersión

- El tiempo de acceso es constante no importa cuantos elementos almacena.



Algoritmo de búsqueda primero en profundidad

2018-09-13

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

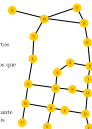
Algoritmo primero en profundidad (Depth-first search - DFS)

Usa dos estructuras de datos:

- Agenda: Lista de estados descubiertos pero no expandidos.
- Expandidos: La lista con los estados que ya fueron expandidos.

La lista de expandidos es un conjunto basado en una tabla de dispersión

- El tiempo de acceso es constante no importa cuantos elementos almacena.



La segunda estructura de datos que usaremos es un conjunto basado en una tabla de dispersión, en inglés *hash set*.

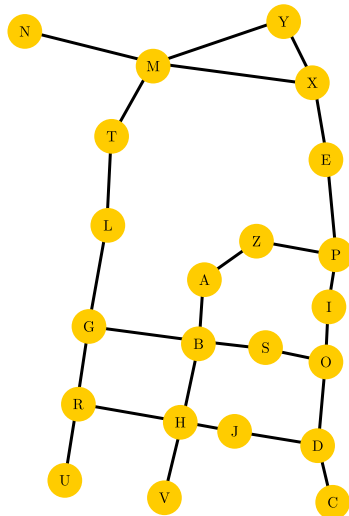
Dado que no queremos repetir el trabajo y expandir dos veces un mismo nodo, ni tampoco queremos que el agente quede atrapado en un ciclo infinito, usaremos esta lista para recordar las expansiones anteriores.

La razón de que utilicemos una tabla de dispersión para almacenar estos estados es que deseamos verificar de manera eficiente la pertenencia de un estado a esta lista.

Una tabla de dispersión puede verificar la pertenencia de un estado al conjunto en tiempo constante, esto es, se tardará lo mismo no importa si tenemos un estado o cien millones de estados.

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

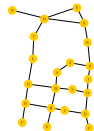


Algoritmo de búsqueda primero en profundidad

2018-09-13

└ Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad (Depth-first search)

Algoritmo primero en profundidad
(Depth-first search - DFS)

Ilustremos el funcionamiento del algoritmo resolviendo el problema de encontrar la ruta de metro Bellas Artes a metro Pino Suarez.

Esto es, la ruta del nodo B a nodo P en el grafo.

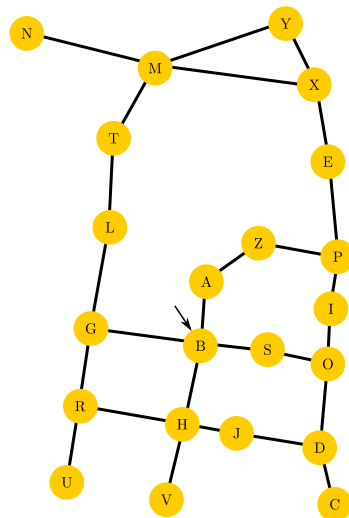
- └ Encontrar ruta de B a P usando algoritmo DFS

El primer paso del algoritmo es verificar la condición trivial.
Esto es determinar si la condición de paro se satisface para el nodo al.

En este caso la condición de paro se cumple para el nodo P y no para el nodo B .

Agregamos el nodo inicial a la agenda.

Ahora vamos a entrar a repetir los pasos siguientes hasta que la meta se cumpla o la agenda ya no tenga nodos por expandir.



Expandidos

- └ Encontrar ruta de B a P usando algoritmo DFS

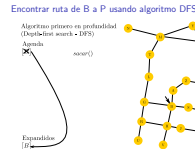
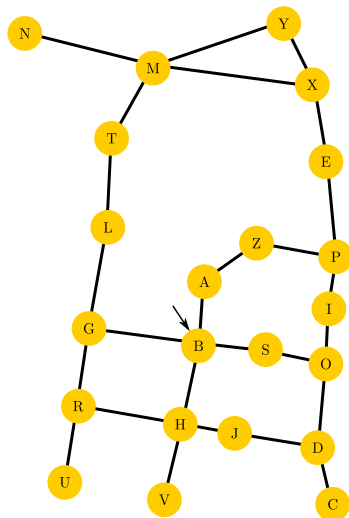
Sacamos el siguiente nodo a expandir y lo colocamos en el conjunto de nodos expandidos.

En este caso, el único elemento de la pila, el nodo B.

sacar()

$$[\mathcal{B}]$$

Expandidos

$$[B]$$


2018-09-13

- └ Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad

Depth
Ascending

[X] *expand(B)*



Expandidos
[B]

Expandir B , significa agregar a la agenda la lista de sus estados sucesores.

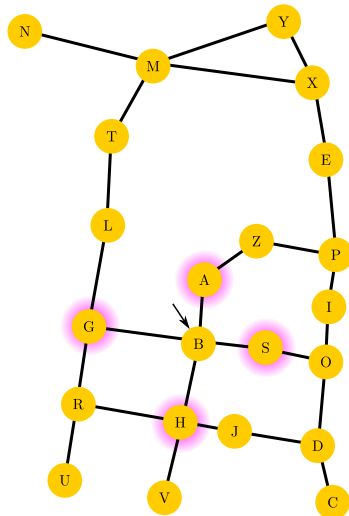
Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\boxed{B} *expandir(B)*
 $[G_B \quad A_B \quad S_B \quad H_B]$

Expandidos
 $[B]$



Algoritmo de búsqueda primero en profundidad

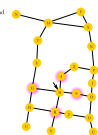
2018-09-13

└ Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 \boxed{B} *expandir(B)*
 $[G_B \quad A_B \quad S_B \quad H_B]$

Expandidos
 $[B]$



Observemos en el mapa todas las estaciones a las que podemos llegar desde B.

Verificamos si alguno es el objetivo. No es el caso, deseamos llegar a P.

Vamos a agregar los estados sucesores a la agenda siempre que no esten en el conjunto de estados expandidos.

En nuestra notación el subíndice indica de que estado venimos. Hemos terminado una iteración.

Continuamos iterando.

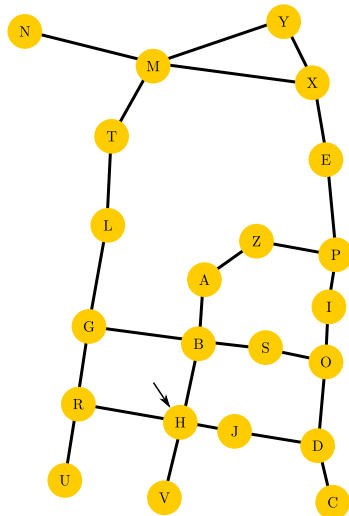
Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$

Expandidos
 $[B \ H]$



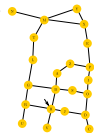
Algoritmo de búsqueda primero en profundidad

2018-09-13

└ Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 $[X]$ *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$
 Expandidos
 $[B \ H]$



Sacamos un estado del tope de la pila, en este caso el estado H .
 Lo agregamos a la lista de expandidos.

En este momento la frontera de la búsqueda tiene 4 estados.

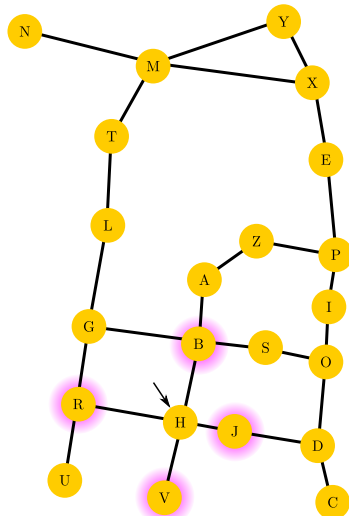
Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ $expandir(H)$
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$

Expandidos
 $[B \ H]$



Algoritmo de búsqueda primero en profundidad

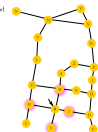
2018-09-13

└ Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 $[X]$ $expandir(H)$
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$

Expandidos
 $[B \ H]$



Expandimos el nodo H .

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$

$expandir(H)$

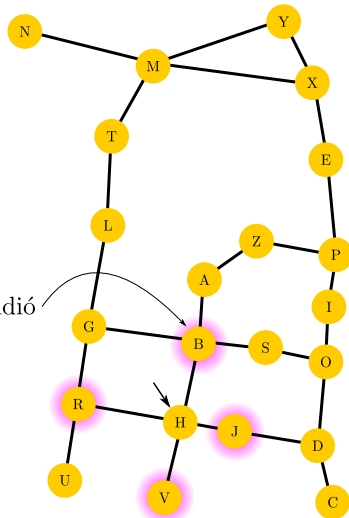
$[G_B \ A_B \ S_B \ \cancel{X_B}]$

$[G_B \ A_B \ S_B \ R_H \ J_H \ V_H]$

B es sucesor de H pero ya se expandió

Expandidos

$[B \ H]$

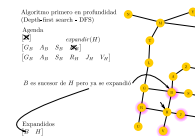


Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



Para cada sucesor de H verificamos si alguno es el objetivo.
Ninguno es el objetivo.

Agregamos a la agenda R , J , y V .

El nodo B es sucesor pero ya se expandió por lo cual no lo agregamos a la agenda.

Terminamos la segunda iteración.

Continuamos con la siguiente iteración.

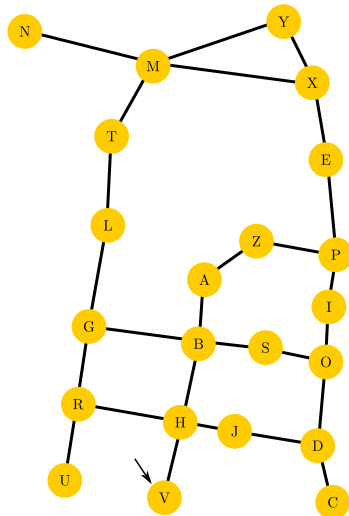
Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X_H}]$

Expandidos
 $[B \ H \ V]$

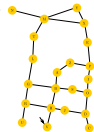


Algoritmo de búsqueda primero en profundidad

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X_B}]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X_H}]$
 Expandidos
 $[B \ H \ V]$



El tope de la pila es el nodo V , lo extraemos y agregamos a la lista de expandidos.

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\emptyset $expandir(V)$

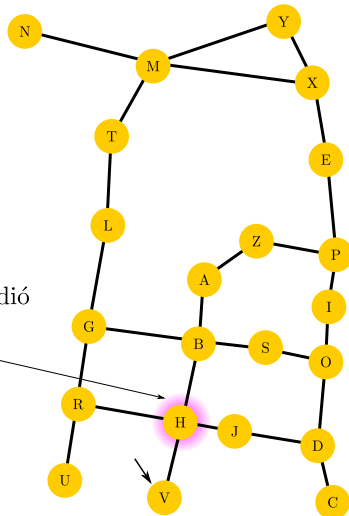
$[G_B \ A_B \ S_B \ \cancel{H}_B]$

$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{H}_H]$

$[G_B \ A_B \ S_B \ R_H \ J_H]$

El único sucesor de V ya se expandió

Expandidos
 $[B \ H \ V]$

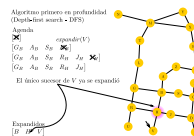


Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



El único sucesor del nodo B ya se expandió.
Entonces no agregamos sucesores a la agenda.
Siguiendo iteración...

Encontrar ruta de B a P usando algoritmo DFS

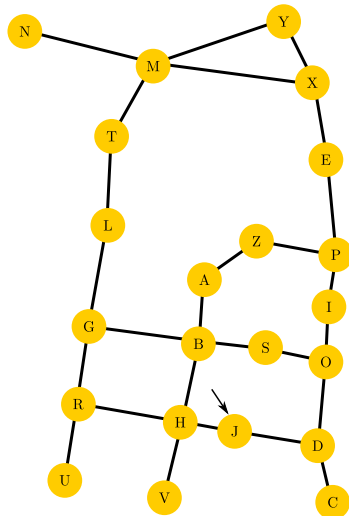
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

[X]	<i>sacar()</i>
$[G_B \ A_B \ S_B \ \cancel{X}_B]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$	

Expandidos

$[B \ H \ V \ J]$



Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

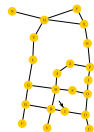
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

[X]	<i>sacar()</i>
$[G_B \ A_B \ S_B \ \cancel{X}_B]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$	

Expandidos

$[B \ H \ V \ J]$



El tope de la pila es J .

Lo sacamos y agregamos a expandidos.

Encontrar ruta de B a P usando algoritmo DFS

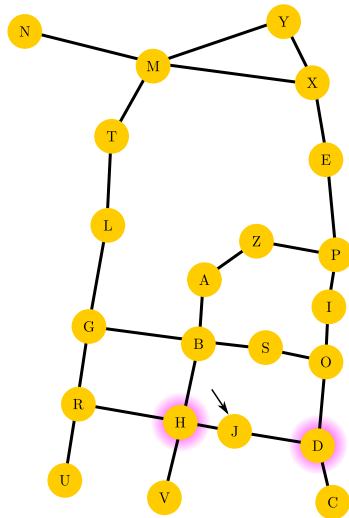
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[\emptyset]$ $expandir(J)$
 $[G_B \ A_B \ S_B \ \cancel{H}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{H}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{H}_H]$
 $[G_B \ A_B \ S_B \ R_H \ D_J]$

Expandidos

$[B \ H \ V \ J]$



Algoritmo de búsqueda primero en profundidad

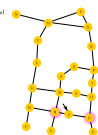
2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 $[\emptyset]$ $expandir(J)$
 $[G_B \ A_B \ S_B \ \cancel{H}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{H}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{H}_H]$
 $[G_B \ A_B \ S_B \ R_H \ D_J]$

Expandidos
 $[B \ H \ V \ J]$



El único sucesor de J no expandido previamente es D .

El nodo D no es el objetivo.

Termina la iteración.

Continuamos...

Encontrar ruta de B a P usando algoritmo DFS

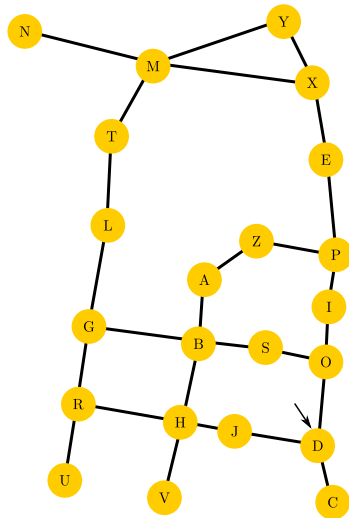
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

[X]	<i>sacar()</i>
$[G_B \ A_B \ S_B \ \cancel{X}_B]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$	

Expandidos

$[B \ H \ V \ J \ D]$



Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

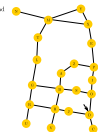
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

[X]	<i>sacar()</i>
$[G_B \ A_B \ S_B \ \cancel{X}_B]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$	

Expandidos

$[B \ H \ V \ J \ D]$



El tope de la pila es D .

Lo sacamos y ponemos en expandidos.

Encontrar ruta de B a P usando algoritmo DFS

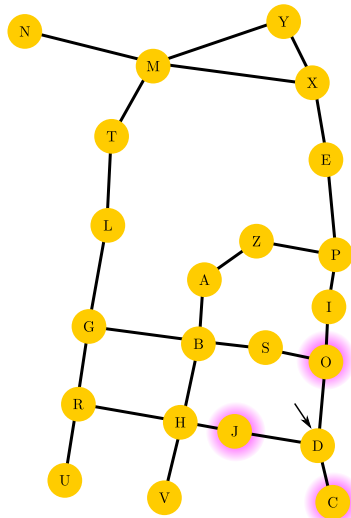
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ *expandir(D)*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ C_D]$

Expandidos

$[B \ H \ V \ J \ D]$



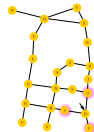
Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 $[X]$ *expandir(D)*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ C_D]$
 Expandidos
 $[B \ H \ V \ J \ D]$



D tiene 3 sucesores.

Ninguno de ellos el objetivo.

O y C se agregan a la agenda.

Siguiente iteración...

Encontrar ruta de B a P usando algoritmo DFS

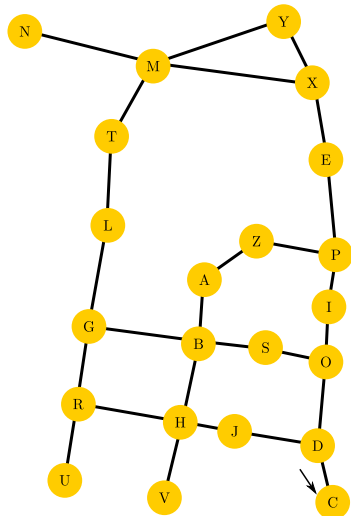
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\emptyset	<i>sacar()</i>
$[G_B \ A_B \ S_B \ \cancel{B}]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{H}]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{H}]$	
$[G_B \ A_B \ S_B \ R_H \ \cancel{J}]$	
$[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{D}]$	

Expandidos

$[B \ H \ V \ J \ D \ C]$



Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\emptyset

$[G_B \ A_B \ S_B \ \cancel{B}]$

$[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{H}]$

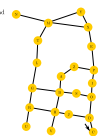
$[G_B \ A_B \ S_B \ R_H \ \cancel{H}]$

$[G_B \ A_B \ S_B \ R_H \ \cancel{J}]$

$[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{D}]$

Expandidos

$[B \ H \ V \ J \ D \ C]$



El tope es el nodo C .

Se saca y agrega a expandidos.

Encontrar ruta de B a P usando algoritmo DFS

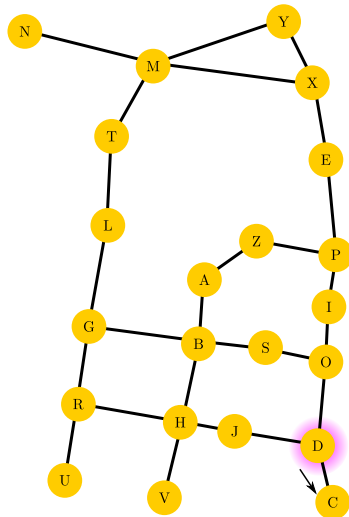
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\emptyset *expandir(C)*
 $[G_B \ A_B \ S_B \ \emptyset_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \emptyset_H]$
 $[G_B \ A_B \ S_B \ R_H \ \emptyset_H]$
 $[G_B \ A_B \ S_B \ R_H \ \emptyset_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \emptyset_D]$
 $[G_B \ A_B \ S_B \ R_H \ O_D]$

Expandidos

$[B \ H \ V \ J \ D \ C]$



Algoritmo de búsqueda primero en profundidad

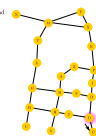
2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)
 Agenda
 \emptyset *expandir(C)*
 $[G_B \ A_B \ S_B \ \emptyset_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \emptyset_H]$
 $[G_B \ A_B \ S_B \ R_H \ \emptyset_H]$
 $[G_B \ A_B \ S_B \ R_H \ \emptyset_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \emptyset_D]$
 $[G_B \ A_B \ S_B \ R_H \ O_D]$

Expandidos
 $[B \ H \ V \ J \ D \ C]$



C no tiene sucesores que no se hayan expandido.
Termina esta iteración.

Encontrar ruta de B a P usando algoritmo DFS

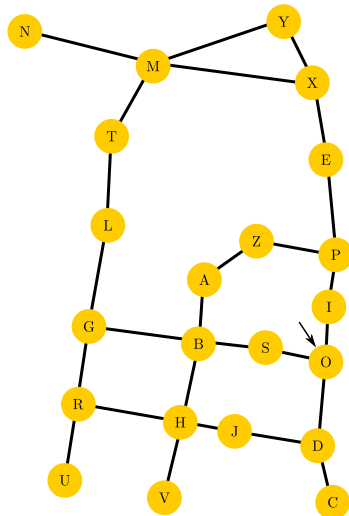
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

\emptyset	$sacar()$
$[G_B \ A_B \ S_B \ \emptyset_B]$	
$[G_B \ A_B \ S_B \ R_H \ J_H \ \emptyset_H]$	
$[G_B \ A_B \ S_B \ R_H \ \emptyset_H]$	
$[G_B \ A_B \ S_B \ R_H \ \emptyset_J]$	
$[G_B \ A_B \ S_B \ R_H \ O_D \ \emptyset_D]$	
$[G_B \ A_B \ S_B \ R_H \ \emptyset_D]$	

Expandidos

$[B \ H \ V \ J \ D \ C \ O]$

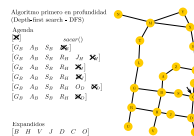


Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



Ahora el tope de la pila es el nodo O .
Lo sacamos y agregamos a expandidos.

Encontrar ruta de B a P usando algoritmo DFS

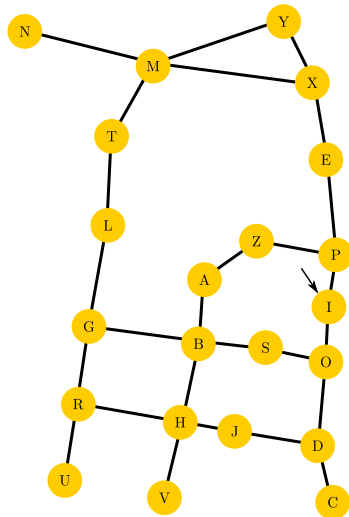
Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ S_O \ \cancel{X}_O]$

Expandidos

$[B \ H \ V \ J \ D \ C \ O \ I]$



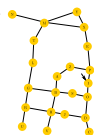
Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
 (Depth-first search - DFS)
 Agenda
 $[X]$ *sacar()*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ S_O \ \cancel{X}_O]$
 Expandidos
 $[B \ H \ V \ J \ D \ C \ O \ I]$



El tope de la pila es el nodo I .

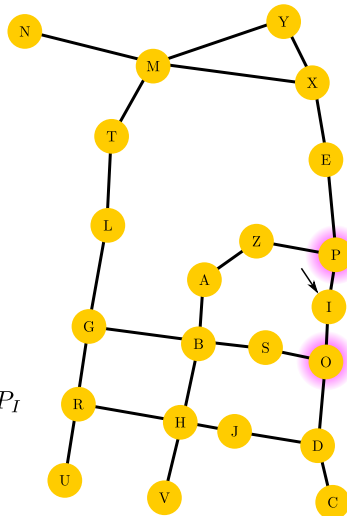
Lo sacamos y agregamos a expandidos.

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

$[X]$ *expandir(I)*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ S_O \ \cancel{X}_O] \rightarrow P_I$
 $[G_B \ A_B \ S_B \ R_H \ S_O]$
 Expandidos
 $[B \ H \ V \ J \ D \ C \ O \ I]$

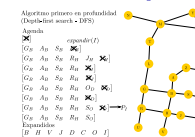


Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



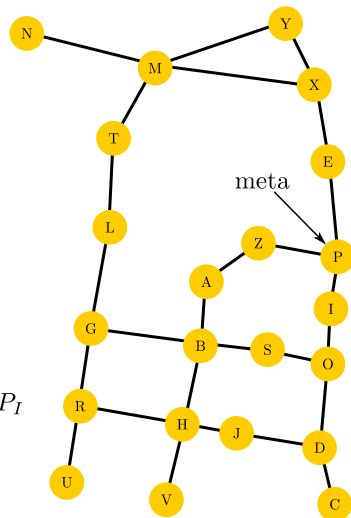
Al expandirlo descubrimos a P .

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

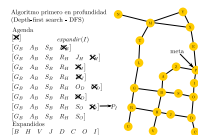
$[X]$ *expandir(I)*
 $[G_B \ A_B \ S_B \ \cancel{X}_B]$
 $[G_B \ A_B \ S_B \ R_H \ J_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_H]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_J]$
 $[G_B \ A_B \ S_B \ R_H \ O_D \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ \cancel{X}_D]$
 $[G_B \ A_B \ S_B \ R_H \ S_O \ \cancel{X}_O] \rightarrow P_I$
 $[G_B \ A_B \ S_B \ R_H \ S_O]$
 Expandidos
 $[B \ H \ V \ J \ D \ C \ O \ I]$



Algoritmo de búsqueda primero en profundidad

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



P es el nodo objetivo.

En el algoritmo DFS la primera solución encontrada es la que regresaremos.

Encontrar ruta de B a P usando algoritmo DFS

Algoritmo primero en profundidad
(Depth-first search - DFS)

Agenda

ruta = [B H J D O I P]

[X]

[G_B A_B S_B ~~X_B~~]

[G_B A_B S_B R_H J_H ~~X_H~~]

[G_B A_B S_B R_H ~~X_H~~]

[G_B A_B S_B R_H ~~X_J~~]

[G_B A_B S_B R_H O_D ~~X_D~~]

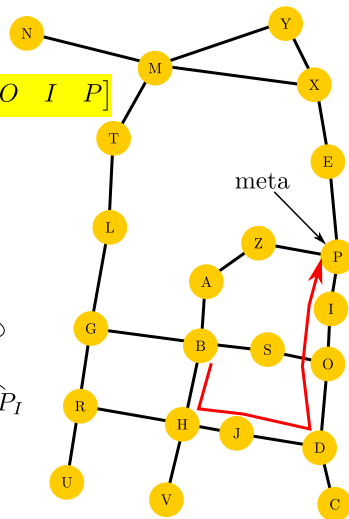
[G_B A_B S_B R_H ~~X_D~~]

[G_B A_B S_B R_H S_O ~~X_O~~]

[G_B A_B S_B R_H S_O]

Expandidos

[B H V J D C O I]

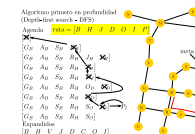


Algoritmo de búsqueda primero en profundidad

2018-09-13

Encontrar ruta de B a P usando algoritmo DFS

Encontrar ruta de B a P usando algoritmo DFS



Para recuperar la ruta, basta con recorrer los nodos predecesores hasta llegar al nodo inicial.

La referencia al nodo padre o predecesor será necesaria para recuperar la ruta.

Nótese que la ruta no incluye todos los nodos expandidos.

También observese que en este caso DFS no nos regresó la ruta óptima en número de pasos.