Simulazione di Protocollo di Routing

Università degli studi di Bologna, Dipartimento di Informatica - Scienza e Ingegneria - DISI

A.A. 2024-2025

Laurea Ing. e scienze informatiche

Corso: programmazione di reti

RELAZIONE PROGETTO LABORATORIO TRACCIA N.2

Prof: Andrea Piroddi

“Simulazione di Protocollo di Routing”

Ludovico Riccio

Matricola 1069058

▼ 1. Obiettivi e consegna

Descrizione della consegna

Il progetto di laboratorio prevede la simulazione di un protocollo di routing, con particolare focus sull'implementazione e sul funzionamento di un algoritmo di Distance Vector Routing, come il protocollo RIP.

Obiettivi richiesti

1. **Creazione dello script**: Implementare uno script Python che simuli il comportamento del protocollo di routing RIP.
    - Gestione delle tabelle di routing per i nodi della rete.
    - Scambio dinamico di informazioni di routing tra i nodi.
    - Calcolo e aggiornamento delle distanze per determinare il percorso migliore.
2. **Simulazioni**:
    - Configurare una rete iniziale con nodi e collegamenti.
    - Simulare guasti e osservare l'aggiornamento automatico delle tabelle di routing.
    - Misurare i tempi di convergenza.
3. **Documentazione**:
    - Fornire una relazione dettagliata che spieghi il funzionamento dello script e includa:
        - Configurazioni iniziali e modifiche durante le simulazioni.
        - Output dei risultati.
        - Analisi dei tempi di convergenza.
        - Problematiche affrontate e risoluzioni adottate.

▼ 2. Accenni di teoria

Protocollo di routing

Definizione

Un protocollo di routing è un insieme di regole che permettono ai router ed altri dispositivi di rete di comunicare tra loro per determinare il percorso migliore per instradare i pacchetti dati attraverso una rete.

Le caratteristiche principali di un protocollo di routing sono le seguenti:

- Permette ai nodi della rete di scambiarsi informazioni sulla topologia della rete, come ad esempio le distanze tra i nodi, o la disponibilità dei collegamenti. Mediante tali informazioni si costruiscono (e si aggiornano) le tabelle di routing.
- Utilizza algoritmi per determinare il percorso ottimale, come Bellman-Ford o Dijkstra.
- I percorsi sono aggiornati automaticamente in caso di cambiamenti della topologia (come collegamenti nuovi, guasti)
- Sono progettati per gestire reti di qualsiasi dimensioni, infatti devono avere una gran scalabilità

Un protocollo di routing può essere *statico* o *dinamico*.

Nel primo caso, i percorsi vengono configurati manualmente e non cambiano dinamicamente, risulta evidente che sono semplici con un basso overhead ma hanno lo svantaggio di non essere dinamici.

Nel secondo caso, invece, come suggerisce il nome, i percorsi sono calcolati e aggiornati automaticamente, quindi risultano flessibili e adattabili. Il cambiamento dinamico avviene tramite algoritmi per trovare il miglior percorso. E' evidente la loro maggior complessità e overhead.

I protocolli di routing si possono distinguere in:

- Interior Gateway Protocol (IGP)
- Exterior Gateway Protocol (EGP)

Interior Gateway Protocol

I protocolli IGP sono utilizzati per l'instradamento all'interno di un singolo Autonomous System (AS), ovvero una raccolta di dispositivi di rete che sono sotto il controllo di una singola organizzazione o amministrazione che condividono una politica comune di instradamento o routing.

Vengono impiegati per gestire reti interne, fornendo percorsi ottimizzati all'interno dell'AS (Intra-AS Routing), poiché hanno una visione completa della topologia interna, consentendo percorsi precisi e veloci. Possono essere impiegati per la comunicazione interna tra i router di un'azienda, una reta aziendale o una rete di provider.

Esempi di IGP, sono:

- **RIP (Routing Information Protocol)**: è uno dei protocolli più semplici e viene utilizzato in reti di piccole o medie dimensioni per distribuire informazioni sulle tabelle di routing.

  Si basa su un algoritmo di *distanza-vettore* e utilizza la metrica di hop count (la metrica massima supportata è di 15 hop) per determinare il percorso migliore verso una destinazione.

  Il RIP può riscontrare due problemi principali:

  1. Convergenza lenta: dopo un cambiamento della rete, può impiegare tanto tempo per aggiornare tutte le tabelle di routing;
  2. Count-to-infinity: un ciclo nel routing potrebbe causare un incremento continuo della distanza verso la destinazione, fino a raggiungere il limite massimo di hop, provocando inefficienze nel sistema.

  A seguito di queste problematiche sono state adottate le seguenti soluzioni:

  1. Split Horizon: per evitare che un router annuncia un percorso a un vicino da cui ha appreso quello stesso percorso;
  2. Poison Reverse: se un router riceve informazioni su una rete già presente nella sua tabella, invia un aggiornamento con una metrica pari a 16 (indicando che è irraggiungibile), notificando ai router vicini che quella rete non è più accessibile tramite di esso.
  3. Timer: utilizza un sistema di timer per scadenza e medie dimensioni, che aiuta a mantenere aggiornate le informazioni di routing

- **OSPF (Open Shortest Path First):** è un protocollo basato su un algoritmo di *link-state* che trova il percorso più breve per il traffico all'interno di una rete IP.

E’ ampiamente utilizzato nei grandi ambienti di rete perché è efficiente, veloce e supporta reti di grandi dimensioni

- **EIGGRP (Enhanced Interior Gateway Riuting Protocol):** utilizza un algoritmo di distanza-vettore più avanzato rispetto al **RIP**, con alcune caratteristiche di *link-state.* E’ più veloce e scalabile di RIP e supporta reti più complesse. Cisco ne è il proprietario di tale protocollo.

Exterior Gateway Protocol (EGP)

I protocolli EGP sono progettati per gestire l’inatradamento tra diversi **AS** e non all’interno di un singolo come avviene per gli **IGP**.

Proprio per questo sono fondamentali per l’interconnessione di reti globali, come Internet.

Gestiscono informazioni di topologia limitate, scambiando solo le rotte necessarie tra AS; supportano la scalabilità mondiale, evitando di dover conoscere la topologia completa della rete mondiale.

Esempio di EGP:

- **BGP (Border Gateway Protocol):** è un protocollo basato sul *path vector*, sceglie il percorso migliore in base a una serie di attributi di percorso, come il numero di AS attraversati, la preferenza amministrativa e le politiche di routing configurate, che possono essere il numero di AS attraversati, preferenze locali. Talvolta tali politiche possono essere complesse e complicate, richiedendo un’estrema attenzione per evitare errori che possono compromettere il funzionamento della rete

  Supporta politiche di routing complesse per la gestione del traffico e delle priorità; non converge velocemente ma è estremamente scalabile e robusto.

Distance Vector Routing

Come abbiamo visto per il RIP, il distance vector (distanza-vettore) è un metodo per determinare i percorsi ottimali tra nodi di una rete. Con questo metodo ogni nodo mantiene una tabella con la distanza minima per raggiungere tutti gli altri nodi e aggiorna queste informazioni scambiandole con i vicini.

Ogni nodo ha una tabella di routing che include:

- Le destinazioni nella rete
- La distanza verso ogni destinazione
- Il next hop da cui inoltrare i pacchetti

I nodi si scambiano le proprie tabelle di routing con i nodi vicini, utilizzando aggiornamenti periodici o eventi di cambiamento nella rete

Il distance vector si basa sull'algoritmo *Bellman-Ford,* che calcola il percorso di costo minimo in modo iterativo.

Funzionamento

Ogni nodo conosce la distanza a sé stesso e la distanza verso i suoi vicini diretti; ogni nodo invia ai vicini il proprio distanza-vettore che contiene le distanze attuali verso tutte le distanze conosciute.

A seguito di ciò ogni nodo ricevente aggiorna la propria tabella in base al vettore ricevuto, confrontando il costo dei percorsi ricevuti con quelli già conosciuti, se risulta più economico allora la tabella viene aggiornata con il nuovo costo e il nuovo next hop.

Il processo viene ripetuto iterativamente fino a raggiungere la convergenza, ovvero quando tutte le tabelle di routing non cambiano più.

Analizziamo i pro e i contro.

Pro:

- E' facile da implementare e richiede risorse computazionali limitate
- Funziona anche con informazioni incomplete o iniziali
- Si aggiorna automaticamente in caso di modifiche nella rete

Contro:

- A seguito di un cambiamento nella rete, il distance vector routing ouò impiegare tempo a propagare le informazioni aggiornate
- In assenza di tecniche di prevenzione, i nodi possono scambiarsi erroneamente informazioni obsolete, causando cicli infiniti:
    - Nodo A invia a B che un percorso è disponibile a costo infinito, nodo B aggiorna il percorso ma lo considera ancora valido

Soluzioni ai problemi:

Come abbiamo visto per il RIP, le soluzioni possibili sono:

- Split horizon
- Posion reverse
- Triggered updates: gli aggiornamenti non avvengono solo periodicamente, ma anche quando si rilevano cambiamenti.

Il RIP è il metodo utilizzato per lo svolgimento del progetto.

▼ 3. Implementazione

Struttura

Il progetto, come da consegna, è stato realizzato in Python, la struttura è la seguente:

- Classe `Node`:
  - Rappresenta un nodo nella rete
  - Ogni nodo mantiene una tabella di routing, che contiene:
    - Le destinazioni conosciute
    - La distanza verso ciascuna destinazione
    - Il *next hop* attraverso cui instradare i pacchetti
  - Metodi principali:
    - `updateRoutingTable`: aggiorna la tabella di routing in base alle informazioni ricevute dai nodi vicini
    - `shareTable`: condivide la tabella di routing con i vicini
    - `disconnectNeighbor`: simula un guasto disconnettendo un vicino
- Classe `Network`:
  - rappresenta l'intera rete come un insieme di nodi e connessioni
  - Gestisce le simulazioni del protocollo RIP, inclusi:
    - La creazione dei nodi e delle connessioni
    - La simulazione del guasto su un collegamento
    - Il calcolo dei tempi di convergenza
  - Metodi principali:

- `simulateIteration`: simula un'iterazione del protocollo RIP
- `simulateFailure`: simula un guasto tra due nodi
- `logRoutingTables`: registra lo stato delle tabelle di routing in un file di log
- `printRountingTables`: stampa lo stato attuale delle tabelle di routing

Funzionamento del protocollo RIP

1. **Stato iniziale:**
    a. Ogni nodo conosce solo sé stesso e i propri vicini diretti
    b. Le connessioni tra i nodi vengono configurate con un costo specifico
2. **Simulazioni delle iterazioni:**
    a. I nodi scambiano le proprie tabelle di routing con i vicini
    b. Ogni nodo aggiorna la propria tabella confrontando i costi ricevuti con quelli già noti
    c. Il processo continua finché le tabelle di routing non convergono, ovvero non subiscono più modifiche
3. **Gestione dei guasti:**
    a. Viene simulata la disconnessione di un collegamento tra due nodi
    b. I nodi rilevano il guasto e aggiornano le proprie tabelle per escludere il percorso non più disponibile
    c. Il protocollo RIP ricalcola automaticamente i percorsi alternativi
4. **Calcolo del tempo di convergenza:**
    a. Viene misurato il tempo impiegato dalla rete per convergere, sia nello stato iniziale che dopo un guasto

▼ 4. Esempio di simulazione

Configurazione iniziale

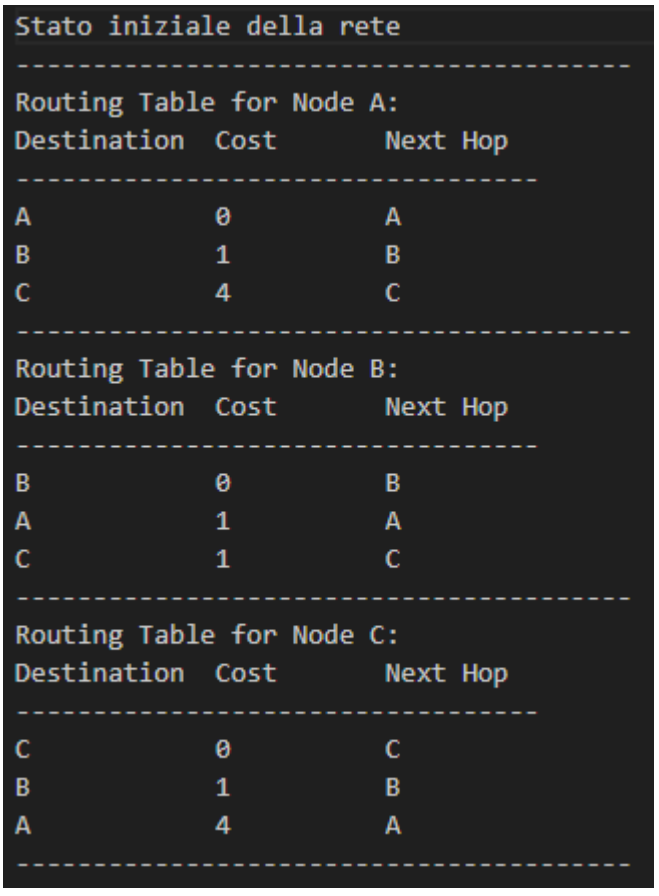
La rete configurata per il progetto comprende tre nodi e le seguenti connessioni:

- Nodi:
  - A, B, C
- Connessione e rispetti costi:
  - A ←→ B, con costo 1
  - B ←→ C, con costo 1
  - A ←→ C, con costo 4

Questa topologia semplice permette di simulare l'apprendimento dei percorsi e la gestione dei guasti da parte del protocollo RIP.

Stato iniziale della rete

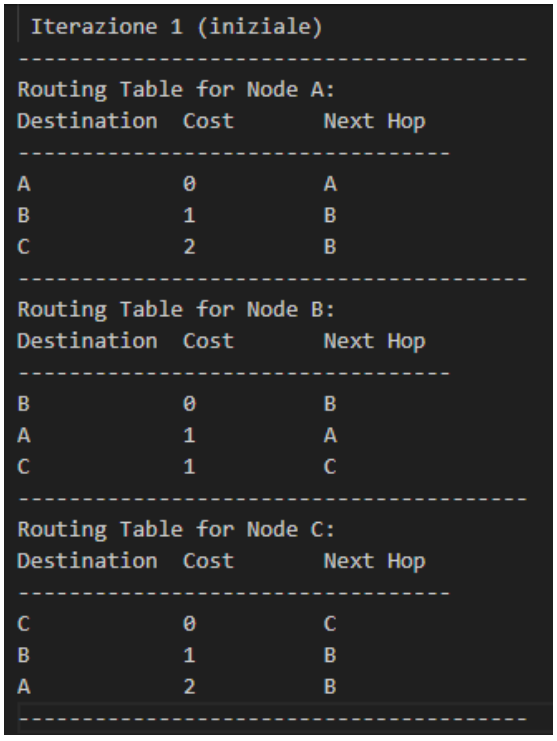
All'avvio del programma, ogni nodo conosce solo sé stesso e i propri vicini diretti. Le tabelle di routing iniziali sono le seguenti:



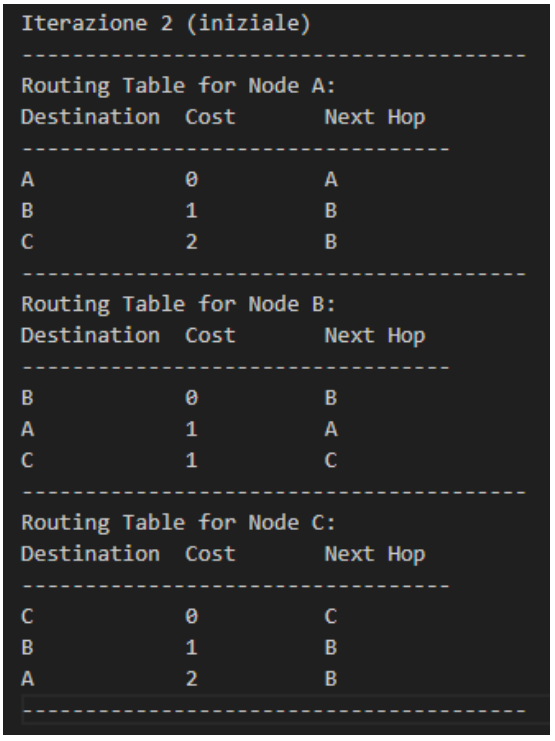
Dati estrapolati dal file log

Simulazione del protocollo RIP: stato iniziale

Durante le iterazioni iniziali del protocollo RIP, i nodi scambiano le proprie tabelle di routing e aggiornano i percorsi in base ai costi ricevuti. Dopo 2 iterazione, la rete converge e le tabelle di routing risultano come nell'immagine sottostante.



```
Iterazione 1 (iniziale)
--------------------------------------
Routing Table for Node A:
Destination  Cost       Next Hop
---------------------------------
A            0          A
B            1          B
C            2          B
--------------------------------------
Routing Table for Node B:
Destination  Cost       Next Hop
---------------------------------
B            0          B
A            1          A
C            1          C
--------------------------------------
Routing Table for Node C:
Destination  Cost       Next Hop
---------------------------------
C            0          C
B            1          B
A            2          B
--------------------------------------
```



```
Iterazione 2 (iniziale)
--------------------------------------
Routing Table for Node A:
Destination  Cost       Next Hop
---------------------------------
A            0          A
B            1          B
C            2          B
--------------------------------------
Routing Table for Node B:
Destination  Cost       Next Hop
---------------------------------
B            0          B
A            1          A
C            1          C
--------------------------------------
Routing Table for Node C:
Destination  Cost       Next Hop
---------------------------------
C            0          C
B            1          B
A            2          B
--------------------------------------
```

Come possiamo osservare la rete converge dopo 2 iterazioni.

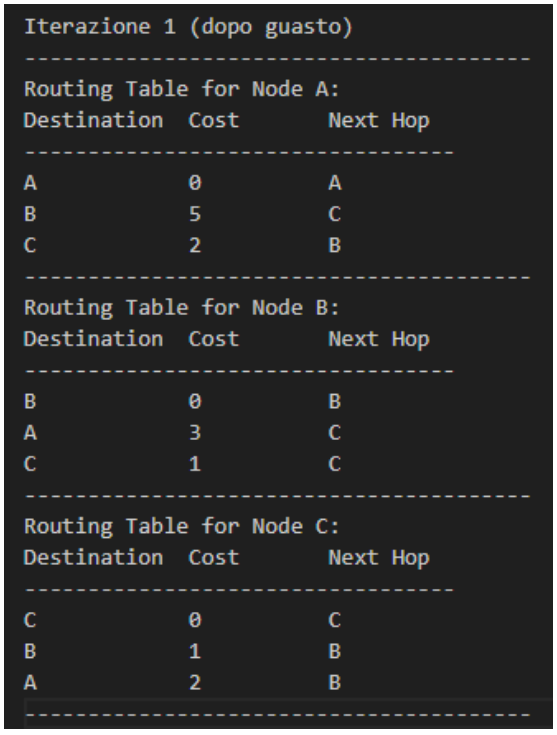
Simulazione di un guasto

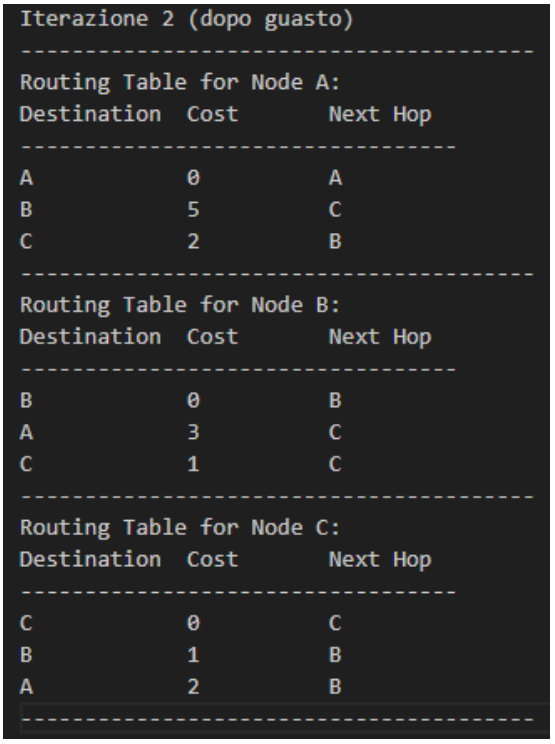
Viene simulato un guasto tra i nodi A e B. Pertanto, i percorsi che passano attraverso il collegamento A — B vengono invalidati e il protocollo RIP calcola automaticamente percorsi alternativi:

```
Guasto tra A e B
---------------------------------------
Routing Table for Node A:
Destination  Cost       Next Hop
---------------------------------
A            0          A
B            inf        N/A
C            2          B
---------------------------------------
Routing Table for Node B:
Destination  Cost       Next Hop
---------------------------------
B            0          B
A            inf        N/A
C            1          C
---------------------------------------
Routing Table for Node C:
Destination  Cost       Next Hop
---------------------------------
C            0          C
B            1          B
A            2          B
---------------------------------------
```

Simulazione del protocollo RIP: dopo il guasto

Dopo il guasto, i nodi scambiano nuovamente le tabelle di routing e convergono su nuovi percorsi in 2 iterazioni:





Tempi di convergenza

I tempi di convergenza sono:

- Convergenza iniziale: 2 iterazioni, t = 0,03 secondi
- Convergenza dopo il guasto: 2 iterazioni, t = 0,04 secondi
- Tempo totale: 0,07 secondi

▼ 5. Scalabilità

Il protocollo RIP è progettato principalmente per reti di piccole e medie dimensioni. La sua semplicità lo rende facilmente implementabile, ma comporta alcune limitazioni in termini di scalabilità.

Limiti di scalabilità

RIP utilizza il numero di hop come metrica per determinare il percorso migliore, ma il limite massimo di 15 hop rende impossibile raggiungere qualsiasi rete che superi tale numero. Questo vincolo restringe la dimensione massima della rete supportata e ne limita l'applicabilità in reti più grandi.

In caso di guasti nella rete, il tempo richiesto per la convergenza aumenta proporzionalmente alla dimensione della rete, a causa del metodo di aggiornamento iterativo basato su aggiornamenti tra i vicini. Con il crescere di dimensione della rete tale problema risulterà sempre più evidente.

RIP invia aggiornamenti periodici dell'intera tabella di routing tramite broadcast a tutti i vicini. In reti più grandi, questo approccio può comportare un significativo overhead di rete, riducendo l'efficienza complessiva.
In assenza di meccanismi come
split horizon e *poison reverse*, le reti più grandi possono soffrire di cicli infiniti di routing, in cui i pacchetti continuano a circolare senza raggiungere la destinazione.

Miglioramenti per la scalabilità

Le reti possono essere suddivise in aree o domini più piccoli, ciascuno gestito da un proprio protocollo RIP. L'interscambio di rotte tra le aree può essere invece affidato a protocolli più scalabili come OSPF o BGP. Questa tecnica consente di sfruttare la semplicità di RIP in ambiti circoscritti, garantendo al contempo la scalabilità generale della rete.

Come trattato nella sezione "Accenni di Teoria", protocolli come OSPF e BGP sono progettati per gestire reti di grandi dimensioni e sono più adatti per reti complesse. Essi offrono tempi di convergenza più rapidi e una gestione del traffico più efficiente.
Accorgimenti come
split horizon, *poison reverse* e *triggered updates* possono migliorare l'efficienza di RIP. Questi meccanismi riducono il tempo di convergenza, prevengono cicli infiniti e ottimizzano la propagazione delle informazioni di routing.

Sebbene RIP presenti limiti di scalabilità significativi, rimane una scelta valida per reti più piccole o scenari controllati, dove la sua semplicità e leggerezza rappresentano vantaggi. Per reti più grandi o complesse, è necessario considerare protocolli alternativi più adatti alle esigenze di scalabilità e prestazioni.

▼ 6. Conclusioni

Il progetto dimostra il funzionamento del protocollo RIP in una rete simulata, ponendo il focus sull'algoritmo di distanza-vettore per la gestione dinamica delle tabelle di routing. Dalla simulazione si può osservare il processo di convergenza del protocollo, sia in condizioni di rete stabile che in scenari di guasti, mostrando come il protocollo utilizzato possa ricalcolare percorsi alternativi e mantenere la connettività.

I problemi riscontrati durante lo sviluppo sono stati: la gestione dei nodi disconnessi, l'ottimizzazione del tempo di convergenza e formattazione nel file di log, a seguito di un'attenta analisi sono state trovate le corrispettive risoluzioni. Osserviamo:

Gestione del guasto: disconnessione tra nodi

Inizialmente, la simulazione del guasto non aggiornava correttamente le tabelle di routing, causano percorsi infiniti o incoerenti.

Pertanto è stato implementato un sistema di aggiornamento che imposta il costo dei collegamenti guasti a infinito, segnalando ai nodi vicini l'impossibilità di raggiungere determinate destinazioni.

L'uso di *poison reverse* e l'aggiornamento iterativo hanno assicurato che le tabelle convergessero rapidamente a uno stato consistente

Tempi di convergenza non chiari

La misurazione dei tempi di convergenza risultava inaccurata nelle prime implementazioni.

E' stato introdotto un sistema di misurazione separato per la convergenza iniziale e per quella post - disconnessione (guasto simulato), con una gestione accurata dei timer.

La separazione ha permesso di isolare i tempi di convergenza in scenari diversi, migliorando l'analisi delle prestazioni del protocollo

Problemi di formattazione nel file di log

I log delle tabelle di routing presentavano ripetizioni o incongruenze nei dati.

E' stato adottato un sistema di log strutturato con timestamp e separazione chiara degli eventi per ogni iterazione.

Un file di log leggibile facilità il debug e offre una documentazione accurata dello stato della rete durante la simulazione

Sviluppo futuro

1. **Aggiunta di una GUI interattiva**: integrare una GUI per rappresentare visivamente la rete simulata, consentendo agli utenti di monitorare in tempo reale i cambiamenti delle tabelle di routing e di interagire con la rete, ad esempio simulando guasti o aggiungendo nodi e connessioni
2. **Estensione della simulazione a reti più grandi:** ampliare il supporto per reti con un numero maggiore di nodi e connessioni, valutando l'impatto delle dimensioni della rete sui tempi di convergenza e sull'efficienza complessiva
3. **Simulazioni di protocolli alternativi:** affiancare al RIP la simulazione di protocolli più avanzati, come OSPF o BGP, per confrontare le prestazioni e analizzare scenari più complessi che richiedono una maggiore scalabilità e velocità di convergenza
4. **Aggiunta di metriche avanzate:** calcolare metriche aggiuntive durante le simulazioni, come il traffico totale generato dagli aggiornamenti o il consumo di risorse computazionali, per valutare l'efficienza del protocollo