Simulatore di Routing a Vettore di Distanza

Lucio Baiocchi, Luca Camillini, Andrea De Carli

December 11, 2024

Introduzione

Il progetto realizzato consiste in un simulatore di routing a vettore di distanza, sviluppato in Python utilizzando la libreria Tkinter per l'interfaccia grafica. L'obiettivo principale è fornire uno strumento educativo per esplorare e comprendere il funzionamento dell'algoritmo di routing a vettore di distanza, un metodo fondamentale nelle reti per determinare i percorsi ottimali tra i nodi. Attraverso il simulatore, gli utenti possono creare una rete, aggiungere nodi e collegamenti, assegnare costi, e osservare in tempo reale come i nodi aggiornano dinamicamente le proprie tabelle di routing. Questo progetto si pone come una base per l'apprendimento dei principi di routing e per lo studio delle dinamiche delle reti.

Architettura del Progetto

Il progetto è organizzato in quattro moduli principali per garantire modularità e manutenibilità. Ogni modulo ha un compito specifico: il modulo `node.py` definisce i nodi della rete, il modulo `routing_table.py` gestisce l'aggiornamento delle tabelle di routing, il modulo `gui.py` si occupa dell'interfaccia grafica e dell'interazione con l'utente, mentre il modulo `main.py` rappresenta il punto di ingresso dell'applicazione. Questa architettura separa la logica della simulazione dalla presentazione grafica, facilitando l'espansione del progetto con nuove funzionalità.

Spiegazione dell'Architettura

Ogni nodo della rete è rappresentato dalla classe `Node`, che contiene informazioni sui vicini e sulla tabella di routing. La tabella di routing viene

aggiornata dinamicamente tramite l'algoritmo a vettore di distanza, che confronta i costi dei percorsi esistenti con quelli ricevuti dai vicini. La gestione complessiva dell'aggiornamento delle tabelle di routing è affidata al modulo `routing_table.py`, che coordina il processo globale di propagazione delle modifiche. L'interfaccia grafica, sviluppata in `gui.py`, utilizza Tkinter per consentire agli utenti di aggiungere nodi, collegamenti e modificare la topologia della rete in tempo reale. Infine, il modulo `main.py` avvia l'applicazione, creando la finestra principale e collegando i componenti.

Descrizione del Codice

Il modulo `node.py` contiene la definizione della classe `Node`. Ogni nodo mantiene una lista di vicini, rappresentati da altri nodi collegati, e una tabella di routing che memorizza il costo minimo per raggiungere ogni altro nodo. La funzione `add_neighbor` permette di aggiungere un vicino a un nodo con un costo definito, mentre la funzione `update_routing_table` aggiorna la tabella di routing del nodo corrente, confrontando i costi esistenti con quelli ricevuti dai vicini. Di seguito è riportato un esempio della classe `Node`:

```
1  class Node:
2      def __init__(self, name):
3          self.name = name
4          self.routing_table = {name: (0, name)}
5          self.neighbors = {}
6
7      def add_neighbor(self, neighbor, cost):
8          self.neighbors[neighbor] = cost
9
10     def update_routing_table(self):
11         updated = False
12         for neighbor, cost_to_neighbor in self.neighbors.items():
13             for dest, (cost_to_dest, next_hop) in neighbor.
                   routing_table.items():
14                 new_cost = cost_to_neighbor + cost_to_dest
15                 if dest not in self.routing_table or new_cost < self.
                       routing_table[dest][0]:
16                     self.routing_table[dest] = (new_cost, neighbor.
                           name)
17                     updated = True
18         return updated
```

Listing 1: Definizione della classe Node (`node.py`)

Capacità di Diagnosi e Problem Solving

Durante lo sviluppo del progetto, sono stati affrontati diversi problemi tecnici che hanno richiesto un'analisi approfondita e una strategia di risoluzione. Un problema significativo riguardava l'aggiornamento delle tabelle di routing quando il costo di un collegamento veniva modificato. Inizialmente, le tabelle non si aggiornavano correttamente, causando un disallineamento tra la topologia della rete visualizzata e le informazioni di routing. Per risolvere questo problema, è stato introdotto un controllo esplicito nella funzione `on_link_click`, che forza l'aggiornamento delle tabelle di routing tramite una chiamata diretta alla funzione `update_routing_tables`. Questo approccio ha garantito che tutte le tabelle venissero ricalcolate immediatamente dopo ogni modifica.

Un altro problema è emerso durante la gestione dei collegamenti duplicati tra nodi. Senza un controllo adeguato, l'applicazione permetteva di creare più collegamenti tra gli stessi nodi, causando incoerenze nella rete. Per affrontare questa situazione, è stato aggiunto un controllo nella funzione `add_link` per verificare se un collegamento tra due nodi esiste già. Se il collegamento è presente, l'applicazione mostra un messaggio di errore all'utente e impedisce la creazione del collegamento.

1 Funzionamento GUI

Tramite i pulsanti inferiori è possibile aggiungere un nodo o un collegamento. Sono stati inseriti controlli per non inserire un collegamento tra due nodi più volte, o che venga inserito un collegamento con peso negativo. È possibile visualizzare le tabelle di routing con il tasto destro sopra un nodo per visualizzare in automatico le tabelle di routing del nodo selezionato. Le tabelle si aggiornano automaticamente ogni volta che viene aggiunto un nodo o viene aggiunto un collegamento. Per modificare un collegamento è possibile rimuoverlo dopo averlo selezionato cliccandoci sopra con il tasto sinistro. Inoltre è possibile posizionare in nodi trascinandoli. I nodi si possono rinominare facendo doppio click su quello da modificare. Di seguito alcune immagini relative all'utilizzo dell'interfaccia grafica.

2 Conclusioni

Il simulatore di routing a vettore di distanza rappresenta un potente strumento educativo per comprendere i principi fondamentali del routing nelle reti. Grazie alla sua interfaccia grafica intuitiva e alla capacità di aggiornare

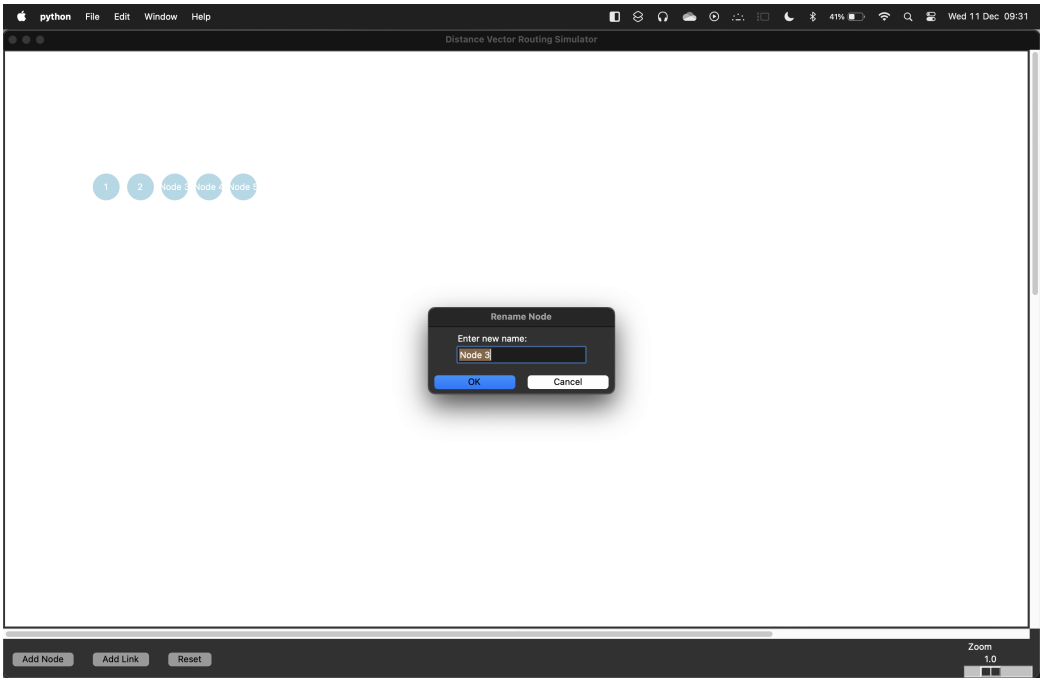

Figure 1: Rinominazione nodo

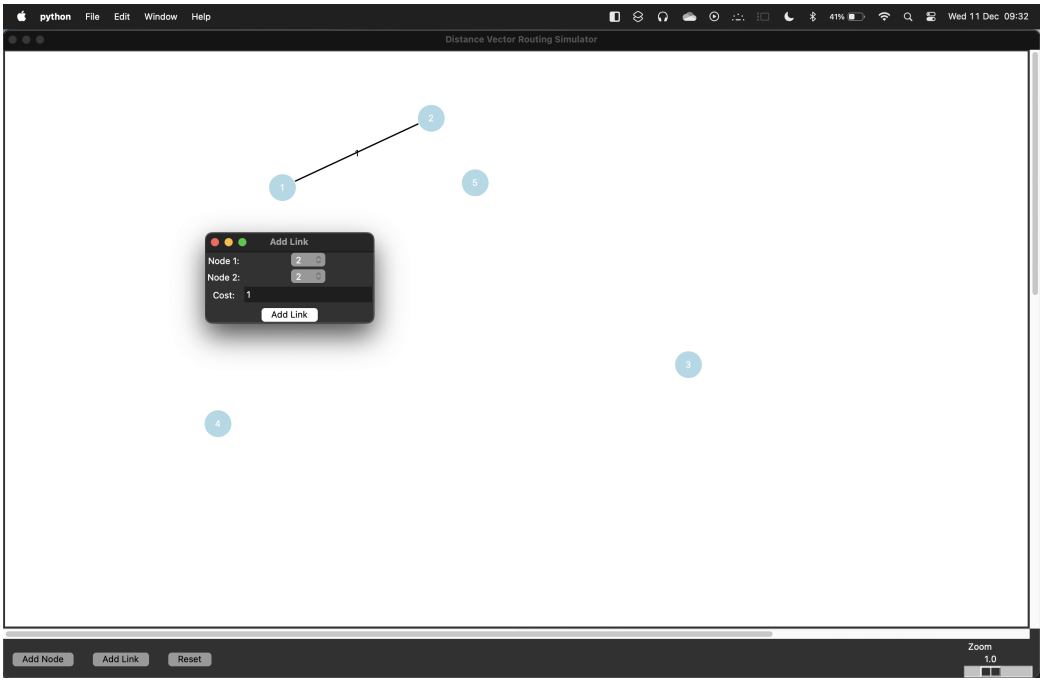

Figure 2: Aggiunta collegamento

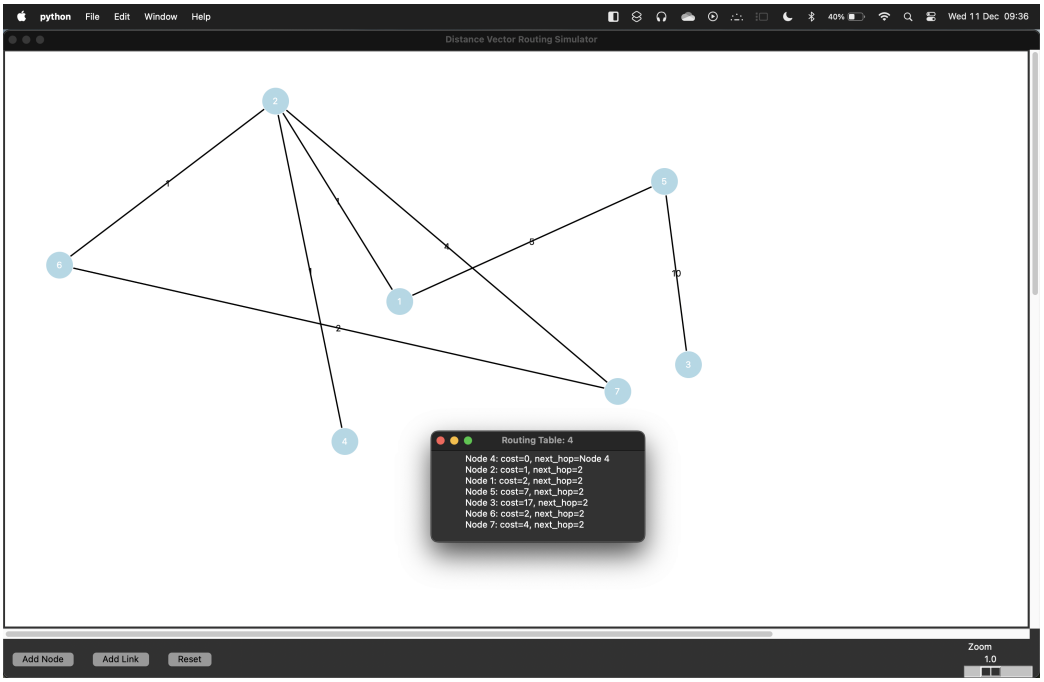

Figure 3: Tabella routing di un nodo

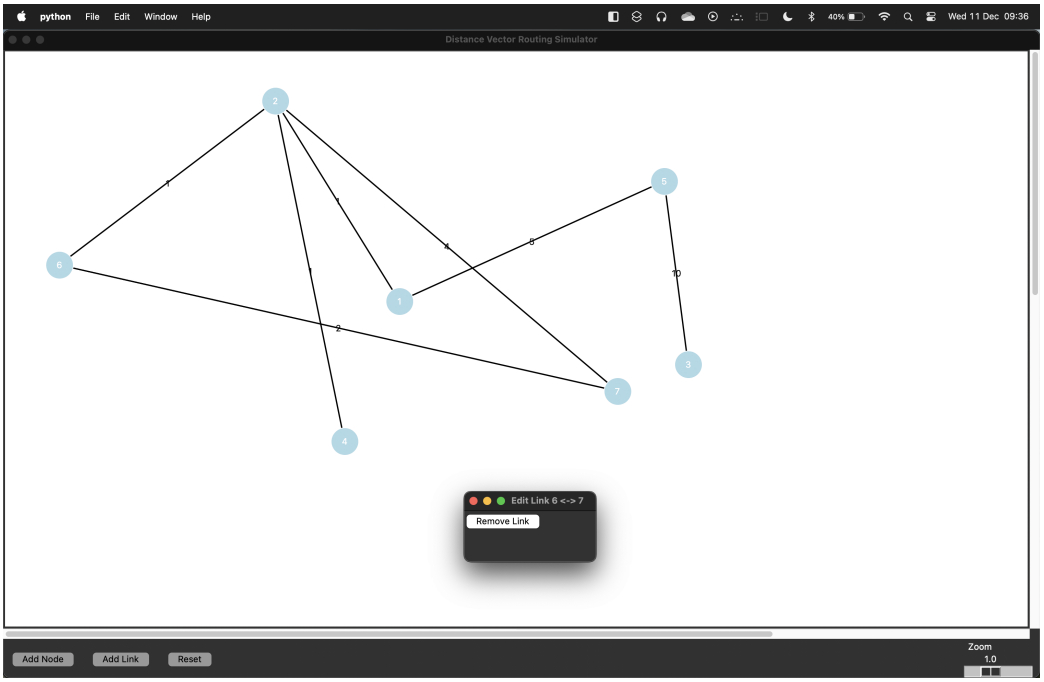

Figure 4: Rimozione collegamento

dinamicamente le tabelle di routing, il progetto offre un'esperienza interattiva e coinvolgente. Le funzionalità di aggiunta e rimozione di nodi e collegamenti, insieme al controllo dello zoom, rendono l'applicazione flessibile e facile da usare. In futuro, il progetto potrebbe essere ampliato con funzionalità aggiuntive, come il salvataggio della topologia della rete o l'implementazione di altri algoritmi di routing. Questo progetto rappresenta una solida base per ulteriori sviluppi e studi nel campo delle reti.