

# Relazione Progetto Reti 2024

Gianmaria Di Fronzo 1077802 - Riccardo Polazzi 1081320

Relazione sul Progetto: Simulazione di Protocollo di Routing in Python

---

## Introduzione

Il progetto ha l'obiettivo di implementare e simulare un protocollo di routing, nello specifico il **Distance Vector Routing**. Questo protocollo consente a ogni nodo di aggiornare la propria tabella di routing in base alle informazioni ricevute dai vicini. L'approccio si basa sull'algoritmo di Bellman-Ford per calcolare i percorsi più brevi in una rete connessa.

Gli obiettivi principali sono:

1. Creare e gestire tabelle di routing per ogni nodo.
  2. Implementare la logica di aggiornamento basata sul costo minimo.
  3. Documentare ogni fase e fornire output chiari per verificare il funzionamento.
- 

## Descrizione del Codice

Classe **Network**

La classe **Network** modella la rete, composta da nodi, collegamenti e tabelle di routing. I metodi principali sono:

1. **\_\_init\_\_**
  - Inizializza la rete con i nodi dati e costruisce:
    - Il grafo della rete come un dizionario annidato `{nodo: {vicino: costo}}`.
    - Tabelle di routing inizialmente impostate con distanze infinite verso tutti i nodi.
2. **add\_link**
  - Aggiunge un collegamento bidirezionale tra due nodi con un costo associato.
  - Aggiorna il grafo della rete.
3. **initialize\_tables**
  - Inizializza le tabelle di routing di ciascun nodo:
    - La distanza verso se stesso è impostata a 0.
    - Ogni altro nodo ha distanza iniziale  $\infty$  (dato dal costruttore).
4. **update\_routing**
  - Implementa la logica del Distance Vector Routing:
    - Ogni nodo verifica se una nuova rotta verso una destinazione (attraverso un vicino) ha un costo inferiore rispetto al percorso attuale.
    - Se viene trovato un percorso più breve, la tabella di routing viene aggiornata.

## 5. **simulate**

- Simula il protocollo iterando sull'aggiornamento delle tabelle di routing fino a convergenza:
  - Ad ogni iterazione, stampa le tabelle di routing per monitorare i progressi.
  - Termina quando non ci sono più aggiornamenti.

### Simulazione della Rete

#### 1. **Creazione della rete**

- La rete è composta da 8 nodi: A, B, C, D, E, F, G, H.
- I collegamenti tra i nodi e i rispettivi costi sono definiti con il metodo `add_link`.

#### 2. **Esecuzione**

- La simulazione mostra l'evoluzione delle tabelle di routing a ogni iterazione.
- La rete converge in un numero finito di iterazioni.

---

### Valutazione della Soluzione

#### 1. **Completezza della Soluzione**

- Il codice implementa tutte le funzionalità richieste:
  - Creazione della rete.
  - Calcolo delle rotte più brevi usando il protocollo Distance Vector Routing.
  - Output dettagliato delle tabelle di routing.

#### 2. **Chiarezza della Documentazione**

- Il codice è ben documentato e le funzioni hanno nomi esplicativi.
- Gli output intermedi aiutano a comprendere il funzionamento passo dopo passo.

#### 3. **Accuratezza Tecnica**

- L'algoritmo segue correttamente le regole del Distance Vector Routing:
  - L'aggiornamento delle tabelle si basa sul principio del minimo costo.

$$D_j^h = \min_i \{D_i^{h-1} + d_{ij}, D_j^{h-1}\}$$

- La convergenza è garantita quando non ci sono ulteriori miglioramenti.
  - L'implementazione rispetta l'approccio distribuito tipico di questo protocollo.
- #### 4. **Capacità di Diagnosi e Problem Solving**
- La struttura modulare consente di isolare facilmente eventuali problemi.
  - Gli output iterativi permettono di verificare il corretto aggiornamento delle tabelle.
-

## Conclusioni

Il progetto rappresenta un esempio efficace di simulazione di un protocollo di routing. L'implementazione è completa, tecnicamente accurata e ben documentata. L'output del codice permette di analizzare in dettaglio l'evoluzione delle tabelle di routing, facilitando la comprensione dei principi del Distance Vector Routing.

Possibili miglioramenti includono:

- Aggiunta di test automatizzati per verificare la correttezza delle tabelle.
- Espansione del modello per simulare guasti ai nodi o cambiamenti dinamici nei costi dei collegamenti.

Nel complesso, il progetto soddisfa pienamente gli obiettivi prefissati, dimostrando competenze solide in programmazione di reti.