ISSN 0798 1015

REVISTA ESPACIOS

Vol. 40 (Nº 40) Año 2019. Pág. 14

Programación dinámica en el cálculo de la ruta óptima de una topología de red: Caso de Estudio

Dynamic programming in the calculation of the optimal route of a network topology

ACOSTA, Alberto 1; SALCEDO, Octavio 2 y RIVAS, Edwin 3

Recibido: 29/07/2019 • Aprobado: 10/11/2019 • Publicado 18/11/2019

Contenido

RESUMEN:
En este artículo se presenta un caso de estudio en el cual se quiere encontrar la ruta más corta, proporcionada por las rutas existentes entre dos nodos: fuente y destino. En primer lugar, la metodología de solución se diseñó usando la programación dinámica determinística PDD. Posteriormente se implementaron simulaciones en los diferentes nodos del sistema con el fin de evaluar las rutas del grafo, el costo total de ruta, y el vector de vértices de la ruta óptima.
Palabras clave: Procesos de Márkov, programación dinámica, grafos

ABSTRACT:
This article presents a case study in which you want to find the shortest route, provided by the existing routes between two nodes: source and destination. First, the solution methodology was designed using the dynamic deterministic PDD programming. Subsequently, simulations were implemented in the different nodes of the system in order to evaluate the graph routes, the total route cost, and the vector of vertices of the optimal route.
Keywords: Markov Processes, Dynamic Programming, Graphs

1. Introducción

La utilidad de procesos markovianos en la actualidad toma mayor relevancia en la medida que Internet crece exponencialmente y es utilizada en muchos escenarios como en el comercio electrónico, redes sociales y la trasmisión cada día más creciente a nivel de streaming que representa aproximadamente entre el 70 % y 80 % en países desarrollados. Por ello es necesario modelar utilizando grafos que permitan obtener mejores rutas a la hora de desarrollarse servicios y aplicaciones sobre Internet.

Por tanto, la programación dinámica es un método que permite resolver un problema de ***n*** variables dividiéndolo en ***n*** problemas de una variable cada uno. [1] La solución óptima de cada etapa se toma como variable de entrada de la etapa siguiente. Dicho de otro modo, el estado de la siguiente etapa está determinado por: el estado y la política de decisión de la etapa actual. Aplicando las características y estructura básicas de la programación dinámica, que se muestran

en la figura 1, el proceso se concluye iterativamente en n etapas, el objetivo es encontrar una combinación de decisiones que optimicen una cierta cantidad asociada con un sistema.

2. Metodología

Un grafo es un par de conjuntos, en donde uno de ellos contiene los vértices o nodos, mientras que el segundo es el conjunto de ramas o arcos y se encuentra formado por pares de elementos que se encuentran conectados entre sí. (Maurette & Ojea I., 2006)

Se propone un método de tres etapas el cual se entiende como un método que explora diferentes técnicas, heurísticas y exactas, para encontrar la solución. (Problemas & Escolares, 2018)

En este apartado se presentan algunas características básicas que se aplican a los problemas de programación dinámica (Taha, 2012, p. 429):

> Un problema se puede dividir en etapas, cada una de las cuales requiere una política de decisión, esto es, qué destino elegir para la siguiente etapa.
>
> Cada etapa tiene cierto número de estados asociados con su inicio. En general, los estados son las distintas condiciones posibles en las que se puede encontrar el sistema en cada etapa del problema.
>
> El objetivo de la política de decisión en cada etapa es transformar el estado actual en un estado asociado con el inicio de la siguiente etapa.
>
> Este procedimiento sugiere que los problemas de programación dinámica se pueden interpretar en términos de redes; en donde cada nodo de la red corresponde a un estado. La red consistiría en columnas de nodos, donde cada columna corresponde a una etapa, de manera tal que el flujo de salida de un nodo sólo puede dirigirse a un nodo de la siguiente columna a la derecha. En la mayoría de los casos, el objetivo consiste en encontrar la trayectoria más corta o la más larga de la red.
>
> El procedimiento de solución está diseñado para encontrar una política óptima para manejar el problema completo, es decir, una receta para elaborar la política de decisión óptima para cada etapa, en cada uno de los estados posibles. La PD (política de decisión) presenta este tipo de receta política sobre qué hacer en todas las circunstancias posibles (a esto se debe que la decisión real que se toma al llegar a un estado específico se llama política de decisión).
>
> Dado el estado actual, una política óptima para las etapas restantes es independiente de la política adoptada en etapas anteriores. Por lo tanto, la decisión óptima inmediata depende sólo del estado actual y no de cómo se llegó ahí. Éste es el principio de optimalidad de la PD. En general, en los problemas de PD, el conocimiento del estado actual del sistema expresa toda la información sobre su comportamiento anterior, información que es necesaria para determinar la política óptima de ahí en adelante. esta definición obedece a la propiedad markoviana, la cual establece que la probabilidad condicional de cualquier "evento" futuro dados cualquier "evento" pasado y el estado actual , es independiente de los eventos pasados y sólo depende del estado actual del proceso (Kashif, Talib, Ayub, Umer, & Ullah, 2017). Un problema que carezca de esta propiedad no se puede formular como un problema de programación dinámica.
>
> El procedimiento de solución comienza cuando se determina la política para la última etapa. Es decir, la política óptima para la última etapa prescribe la política óptima de decisión para cada estado posible en esa etapa. es común que la decisión de este problema de una etapa sea trivial.
>
> Se dispone de una relación recursiva que identifica la política óptima para la etapa n, dada la política óptima para la etapa . La forma precisa de la relación recursiva difiere de un problema a otro, pero se usará una notación específica que muestra en la figura 1.
>
> Cuando se usa esta relación recursiva, el procedimiento de solución comienza al final y se mueve hacia atrás etapa por etapa para encontrar cada vez la política óptima para esa etapa hasta que encuentra la política óptima desde la etapa inicial. Esta política óptima lleva de inmediato a una solución óptima para el problema completo, es decir, para el estado inicial , después para el estado que resulta, y así sucesivamente hasta para el estado resultante.

A continuación, observaremos el modelo estático y determinístico (Sudakova, Agarkov, & Tarasyev, 2018):

El modelo estático utiliza la estructura del modelado estocástico, que requiere el conocimiento de las características probabilísticas de los principales parámetros del modelo y las condiciones especiales para la implementación del proceso considerado. Para este se requieren condiciones muy estrictas para el uso del aparato de modelado estocástico, que, en la práctica, no son factibles. (Shorikov, Sudakova, Agarkov, & Tarasyev, 2018)

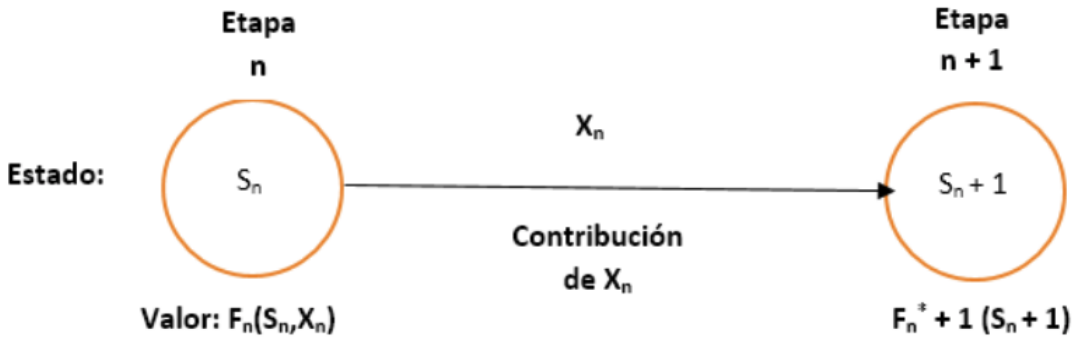
El proceso estocástico es un proceso aleatorio el cual evoluciona con el tiempo y se representa por un conjunto de funciones aleatorias. El proceso Markoviano son procesos estocásticos de decisión los cuales se basan en el concepto de que la acción a formar en un estado determinado, en un instante determinado, depende solo del estado en que se encuentre el sistema al momento de tomar la decisión (Jiménez, Mercado Caruso, Oliveira, Crissien, & Coronado-Hernandez, 2017)

El modelo determinista tiene en cuenta que la mayoría de los parámetros de los procesos considerados son deterministas y que las limitaciones geométricas de los parámetros indeterminados a priori se conocen por el experimento: los conjuntos de sus posibles valores. En los casos en que, para algunos parámetros indefinidos, en donde es posible formar restricciones geométricas sobre sus posibles valores y utilizar un enfoque determinista.

Por lo tanto, se puede concluir que el enfoque determinista y los métodos correspondientes de la teoría matemática de los procesos de control tienen aplicaciones más amplias para las tareas de toma de decisiones, (Sudakova, Agarkov, & Tarasyev, 2018)

2.1. Estructura básica de PDD

Figura 1
Estructura básica de PDD



Los datos de la figura 1, se interpretan de la siguiente manera:

n= Número de etapa

Sn = Estado actual de la etapa n

Xn = variable de decisión de la etapa

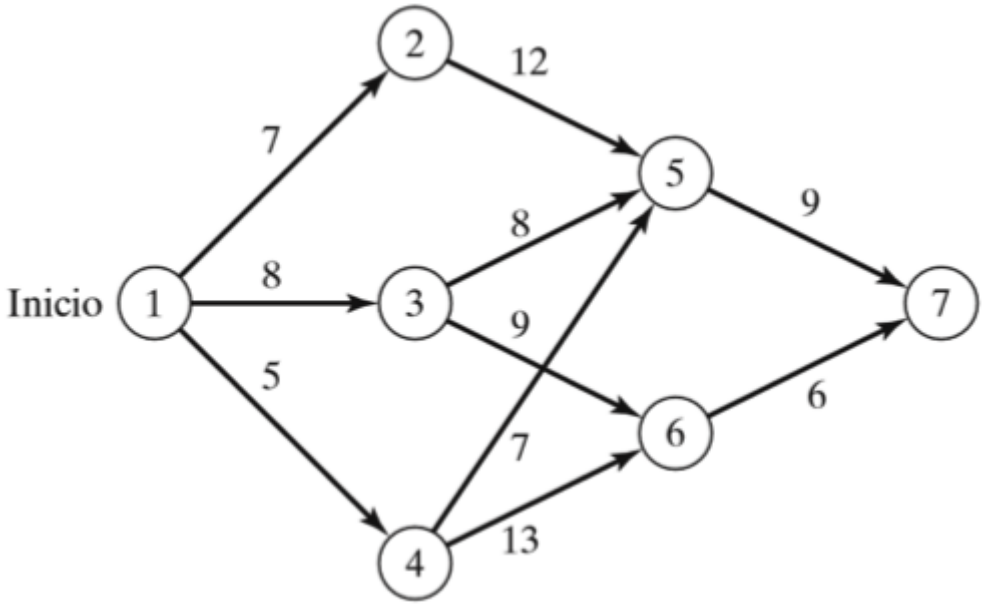
Fn (Sn, Xn) = contribución a la función objetivo de los estados n,n + 1,n+2,…,N , si el sistema se encuentra en el estado Sn en la etapa , la decisión inmediata es y en adelante se toman decisiones óptimas.

3. Resultados

3.1. Caso de estudio

Se desea seleccionar la ruta por carretera más corta entre dos ciudades. La topología de red que se muestra en la figura 3 proporciona las posibles rutas entre la ciudad de inicio en el nodo 1 y la ciudad destino en el nodo 7. Las rutas pasan por ciudades intermedias designadas por los nodos 2 a 6.

Figura 2
Red de rutas para el ejemplo 1

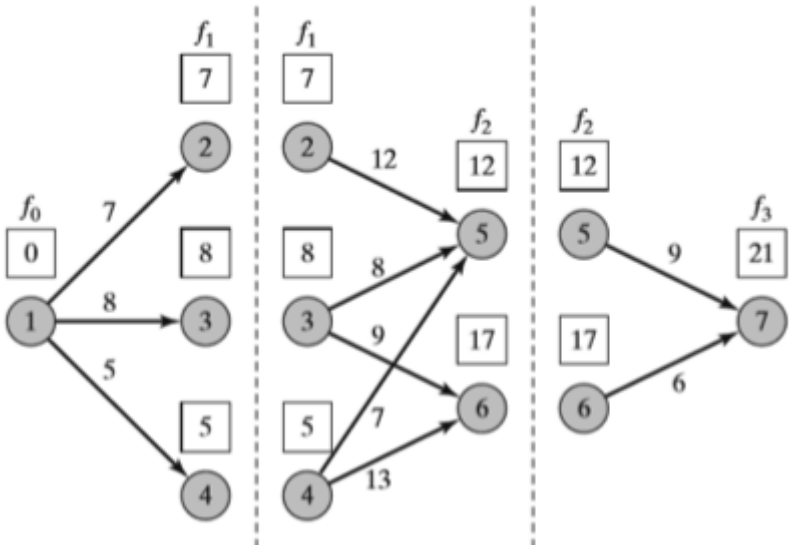


Se puede resolver este problema enumerando todas las rutas entre los nodos 1 y 7 (hay 5 rutas). Sin embargo, la enumeración exhaustiva es computacionalmente insoluble en redes grandes.

Utilizando PD, primero se descompone en *etapas* como se indica mediante las rayas verticales en la figura 3 y se realiza por separado los cálculos en cada etapa.

La idea general para determinar la ruta más corta es calcular las distancias (acumulativas) más cortas a todos los nodos terminales de una etapa, y luego utilizarlas como datos de entrada de la etapa inmediatamente siguiente. Partiendo del nodo 1, la etapa 1 llega a tres nodos terminales (2,3 y 4).

Figura 3
Descomposición por etapas de la red del ejemplo 1



3.2. Resumen etapa 1

Distancia más corta del nodo 1 al nodo 2 = 7 Kilómetros (desde el nodo 1)

Distancia más corta del nodo 1 al nodo 3 = 8 Kilómetros (desde el nodo 1)

Distancia más corta del nodo 1 al nodo 4 = 5 Kilómetros (desde el nodo 1)

La etapa 2 tiene dos nodos terminales, 5 y 6. La figura 2 muestra que se puede llegar al nodo 5 desde los nodos 2, 3 y 4 por las rutas (2,5), (3,5) y (4,5). Esta información, junto con los resultados resumidos (distancias más cortas) en la etapa 1, determina la distancia (acumulativa) más corta al nodo 5.

En (Hillier & Lieberman, 2015), se definen las ecuaciones de distancia acumulativa más corta.

$$\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ 5\end{pmatrix} = min_{i=2,3,4}\left\{\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ i\end{pmatrix} + \begin{pmatrix}Distancias\ del\\ nodo\ i\ al\ nodo\ 5\end{pmatrix}\right\}$$

$$= min\begin{Bmatrix}7+12=19\\ 8+8=16\\ 5+7=12\end{Bmatrix} = 12(desde\ el\ nodo\ 4) \qquad (1)$$

Distancia acumulativa más corta al nodo 5

Como se puede apreciar en la figura 2, solamente desde los nodos 3 y 4, se puede llegar al nodo 6 (distancia acumulativa más corta)

Distancia acumulativa más corta al nodo 6

$$\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ 6\end{pmatrix} = min_{i=3,4}\left\{\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ i\end{pmatrix} + \begin{pmatrix}Distancia\ del\\ nodo\ i\ al\ nodo\ 6\end{pmatrix}\right\}$$

$$= min\begin{Bmatrix}8+9=17\\ 5+13=18\end{Bmatrix} = 17(desde\ el\ nodo\ 3) \qquad (2)$$

3.3. Resumen etapa 2

Distancia más corta del nodo 1 al nodo 5 = 12 Kilómetros (desde el nodo 4)

Distancia más corta del nodo 1 al nodo 6 = 17 Kilómetros (desde el nodo 3)

El último paso es considerar la etapa 3. Se puede llegar al nodo de destino 7 desde el nodo 5 o 6. Utilizando los resultados resumidos desde la etapa 2 y las distancias de los nodos 5 y 6 al nodo 7.

Distancia acumulativa más corta al nodo 7

$$\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ 7\end{pmatrix} = min_{i=5,6}\left\{\begin{pmatrix}Distancia\ mas\ corta\\ al\ nodo\ i\end{pmatrix} + \begin{pmatrix}Distancia\ del\\ nodo\ i\ al\ nodo\ 7\end{pmatrix}\right\}$$

$$= min\begin{Bmatrix}12+9=21\\ 17+6=32\end{Bmatrix} = 21(desde\ el\ nodo\ 5) \qquad (3)$$

3.4. Resumen etapa 3

Distancia más corta del nodo 1 al nodo 7 = 21 Kilómetros (desde el nodo 5)

El resumen de la etapa 3 muestra que la distancia más corta entre los nodos 1 y 7 es de 21 Kilómetros. Para determinar la ruta óptima se toma el resumen de la etapa 3, donde el nodo 7 se conecta con el nodo 5; en el resumen de la etapa 2, el nodo 4 se conecta al nodo 5, y en el resumen de la etapa 1, el nodo 4 se conecta al nodo 1. Por lo tanto, la ruta más corta para el ejemplo 1.1 es 1→4→5→7.

El ejemplo revela las propiedades básicas de los cálculos de PD.

Los cálculos en cada etapa son una función de las rutas factibles de dicha etapa, y solo de esa etapa. (Principio de optimización)

Una etapa actual está conectada a la etapa inmediatamente precedente sólo (sin tener en cuenta las etapas anteriores) con base en el resumen de distancias más cortas de la etapa inmediatamente precedente.

Se presenta la solución para el mismo caso de estudio utilizando la herramienta Matlab. Se explicará detalladamente las secciones del algoritmo y se mostrarán los procesos usados para el recorrido y evaluación de las rutas del grafo.

Definición del grafo

Figura 4
Definición de vértices y aristas del grafo en Matlab

```
%% a b c d e f g
GG=[0 7 8 5 0 0 0; %a
    0 0 0 0 12 0 0; %b
    0 0 0 0 8 9 0; %c
    0 0 0 0 7 13 0; %d
    0 0 0 0 0 0 9; %e
    0 0 0 0 0 0 6; %f
    0 0 0 0 0 0 0]; %g

VV=[1 -4  0; %a
    2 -2  4; %b
    3 -2  0; %c
    4 -2 -4; %d
    5  2  1; %e
    6  2 -1; %f
    7  4  0]; %g
```

En la Figura 4 se presenta la definición matricial de la matriz de adyacencia del grafo ponderado expuesto en la Figura 1. Seguido de la representación matricial de la posición en el plano de cada uno de los vértices del mismo.

Figura 5
Trazo vértices del grafo en Matlab

```
Figure (100);
plot(VV(:,2),VV(:,3),'*b');
hold on

Y=[];
GG=triu(GG); %se obtiene la diagonal superior
for i=1:1:size(GG,1) %recorre todo el grafo para generar los caminos
    for j=1:1:size(GG,1)
        if GG(i,j)~=0
            Y[VV(i,2:3);
            VV(j,2:3)]
            plot(Y(:,1),Y(:,2),'-r')
            hold on
        end
    end
end
```

La Figura 5 se representa los métodos empleados para el trazo de los vértices del grafo y el cálculo y posterior del trayecto de los caminos de un vértice a otro.

Figura 6
Recorrido del grafo y generación de caminos

```
D=[];
n=0;
for i=1:1:size(GG,1) %recorre todo el grafo para generar los caminos
    for j=1:1:size(GG,1)
        if GG(i,j)~=0
            n=n+GG(i,j);
            D(n,:)=[i j];
        end
    end
end

costo=0;
costoTot=0;
vertSig=1;
rutaV=[1] %vector que guarda el recorrido realizado
GG
```

La Figura 6 representa el recorrido del grafo creado y la generación de los caminos existentes en el mismo almacenado de manera matricial en el vector D. Así mismo como la definición de algunas variables útiles para el cálculo de la ruta: el costo relativo de un vértice al siguiente (), el costo ponderado del valor total de la ruta (), en apuntador al siguiente vértice a examinar (), la variable se inicializa en 1 debido a que éste es el vértice de inicio de la ruta, y un arreglo que almacenará los vértices que pertenecen a la ruta más corta del grafo (), representada en la figura 6.

Figura 7
Núcleo del código

```
for i=1:1:(size(D,1)-3)
    if D(i,1) ~= 0
        if D(i,1) == vertSig
            %VV(D(i,1),:)
            costo = GG(D(i,1), D(i,2)):
            for j=1:1:(size(D,1)-3)
                if (D(j,1) == D(i,1)) & (D(j,2) ~= D(i,2))
                    if (costo == 0 | costo>GG(D(j,1), D(j,2)))
                        costo=GG(D(j,1)), D(j,2));
                        vertSig = D(j,2);
                    end
                end
            end
            if vertSig == D(i,1)
                vertSig = D(i,2);
            end
            costoTot = costoTot + costo;
            costo
            costoTot
            costo = 0;
            vertSig
            rutaV(end+1) = vertSig
        end
    end
end
```

La figura 7 representa el núcleo del código. Se determina el recorrido por el vector de caminos generado anteriormente, dicho vector es una dupla ordenada cuyos valores representan si hay o no un camino de un vértice a otro; si no lo hay, existirá en el vector D una fila [0 0].

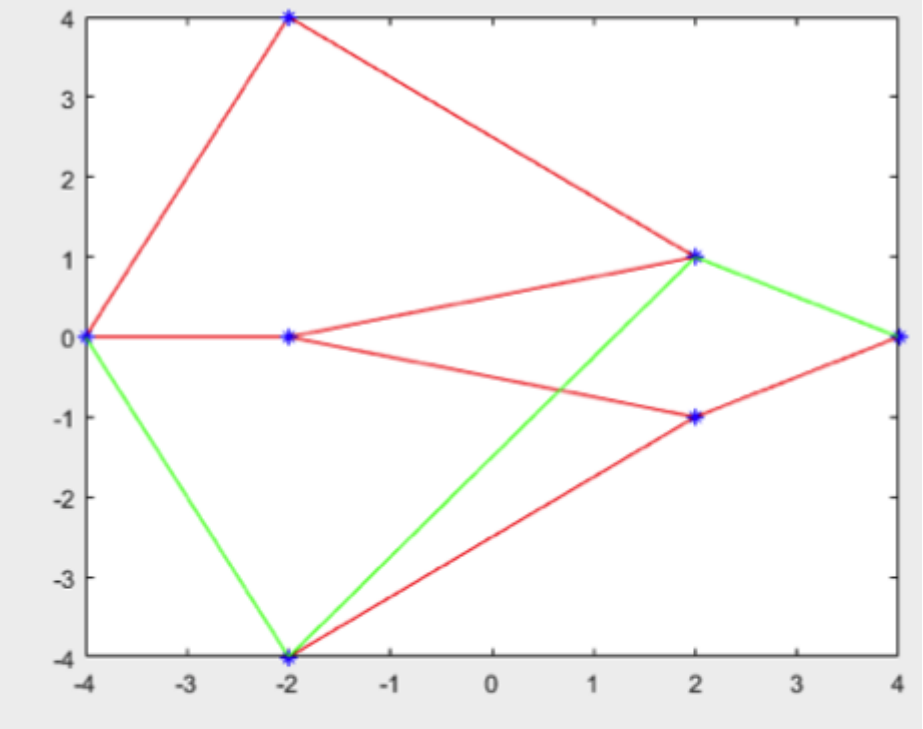
Figura 8
Representación de la ruta más corta

```
costoTot
rutaV

%%Graficacion de ruta corta
figure(100);
plot(VV(:,2),VV(:,3),'*b'); %vertices del grafo
hold on
for i=1:1:(size(rutaV,2)-1) %recorre el vector con los vertices de ruta
    Y=[VV(rutaV(i),2:3);
    VV(rutaV(i+1),2:3)];
    plot(Y(:,1),Y(:,2),'-g')
    hold on
end
```

Finalmente, se imprime en consola el costo total de la ruta, el vector de vértices de la ruta óptima y se traza el grafo de la figura 9, el cual muestra la ruta generada por el algoritmo.

Figura 9
Ejecución del programa en Matlab




```
costoTot =

    21


rutaV =

     1     4     5     7
```

4. Conclusiones

Se palpa la pertinencia de la utilización de modelos markovianos en el modelamiento típico de los servicios y aplicaciones que actualmente se utilizan sobre Internet.

La utilización de la programación dinámica es una herramienta interesante en escenarios que pretenden interpretar el tráfico en las redes de datos.

Las conclusiones obtenidas representan la importancia que tiene el encaminamiento en la compresión del tráfico que actualmente es soportado por Internet.

Se verificó que los resultados obtenidos en las simulaciones para los cálculos de: ruta más corta en los diferentes nodos de la topología de red, costo total de ruta y el vector de vértice obtenidos a través de la simulación, presentaron los mismos valores que los que se obtuvieron utilizando las diferentes propiedades de la PDD. De manera similar se pudo comprobar y verificar que los resultados de la simulación realizada para el cálculo de la ruta óptima, fueron los mismos, que los obtenidos utilizando las propiedades de los procesos de Márkov.

Un estudio previo de PDD en los sistemas de telecomunicaciones con el fin de reducir el número de cálculos computacionales y minimizar el número de nodos en la red, multiplicaría las ventajas de incluir PDD, mejorando de esta forma el desempeño en una topología red de telecomunicaciones.

Referencias bibliográficas

Hillier, F. S., & Lieberman, G. J. (2015). Introducción a la Investigación de operaciones. In *Journal of Chemical Information and Modeling* (Vol. 53). https://doi.org/10.1017/CBO9781107415324.004

Jiménez, J. R., Mercado Caruso, N., Oliveira, H., Crissien, T., & Coronado-Hernandez, J. R. (2017). Models Markovian to CSP Plans for acceptance sampling. *Espacios*, *38*.

Kashif, M., Talib, R., Ayub, N., Umer, M., & Ullah, S. (2017). OSPF vs EIGRP: A Comparative Analysis of CPU Utilization using OPNET. *International Journal of Advanced Computer Science and Applications*, *8*(7), 468–471. https://doi.org/10.14569/ijacsa.2017.080764

Maurette, M., & Ojea I. (2006). *Programación dinámica determinística*. 2–5. Retrieved from http://cms.dm.uba.ar/materias/1ercuat2009/optimizacion/Maurette_Ojea.pdf

Problemas, L., & Escolares, B. (2018). *Método de tres fases para la solución del ruteo de buses escolares Three-phase method for the solution of school bus routing*. Retrieved from https://www.revistaespacios.com/a18v39n50/a18v39n50p06.pdf

Shorikov, A. F., Sudakova, A. E., Agarkov, G. A., & Tarasyev, A. A. (2018). A deterministic dynamic model of optimizing university training structure. *IFAC-PapersOnLine*, *51*(2), 121–125. https://doi.org/https://doi.org/10.1016/j.ifacol.2018.03.021

Taha, H. a. (2012). Investigación de operaciones - Modelo de transbordo. In *Investigación de Operaciones*. https://doi.org/10.1017/CBO9781107415324.004

---

1. Docente investigador, Msc en Teleinformática y Director grupo de investigación TRHISCUD, Especialización Gestión de proyectos en Ingeniera Facultad de Ingeniería. Universidad Distrital Francisco José de Caldas. aacosta@udistrital.edu.co

2. Profesor Titular y Director del grupo de investigación Internet Inteligente, Universidad Distrital Francisco José de Caldas, Facultad de Ingeniería, osalcedo@udistrital.edu.co. Profesor Asistente, Universidad Nacional de Colombia, Departamento de Ingeniería de Sistemas E Industrial. Facultad de Ingeniería, ojsalcedop@unal.edu.co

3. Profesor Investigador y Director del grupo GCEM. Facultad de Ingeniería. Universidad Distrital Francisco José de Caldas. Ingeniero Electricista- MsC y Ph.D en Ingeniería Eléctrica, Electrónica y Automática. erivas@udistrital.edu.co

---

Revista ESPACIOS. ISSN 0798 1015
Vol. 40 (Nº 40) Año 2019

[Índice]

[En caso de encontrar algún error en este website favor enviar email a webmaster]