

Algoritmo de búsqueda con profundidad limitada

Stalin Muñoz Gutiérrez

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

Como ya hemos comentado con anterioridad, los algoritmos DFS y BFS son la base para construir algoritmos más sofisticados.

Ahora toca el turno del algoritmo de búsqueda con profundidad limitada, también conocido como DLS por sus siglas en inglés que significan Depth-limited search.

2018-09-24

└ Algoritmo de búsqueda con profundidad
limitada (Depth-limited search)

Podemos definir el algoritmo DLS como un algoritmo DFS con ciertas modificaciones.

Algoritmo de búsqueda con profundidad limitada (Depth-limited search)

DLS definido a partir de DFS

El algoritmo DLS funciona como un algoritmo DFS con dos diferencias:

Algoritmo de búsqueda con profundidad limitada (Depth-limited search)

La primera modificación consiste en que el algoritmo no puede sobrepasar una cota de profundidad que se le establece al invocarlo. Esto es, todo estado que este más allá de la cota de profundidad será ignorado por el algoritmo.

Algoritmo de búsqueda con profundidad limitada (Depth-limited search)

DLS definido a partir de DFS

El algoritmo DLS funciona como un algoritmo DFS con dos diferencias:

- 1 Tiene un parámetro de entrada adicional, denominado **profundidad de corte** o **cota de profundidad**. El algoritmo al explorar el grafo de estados-acciones no puede superar esta cota, es decir, no se expanden nodos que tengan una profundidad igual o mayor que la cota.

DLS definido a partir de DFS
El algoritmo DLS funciona como un algoritmo DFS con dos diferencias:

- Tiene un parámetro de entrada adicional, denominado **profundidad de corte** o **cota de profundidad**. El algoritmo al explorar el grafo de estados-acciones no puede superar esta cota, es decir, no se expanden nodos que tengan una profundidad igual o mayor que la cota.
- No existe el conjunto de expandidos.

Algoritmo de búsqueda con profundidad limitada (Depth-limited search)

Con DFS nos sorprendió que el crecimiento de la agenda o frontera de búsqueda es lineal con la profundidad de la exploración.

Sin embargo, el tener un conjunto de nodos explorados resultaba en un consumo de memoria exponencial.

La segunda modificación a DFS propone una respuesta a este problema ya que en DLS no se utiliza un conjunto de nodos expandidos.

Es decir toda, la memoria utilizada es la que consume la pila de la frontera de búsqueda.

Algoritmo de búsqueda con profundidad limitada (Depth-limited search)

DLS definido a partir de DFS

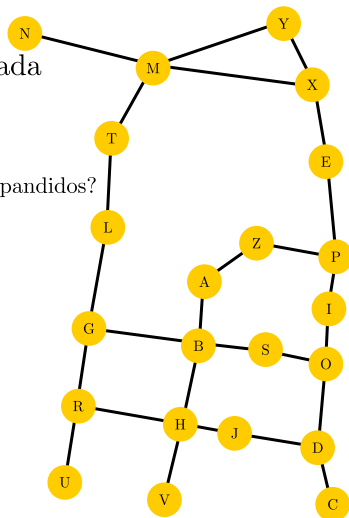
El algoritmo DLS funciona como un algoritmo DFS con dos diferencias:

- 1 Tiene un parámetro de entrada adicional, denominado **profundidad de corte** o **cota de profundidad**. El algoritmo al explorar el grafo de estados-acciones no puede superar esta cota, es decir, no se expanden nodos que tengan una profundidad igual o mayor que la cota.
- 2 No existe el conjunto de expandidos.

¿y el conjunto de expandidos?

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?



¿y el conjunto de expandidos?

¿y el conjunto de expandidos?

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?



Queremos entender cuales son las consecuencias de prescindir del conjunto de expandidos.

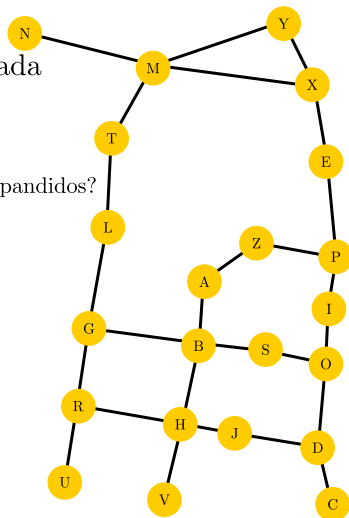
Recordemos la razón por la que fue incluido en primer lugar.

└ ¿y el conjunto de expandidos?

La primera es no trabajar doble.

Si un estado ya fue expandido, es infructuoso volver a hacerlo en

- Para evitar trabajar de más al volver a explorar estados ya conocidos.



¿y el conjunto de expandidos?

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?

- Para evitar trabajar de más al volver a ciertos estados ya conocidos.

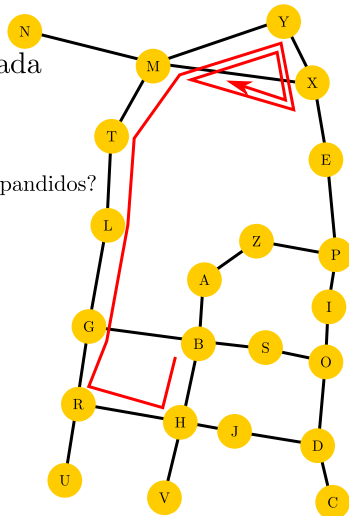


¿y el conjunto de expandidos?

Algoritmo con profundidad limitada (Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?

- Para evitar trabajar de más al volver a explorar estados ya conocidos.
- Para evitar caer en ciclos infinitos.



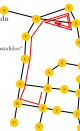
Algoritmo de búsqueda con profundidad limitada

¿y el conjunto de expandidos?

¿y el conjunto de expandidos?

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

- ¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?
- Para evitar trabajar de más al volver a explorar estados ya conocidos.
 - Para evitar caer en ciclos infinitos.





La razón más importante, es la de evitar que el agente caiga en un ciclo infinito.

¿y el conjunto de expandidos?

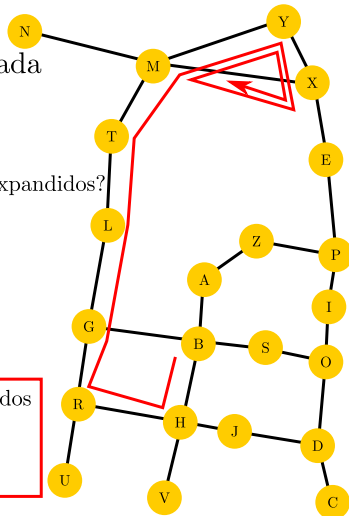
Algoritmo con profundidad limitada (Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?

 Para evitar trabajar de más al volver a explorar estados ya conocidos.

 Para evitar caer en ciclos infinitos.

Si monitoreamos la profundidad de los nodos y acotamos a una profundidad máxima sólo se puede dar un número finito de vueltas en un ciclo.




¿y el conjunto de expandidos?

¿y el conjunto de expandidos?

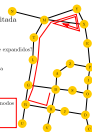
Algoritmo con profundidad limitada
(Depth-limited search - DLS)

¿Podemos simplemente quitar la lista de expandidos?
¿Por qué la incluimos originalmente?

 Para evitar trabajar de más al volver a explorar estados ya conocidos.

 Para evitar caer en ciclos infinitos.

Si monitoreamos la profundidad de los nodos y acotamos a una profundidad máxima sólo se puede dar un número finito de vueltas en un ciclo.



Al controlar la profundidad máxima de la búsqueda, DLS no podrá caer en un ciclo infinito.

A lo más dará vueltas un número finito de veces.

Esto puede sonar muy ineficiente.

Sin embargo el ahorro en memoria es sustancial.

Además, se tiene control total sobre los recursos computacionales del algoritmo a través de la cota de profundidad.

Ejemplo: Encontrar ruta de A a P

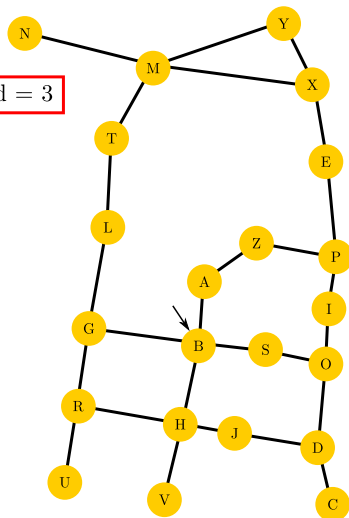
Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

[B]

cota de profundidad = 3

← Estado inicial



Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada

(Depth-limited search - DLS)

Agenda

[B]

cota de profundidad = 3

← Estado inicial



Vamos a ilustrar la operación del algoritmo DLS en el ejemplo de encontrar la ruta de la estación Bellas Artes a la estación Pino Suarez.

El algoritmo agrega el estado inicial a la agenda.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

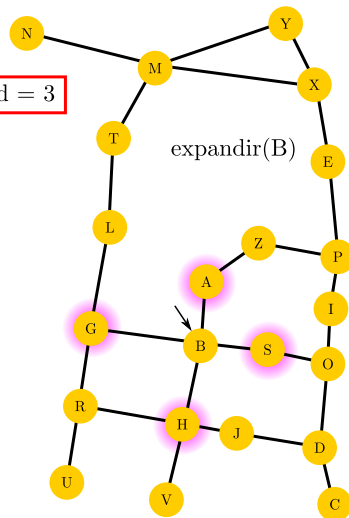
Agenda

cota de profundidad = 3

$[\emptyset]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ H_B^1]$

Anotamos la profundidad
de cada nodo.

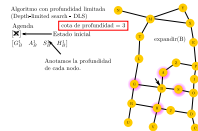


Algoritmo de búsqueda con profundidad limitada

2018-09-24

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Expandimos a B , B tiene 4 estados sucesores. B , A , S y H .

Agregamos los cuatro a la agenda.

Indicamos en el subíndice el estado del que vienen.

Como superíndice la profundidad del vértice.

Por ser la primera expansión, todos tienen profundidad 1.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

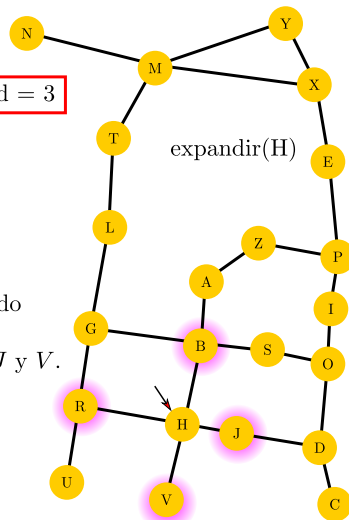
$[X]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ V_H^2]$

Al expandir H sabemos que B es el estado
que le dió origen.

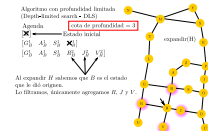
Lo filtramos, únicamente agregamos R , J y V .



Algoritmo de búsqueda con profundidad limitada

— Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Sacamos al estado H y lo expandimos.

Sus sucesores R , J y V tienen profundidad 2.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

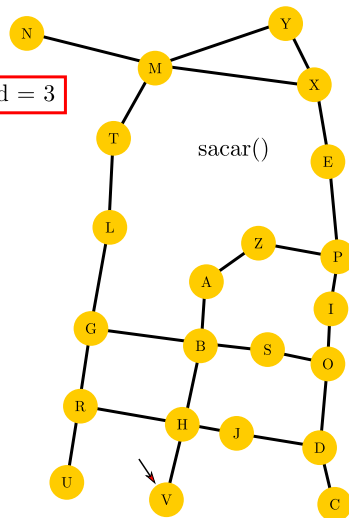
Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ V_H^2]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Sacamos el tope de la pila, el estado V.

Ejemplo: Encontrar ruta de A a P



Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

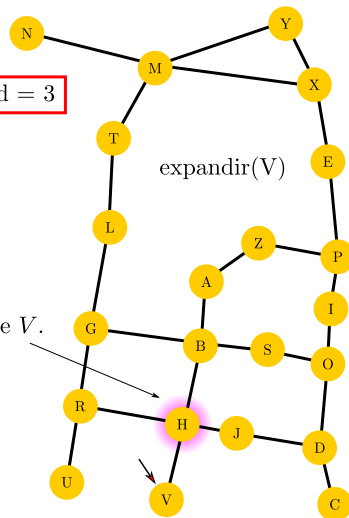
$\left[\emptyset \right] \leftarrow$ Estado inicial

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \emptyset_B^1 \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad J_H^2 \quad \emptyset_H^2 \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad J_H^2 \right]$

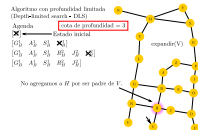
No agregamos a H por ser padre de V .



Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



No hay sucesores de V que tengamos que agregar.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

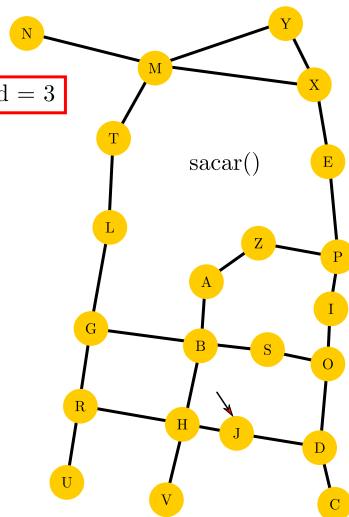
cota de profundidad = 3

$[X]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



En el siguiente paso, se saca a J de la agenda.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

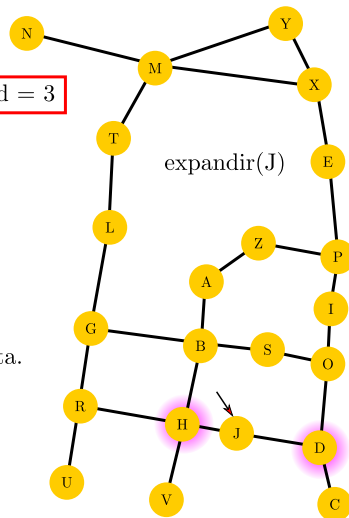
$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ D_J^3]$

La profundidad de J alcanzó la cota.



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Con su expansión se agrega el estado D .
 D ha alcanzado la cota de profundidad.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

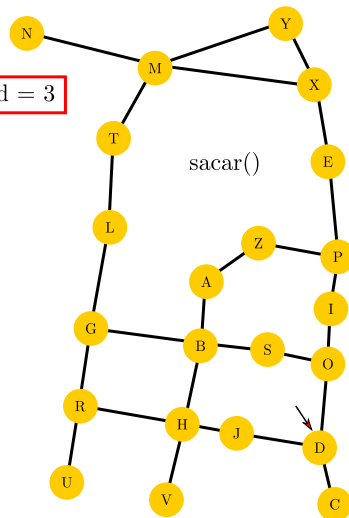
$[\mathcal{X}] \leftarrow$ Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ \mathcal{X}_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \mathcal{X}_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \mathcal{X}_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \mathcal{X}_J^3]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Sacamos a D de la agenda.

Ejemplo: Encontrar ruta de A a P



Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

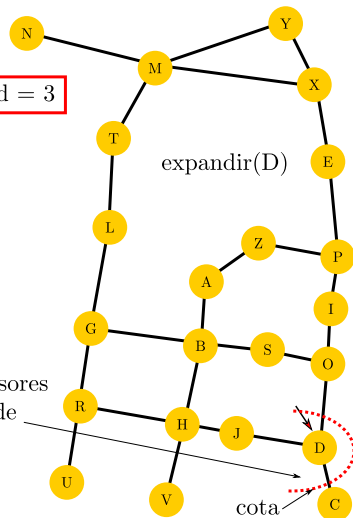
$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_J^3]$

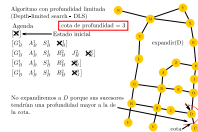
No expandiremos a D porque sus sucesores tendrían una profundidad mayor a la de la cota.



Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



DLS no puede agregar nodos que superen la cota de profundidad.
Los sucesores de D son ignorados.
El nodo simplemente no se expande.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

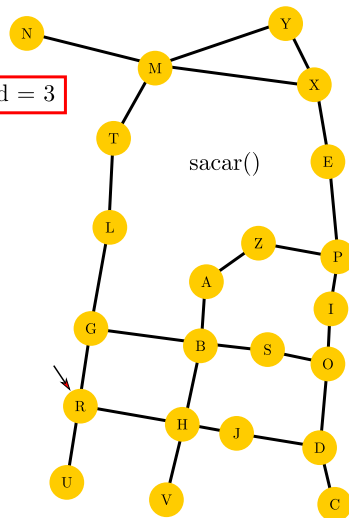
$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada

(Depth-limited search - DLS)

Agenda

Estado inicial

cota de profundidad = 3

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$



El tope de la pila ahora es R .
Lo sacamos.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[\emptyset]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ \emptyset_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \emptyset_H^2]$

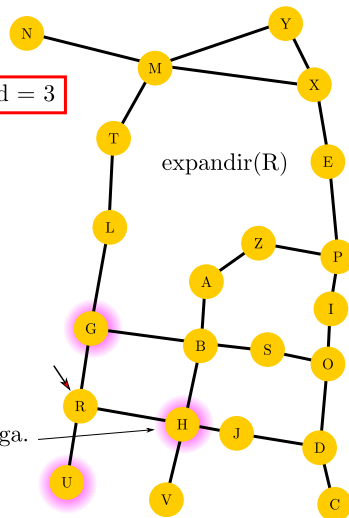
$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \emptyset_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \emptyset_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ \emptyset_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ U_R^3]$

H es el padre de R y no se agrega.



Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Al expandirlo agregamos a G y a U.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

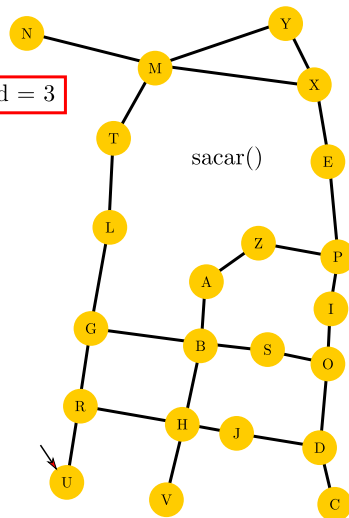
$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ X_R^3]$

Se saca U no hay expansión
ya se alcanzó la cota.



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Sacamos de la agenda a U .

No se expande porque su profundidad es 3.

Sus sucesores superarían la cota.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$\left[\emptyset \right] \leftarrow$ Estado inicial

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{H_B^1} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad J_H^2 \quad \cancel{V_H^2} \right]$

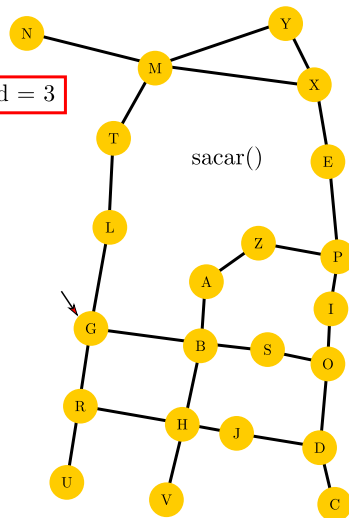
$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad \cancel{J_H^2} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad \cancel{J}^3 \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{R_H^2} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad G_R^3 \quad \cancel{V_R^3} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{G_R^3} \right]$



sacar()

Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada

(Depth-limited search - DLS)

Agenda

$\left[\emptyset \right] \leftarrow$ Estado inicial

cota de profundidad = 3

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{H_B^1} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad J_H^2 \quad \cancel{V_H^2} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad \cancel{J_H^2} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad R_H^2 \quad \cancel{J}^3 \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{R_H^2} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad G_R^3 \quad \cancel{V_R^3} \right]$

$\left[G_B^1 \quad A_B^1 \quad S_B^1 \quad \cancel{G_R^3} \right]$



Ahora sacamos a G.

Tampoco se expande.

Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$ ← Estado inicial

$[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^2]$

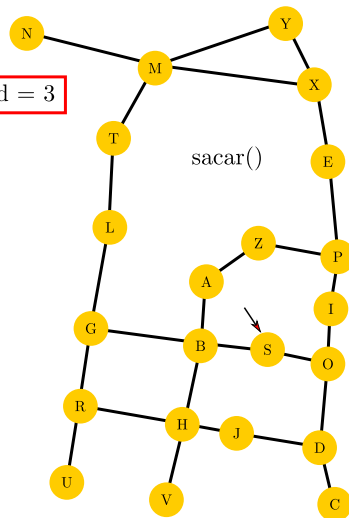
$[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_J^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$

$[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ X_R^3]$

$[G_B^1 \ A_B^1 \ S_B^1 \ X_R^3]$

$[G_B^1 \ A_B^1 \ X_B^1]$



sacar()

Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



El siguiente estado en salir es S.

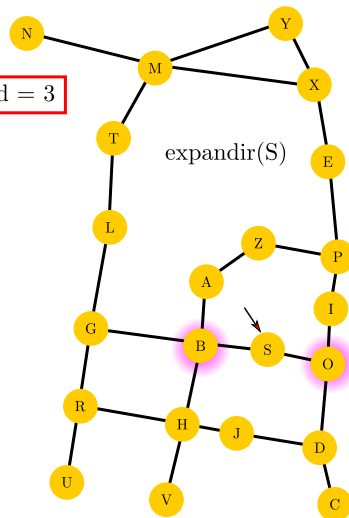
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

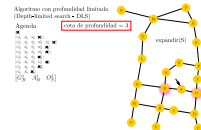
$\begin{bmatrix} \times \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_B^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & J_H^2 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & G_R^3 & \times_R^3 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_R^3 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & \times_B^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & O_S^2 \end{bmatrix}$



Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



Al expandir a S se agrega O a la agenda.

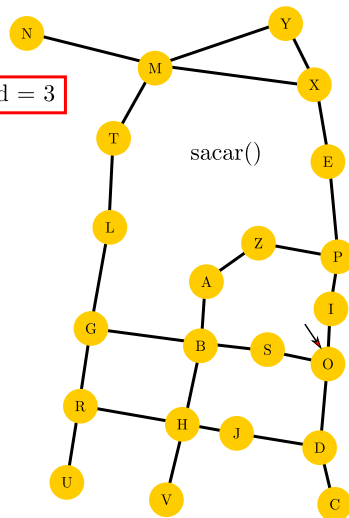
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$\{\}$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ \{ \}]$



sacar()

Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Se saca O.

Ejemplo: Encontrar ruta de A a P



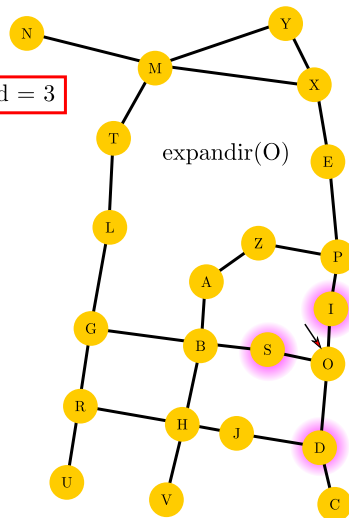
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$\begin{bmatrix} \times \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_B^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & J_H^2 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_H^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & G_R^3 & \times_R^3 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & S_B^1 & \times_R^3 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & \times_B^1 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & \times_S^2 \end{bmatrix}$
 $\begin{bmatrix} G_B^1 & A_B^1 & I_O^3 & D_O^3 \end{bmatrix}$



expandir(O)

Algoritmo de búsqueda con profundidad limitada

└ Ejemplo: Encontrar ruta de A a P

Agregamos dos sucesores, el estado I y el estado D .

Ejemplo: Encontrar ruta de A a P



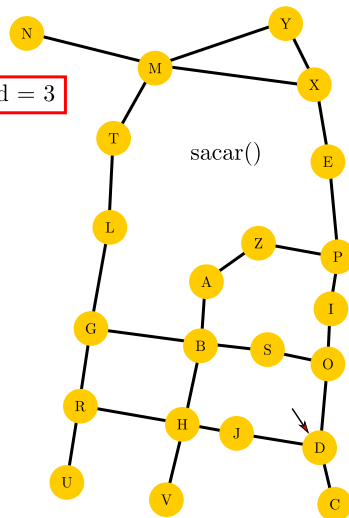
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

\times
 $[G_B^1 \ A_B^1 \ S_B^1 \ \times_B^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \times_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \times_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \times_H^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \times_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ \times_R^3]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \times_R^3]$
 $[G_B^1 \ A_B^1 \ \times_B^1]$
 $[G_B^1 \ A_B^1 \ \times_S^2]$
 $[G_B^1 \ A_B^1 \ I_O^3 \ \times_O^3]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

En el tope queda D .

Lo sacamos.

No se expandirá dado que su profundidad es igual a la cota.

Ejemplo: Encontrar ruta de A a P



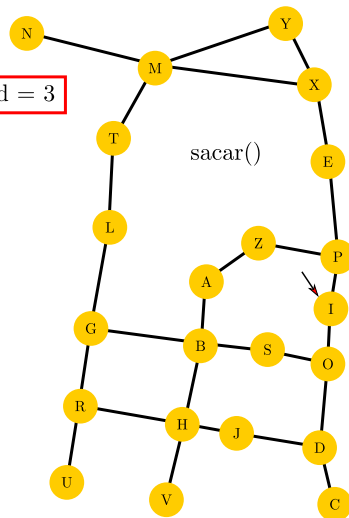
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$\{\}$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ I_O^3 \ \{ \}]$
 $[G_B^1 \ A_B^1 \ \{ \}]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Ejemplo: Encontrar ruta de A a P



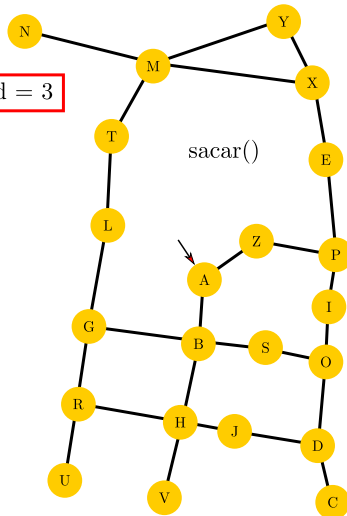
Similarmente para I , no agregamos sucesores.

- └ Ejemplo: Encontrar ruta de A a P

El tope de la pila ahora es A .
Lo sacamos.

cota de profundidad = 3

sacar()



El tope de la pila ahora es A .
Lo sacamos.

[illegible]

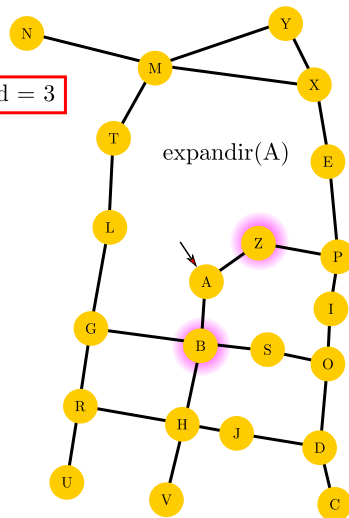
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ X_R^3]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_R^3]$
 $[G_B^1 \ A_B^1 \ X_B^1]$
 $[G_B^1 \ A_B^1 \ X_S^2]$
 $[G_B^1 \ A_B^1 \ I_O^3 \ X_O^3]$
 $[G_B^1 \ A_B^1 \ X_O^3]$
 $[G_B^1 \ X_B^1]$
 $[G_B^1 \ Z_A^2]$



Algoritmo de búsqueda con profundidad limitada

2018-09-24

└ Ejemplo: Encontrar ruta de A a P

Al expandir A agregamos a Z .

Ejemplo: Encontrar ruta de A a P



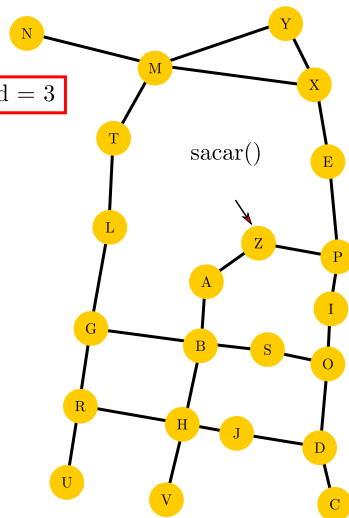
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$\left[\begin{matrix} \times \\ G_B^1 & A_B^1 & S_B^1 & \times_B^1 \end{matrix} \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & R_H^2 & J_H^2 & \times_H^2 \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^2 \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & R_H^2 & \times_H^1 \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & \times_H^2 \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & G_R^3 & \times_R^3 \right]$
 $\left[G_B^1 & A_B^1 & S_B^1 & \times_R^3 \right]$
 $\left[G_B^1 & A_B^1 & \times_B^1 \right]$
 $\left[G_B^1 & A_B^1 & \times_S^2 \right]$
 $\left[G_B^1 & A_B^1 & I_O^3 & \times_O^3 \right]$
 $\left[G_B^1 & A_B^1 & \times_O^3 \right]$
 $\left[G_B^1 & \times_B^1 \right]$
 $\left[G_B^1 & \times_A^2 \right]$



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Z es el tope.
Lo sacamos.

Ejemplo: Encontrar ruta de A a P



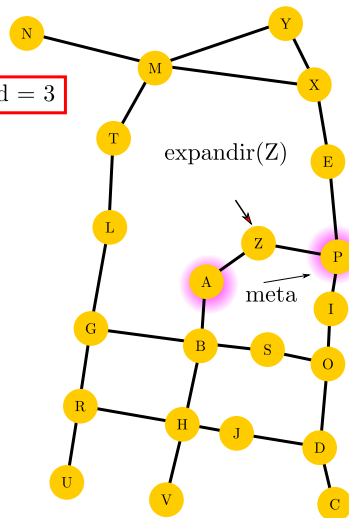
Ejemplo: Encontrar ruta de A a P

Algoritmo con profundidad limitada
(Depth-limited search - DLS)

Agenda

cota de profundidad = 3

$[X]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_B^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ X_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ X_H^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ X_R^3]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ X_R^3]$
 $[G_B^1 \ A_B^1 \ X_B^1]$
 $[G_B^1 \ A_B^1 \ X_S^2]$
 $[G_B^1 \ A_B^1 \ I_O^3 \ X_O^3]$
 $[G_B^1 \ A_B^1 \ X_O^3]$
 $[G_B^1 \ X_B^1]$
 $[G_B^1 \ X_A^2]$
 $[G_B^1] \rightarrow P_Z$



Algoritmo de búsqueda con profundidad limitada

2018-09-24

└ Ejemplo: Encontrar ruta de A a P

Al expandirlo descubrimos a P.
Nuestra meta.

Ejemplo: Encontrar ruta de A a P



Ejemplo: Encontrar ruta de A a P

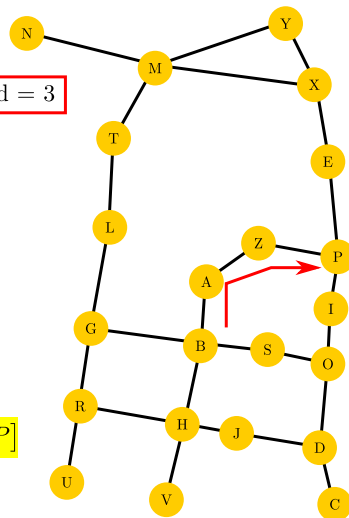
Algoritmo con profundidad limitada
(Depth-limited search - DLS)

cota de profundidad = 3

Agenda

$[G_B^1 \ A_B^1 \ S_B^1 \ \text{X}_B^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ J_H^2 \ \text{X}_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \text{X}_H^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ R_H^2 \ \text{X}_H^1]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \text{X}_H^2]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ G_R^3 \ \text{X}_R^3]$
 $[G_B^1 \ A_B^1 \ S_B^1 \ \text{X}_R^3]$
 $[G_B^1 \ A_B^1 \ \text{X}_B^1]$
 $[G_B^1 \ A_B^1 \ \text{X}_S^2]$
 $[G_B^1 \ A_B^1 \ I_O^3 \ \text{X}_O^3]$
 $[G_B^1 \ A_B^1 \ \text{X}_O^3]$
 $[G_B^1 \ \text{X}_B^1]$
 $[G_B^1 \ \text{X}_A^2]$
 $[G_B^1] \rightarrow P_Z$

ruta = [B A Z P]



Algoritmo de búsqueda con profundidad limitada

Ejemplo: Encontrar ruta de A a P

Antes de terminar, recuperamos la ruta.

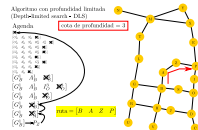
P viene de Z.

Z viene de A.

A viene de B.

La ruta es B,A,Z,P.

Ejemplo: Encontrar ruta de A a P



Análisis asintótico del algoritmo DLS

1 Memoria.

Algoritmo de búsqueda con profundidad limitada

2018-09-24

Análisis asintótico del algoritmo DLS

Análisis asintótico del algoritmo DLS

■ Memoria.

Con DLS aprovechamos la característica más atractiva del algoritmo DFS.

El crecimiento lineal de la frontera de búsqueda.

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

└ Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

2 Tiempo.

└ Análisis asintótico del algoritmo DLS

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

2 Tiempo.

- $O(b^c)$

└ Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$
- Tiempo.
 - $O(b^c)$

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

2 Tiempo.

- $O(b^c)$

3 Calidad.

Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$
- Tiempo.
 - $O(b^c)$
- Calidad.

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

2 Tiempo.

- $O(b^c)$

3 Calidad.

- Depende de c .
Es óptima únicamente si $c = d$.

Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$
- Tiempo.
 - $O(b^c)$
- Calidad.
 - Depende de c .
Es óptima únicamente si $c = d$.

Análisis asintótico del algoritmo DLS

- 1 Memoria.
 - $O(cb)$
- 2 Tiempo.
 - $O(b^c)$
- 3 Calidad.
 - Depende de c .
Es óptima únicamente si $c = d$.
- 4 Completez.

└ Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$
- Tiempo.
 - $O(b^c)$
- Calidad.
 - Depende de c .
Es óptima únicamente si $c = d$.
- Completez.

Análisis asintótico del algoritmo DLS

1 Memoria.

- $O(cb)$

2 Tiempo.

- $O(b^c)$

3 Calidad.

- Depende de c .
Es óptima únicamente si $c = d$.

4 Completez.

- No es completo para toda $c < d$

Análisis asintótico del algoritmo DLS

- Memoria.
 - $O(cb)$
- Tiempo.
 - $O(b^c)$
- Calidad.
 - Depende de c .
Es óptima únicamente si $c = d$.
- Completez.
 - No es completo para toda $c < d$

Algoritmos tipo DFS

Algoritmo	DFS	DLS	ID	DFBB
Memoria	$O(b^m)$	$O(bc)$	$O(bd)$	$O(bc)$
Tiempo	$O(b^m)$	$O(b^c)$	$O(b^d)$	$O(b^c)$
Calidad	Subóptima	Subóptima	Óptima	Óptima
Completez	Incompleto	Incompleto	Completo	Incompleto

Algoritmos tipo DFS

Algoritmo	DFS	DLS	ID	DFBB
Memoria	$O(b^m)$	$O(bc)$	$O(bd)$	$O(bc)$
Tiempo	$O(b^m)$	$O(b^c)$	$O(b^d)$	$O(b^c)$
Calidad	Subóptima	Subóptima	Óptima	Óptima
Completez	Incompleto	Incompleto	Completo	Incompleto