CC3067, Redes
Sección 20

Ricardo Méndez, 21289
Sara Echeverría, 21371
Melissa Pérez Alarcón, 21385

LABORATORIO 03

Algoritmos de Enrutamiento

Enlace al repositorio

https://github.com/bl33h/routingAlgorithms

Descripción de la práctica

La práctica consiste en la implementación y prueba de dos algoritmos de enrutamiento, a elección del equipo: Flooding y Link State Routing (LSR). Flooding asegura la entrega de mensajes a todos los nodos de la red reenviando cada mensaje a todos los vecinos, mientras que LSR utiliza el algoritmo de Dijkstra para calcular la ruta más corta a cada nodo basado en la información de topología compartida. El objetivo principal es comprender cómo estos algoritmos operan en una red, evaluando su rendimiento y eficiencia en un entorno controlado.

Descripción de los algoritmos utilizados

- *Flooding:* El algoritmo en cuestión radica en la propagación de información por medio de una red, en este caso el servidor alumchat.lol, enviándola a todos los nodos vecinos sin verificar el destino específico. Este proceso lo realiza cada nodo que recibe la data, a excepción del emisor (GeeksforGeeks, 2023).

- *Link State Routing*: Este permite a cada router enviar información sobre el estado de sus enlaces a toda la red (alumchat.lol). De esta manera, cada direccionador construye un mapa de la red y puede calcular la ruta más corta a cada destino usando Dijkstra.

Resultados

- Flooding

```
Configuración de nodos: {
  A: { D: 5, E: 4 },
  B: { C: 3, D: 2, E: 6, F: 1 },
  C: { B: 3, D: 4, E: 5 },
  D: { A: 5, B: 2, C: 4, E: 3 },
  E: { A: 4, B: 6, C: 5, D: 3, F: 2 },
  F: { B: 1, E: 2 }
}
Configuración de nombres: {
  A: 'grupo12@alumchat.xyz',
  B: 'grupo8@alumchat.xyz',
  C: 'grupo11@alumchat.xyz',
  D: 'grupo10@alumchat.xyz',
  E: 'grupo7@alumchat.xyz',
  F: 'grupo9@alumchat.xyz'
}
Conectado como men21289-test@alumchat.lol/84ec3a04-fa23-4cf0-a266-c663f1f943e7
Iniciando algoritmo Flooding...
Enviando mensaje de Flooding: {
  type: 'flooding',
  from: 'A',
  hops: 0,
  payload: 'Mensaje de prueba utilizando Flooding'
}
names recibido en startFlooding: {
  A: 'grupo12@alumchat.xyz',
  B: 'grupo8@alumchat.xyz',
  C: 'grupo11@alumchat.xyz',
  D: 'grupo10@alumchat.xyz',
  E: 'grupo7@alumchat.xyz',
  F: 'grupo9@alumchat.xyz'
}
Enviando mensaje de Flooding a grupo8@alumchat.xyz: {
  type: 'flooding',
  from: 'A',
  to: 'grupo8@alumchat.xyz',
  hops: 1,
  payload: 'Mensaje de prueba utilizando Flooding',
  id: 'A-Mensaje de prueba utilizando Flooding-1'
}
```

```
Enviando mensaje de Flooding a grupo11@alumchat.xyz: {
  type: 'flooding',
  from: 'A',
  to: 'grupo11@alumchat.xyz',
  hops: 1,
  payload: 'Mensaje de prueba utilizando Flooding',
  id: 'A-Mensaje de prueba utilizando Flooding-1'
}
Enviando mensaje de Flooding a grupo10@alumchat.xyz: {
  type: 'flooding',
  from: 'A',
  to: 'grupo10@alumchat.xyz',
  hops: 1,
  payload: 'Mensaje de prueba utilizando Flooding',
  id: 'A-Mensaje de prueba utilizando Flooding-1'
}
Enviando mensaje de Flooding a grupo7@alumchat.xyz: {
  type: 'flooding',
  from: 'A',
  to: 'grupo7@alumchat.xyz',
  hops: 1,
  payload: 'Mensaje de prueba utilizando Flooding',
  id: 'A-Mensaje de prueba utilizando Flooding-1'
}
Enviando mensaje de Flooding a grupo9@alumchat.xyz: {
  type: 'flooding',
  from: 'A',
  to: 'grupo9@alumchat.xyz',
  hops: 1,
  payload: 'Mensaje de prueba utilizando Flooding',
  id: 'A-Mensaje de prueba utilizando Flooding-1'
}
```

- Link State Routing

```
Configuración de nodos: {
  A: { D: 5, E: 4 },
  B: { C: 3, D: 2, E: 6, F: 1 },
  C: { B: 3, D: 4, E: 5 },
  D: { A: 5, B: 2, C: 4, E: 3 },
  E: { A: 4, B: 6, C: 5, D: 3, F: 2 },
  F: { B: 1, E: 2 }
}
Configuración de nombres: {
  A: 'grupo12@alumchat.xyz',
  B: 'grupo8@alumchat.xyz',
  C: 'grupo11@alumchat.xyz',
  D: 'grupo10@alumchat.xyz',
  E: 'grupo7@alumchat.xyz',
  F: 'grupo9@alumchat.xyz'
}
Conectado como men21289-test@alumchat.lol/e0cc63b4-c38a-400e-b07f-758f677bc59c
Iniciando algoritmo Link State Routing...
Enviando mensaje de LSR: {
  type: 'lsr',
  payload: 'Mensaje de prueba utilizando Link State Routing',
  hops: 0,
  to: 'B'
}
```

```
Graph configured: {
  A: Node { name: 'A', neighbors: { D: 5, E: 4 } },
  B: Node { name: 'B', neighbors: { C: 3, D: 2, E: 6, F: 1 } },
  C: Node { name: 'C', neighbors: { B: 3, D: 4, E: 5 } },
  D: Node { name: 'D', neighbors: { A: 5, B: 2, C: 4, E: 3 } },
  E: Node { name: 'E', neighbors: { A: 4, B: 6, C: 5, D: 3, F: 2 } },
  F: Node { name: 'F', neighbors: { B: 1, E: 2 } }
}
Iniciando cálculo de rutas desde A hacia grupo8@alumchat.xyz
Procesando nodo: A
Revisando vecino D de A con distancia 5
Actualizada distancia de D a 5
Revisando vecino E de A con distancia 4
Actualizada distancia de E a 4
Procesando nodo: E
Revisando vecino A de E con distancia 8
Revisando vecino B de E con distancia 10
Actualizada distancia de B a 10
Revisando vecino C de E con distancia 9
Actualizada distancia de C a 9
Revisando vecino D de E con distancia 7
Revisando vecino F de E con distancia 6
Actualizada distancia de F a 6
Procesando nodo: F
Revisando vecino B de F con distancia 7
Actualizada distancia de B a 7
Revisando vecino E de F con distancia 8
Procesando nodo: B
Revisando vecino C de B con distancia 10
Revisando vecino D de B con distancia 9
Revisando vecino E de B con distancia 13
Revisando vecino F de B con distancia 8
```

Discusión

El objetivo del laboratorio es implementar y probar algoritmos de enrutamiento, en este caso Flooding y Link State Routing (LSR) en una red simulada usando XMPP.

En el algoritmo de Flooding, los nodos reciben y reenvían mensajes a través de la red, es decir que cada nodo conectado está enviando su mensaje a todos sus vecinos. Según los resultados obtenidos, se está simulando la propagación completa de esos mensajes correctamente. Además, se está garantizando que el mensaje llegue a todos los nodos de la red a pesar de que implique una duplicación de mensajes.

En el caso del algoritmo LSR, cada nodo conoce la topología de la red a partir de la información recibida de otros nodos para calcular la mejor ruta a cualquier otro nodo utilizando Dikstra. Como se puede ver en los resultados, cumple con ese comportamiento al procesar los nodos, calcular las distancias y actualizar las rutas a partir del nodo de origen hasta destino. En el resultado capturado, se puede ver que el nodo origen es A y el de destino es B, donde calculó la ruta con los pesos totales de: $A \rightarrow E\,[4]$, $E \rightarrow F[6]$, $F \rightarrow B[7]$. Esto indica que el algoritmo está calculando correctamente las rutas más cortas en la red utilizando la información de los vecinos.

Ambos algoritmos están cumpliendo con las funcionalidades esperadas del laboratorio, pueden llegar a ser mejorados como en su rendimiento y eficiencia, pero al ser una red controlada no afecta en este entorno de aprendizaje.

Conclusiones

- Los dos algoritmos implementados, Flooding y Link State Routing, demuestran funcionar correctamente en el entorno simulado. Aún queda realizar las pruebas en un ambiente de prueba con los otros nodos.
- El algoritmo de Flooding logró enviar los mensajes a todos los nodos de la red cumpliendo el objetivo principal.
- El algoritmo LSR utilizó la información de la topología para el cálculo de las rutas más cortas usando Dijkstra.
- Aunque ambos algoritmos funcionaron correctamente, existen áreas de mejora, especialmente en términos de eficiencia y optimización.

Comentarios

- En esta práctica el mayor desafío fue la comprensión de cómo utilizar las topologías y la información proporcionada, ya que la implementación de los algoritmos en sí, fue posible realizarla.
- Ha sido un laboratorio donde se ha podido experimentar los desafíos y compromisos involucrados en el diseño de algoritmos funcionales de red.

Referencias

GeeksforGeeks. (2023). Fixed and Flooding Routing algorithms. GeeksforGeeks. https://www.geeksforgeeks.org/fixed-and-flooding-routing-algorithms/