

Traveling Salesman Problem con Ant Colony Optimization

1. Obiettivo del progetto

Lo scopo del lavoro è stato affrontare il **Traveling Salesman Problem (TSP)**, un classico problema di ottimizzazione combinatoria in cui bisogna determinare il percorso minimo che consente a un venditore di visitare un insieme di città una sola volta e poi tornare al punto di partenza.

Sono stati analizzati e confrontati due approcci principali:

- **Brute Force / Exhaustive Search**, che trova la soluzione ottima (tra tutte le combinazioni possibili), con costi computazionali elevati.
- **Ant Colony Optimization (ACO)**, una metaeuristica ispirata al comportamento collettivo delle formiche.

2. Implementazione dell'algoritmo Ant Colony Optimization

L'algoritmo ACO è basato sull'idea che le formiche, durante la ricerca del cibo, rilasciano **feromoni** lungo i percorsi migliori, aumentando la probabilità che altre formiche seguano lo stesso tragitto.

2.1 Costruzione della soluzione

Ogni formica costruisce un tragitto/percorso scegliendo la prossima città in base a:

- **Intensità del feromone (τ)**: misura quanto un percorso è stato considerato buono nelle iterazioni precedenti.
- **Visibilità (η)**: inversamente proporzionale alla distanza tra le città.

La probabilità di scelta è definita da:

$$P(i, j) \propto (\tau_{ij})^\alpha \cdot (\eta_{ij})^\beta$$

dove:

- α controlla l'importanza del feromone
- β controlla l'importanza della distanza

2.2 Aggiornamento dei feromoni

Al termine di ogni iterazione:

- Una parte del feromone evapora (per evitare convergenza precoce)
- Le formiche rilasciano nuovo feromone sui percorsi migliori

2.3 Parametri principali

I parametri utilizzati nell'implementazione includono:

- Numero di formiche
- Numero di iterazioni
- α e β (influenza feromone/distanza)
- ρ (evaporazione)
- Quantità di feromone depositata

3. Risultati sperimentali

L'algoritmo Ant Colony Optimization è stato testato su diverse istanze del Traveling Salesman Problem, confrontandolo con l'approccio Brute Force quando computazionalmente possibile.

3.1 Confronto con Brute Force (12 città)

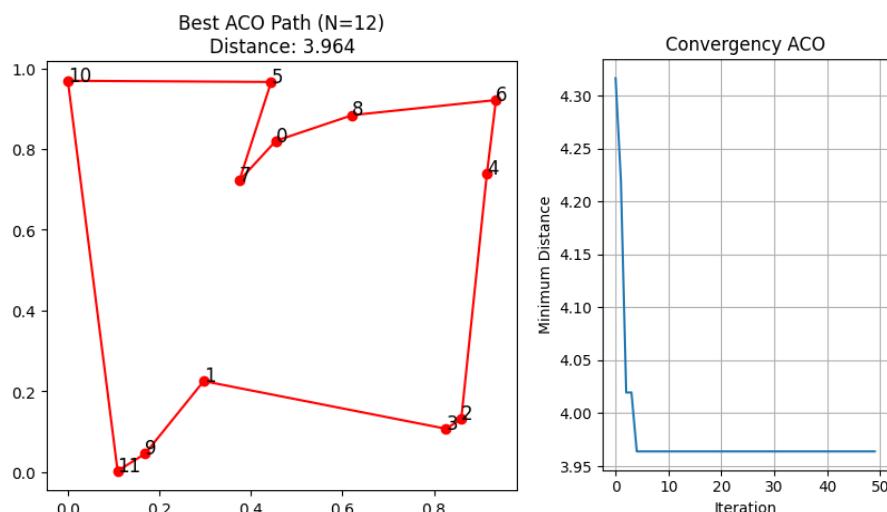
Nel caso di 12 città, è stato possibile confrontare direttamente ACO e Brute Force. I risultati ottenuti sono i seguenti:

- **Brute Force:**
 - Lunghezza del percorso: **3.3132**
 - Tempo di esecuzione: **79.92 secondi**
- **Ant Colony Optimization:**
 - Lunghezza del percorso: **3.3132**
 - Tempo di esecuzione: **0.29 secondi**

L'algoritmo ACO è riuscito a trovare la stessa soluzione ottima del Brute Force, ma con un tempo di esecuzione drasticamente inferiore (circa 270 volte più veloce).

3.2 Analisi della convergenza

Durante le iterazioni dell'algoritmo ACO si osserva una progressiva riduzione della lunghezza del percorso migliore.



Il grafico dell'andamento della soluzione migliore in funzione delle iterazioni mostra come il meccanismo dei feromoni guidi la convergenza verso soluzioni sempre più efficienti.

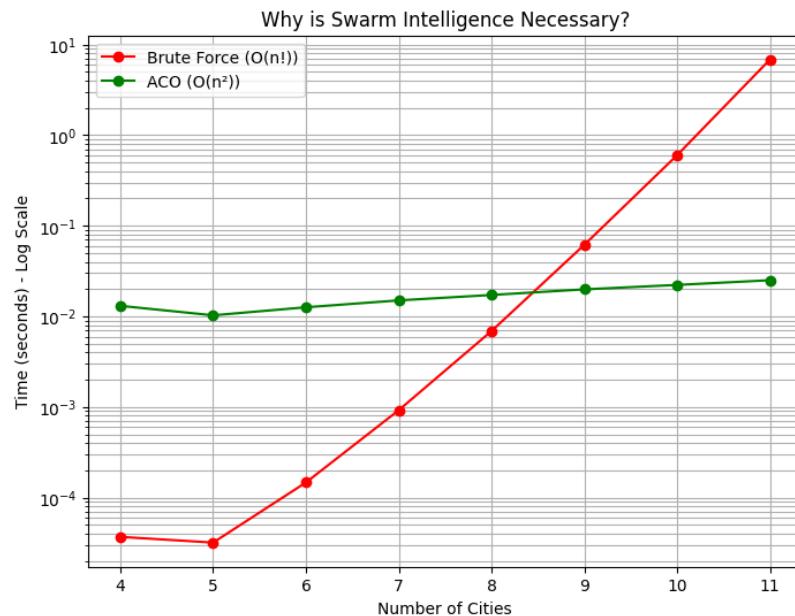
3.3 Scalabilità dell'algoritmo

Per valutare la scalabilità dell'approccio ACO, è stato eseguito un test su un'istanza di 50 città:

- **ACO**
 - Lunghezza del percorso: 6.5142
 - Tempo di esecuzione: 1.50 secondi

In questo caso il Brute Force risulta impraticabile, poiché richiederebbe tempi di calcolo dell'ordine di miliardi di anni.

Questo evidenzia come l'**Ant Colony Optimization** sia particolarmente adatto alla risoluzione di problemi **TSP di grandi dimensioni**.



Se volessimo misurare la scalabilità utilizzando 50 città:

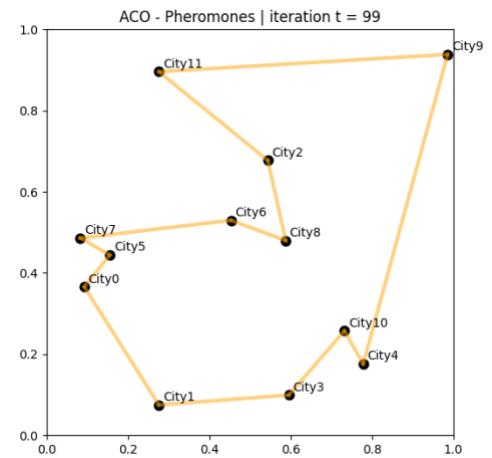
- ACO:
 - 6.1641 in 1.4569 seconds
- Brute Force:
 - Ci metterebbe miliardi di anni a trovare una soluzione

3.4 Analisi dell'evoluzione dei feromoni

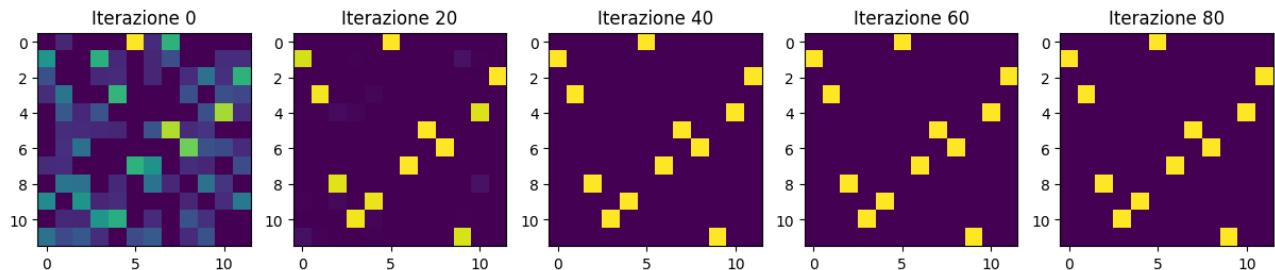
È stata effettuata un'analisi più approfondita includendo la visualizzazione dell'evoluzione, nello spazio, dei feromoni nel tempo.

All'inizio dell'esecuzione, i livelli di feromone associati agli archi tra le città sono inizializzati in modo uniforme. In questa fase, la scelta dei percorsi da parte delle formiche è fortemente influenzata dalla componente di visibilità (verso inversamente proporzionale alla distanza), e il comportamento risulta prevalentemente esplorativo.

Con il procedere delle iterazioni, gli archi che fanno parte di soluzioni più brevi ricevono una maggiore quantità di feromone, mentre sugli altri archi il feromone evapora progressivamente.



Evoluzione dei feromoni nel tempo



4. Conclusioni

Il progetto dimostra l'efficienza dell'**Ant Colony Optimization** come tecnica meta-euristica per risolvere il TSP.

Rispetto al brute force, ACO offre:

- Tempi computazionali molto più bassi
- Soluzioni di buona qualità anche con molte città
- Un modello flessibile e ottimizzabile tramite parametri

Per cui risulta un approccio valido per affrontare problemi complessi di ottimizzazione combinatoria.

Codice visionabile sotto la repository github al link:

https://github.com/GuerineHoussem/AI_Project