

UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

UNIVERSITÀ DEGLI STUDI DI MODENA

E REGGIO EMILIA

Dipartimento di Ingegneria "Enzo Ferrari"

Corso di Laurea Magistrale in Ingegneria Informatica (LM-32)

Analisi dei Trade-off di Posizionamento di Microservizi Multi-Versione in Infrastrutture Edge

Relatore:
Prof. Riccardo Lancellotti

Candidato:
Mario Coppola
Matricola 207913

ANNO ACCADEMICO 2025/2026

Indice

1 Introduzione **1**

1.1 Sfide dell'Ottimizzazione 1

1.2 Obiettivo 2

1.3 Panoramica 3

2 Stato dell'Arte e Tecnologie **4**

2.1 Paradigma *Fog* ed *Edge Computing* 4

2.1.1 Il Secondo Strato: *Edge Layer* 5

2.1.2 Il Terzo Strato: *Fog* Layer 6

2.1.3 Vincoli delle Applicazioni Reali 6

2.2 YOLO: Architettura, Standard De Facto e Implementazioni Computazionali 7

2.2.1 YOLO come Standard Attuale 8

2.2.2 Il Paradigma Multi-Versione 8

2.3 Euristiche Evolutive 9

2.3.1 Principi Fondamentali degli Algoritmi Genetici 10

2.4 Teoria dei Problemi Multi-Obiettivo 10

2.4.1 Formalizzazione del Problema Multi-Obiettivo 11

2.4.2 Limiti delle Tecniche di Scalarizzazione Classiche 11

2.4.3 Dominanza Paretiana e Fronte di Pareto 12

2.4.4 L'Euristica Evolutiva NSGA-II 12

2.4.5 Metriche Quantitative di Valutazione del Fronte 13

3 Modellazione Formale del Problema **15**

3.1 Notazione, Insiemi e Variabili Decisionali 16

3.1.1 Insiemi e Indici Fondamentali 16
3.1.2 Flussi di Dati e Parametri del Modello 17
3.1.3 Variabili Decisionali 17
3.1.4 Tabella Riassuntiva della Notazione 19
3.2 *Quality Model* del Microservizio 19
3.2.1 Matrice di Confusione Ponderata sul Traffico 20
3.2.2 Accuratezza Globale del Sistema 21
3.2.3 Funzione Obiettivo sulla Qualità 21
3.3 Modello delle Performance 22
3.3.1 Tempo di Risposta dell'Istanza di Microservizio 22
3.3.2 Tempo di Risposta Medio Pesato dell'Infrastruttura 23
3.3.3 Funzione Obiettivo Prestazionale 24
3.4 Modello Prestazionale e Teoria delle Code 24
3.4.1 Motivazione dell'Adozione del Modello *M/G/1* 25
3.4.2 Tasso di Arrivo Aggregato sul Nodo 26
3.4.3 Miscela delle Distribuzioni dei Tempi di Servizio 26
3.4.4 Fattore di Utilizzo e Condizione di Stabilità 27
3.4.5 Formula di Pollaczeck-Khinchine per il Tempo Medio di Attesa 27
3.5 Definizione Completa del Problema di Ottimizzazione 28
3.5.1 Modello Energetico dei Nodi 28
3.5.2 Formulazione delle Funzioni Obiettivo 29
3.5.3 Vincoli del Modello 30

4 Algoritmo Genetico e Implementazione **31**
4.1 Rappresentazione del Cromosoma 32
4.1.1 Struttura della Codifica a Coppie di Geni 32
4.1.2 Decodifica delle Variabili Decisionali del Modello 33
4.1.3 Vantaggi Algoritmici della Rappresentazione 34
4.2 Funzioni di Fitness 34
4.2.1 Computazione dei Parametri di Rete e dei Nodi 34

4.2.2 Struttura del Vettore di Fitness Multi-Obiettivo 35
4.2.3 Proprietà della Formulazione Vettoriale 36
4.3 Operatori Genetici 36
4.3.1 Inizializzazione della Popolazione 37
4.3.2 Operatore di *Crossover* 37
4.3.3 Operatore di Mutazione 38
4.3.4 Selezione Ambientale 39
4.4 Gestione dell'Ammissibilità 40
4.4.1 Procedura di Riparazione del Sovraccarico 41
4.4.2 Gestione dell'Inammissibilità Residua e Vincoli di SLA 42

5 Campagna Sperimentale e Analisi dei Risultati **43**
5.1 Set-up Applicativo di Riferimento 44
5.1.1 Modello del Carico Nominale 45
5.1.2 Infrastruttura di Rete e Nodi di Calcolo RSU 45
5.1.3 Caratterizzazione dei Microservizi Multi-Versione 47
5.2 Calibrazione dei Parametri dell'Algoritmo Genetico 49
5.2.1 Spazio di Esplorazione dei Parametri 49
5.2.2 Criteri di Valutazione e Configurazione Ottima 49
5.2.3 Protocollo di Validazione dei Risultati 50
5.3 Analisi del Fronte Pareto sotto Carico Uniforme 51
5.3.1 Struttura dello Spazio di Ricerca 51
5.3.2 Analisi delle Proiezioni Bidimensionali 53
5.3.3 Rilevanza per la Pianificazione dell'Infrastruttura 54
5.4 Analisi del Fronte di Pareto in Scenari Eterogenei 55
5.4.1 Dimensionamento dell'Eterogeneità Hardware 55
5.4.2 Modellazione del Carico Eterogeneo 56
5.4.3 Confronto Comparativo e Validazione *Multi-Seed* 56

6 Conclusioni e Sviluppi Futuri **59**
6.1 Sviluppi Futuri 60

Elenco delle Figure

2.1 Piramide *Cloud*, *Fog* ed *Edge Computing* 5
2.2 Esempio di Output di una Rete YOLO 7

4.1 Rappresentazione del Cromosoma 32
4.2 Esempio di *Two-Point Crossover* 38
4.3 Esempio di Mutazione Casuale 39

5.1 Posizione delle RSU 44
5.2 Distribuzione delle Distanze tra RSU 46
5.3 Rapporto Raggio vs. Copertura 47
5.4 F1-Score vs. Latenza 48
5.5 Fronte di Pareto Tridimensionale 52
5.6 Rappresentazione Bidimensionale del Fronte di Parteo 54
5.7 Analisi Comparativa delle Metriche di Interesse 57

Elenco degli Algoritmi

1 Algoritmo Genetico Multi-Obiettivo per *Placement* Multi-Versione 40

Elenco delle Equazioni

2.1 Equazione 11
2.2 Equazione 11
2.3 Equazione 12
2.4 Equazione 12
2.5 Equazione 12
2.6 Equazione 12
2.7 Equazione 13
2.8 Equazione 14
3.1 Equazione 17
3.4 Equazione 20
3.5 Equazione 20
3.6 Equazione 20
3.7 Equazione 20
3.8 Equazione 21
3.9 Equazione 21
3.10 Equazione 22
3.11 Equazione 23
3.12 Equazione 24
3.13 Equazione 24
3.14 Equazione 24
3.15 Equazione 26
3.16 Equazione 26
3.17 Equazione 26
3.18 Equazione 27

3.19 Equazione 27
3.20 Equazione 27
3.21 Equazione 27
3.22 Equazione 28
3.23 Equazione 29
3.24 Equazione 29
3.25 Equazione 29
3.26 Equazione 30
3.27 Equazione 30
3.28 Equazione 30
3.29 Equazione 30
3.30 Equazione 30
4.1 Equazione 32
4.2 Equazione 33
4.5 Equazione 35
4.6 Equazione 35
4.7 Equazione 35
4.8 Equazione 36
4.9 Equazione 36
4.11 Equazione 42
5.1 Equazione 56

Elenco delle tabelle

3.1 Notazione Usata per la Formulazione del Problema 19

5.1 Tipologie di Nodi Edge 45
5.2 Parametri dei Microservizi 48
5.3 Tabella Riassuntiva dei Parametri del Problema 50

1. Introduzione

La rapida proliferazione dei dispositivi dell'*Internet of Things* (**IoT**), assieme alla diffusione di applicazioni sensibili alla latenza, ha guidato una rapida adozione dell'*Edge Computing*. Questo paradigma sposta le risorse computazionali in prossimità delle sorgenti di generazione dei dati, riducendo sensibilmente i ritardi di rete e il consumo di banda. Tuttavia, il *deployment* di applicazioni in certi ambienti lascia aperte diverse questioni, a causa della natura fortemente eterogenea del problema, a cui si aggiungono i vincoli sulle risorse delle infrastrutture perimetrali.

Le moderne applicazioni, quali l'analisi video in tempo reale per la *Smart Mobility*, sono sempre più frequentemente progettate impiegando architetture a microservizi. Un aspetto cruciale, e spesso trascurato, risiede nel fatto che i microservizi sono comunemente disponibili in versioni multiple. Ad esempio, compiti di *machine learning*, come l'individuazione di oggetti, possono avvalersi di reti neurali caratterizzare da diverse dimensioni e complessità computazionali.

1.1 Sfide dell'Ottimizzazione

La disponibilità di molteplici versioni per ciascun servizio introduce un compromesso intrinseco e non-lineare. I modelli di dimensioni maggiori garantiscono un'accuratezza superiore a fronte di una maggiore potenza di calcolo, con conseguente incremento dei tempi di risposta e dei consumi energetici dell'infrastruttura.

Di conseguenza, il processo decisionale non si esaurisce nello stabilire dove collocare un determinato microservizio, bensì richiede di selezionare congiuntamente quale versione istanziare attraverso l'infrastruttura perimetrale. Quindi, tale problema deve gestire simultaneamente tre obiettivi in mutuo conflitto:

1. La minimizzazione del tempo di risposta globale dei servizi, comprensivo dei ritardi di trasmissione e dei tempi di attesa nelle code computazionali modellate secondo la teoria *M/G/1*.

2. La massimizzazione dell'accuratezza del servizio offerto.

3. La minimizzazione del consumo energetico complessivo dei nodi Edge eterogenei, tenendo conto sia della potenza statica di *idle* sia di quella dipendente dal fattore di utilizzo.

L'accoppiamento di decisioni di assegnamento discrete con vincoli operativi stringenti, quali la stabilità delle code di calcolo e i limiti sui tempi di risposta fissati dai *Service Level Agreement*, rende il problema *NP-Hard*. Ciò preclude l'impiego di approcci analitici esatti per istanze di dimensione realistiche e impone il ricorso a euristiche evolutive avanzate.

1.2 Obiettivo

L'obiettivo primario di questo lavoro è formulare analiticamente il problema di ottimizzazione multi-obiettivo per l'allocazione di microservizi multi-versione, proponendo un'euristica efficace basata su algoritmi genetici per approssimare il Fronte di Pareto.

Nello specifico, i contributi della tesi si articolano come segue:

- **Formulazione del modello multi-obiettivo**: Viene formalizzato un modello analitico congiunto che ottimizza simultaneamente tre metriche contrastanti: minimizzazione del tempo di risposta, massimizzazione dell'accuratezza e minimizzazione della potenza elettrica assorbita dall'infrastruttura.

- **Progettazione dell'euristica evolutiva:** Si propone un algoritmo genetico multi-obiettivo fondato sul paradigma elitista NSGA-II. La soluzione è integrata da una codifica cromosomica a coppie intere a lunghezza fissa, una strategia ibrida di riparazione del sovraccarico locale e un meccanismo di penalizzazione numerica delle violazioni di SLA.

- **Validazione sperimentale:** L'efficacia del modello e dell'euristica viene validata attraverso un caso d'uso realistico incentrato sulla mobilità veicolare intelligente all'interno del *living lab Modena Automotive Smart Area*. Impiegando microservizi multi-versione per l'analisi dei flussi video acquisiti da telecamere di traffico, è possibile dimostrare la capacità dell'approccio di caratterizzare lo spazio dei compromessi tra prestazioni, energia e qualità.

1.3 Panoramica

La trattazione è organizzata come segue:

1. **Stato dell'arte e tecnologie:** Colloca la proposta nel contesto della letteratura scientifica attraverso l'introduzione dei paradigmi di *Edge* e *Fog Computing*, la caratterizzazione architetturale dei modelli *single-stage Ultralytics YOLO* (e relative varianti) e la trattazione dei fondamenti teorici relativi alle euristiche evolutive e all'ottimizzazione multi-obiettivo.

2. **Modellazione formale del problema:** Espone la formulazione matematica del sistema, definendo la notazione algebrica adottata, ma metrica di qualità aggregata, l'analisi prestazionale dei tempi di attesa basata sulla teoria delle code, la caratterizzazione energetica lineare dei nodi e la formulazione delle tre funzioni obiettivo vincolate.

3. **Algoritmo genetico e implementazione:** Illustra l'architettura dell'euristica evolutiva proposta, approfondendo la codifica cromosomica a coppie di geni (n_i, v_i), la valutazione vettoriale della funzione di *fitness*, gli operatori specializzati di crossover e mutazione, per poi concludere con l'architettura software dell'implementazione.

4. **Campagna sperimentale e analisi dei risultati:** Presenta i risultati empirici ottenuti applicando l'algoritmo alla topologia reale delle RSU del *MASA*. La trattazione illustra la calibrazione degli iperparametri del motore genetico, l'analisi delle proiezioni bidimensionali e tridimensionali del Fronte di Pareto nominale, per poi concludersi con uno studio comparativo delle prestazioni dell'infrastruttura al variare dell'eterogeneità computazionale e sotto regimi di carico asimmetrico.

2. Stato dell'Arte e Tecnologie

Il presente capitolo delinea il quadro teorico e tecnologico alla base del problema di allocazione di microservizi in architetture distribuite [3, 13, 1]. La progettazione di sistemi di elaborazione *Edge* richiede l'uso di paradigmi architetturali avanzati, metodologie formali e strategie di ottimizzazione per riuscire a gestire applicazioni sempre più complesse.

Pertanto, l'analisi si sviluppa lungo quattro aspetti fondamentali:

- **Paradigma *fog* ed *edge computing*:** Evoluzione dai modelli centralizzati di *Cloud Computing* verso le architettura perimetrali a bassa latenza, analizzando la gerarchia computazionale e i vincoli infrastrutturali tipici dei nodi periferici.
- **Modelli *Ultralytics* YOLO:** Caratterizzazione dello stato dell'arte nei modelli di *object detection* a singolo stadio, evidenziando il conflitto intrinseco tra complessità architetturale, tempo di inferenza e accuratezza dell'output.
- **Euristiche evolutive:** Rassegna dei principali algoritmi genetici applicati all'esplorazione di spazi di ricerca combinatori ad alta dimensionalità.
- **Ottimizzazione multi-obiettivo:** Formalizzazione del concetto di dominanza paretiana, fronti di non-dominanza, metriche quantitative di valutazione delle prestazioni per problemi decisionali concorrenti e analisi dell'algoritmo NSGA-II.

2.1 Paradigma *Fog* ed *Edge Computing*

La progressiva diffusione di dispositivi *Internet of Things*, o IoT, e di applicazioni *time-critical* a bassissima latenza, ha messo in luce i limiti strutturali del modello *Cloud Computing* tradizionale. Nonostante i *data-center* centralizzati garantiscano capacità di calcolo e memorizzazione virtualmente illimitate, il trasferimento continuo di grandi moli di dati grezzi genera congestione della rete geografica, instabilità dei tempi di trasferimento e ritardi di propagazione incompatibili con requisiti operativi *real-time*. Di fatti, su grandi distanze geografiche, le informazioni possono impiegare centinaia di millisecondi per raggiungere la destinazione, se a ciò si aggiunge l'utilizzo

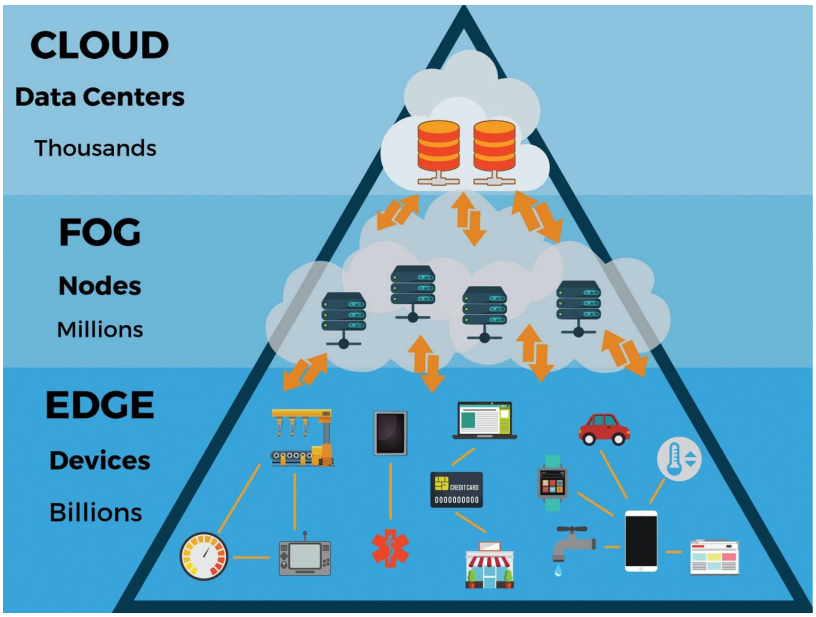


Figura 2.1: Piramide *Cloud*, *Fog* ed *Edge Computing*

di connessioni affidabili, come TCP, la trasmissione risente anche dell'*overhead* introdotto dal *three way-handshake*, circa 1.5 RTT. Dunque, l'approccio *Cloud* ha le capacità per poter soddisfare le richieste dei task in ingresso, ma potrebbe fornire i risultati in tempi troppo elevati, portando alla violazione di *Service Level Agreement*.

2.1.1 Il Secondo Strato: *Edge Layer*

Per mitigare tali problematiche è stato pensato un *layer* che potesse soddisfare le esigenze di task *time-critical* con latenze stringenti. Tale strato è chiamato *Edge Layer*. L'*Edge Computing* sposta le risorse computazionali e di memorizzazione in prossimità logica e fisica delle sorgenti dati, quali telecamere, sensori veicolari, dispositivi mobili. Elaborando le informazioni all'estremità della rete, l'*Edge Computing* riduce sensibilmente il traffico e garantisce tempi di risposta deterministici dell'ordine dei millisecondi, abilitando applicazioni come: *healthcare*, guida autonoma, industria 4.0. Nonostante il vantaggio significativo, l'*Edge layer* presenta ancora dei limiti non indifferenti. La limitata potenza computazionale e le scarse abilità di aggregazione delle richieste rendono il gravo computazionale sul *Cloud* ancora importante, anche se notevolmente ridotto.

2.1.2 Il Terzo Strato: *Fog* Layer

Quindi, osservando i due estremi, è stato necessario inserire uno strato intermedio che sia in grado di gestire le richieste di applicazioni con vincoli stringenti dal punto di vista energetico e computazionale. Il *Fog Computing* estende e generalizza l'approccio *Edge* introducendo un *layer* costituito da piccoli *data-center* organizzati in domini. Mentre la gestione della singola isola computazionale è puramente centralizzata, coordinata da un *controller* centrale, l'interazione tra i *data-center* è realizzata mediante algoritmi distribuiti che, sfruttando una stretta comunicazione, riescono a coordinare i diversi domini. Il mondo *Fog*, inoltre, non rappresenta solo un nuovo strato di computazione, bensì un ulteriore livello di aggregazione delle richieste. Nello specifico, i task da indirizzare al *data-center* centrale sono filtrati e aggregati con una granularità superiore rispetto allo strato *Edge*, evitando quindi di inoltrare richieste duplicate o grezze. Grazie a queste piccole accortezze, la mole di dati inviata al *Cloud* è drasticamente ridotta.

2.1.3 Vincoli delle Applicazioni Reali

Nelle implementazioni reali, le applicazioni vengono tipicamente decomposte in microservizi containerizzati per garantire modularità e portabilità. Ciononostante, l'efficienza complessiva dell'architettura rimane condizionata da vincoli fisici e strutturali:

- **Eterogeneità delle risorse:** I nodi computazionali dispongono di acceleratori hardware disomogenei, determinando velocità di elaborazione, capacità di *throughput* e costi energetici marcatamente differenti.
- **Dislocazione topologica:** La comunicazione tra sensori e nodi Fog si basa su reti *wireless* caratterizzate da ritardi di propagazione variabili, a seconda della distanza fisica, rendendo l'allocazione spaziale dei microservizi un fattore discriminante quanto la pura potenza di calcolo.



Figura 2.2: Esempio di Output di una Rete YOLO

2.2 YOLO: Architettura, Standard De Facto e Implementazioni Computazionali

I primi modelli di successo per l'*object detection*, quali la famiglia di R-CNN e i suoi successori *Fast R-CNN* e *Faster R-CNN*, adottavano un approccio a due stadi. In questi sistemi, una prima rete neurale, detta *Region Proposal Network*, seleziona e propone regioni d'interesse all'interno dell'immagine, mentre un secondo classificatore elabora ciascuna regione candidata per estrarne le etichette di classe e affinare le coordinate spaziali. Sebbene questo paradigma garantisca elevati livelli di accuratezza, il costo computazionale associato e la natura sequenziale dei due stadi generano latenze di inferenza elevate.

Per superare tale collo di bottiglia temporale, è stato introdotto il paradigma *single-stage*, o *single-shot*, di cui la famiglia dei modelli YOLO rappresenta il principale esponente e standard [15, 9]. A differenza dei modelli a due stadi, YOLO riformula l'*object detection* come un unico problema di regressione *end-to-end*. Un'unica rete neurale convoluzionale elabora un'intera immagine d'ingresso in un solo passaggio, predicendo simultaneamente le coordinate spaziali di molteplici *bounding boxes* e le relative probabilità di classe.

2.2.1 YOLO come Standard Attuale

Nel panorama attuale, le implementazioni della famiglia YOLO hanno consolidato la loro leadership per diverse ragioni:

- **Ottimizzazione delle componenti architetturali:** L'integrazione di *backbone* moderni basati su modelli *Cross Stage Partial* o blocchi con convoluzioni riparametrizzate, consente un'estrazione delle caratteristiche geometriche e visive estremamente efficiente a parità di parametri.
- **Architettura *anchor-free*:** L'eliminazione delle *anchor boxes* a favore della predizione diretta delle distanze dai punti centrali semplifica il calcolo della *loss function*. Questo approccio, che supera i riquadri predefiniti della famiglia *Faster R-CNN*, migliora sensibilmente la convergenza durante l'addestramento e la robustezza nel rilevamento a diverse scale.
- **Supporto nativo per l'inferenza perimetrale:** I modelli sono concepiti per essere facilmente esplorati e quantizzati verso formati di *runtime* per l'*Edge*, come TensorRT per GPU NVIDIA, massimizzando il *throughput* computazionale sulle piattaforme hardware *embedded* e sulle *Road Side Unit*.

2.2.2 Il Paradigma Multi-Versione

Una caratteristica centrale della libreria Ultralytics YOLO risiede nella disponibilità del modello in diverse versioni architetturali. Nello specifico, variando sistematicamente la profondità degli strati convoluzionali e il numero di canali estratti, si ottengono le seguenti 5 versioni:

- ***Nano* (N):** Variante ultra-leggera, caratterizzata da un ridotto numero di parametri e un basso volume di operazioni *floating-point*. È concepita per dispositivi con stringenti vincoli termici e computazionali, offrendo tempi di inferenza minimi ma sacrificando la capacità di discriminare oggetti complessi o parzialmente occlusi.
- ***Small* (S):** Configurazione compatta e bilanciata, adatta per inferenze rapide su processori *general-purpose* con un lieve incremento dell'accuratezza.

- ***Medium* (M):** Variante intermedia che bilancia in modo uniforme l'onere computazionale e le capacità percettive del modello.

- ***Large* (L):** Modello ad alta capacità espressiva, in grado di garantire una precisione elevata al prezzo di un incremento sensibile del carico computazionale.

- ***Extra-Large* (XL):** Configurazione a massima complessità parametrica, studiata per massimizzare il tasso delle predizioni corrette riducendo il numero di falsi allarmi. Tuttavia, questa versione richiede un'elevata potenza di calcolo e tempi di elaborazione consistenti.

Quindi, utilizzando diverse versioni della rete YOLO, si trasforma l'assegnazione dei microservizi in una complessa decisione congiunta. Non si tratta unicamente di stabilire dove posizionare il servizio computazionale, ma di selezionare quale versione risponda in modo ottimale al triplice compromesso tra qualità visiva, latenza dell'anello di controllo e budget energetico dell'infrastruttura.

2.3 Euristiche Evolutive

La risoluzione di problemi di ottimizzazione combinatoria in ambienti *Edge/Fog* presenta notevoli complessità. Come formalizzato nel **Capitolo 3**, determinare congiuntamente posizione e versione del microservizio su nodi eterogenei costituisce un problema decisionale non lineare e vincolato. Tale formulazione ricade nella classe dei problemi *NP-Hard*. All'aumentare del numero di sorgenti $|\mathcal{M}|$, dei nodi computazionali $|\mathcal{E}|$ e delle varianti implementative disponibili $|\mathcal{V}_m|$, la cardinalità dello spazio di ricerca cresce in modo esponenziale secondo l'espressione $\prod_{m\in\mathcal{M}} (|\mathcal{E}| \cdot |\mathcal{V}_m|)$.

Data la grandezza dello spazio di ricerca, l'applicazione di metodi esatti di programmazione matematica, come il *Branch and Bound* o la programmazione lineare intera, risulta impraticabile. Spesso, questo approccio è usato unicamente per istanze su piccola scala, poiché non garantisce i requisiti minimi per applicazioni reali. Per superare tale barriera computazionale, si ricorre a metodologie euristiche e meta-euristiche, capaci di identificare configurazioni ad elevata qualità sub-ottima in tempi trattabili [4, 5, 12].

2.3.1 Principi Fondamentali degli Algoritmi Genetici

Tra le meta euristiche a base di popolazione, gli algoritmi genetici rappresentano una delle classi più studiate e consolidate. Ispirati ai meccanismi della selezione e della genetica mendeliana, gli algoritmi genetici operano facendo evolvere iterativamente un insieme di soluzioni candidate verso regioni a maggiore *fitness* nello spazio delle soluzioni.

Il ciclo evolutivo si articola attraverso passaggi sequenziali ben definiti:

- **Rappresentazione/Cromosoma:** Ciascuna soluzione potenziale viene codificata in una struttura dati vettoriale o a stringa. I singoli elementi, anche detti geni, codificano le decisioni atomiche del problema.

- **Valutazione della *fitness*:** Per quantificare l'adeguatezza e il rispetto dei vincoli operativi, ciascun individuo viene decodificato e valutato rispetto a uno o più funzioni di costo.

- **Crossover:** Operatore stocastico che combina il patrimonio genetico di due individui per generare nuova prole. Questa operazione favorisce l'esplorazione combinatoria dello spazio di ricerca, promuovendo la convergenza e l'aggregazione di blocchi costruttivi promettenti verso soluzioni a maggiore *fitness*.

- **Mutazione:** Operatore con bassa probabilità di attivazione che altera casualmente uno o più geni di un individuo. L'operazione promuove la diversità genetica e l'esplorazione di nuove aree dello spazio decisionale, evitando la convergenza prematura verso ottimi locali.

2.4 Teoria dei Problemi Multi-Obiettivo

Nei problemi di ottimizzazione complessi applicati a contesti reali, raramente la qualità di una configurazione ammissibile può essere ricondotta a una singola metrica prestazionale scalare. Più frequentemente, la progettazione del sistema richiede di bilanciare molteplici criteri intrinsecamente antagonisti. Nel caso dell'allocazione di servizi multi-versione in ambienti *Edge/Fog*, tali dimensioni corrispondono alla latenza di risposta, al consumo energetico dell'infrastruttura e all'accuratezza del modello visivo impiegato.

2.4.1 Formalizzazione del Problema Multi-Obiettivo

Un problema di ottimizzazione multi-obiettivo può essere formulato matematicamente come la minimizzazione simultanea di un vettore di M funzioni obiettivo:

$$\min \mathbf{f}(\mathbf{x}) = (f_1(x), f_2(x), \ldots, f_M(x)) \;\; s.t. \; \mathbf{x} \in \Omega \subseteq \mathbb{R}^D \tag{2.1}$$

ove Ω rappresenta la regione ammissibile dello spazio delle decisioni, delimitata dall'insieme dei vincoli operativi e strutturali del sistema.

A differenza dei problemi mono-obiettivo, in un problema di ottimizzazione multi-obiettivo non esiste una soluzione singola in grado di minimizzare simultaneamente tutte le componenti del vettore $\mathbf{f}(\mathbf{x})$. Il miglioramento di un obiettivo comporta inevitabilmente il degrado di almeno uno degli altri.

2.4.2 Limiti delle Tecniche di Scalarizzazione Classiche

L'approccio tradizionale per affrontare problemi multi-criterio consiste nella scalarizzazione, ovvero nella riconduzione del vettore obiettivo a una singola funzione scalare mediante una combinazione lineare pesata:

$$F(x) = \sum_{k=1}^{M} w_k f_k(x), \text{ con } \sum_{k=1}^{M} w_k = 1, \; w_k \geq 0 \tag{2.2}$$

Sebbene computazionalmente semplice, tale metodologia presenta forti limitazioni teoriche e pratiche:

- **Difficoltà nella calibrazione a priori dei pesi:** È estremamente complesso assegnare pesi w_k coerenti a metriche dimensionalmente eterogenee e caratterizzate da unità di misura incompatibili.
- **Incapacità di individuare ottimi in regioni non convesse:** I metodi a somma pesata falliscono sistematicamente nel campionare punti appartenenti a porzioni concave del fronte di compromesso, indipendentemente dalla combinazione scelta.
- **Perdita della granularità decisionale:** Riducendo lo spazio delle soluzioni a un singolo valore scalare, si oscurano le relazioni di *trade-off* tra gli obiettivi in gioco.

2.4.3 Dominanza Paretiana e Fronte di Pareto

Pertanto, al fine di superare i limiti della scalarizzazione, si ricorre al criterio di dominanza paretiana, formalizzato originariamente da Vilfredo Pareto.

Dati due vettori di decisione ammissibili $\mathbf{x}_a, \mathbf{x}_b \in \Omega$, si definisce che $\mathbf{x}_a$ domina in senso paretiano $\mathbf{x}_b$, definito con $\mathbf{x}_a \prec \mathbf{x}_b$, se e solo se:

1. La soluzione $\mathbf{x}_a$ non è peggiore di $\mathbf{x}_b$ su alcuna delle funzioni obiettivo:

$$\forall k \in \{1, \ldots, M\}, \; f_k(\mathbf{x}_a) \leq f_k(\mathbf{x}_b) \tag{2.3}$$

2. La soluzione $\mathbf{x}_a$ è strettamente migliore di $\mathbf{x}_b$ rispetto ad almeno un obiettivo:

$$\exists j \in \{1, \ldots, M\} : \; f_j(\mathbf{x}_a) < f_j(\mathbf{x}_b) \tag{2.4}$$

Una soluzione $\mathbf{x}^* \in \Omega$ si dice Pareto-ottima se non esiste alcuna altra soluzione ammissibile $\mathbf{x} \in \Omega$ tale che $\mathbf{x} \prec \mathbf{x}^*$.

L'insieme di tutti i punti Pareto-ottimi nello spazio delle decisioni costituisce il Pareto Set, detto anche $\mathcal{PS}$, espresso come:

$$\mathcal{PS} = \left\{ \mathbf{x}^* \in \Omega | \nexists \mathbf{x} \in \Omega : \mathbf{x} \prec \mathbf{x}^* \right\} \tag{2.5}$$

L'immagine del Pareto Set proiettata nello spazio delle funzioni obiettivo è denominata Fronte di Pareto, o $\mathcal{PF}$:

$$\mathcal{PF} = \left\{ \mathbf{f}(\mathbf{x}^*) \in \mathbb{R}^M | \mathbf{x}^* \in \mathcal{PS} \right\} \tag{2.6}$$

Il fronte di Pareto racchiude la totalità dei migliori compromessi reciprocamente non dominati.

2.4.4 L'Euristica Evolutiva NSGA-II

Per approssimare con accuratezza il Fronte di Pareto su problemi NP-Hard, si ricorre ai *Multi-Objective Evolutionary Algorithms*. Tra questi, il *Non-dominated Sorting Genetic Algorithm II*, o **NSGA-II**, rappresenta lo standard metodologico di riferimento [7] grazie alla combinazione di tre meccanismi operativi:

- ***Fast Non-Dominated Sorting*:** A ogni iterazione, le soluzioni candidate vengono ripartite in fronti sequenziali di dominanza decrescente $(F_1, F_2, \dots)$. Il primo fronte (F_1) contiene le soluzioni reciprocamente non dominate del pool corrente. Rimosso F_1, il fronte F_2 raccoglie le soluzioni non dominate tra quelle residue, fino a esaurimento della popolazione.

- **Elitismo** $(\mu + \lambda)$**:** I candidati della generazione precedente (P_{g-1}) vengono aggregati ai nuovi discendenti generati (Q_g), formando un insieme esteso di $2N_{pop}$ individui. La selezione dei sopravvissuti per la generazione attuale (P_g) avviene ammettendo in blocco i fronti a rango più favorevole, garantendo che le configurazioni ottime individuate non vadano perse nel corso del processo evolutivo.

- ***Crowding distance*:** Qualora l'inserimento dell'ultimo fronte non dominato determini il superamento della popolazione N_{pop}, la selezione delle soluzioni residue avviene calcolando la *crowding distance*, o distanza di affollamento. Questa grandezza stima la densità locale dei punti nello spazio delle funzioni obiettivo, favorendo le soluzioni isolate e garantendo un campionamento uniforme ed esteso dell'intero fronte paretiano, senza introdurre parametri soggettivi di nicchia.

2.4.5 Metriche Quantitative di Valutazione del Fronte

Poiché gli algoritmi stocastici generano fronti approssimati, l'analisi delle prestazioni richiede indicatori quantitativi capaci di misurare convergenza, dispersione ed estensione del fronte indicato:

1. **Ipervolume (HV):** Misura il volume della regione degli obiettivi racchiusa tra l'insieme delle soluzioni non dominate e un punto di riferimento prefissato $\mathbf{r}$:

$$HV(A, \mathbf{r}) = \Lambda\left(\bigcup_{x \in A} [\mathbf{f}(\mathbf{x}), \mathbf{r}]\right) \tag{2.7}$$

   ove $\Lambda(\cdot)$ denota la misura di Lebesgue.

   L'ipervolume rappresenta la metrica più completa poiché valuta congiuntamente la convergenza, ovvero la prossimità al fronte teorico, e la diversità, ossia l'ampiezza della copertura. Un valore elevato di ipervolume corrisponde a un fronte paretiano quantitativamente superiore.

2. ***Spacing*/Uniformità geometrica:** Valuta l'omogeneità della distribuzione dei punti lungo il fronte, misurando la variabilità delle distanze relative tra soluzioni adiacenti:

$$S = \sqrt{\frac{1}{|A| - 1} \sum_{i=1}^{|A|} \left(d_i - \bar{d}\right)^2} \tag{2.8}$$

ove d_i indica la distanza minima tra la soluzione i-esima e la più vicina tra le rimanenti nello spazio normalizzato, mentre $\bar{d}$ è la distanza media.

Valori contenuti di *spacing* attestano una campionatura regolare e l'assenza di accumuli anomali o discontinuità nel fronte.

3. Modellazione Formale del Problema

In questo capitolo viene formalizzato il modello analitico e matematico posto alla base dell'allocazione di microservizi multi-versione all'interno di un'infrastruttura *Fog* ed *Edge Computing* eterogenea. Il *deployment* di applicazioni distribuite su nodi computazionali perimetrali richiede la caratterizzazione esplicita dei vincoli fisici, delle risorse computazionali e della natura stocastica dei flussi informativi elaborati.

Nel contesto considerato, la complessità decisionale non si esaurisce nell'individuare quale nodo fisico debba ospitare una determinata componente applicativa, ma include la selezione della specifica variante di microservizi da istanziare. L'introduzione di modelli neurali multi-versione, quali le diverse taglie della famiglia YOLO, genera un accoppiamento stretto e conflittuale tra accuratezza dell'output, tempi di elaborazione locale, congestione delle code di calcolo e potenza elettrica assorbita dall'infrastruttura.

Per cogliere tali dinamiche e tradurle in un problema di ottimizzazione multi-versione ben posto, la trattazione formale è organizzata lungo le seguenti direttrici:

- **Insiemi di riferimento e notazione algebrica:** Formalizzazione del dominio delle sorgenti di traffico, dell'insieme dei nodi *Fog*, dei microservizi e delle loro versioni, unitamente alla definizione dello spazio delle variabili decisionali binarie relative ad allocazione e stato operativo del server.

- **Modello di qualità dell'elaborazione:** Caratterizzazione quantitativa dell'accuratezza aggregata del servizio attraverso l'unione ponderata delle matrici di confusione associate a ciascuna variante del servizio.

- **Modello prestazionale e teoria delle code (M/G/1):** Descrizione della dinamica temporale del sistema, includendo la modellazione dei tempi di servizio come miscela di distribuzioni, il calcolo dei ritardi di accodamento mediante la formula di Pollaczek-Khinchine e l'integrazione della latenza di trasmissione di rete.

- **Formulazione analitica del problema multi-obiettivo:** Definizione congiunta delle tre funzioni obiettivo di minimizzazione e dei vincoli operativi di unicità di allocazione, stabilità asintotica delle code e rispetto degli accordi sul livello di servizio (**SLA**).

L'insieme di queste componenti fornisce il substrato matematico necessario per la valutazione rigorosa delle soluzioni e costituisce il fondamento su cui si innestano le funzioni di fitness e i meccanismi di riparazione dell'algoritmo genetico descritto **Capitolo 4**.

3.1 Notazione, Insiemi e Variabili Decisionali

La formulazione formale del problema di allocazione richiede la definizione preliminare degli insiemi di entità che compongono l'architettura perimetrale, dei parametri fisici e computazionali che ne descrivono il funzionamento, e dello spazio delle variabili decisionali che codificano le possibili strategie di dispiegamento.

3.1.1 Insiemi e Indici Fondamentali

Il sistema *Fog/Edge* è modellato attraverso quattro insiemi primari discreti:

- **Sorgenti di dati ($\mathcal{S}$):** Rappresenta l'insieme dei dispositivi generatori di traffico dislocati sul territorio. Ad ogni sorgente $s_m \in \mathcal{S}$ è associato un microservizio $m \in \mathcal{M}$.
- **Nodi computazionali *Fog/Edge* ($\mathcal{E}$):** Definisce l'insieme dei server perimetrali[1] disponibili per l'esecuzione dei carichi applicativi, indicati con $e \in \mathcal{E}$.
- **Microservizi applicativi ($\mathcal{M}$):** Rappresenta l'insieme delle componenti applicative indipendenti che devono essere istanziate sull'infrastruttura, ciascuna indicata con $m \in \mathcal{M}$.
- **Versioni del microservizio ($\mathcal{V}_m$):** Per ogni microservizio $m \in \mathcal{M}$, indica l'insieme delle implementazioni o taglie del modello disponibili. Un'istanza specifica del microservizio m configurata secondo la versione $v \in \mathcal{V}_m$ viene denotata con m_v.

[1] *Road Side Units*.

3.1.2 Flussi di Dati e Parametri del Modello

I requisiti e le proprietà di elaborazione associati a ciascun elemento sono quantificati attraverso i seguenti parametri scalari e stocastici:

- **Tasso di generazione del carico (λ_m):** Frequenza media con cui la sorgente s_m invia richieste di elaborazione al microservizio m, espressa in richieste al secondo. Il carico cumulativo offerto all'intera infrastruttura è:

$$\Lambda = \sum_{m \in \mathcal{M}} \lambda_m \tag{3.1}$$

- **Tempi di servizio (S_m^v, σ_m^v):** Il tempo di computazione necessario all'istanza m_v per processare una singola richiesta su una macchina di riferimento è trattato come una variabile casuale con valore atteso S_m^v e deviazione standard σ_m^v.

- **Metriche di accuratezza del modello ($\{TP, FP, TN, FN\}_m^v$):** Frazioni medie di veri positivi, falsi positivi, veri negativi e falsi negativi registrate sperimentalmente dalla versione v del microservizio m sul dataset di riferimento.

- ***Speedup* Computazionale del Nodo (A_e):** Fattore adimensionale che descrive la capacità di calcolo del nodo e rispetto a una macchina base, nel caso specifico $A_e = 1.0$ per i nodi *Standard* e $A_e = 1.5$ per i nodi *Hi-perf*, riducendo proporzionalmente il tempo di esecuzione effettivo.

- **Ritardo di trasmissione della rete ($\delta_{s_m,e}$):** Latenza di comunicazione introdotta dal collegamento di rete per trasferire i dati dalla sorgente s_m al nodo di calcolo e.

- **Assorbimento energetico (P_e^1, P_e^0):** Potenza assorbita dal nodo e in condizioni di *idle*[2] e di piena utilizzazione computazionale, $\rho_e = 1$.

3.1.3 Variabili Decisionali

L'arbitraggio dell'allocazione e della selezione delle versioni è governato da due famiglie di variabili decisionali binarie:

[2]Nodo acceso ma senza carico computazionale.

1. **Allocazione del microservizio ($x_{m,v,e} \in \{0,1\}$):**

$$x_{m,v,e} = \begin{cases} 1 & \text{se la versione } v \in \mathcal{V}_m \text{ del microservizio } m \in \mathcal{M} \text{ è allocata sul nodo } e \in \mathcal{E}, \\ 0 & \text{altrimenti,} \end{cases} \tag{3.2}$$

   Questa variabile a tre dimensioni identifica congiuntamente la collazione fisica e il modello computazionale associato a ciascun flusso applicativo.

2. **Stato operativo del nodo ($y_e \in \{0,1\}$):**

$$y_e = \begin{cases} 1 & \text{se il nodo Fog } e \in \mathcal{E} \text{ è acceso e ospita almeno un microservizio,} \\ 0 & \text{se il nodo è spento,} \end{cases} \tag{3.3}$$

   Tale variabile consente di definire in modo puntuale lo stato di un nodo e, di conseguenza, il peso dello stesso sull'infrastruttura.

3.1.4 Tabella Riassuntiva della Notazione

Simbolo	Spiegazione
Insiemi e Indici	
$\mathcal{S}$	Insieme delle Sorgenti Dati
$\mathcal{E}$	Insieme degli Edge Nodes
$\mathcal{M}$	Insieme dei Microservizi
$\mathcal{V}_m$	Insieme delle Versioni di un microservizio m
s_m	Sorgente Dati per un Microservizio m
e	Un Edge Node
m	Un Microservizio
m_v	Microservizio m, Versione v
Microservizi	
S_m^v	Tempo di Servizio per il Microservizio m, Versione v
σ_m^v	Deviazione Standard del Tempo di Servizio di m_v
λ_m	Tasso di Richieste per il Microservizio m
$\{T,F\}\{P,N\}_m^v$	Tasso di True/False Positive/Negative di m_v
$T_{m,v,e}$	Tempo di Risposta del Microservizio m_v
Flussi Dati	
λ_m	Dati Processati dal Microservizio m
λ_e	Data Rate in Ingresso sul Nodo e
Λ	Data Rate Totale in Ingresso
Nodi Edge	
ρ_e	Utilizzazione di Nodi Edge e
P_e^0	Consumo di Potenza quando e è Idle
P_e^1	Consumo di Potenza quando e è Pienamente Usato
S_e	Tempo di Servizio Medio sul Nodo Edge e
σ_e	Deviazione Standard sul Tempo di Servizio sul Nodo e
W_e	Tempo di Attesa Medio sul Nodo Edge e
$\delta_{s,e}$	Ritardo di Rete tra Sorgente s e Nodo Edge e
A_e	Speedup del Nodo Edge e
Variabili Decisionali Binarie	
$x_{m,v,e}$	Microservizio m_v sul nodo Edge e se uguale a uno
y_e	Nodo Edge e è acceso se uguale a 1

Tabella 3.1: Notazione Usata per la Formulazione del Problema

3.2 *Quality Model* del Microservizio

La valutazione della qualità del servizio all'interno di un'architettura perimetrale multi-versione differisce dalle formulazioni tradizionali, focalizzate principalmente su metriche di disponibilità o affidabilità della rete. Nel contesto analizzato, la presenza di differenti taglie del modello

neurale YOLO, dalla variante *Nano* a quella *Extra-Large*, introduce una variabilità intrinseca nell'accuratezza con cui le immagini dei flussi video vengono elaborate e classificate.

L'obiettivo di questa sezione è quantificare formalmente il rendimento qualitativo del sistema, tenendo conto che ciascun microservizio può operare con una specifica versione e processare un volume di richieste disomogeneo.

3.2.1 Matrice di Confusione Ponderata sul Traffico

Ciascuna versione $v \in \mathcal{V}_m$ associata a un microservizio $m \in \mathcal{M}$ è profilata empiricamente attraverso i quattro parametri standard di classificazione ricavati sul dataset di prova: TP_m^v, FP_m^v, TN_m^v e FN_m^v.

Poiché le diverse sorgenti s_m trasmettono dati con tassi di arrivo λ_m potenzialmente eterogenei, l'impatto di un microservizio sulla qualità complessiva deve essere pesato in base alla quota di traffico effettivamente gestita rispetto al carico totale $\Lambda = \sum_{m \in \mathcal{M}} \lambda_m$.

I tassi aggregati a livello di infrastruttura vengono formalizzati correlando le variabili decisionali di assegnamento, $x_{m,v,e}$, ai parametri empirici di ciascuna variante:

$$TP = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} x_{m,v,e} TP_m^v \tag{3.4}$$

$$FP = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} x_{m,v,e} FP_m^v \tag{3.5}$$

$$TN = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} x_{m,v,e} TN_m^v \tag{3.6}$$

$$FN = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} x_{m,v,e} FN_m^v \tag{3.7}$$

Questo schema di aggregazione assicura che le scelte di *deployment* applicate alle sorgenti caratterizzate da un elevato tasso di generazione λ_m abbiano un peso proporzionalmente maggiore rispetto a flussi a bassa frequenza di arrivo.

3.2.2 Accuratezza Globale del Sistema

Nel contesto dell'applicazione di riferimento, i pixel o le porzioni di immagini che non contengono elementi di interesse rappresentano la classe di sfondo, meglio definito come *background*. Essendo quest'ultimo la componente visiva statisticamente preponderante all'interno di ciascun fotogramma, il computo esplicito dei *Veri Negativi* altererebbe significativamente la valutazione, poiché qualunque modello tenderebbe a raggiungere valori di accuratezza prossimi al 100% mascherando le reali discrepanze prestazionali tra modelli leggeri, *Nano*, e modelli ad alta capacità, *Extra-Large*.

Per ovviare a questo *bias* e garantire una metrica fedele all'effettiva capacità di rilevamento del target, nel protocollo di test si esclude la componente dei Veri Negativi ponendo $TN_m^v = 0.0 \; \forall \; (m, v) \in (\mathcal{M}, \mathcal{V}_m)$, come validato sperimentalmente nella profilazione COCO in **Tabella 5.2** [11]. Da ciò consegue che $TN = 0$.

La formulazione dell'accuratezza globale, indicata con Acc, si riduce pertanto al rapporto tra i soli veri positivi rilevati dall'unione complessiva delle predizioni target e degli errori di classificazione:

$$Acc = \frac{TP}{TP + FP + FN} \tag{3.8}$$

3.2.3 Funzione Obiettivo sulla Qualità

All'interno della formulazione multi-obiettivo, la massimizzazione dell'accuratezza Acc viene convertita nella minimizzazione del tasso di errore, ovvero il completamento dell'accuratezza:

$$\min f_1(x) = 1 - Acc = 1 - \frac{TP}{TP + FP + FN} \tag{3.9}$$

La minimizzazione di $f_1(x)$ incentiva l'algoritmo genetico ad allocare le versioni YOLO caratterizzate da un TP_m^v più elevato e da un minore FP_m^v, visibile soprattutto nelle reti *Large* ed *Extra-Large*. Tali modelli comportano tuttavia tempi medi di servizio S_m^v e deviazioni standard σ_m^v superiori, introducendo il *trade-off* fondamentale con i tempi di risposta e la potenza assorbita.

3.3 Modello delle Performance

Il modello delle performance ha lo scopo di determinare analiticamente il tempo di risposta associato all'esecuzione di ciascuna istanza di microservizio sull'infrastruttura perimetrale. Nelle architetture Fog/Edge distribuite, tale grandezza non coincide unicamente con la durata dell'elaborazione computazionale locale, ma incorpora i ritardi di accodamento generati dalla contesa delle risorse e la latenza di comunicazione introdotta dalla trasmissione dei dati attraverso la rete.

3.3.1 Tempo di Risposta dell'Istanza di Microservizio

Considerando un'istanza m_v del microservizio $m \in \mathcal{M}$ configurata secondo la versione $v \in \mathcal{V}_m$ e allocata sul nodo Edge $e \in \mathcal{E} \mid x_{m,v,e} = 1$, il tempo di risposta complessivo, denotato con $T_{m,v,e}$, viene formalizzato come:

$$T_{m,v,e} = \frac{S_m^v}{A_e} + W_e + \delta_{s_m,e} \tag{3.10}$$

L'espressione scompone il tempo di risposta in tre contributi additivi fondamentali:

- **Tempo effettivo di servizio locale (S_m^v/A_e):** Rappresenta il tempo richiesto dal processore del nodo e per completare l'elaborazione di una singola richiesta. Il tempo medio nominale di benchmark S_m^v viene scalato mediante il fattore di accelerazione adimensionale A_e, che modella l'eterogeneità computazionale dei nodi.

  In particolare, per nodi di tipo Standard si ha $A_e = 1.0$, preservando così il tempo di calcolo nominale di riferimento. Per i nodi di tipo *Hi-perf*, invece, si ha $A_e = 1.5$, che introduce uno *speedup* del 50% e contrae il tempo puro di inferenza del microservizio.

- **Tempo medio di attesa in coda (W_e):** Quantifica il ritardo medio trascorso da una richiesta nel buffer del nodo e prima dell'effettivo inizio dell'elaborazione. Poiché su un medesimo nodo possono coesistere più microservizi con differenti tassi di arrivo e tempi di elaborazione, la dinamica di accodamento è descritta come un servitore a coda singola *M/G/1* soggetto a una miscela di distribuzioni, risolto analiticamente mediante la formula di Pollaczek-Khinchine, come spiegato nella **Sezione 3.4**.

- **Ritardo di comunicazione di rete ($\delta_{s_m,e}$):** Descrive la latenza di trasmissione dei dati dalla sorgente di acquisizione fisica $s_m \in \mathcal{S}$ al nodo di calcolo ospitante $e \in \mathcal{E}$. Nello scenario analizzato, tale valore segue una distribuzione binomiale su collegamenti *wireless* a lungo raggio:
  - $\delta_{near} = 2$ ms per nodi fisicamente adiacenti o appartenenti alla medesima area urbana, ritardo applicato a circa il 50% dei collegamenti.
  - $\delta_{far} = 10$ ms per comunicazioni che attraversano tratte geografiche più ampie o soggette a instradamento *multi-hop*, caso applicato alla restante metà dei link.

Questa assunzione si basa su studi passati [5]. Di fatto, si è dimostrato come i benchmark per ottimizzazioni multi-obiettivo possono diventare banali se una metrica domina completamente le altre:

- Se i ritardi di rete sono trascurabili (e.g., < 0.5 ms), l'algoritmo tenderà ad eseguire *offloading* del traffico sul nodo più potente e veloce senza penalità.
- Se i ritardi di rete sono troppo elevati, centinaia di millisecondi, l'algoritmo sarebbe forzato a processare i dati localmente, sulla sorgente dati, per poter soddisfare i vincoli di latenza.

Grazie a questa configurazione, si garantisce che i tempi di trasferimento di rete abbiano lo stesso ordine di grandezza dei tempi di calcolo, rendendo il problema di allocazione computazionalmente non banale.

3.3.2 Tempo di Risposta Medio Pesato dell'Infrastruttura

Poiché ciascuna sorgente s_m genera dati a un tasso λ_m potenzialmente eterogeneo, il tempo di risposta globale dell'intero sistema è definito come la media pesata dei tempi di risposta delle singole istanze sul volume di traffico effettivamente processato.

Normalizzando rispetto al carico cumulativo totale immesso nella rete $\Lambda = \sum_{m \in \mathcal{M}} \lambda_m$, l'indicatore di prestazione aggregato dell'infrastruttura assume la seguente espressione formale:

$$\bar{T} = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} T_{m,v,e} x_{m,v,e} \tag{3.11}$$

Sostituendo la decomposizione temporale della singola istanza $T_{m,v,e}$, si ottiene la forma esplicita:

$$\bar{T} = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} x_{m,v,e} \left(\frac{S_m^v}{A_e} + W_e + \delta_{s_m,e} \right) \tag{3.12}$$

3.3.3 Funzione Obiettivo Prestazionale

All'interno dell'ottimizzazione multi-obiettivo, il modello mira a minimizzare il tempo di risposta medio dell'infrastruttura, formalizzato come la seconda funzione obiettivo del problema:

$$\min f_2(x) = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} T_{m,v,e} x_{m,v,e} \tag{3.13}$$

Inoltre, per ciascun microservizio viene garantito il rispetto di un accordo sul livello di servizio (**SLA**[3]) attraverso il vincolo deterministico:

$$x_{m,v,e} T_{m,v,e} \leq T_m^{SLA} \; \forall m \in \mathcal{M}, \forall v \in \mathcal{V}_m, \forall e \in \mathcal{E} \tag{3.14}$$

dove T_m^{SLA} rappresenta la soglia temporale massima ammissibile per l'anello di retroazione e controllo applicativo.

La minimizzazione di $f_2(x)$ spinge l'algoritmo a preferire versioni leggere di YOLO, come *Nano* o *Small*, con tempi S_m^v ridotti e a collocare i servizi su nodi dotati di accelerazione, ovvero $A_e = 1.5$, con ritardo di rete minimo, pari a δ_{near}. Tuttavia, questo obiettivo entra in diretto contrasto con la massimizzazione dell'accuratezza e il contenimento di consumi energetici, analizzato nella **Sottosezione 3.5.1**.

3.4 Modello Prestazionale e Teoria delle Code

Nella **Sottosezione 3.3.1**, il tempo di risposta di un microservizio è stato espresso considerando il tempo medio di attesa in coda W_e come una componente adattiva. La determinazione analitica di tale grandezza richiede di modellare formalmente la contesa delle risorse computazionali che si

[3] *Service Level Agreement*

instaura quando molteplici microservizi, caratterizzati da eterogeneità di traffico e di complessità algoritmica, vengono allocati sul medesimo nodo Edge $e \in \mathcal{E}$.

Ciascun nodo di elaborazione viene modellato come un sistema di accodamento a servitore singolo con arrivi markoviani e tempi di servizio a distribuzione generale, denotato nella notazione di Kendall come sistema *M/G/1*.

3.4.1 Motivazione dell'Adozione del Modello *M/G/1*

La letteratura classica sui sistemi distribuiti ricorre frequentemente all'ipotesi semplificativa di coda *M/M/1*, in cui sia gli intertempi di arrivo sia i tempi di servizio sono descritti da variabili casuali esponenziali con proprietà di assenza di memoria[4]. Sebbene l'ipotesi poissoniana sui flussi di arrivo rimanga valida per l'aggregazione di flussi asincroni e indipendenti dalle telecamere delle RSU, assumere tempi di esecuzione esponenziale risulterebbe inadeguato e fortemente distorsivo per il caso d'uso in esame.

Per iniziare, il tempo richiesto da un acceleratore per eseguire l'inferenza di una specifica variante neurale non è esponenziale: una distribuzione per cui $\sigma = \mu$, con coefficiente di variazione unitario $C_v = 1$. Come evidenziato dalla **Tabella 5.2**, l'esecuzione di un singolo modello presenta tempi altamente deterministici con una dispersione minima. Ad esempio, per YOLO *Nano* si ha $S^{Nano} = 12.81$ ms con $\sigma^{Nano} = 1.63$ ms, che equivale a un coefficiente di variazione $C_v \approx 0.127$; mentre per YOLO *Extra-Large* si registra $S^{XL} = 51.58$ ms con $\sigma^{XL} = 5.07$ ms, corrispondente a un coefficiente di variazione $C_v \approx 0.098$. Assumere una distribuzione esponenziale sovrastimerebbe gravemente la deviazione standard del singolo processo di circa un'ordine di grandezza.

Inoltre, quando un nodo ospita contemporaneamente istanze eterogenee, come flussi elaborati da YOLO *Nano* e YOLO *Extra-Large*, la densità di probabilità del tempo di servizio sul nodo presenta molteplici picchi distinti, creando una distribuzione detta anche distribuzione multi-modale. Tale profilo probabilistico non è riconducibile a un'esponenziale negativa decrescente, ma costituisce una miscela discreta di distribuzioni generali.

Per questi motivi, l'adozione della coda *M/G/1* si rivela dunque necessaria e statisticamente fedele alla realtà sperimentale, poiché consente di modellare distribuzioni di servizio arbitrarie (G)

[4]Memoryless.

preservando l'impatto congiunto sia della varianza interna a ciascun modello sia della discrepanza media tra le diverse versioni collocate sul medesimo processore.

3.4.2 Tasso di Arrivo Aggregato sul Nodo

Poiché i processi di generazione delle sorgenti seguono distribuzioni di Poisson indipendenti con tassi λ_m, il flusso di traffico cumulativo in ingresso al nodo Edge e, denotato con λ_e, è dato dalla sovrapposizione dei tassi di arrivo di tutti i microservizi assegnati a tale nodo:

$$\lambda_e = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} x_{m,v,e} \lambda_m \tag{3.15}$$

In virtù della proprietà di additività dei processi di Poisson, il flusso di arrivo complessivo al nodo costituisce a sua volta un processo poissoniano omogeneo di intensità λ_e.

3.4.3 Miscela delle Distribuzioni dei Tempi di Servizio

La coesistenza di microservizi differenti associati a diverse varianti neurali implica che il servitore del nodo elabori compiti con requisiti computazionali disomogenei. Il tempo di servizio sul nodo e viene trattato analiticamente come una miscela di distribuzioni, dove la probabilità con cui il processore esegue una richiesta del microservizio m con versione v coincide con la frazione di traffico da essa generata rispetto al volume totale in ingresso al nodo, ossia λ_m/λ_e.

Tenendo conto del fattore di accelerazione locale A_e, il tempo di servizio aggregato S_e sul nodo e è definito dal primo momento della miscela:

$$S_e = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \frac{\lambda_m}{\lambda_e} x_{m,v,e} \frac{S_m^v}{A_e} \tag{3.16}$$

Per ricavare la dispersione statistica dei tempi di esecuzione, si calcola il secondo memento non centrale della miscela:

$$\mathbb{E}\left[S_e^2\right] = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \frac{\lambda_m}{\lambda_e} x_{m,v,e} \frac{(S_m^v)^2 + (\sigma_m^v)^2}{A_e^2} \tag{3.17}$$

Sottraendo il quadrato del valore atteso $(\mathbb{E}[S_e])^2 = S_e^2$, la varianza complessiva del tempo di servizio σ_e^2 sul nodo e è formalizzata come:

$$\sigma_e^2 = \left(\sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_e} \frac{\lambda_m}{\lambda_e} x_{m,v,e} \frac{\left(S_m^v\right)^2 + \left(\sigma_m^v\right)^2}{A_e^2} \right) - S_e^2 \tag{3.18}$$

Questa espressione cattura rigorosamente sia la varianza intrinseca delle singole inferenze $(\sigma_m^v)^2$ sia la varianza indotta dalla diversità tra i tempi di medi di esecuzione $\left(S_m^v\right)$ delle versioni co-allocate sul nodo.

3.4.4 Fattore di Utilizzo e Condizione di Stabilità

A partire dal tasso di arrivo cumulativo λ_e e dal tempo medio di servizio S_e, il fattore di utilizzo del nodo ρ_e è definito da:

$$\rho_e = \lambda_e S_e \tag{3.19}$$

Affinché la coda non diverga e il sistema mantenga una stabilità a regime, il fattore di carico deve risultare strettamente inferiore all'unità:

$$\rho_e < 1, \forall e \in \mathcal{E} \tag{3.20}$$

Nel modello di ottimizzazione, tale condizione viene vincolata imponendo $\rho_e \leq y_e(1-\epsilon)$, dove $\epsilon > 0$ rappresenta un margine di sicurezza che previene saturazioni e impone utilizzazione nulla sui nodi spenti $y_e = 0$.

3.4.5 Formula di Pollaczeck-Khinchine per il Tempo Medio di Attesa

Nel sistema *M/G/1* stazionario, il tempo medio di attesa in coda W_e viene quantificato esattamente mediante la formula di Pollaczeck-Khinchine:

$$W_e = S_e \cdot \frac{1 + \left(\frac{\sigma_e}{S_e}\right)^2}{2} \cdot \frac{\rho_e}{1 - \rho_e} \tag{3.21}$$

La formulazione evidenzia la dipendenza da due componenti strutturali:

- **Il coefficiente di variazione quadratico ($C_e^2 = \sigma_e^2 / S_e^2$):** Rappresenta l'effetto della variabilità temporale. Se tutti i componenti avessero tempi rigorosamente costanti, ad esempio $C_e = 0$

come per le code *M/D/1*, il tempo di attesa risulterebbe dimezzato rispetto a una code *M/M/1*, con $C_e = 1$. All'opposto, l'affiancamento di versioni molto sbilanciate, come *Nano* con *Extra-Large*, accresce C_e penalizzando l'attesa in coda.

- **Il fattore iperbolico $\left(\frac{\rho_e}{1-\rho_e}\right)$:** Comporta una crescita asintotica dei tempi non appena ρ_e si avvicina a 1, disincentivando l'algoritmo genetico dal saturare eccessivamente le singole *Road Side Units*.

3.5 Definizione Completa del Problema di Ottimizzazione

Questa sezione raccoglie e formalizza l'insieme completo delle funzioni obiettivo e dei vincoli operativi che governano l'allocazione ottima dei microservizi multi-versione sull'infrastruttura Edge, integrando il modello energetico dei nodi di calcolo.

3.5.1 Modello Energetico dei Nodi

Il consumo energetico di ciascun nodo $e \in \mathcal{E}$ viene descritto attraverso un modello lineare ampiamente consolidato nella letteratura del settore. La potenza assorbita cresce proporzionalmente al fattore di utilizzo computazionale ρ_e, spaziando da una soglia minima P_e^0, la potenza di una macchina accesa ma in uno stato *idle*, fino a un valore massimo P_0^1, nodo a piena saturazione $\rho_e = 1$.

Un nodo disattivato non produce alcun assorbimento elettrico. Lo stato di accensione è tracciato dalla variabile binaria ausiliaria $y_e \in \{0, 1\}$. La potenza istantanea P_e erogata dal nodo e è formulata come:

$$P_e = y_e P_e^0 + \left(P_e^1 - P_e^0\right) \rho_e \tag{3.22}$$

Questa espressione modella esplicitamente l'eterogeneità energetica dell'infrastruttura, associando parametri empirici dedicati alle *Road Side Unit* di classe *Standard*, $P_e^0 = 200W$, $P_e^1 = 400W$, e di classe *Hi-perf*, $P_e^0 = 350W$, $P_e^1 = 1000W$.

Da notare che i parametri del consumo di potenza in fasi *idle* e di picco, (P_e^0 e P_e^1), sono stati inferiti da misurazioni di potenza empiriche eseguite su sistemi Edge paragonabili, in termini di profilo hardware e carico operativo, paragonabile alle RSU dislocate sul territorio.

3.5.2 Formulazione delle Funzioni Obiettivo

Il problema si configura come un'ottimizzazione multi-obiettivo non lineare vincolata, mirata a minimizzare simultaneamente tre metriche tra loro fortemente contrastanti:

- **Minimizzazione dell'errore di qualità globale:** Massimizza l'accuratezza del servizio mediante la minimizzazione del suo complemento a uno:

$$\min f_1(x) = 1 - Acc = 1 - \frac{TP}{TP + FP + FN} \tag{3.23}$$

  dove TP, FP e FN sono i tassi aggregati e ponderati sul traffico definiti nelle **Equazioni 3.4, 3.5** e **3.7**.

- **Minimizzazione del tempo medio di risposta dell'infrastruttura:** Minimizza la latenza complessiva media pesata sul volume di richieste elaborate:

$$\min f_2(x) = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} T_{m,v,e} x_{m,v,e} \tag{3.24}$$

  dove $T_{m,v,e} = \frac{S_m^v}{A_e} + W_e + \delta_{s_m,e}$ include il ritardo di rete, l'attesa in coda *M/G/1* e il tempo effettivo di servizio.

- **Minimizzazione della potenza complessiva assorbita:** Minimizza il fabbisogno elettrico cumulativo dell'intera infrastruttura perimetrale:

$$\min f_3(x) = \sum_{e \in \mathcal{E}} P_e = \sum_{e \in \mathcal{E}} \left[y_e P_e^0 + (P_e^1 - P_e^0) \rho_e \right] \tag{3.25}$$

3.5.3 Vincoli del Modello

Lo spazio delle soluzioni ammissibili è delimitato da tre ordini di vincoli operativi:

- **Vincolo di assegnamento univoco:** Ciascun microservizio $m \in \mathcal{M}$ deve essere istanziato in una e una sola versione $v \in \mathcal{V}_m$ mappato su un unico nodo di elaborazione $e \in \mathcal{E}$:

$$\sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} x_{m,v,e} = 1 \ \forall m \in \mathcal{M} \tag{3.26}$$

- **Vincolo di stabilità delle code e stato dei nodi:** Impedisce il sovraccarico computazionale garantendo che il fattore di utilizzo rimanga strettamente inferiore all'unità mediante un margine cautelativo $\epsilon > 0$. Al contempo, forza lo spegnimento computazionale $\rho_e = 0$ qualora il nodo risulti non alimentato $y_e = 0$:

$$\rho_e \leq y_e(1 - \epsilon) \ \forall e \in \mathcal{E} \tag{3.27}$$

- **Vincolo di SLA:** Impone che il tempo di risposta di ciascuna istanza allocata non ecceda la soglia temporale massima consentita T_m^{SLA}:

$$x_{m,v,e} T_{m,v,e} \leq T_m^{SLA} \ \forall m \in \mathcal{M}, \ \forall v \in \mathcal{V}_m, \ \forall e \in \mathcal{E} \tag{3.28}$$

- **Dominio delle variabili decisionali:** Formalizza la natura binaria delle decisioni di *deployment* e dello stato dei server:

$$x_{m,v,e} \in \{0, 1\} \ \forall m \in \mathcal{M}, \ \forall v \in \mathcal{V}_m, \ \forall e \in \mathcal{E} \tag{3.29}$$

$$y_e \in \{0, 1\} \ \forall e \in \mathcal{E} \tag{3.30}$$

La combinazione della natura combinatoria delle decisioni binarie (x, y) con le non-linearità introdotte dalla formula di Pollaczeck-Khinchine nella componente di latenza f_2 colloca il problema nella classe dei problemi *NP-Hard*. Questa intrinseca complessità giustifica l'adozione delle euristica evolutiva basata su NSGA-II discussa nel capitolo successivo.

4. Algoritmo Genetico e Implementazione

Il problema di allocazione e selezione delle versioni formulato nel **Capitolo 3** combina le scelte di assegnamento discreto con la contemporanea ottimizzazione di tre funzioni obbiettivo in contrasto reciproco. La presenza congiunta di vincoli stringenti di stabilità sulle code computazionali ($\rho_e < 1$) e limiti massimi sul tempo di risposta (T_m^{SLA}) definisce uno spazio di ricerca combinatorio non lineare, la cui risoluzione esaustiva risulta computazionalmente intrattabile su istanze di dimensioni realistiche.

Per esplorare con efficacia tale dominio decisionale e ricavare un'approssimazione accurata del Fronte di Pareto, è stata progettata un'euristica basata su algoritmi evolutivi multi-obiettivo, integrando lo schema elitista proprio di NSGA-II[1]. A differenza degli approcci tradizionali di ottimizzazione vincolata, l'implementazione proposta adotta meccanismi specificamente modellati sulle peculiarità del problema perimetrale.

La prima caratteristica da analizzare è la rappresentazione compatta e vincolata a livello genetico. La struttura del cromosoma incorpora intrinsecamente il soddisfacimento dei vincoli di unicità nell'assegnamento dei microservizi, riducendo le regioni di inammissibilità sintattica all'interno dello spazio di campionamento.

Altro aspetto da considerare sono gli operatori evolutivi coordinati. I meccanismi di ricombinazione e mutazione intervengono preservando la coerenza logica tra collocazione fisica del nodo di calcolo e la variante computazionale selezionata per ogni servizio.

In ultimo, vi è la gestione ibrida dell'ammissibilità fenomenologica. Le violazioni di saturazione locale causate da un eccessivo accentramento del traffico vengono attenuate tramite procedure mirate di riparazione euristica, limitando il ricorso a penalizzazioni numeriche alle sole infrazioni residue dei vincoli.

Nelle sezioni seguenti viene dettagliata l'architettura interna dell'algoritmo genetico realizzato all'interno di un *framework python*. Si descrivono la codifica adottata per il cromosoma, la valutazione formale dei tre vettori di fitness, la configurazione degli operatori di crossover e mutazione

[1] **Non-dominated Sorting Genetic Algorithm II**.

specializzati e, infine, le strategie algoritmiche volte a preservare l'ammissibilità delle soluzioni non dominate.

4.1 Rappresentazione del Cromosoma

Per consentire l'esplorazione dello spazio delle soluzioni mediante algoritmi genetici, è fondamentale definire una codifica cromosomica che sia compatibile con la natura combinatoria del problema e che consenta di manipolare agevolmente le variabili decisionali.

Il modello matematico formulato nella **Sezione 3.5** impone il vincolo di assegnamento univoco (**Equazione 3.26**), secondo cui ogni microservizio $m \in \mathcal{M}$ deve essere istanziato in una sola versione $v \in \mathcal{V}_m$ e mappato su un unico nodo di elaborazione $e \in \mathcal{E}$. Per incorporare nativamente tale restrizione ed evitare generazione di individui privi di senso sintattico, viene adottata una codifica diretta a lunghezza fissa.

4.1.1 Struttura della Codifica a Coppie di Geni

Fissato un ordinamento convenzionale ma invariante dei microservizi $(m_1, m_2, \ldots, m_{|\mathcal{M}|})$, il cromosoma C è strutturato come la concatenazione ordinata di coppie di geni (n_i, v_i), ciascuna associata ad uno specifico microservizio m_i:

$$C = (n_1, v_1, n_2, v_2, \ldots, n_{|\mathcal{M}|}, v_{|\mathcal{M}|}) \tag{4.1}$$

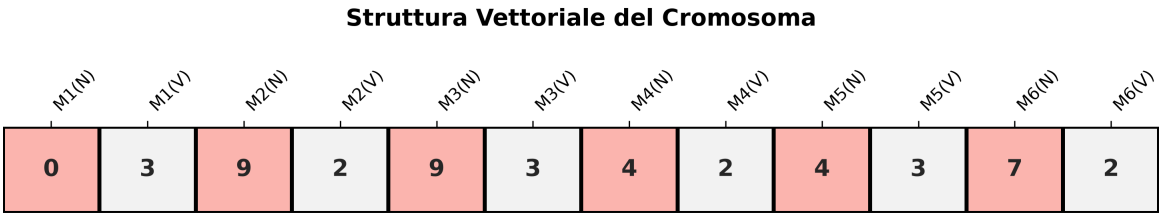


Figura 4.1: Rappresentazione del Cromosoma

Per ciascuna coppia $i \in \{1, \ldots, |\mathcal{M}|\}$, i due geni rappresentano:

- **Il gene di posizionamento** (n_i)**:** Valore intero discreto appartenente all'intervallo $\{1, \ldots, \mathcal{E}\}$, che definisce l'identificativo del nodo *Edge* su cui viene eseguito il microservizio m_i.

- **Il gene di versione** (v_i)**:** Valore intero discreto compreso nell'insieme $\{1, \ldots, |\mathcal{V}_{m_i}|\}$, indicante la taglia del modello neurale selezionata per il microservizio m_i.

Il cromosoma risultante ha pertanto una dimensione complessiva pari a $2 \cdot |\mathcal{M}|$ geni; mentre lo spazio di ricerca esplorabile è dato dal prodotto cartesiano dei domini ammissibili per ciascuna coppia:

$$\prod_{i=1}^{|\mathcal{M}|} \left(\{1, \ldots, |\mathcal{E}|\} \times \{1, \ldots, |V|_{m_i}\} \right) \tag{4.2}$$

Qualsiasi punto campionato all'interno di questo dominio costituisce, per costruzione, un individuo sintatticamente valido.

4.1.2 Decodifica delle Variabili Decisionali del Modello

Dato un individuo C, la transizione dallo spazio genetico allo spazio delle variabili decisionali del problema matematico avviene mediante una procedura di decodifica deterministica a costo computazionale $O(|\mathcal{M}|)$. Specificatamente, sfruttando le seguenti variabili decisionali:

- **Variabili di allocazione dei microservizi** $(x_{m_i,v,e})$**:** L'elemento della matrice tridimensionale binaria assume valore unitario se e solo se la coppia genica corrispondente individua quel nodo e quella versione:

$$x_{m_i,v,e} = \begin{cases} 1 & \text{se } n_i = e \ \wedge \ v_i = v \\ 0 & \text{altrimenti} \end{cases} \tag{4.3}$$

- **Variabili di stato dei nodi *Edge*** (y_e)**:** La variabile binaria indicante l'accensione o lo spegnimento della macchina viene dedotta verificando se al nodo e sia stato assegnato almeno un microservizio:

$$y_e = \begin{cases} 1 & \text{se } \exists i \in \{1, \ldots, |\mathcal{M}|\} : n_i = e \\ 0 & \text{altrimenti} \end{cases} \tag{4.4}$$

Se nessun gene n_i punta al nodo e, quest'ultimo risulta spento, $y_e = 0$, annullando il consumo di potenza *idle* P_e^0 nel calcolo della funzione obiettivo energetica.

4.1.3 Vantaggi Algoritmici della Rappresentazione

La scelta di questa rappresentazione compatta offre notevoli benefici sia analitici che prestazionali:

- **Soddisfacimento implicito del vincolo di assegnamento:** Poiché a ciascun microservizio m_i corrisponde una e una sola coppia genetica (n_i, v_i), il vincolo $\sum_v \sum_e x_{m,v,e} = 1$ è rigorosamente e strutturalmente garantito per qualsiasi individuo generato o ricombinato, senza richiedere operatori di correzione *ad hoc* per la cardinalità dei servizi.

- **Assenza di ridondanze invalide:** A differenza di codifiche binarie dirette della matrice $x_{m,v,e}$, che avrebbero richiesto una stringa di $|\mathcal{M}| \times |\mathcal{V}| \times |\mathcal{E}|$ bit, la rappresentazione a coppie intere abbatte la dimensionalità dello spazio di ricerca, confinandolo unicamente alle configurazioni dove ogni servizio è allocato.

- **Preservazione dell'integrità decisionale:** Come descritto nelle sezioni seguenti, gli operatori di crossover e mutazione agiscono preservando l'associazione logica tra il nodo n_i e la versione v_i, garantendo la consistenza fenomenologica delle soluzioni esplorate.

4.2 Funzioni di Fitness

Una volta convertito il cromosoma C nelle variabili decisionali del problema tramite la procedura di decodifica definita nella sezione precedente, ciascun individuo deve essere valutato rispetto ai tre criteri concorrenti che guidano il processo evolutivo.

La valutazione paretiana richiede la definizione di un vettore di fitness multidimensionale $F(C)$ calcolato a partire dalle metriche statistiche aggregate dei nodi e dei singoli microservizi.

4.2.1 Computazione dei Parametri di Rete e dei Nodi

A partire dal cromosoma decodificato C, vengono calcolate le grandezze prestazionali ed energetiche caratteristiche dell'infrastruttura, quali:

- **Statistiche dei nodi *edge*:** Per ciascun nodo $e \in \mathcal{E}$, si ricavano il tasso di arrivo cumulativo λ_e, il tempo medio di servizio locale S_e, ponderato sui diversi flussi di traffico e scalato dallo *speedup* A_e, la varianza della miscela σ_e^2, il fattore di utilizzo computazionale $\rho_e = \lambda_e S_e$ e il tempo medio di attesa in coda W_e, calcolato mediante la formula di Pollaczek-Khinchine.

- **Tempi di risposta per istanza:** Per ciascun microservizio $m \in \mathcal{M}$, istanziato nella versione v_i e collocato sul nodo n_i, viene calcolato il tempo di risposta come segue:

$$T_{m,v_i,n_i} = \frac{S_m^{v_i}}{A_{n_i}} + W_{n_i} + \delta_{s_m,n_i} \tag{4.5}$$

  In questo modo si calcola un tempo di risposta che combina latenza di rete, attesa in coda e tempo effettivo di inferenza.

- **Accuratezza globale:** Sfruttando i tassi aggregati ponderati $\frac{\lambda_m}{\Lambda}$ definiti nelle **Equazioni 3.4**, **3.6**, **3.5** e **3.7**, si determina l'accuratezza complessiva erogata dall'infrastruttura $Acc(C)$.

4.2.2 Struttura del Vettore di Fitness Multi-Obiettivo

Le grandezze ricavate vengono combinate all'interno del vettore di fitness tridimensionale:

$$F(C) = (f_1(C), f_2(C), f_3(C)) \tag{4.6}$$

I tre componenti del vettore riflettono direttamente le funzioni obiettivo formulate nella **Sezione 3.5**. In particolare:

- **Obiettivo qualitativo** (f_1)**:** Quantifica il completamento a uno dell'accuratezza globale di rilevamento, convertendo la massimizzazione della qualità visiva in un problema di minimizzazione:

$$f_1(C) = 1 - Acc(C) \tag{4.7}$$

- **Obiettivo prestazionale** (f_2)**:** Corrisponde al tempo medio di risposta globale dell'infrastruttura pesato sul volume di traffico complessivo Λ:

$$f_2(C) = \sum_{m \in \mathcal{M}} \sum_{v \in \mathcal{V}_m} \sum_{e \in \mathcal{E}} \frac{\lambda_m}{\Lambda} T_{m,v,e} x_{m,v,e} \tag{4.8}$$

- **Obiettivo energetico** (f_3)**:** Rappresenta la potenza cumulativa istantanea assorbita da tutti i nodi *Road Side Unit* dell'infrastruttura:

$$f_3(C) = \sum_{e \in \mathcal{E}} P_e = \sum_{e \in \mathcal{E}} \left[y_e P_e^0 + \left(P_e^1 - P_e^0 \right) \rho_e \right] \tag{4.9}$$

4.2.3 Proprietà della Formulazione Vettoriale

Poiché tutte e tre le componenti sono formalizzate in termini di minimizzazione, il modello matematico evita il ricorso a tecniche di scalarizzazione a priori, quali combinazioni lineari pesate o metodi a vincolo ϵ.

L'euristica genetica opera direttamente sullo spazio dei vettori non dominati mediante la relazione di dominanza paretiana. Un individuo C_a domina strettamente un individuo C_b $(C_a \prec C_b)$ se e solo se esiste almeno un obiettivo $j \in \{1, 2, 3\}$ per il quale C_a risulta strettamente migliore, ovvero $f_j(C_a) < f_j(C_b)$, e per tutti i restanti obiettivi $k \neq j$ le prestazioni di C_a risultano uguali o migliori rispetto a quelle di C_b, dunque $f_k(C_a) \leq f_k(C_b)$. Questo approccio permette all'algoritmo NSGA-II di esplorare e preservare simultaneamente la totalità delle soluzioni di compromesso tra accuratezza, latenza e consumo energetico.

4.3 Operatori Genetici

Il processo evolutivo è scandito dall'applicazione sequenziale di operatori stocastici progettati per esplorare lo spazio delle soluzioni preservando l'integrità sintattica della codifica cromosomica definita nella **Sezione 4.1**. Per soddisfare la natura accoppiata delle decisioni, nodo ospitante e versioni del modello, gli operatori sono strutturati per manipolare direttamente le coppie di geni (n_i, v_i) associate a ciascun microservizio.

Il flusso evolutivo esegue lo schema iterativo descritto nell'**Algoritmo 1**, articolandosi nelle fasi di inizializzazione, ricombinazione, mutazione e selezione ambientale.

4.3.1 Inizializzazione della Popolazione

La popolazione iniziale P_0, di dimensione N_{pop}, viene generata campionando in modo indipendente e uniforme i valori di ciascuna coppia di geni all'interno dei rispettivi insiemi discreti ammissibili.

In particolare, l'indice del nodo *Edge* n_i viene estratto con distribuzione discreta uniforme nell'insieme $\{1, \ldots, |\mathcal{E}|\}$, allo stesso modo viene estratto l'indice della versione v_i nell'insieme $\{1, \ldots, |V_{m_i}|\}$.

Questo campionamento assicura che ogni individuo iniziale rappresenti una soluzione sintatticamente valida, distribuita uniformemente sull'intero spazio di ricerca.

4.3.2 Operatore di *Crossover*

Per combinare i tratti genetici di due individui genitori preservando la coerenza tra posizionamento e versioni del microservizio, viene impiegato un operatore di crossover a due punti, detto anche *two-point crossover* operante esclusivamente su copie di geni e applicato con probabilità p_c.

Durante la fase di crossover si distinguono due operazioni:

1. Vengono estratti uniformemente due punti di taglio i_1 e i_2 sull'indice dei microservizi tali che $1 \leq i_1 < i_2 \leq |\mathcal{M}|$.

2. La sotto-sequenza contigua di coppie di geni compresa nell'intervallo $[i_1, i_2 - 1]$, denotata con $(n_i, v_i)_i^{i_2-1} = i_1$, viene scambiata tra i due genitori per generare i cromosomi discendenti.

Agire su coppie genetiche complete garantisce che la decisione di allocazione del nodo n_i non venga mai disaccoppiata dalla scelta della relativa versione v_i, assicurando che ciascun discendente mantenga la lunghezza fissa $2 \cdot |\mathcal{M}|$ e rimanga un individuo sintatticamente corretto.

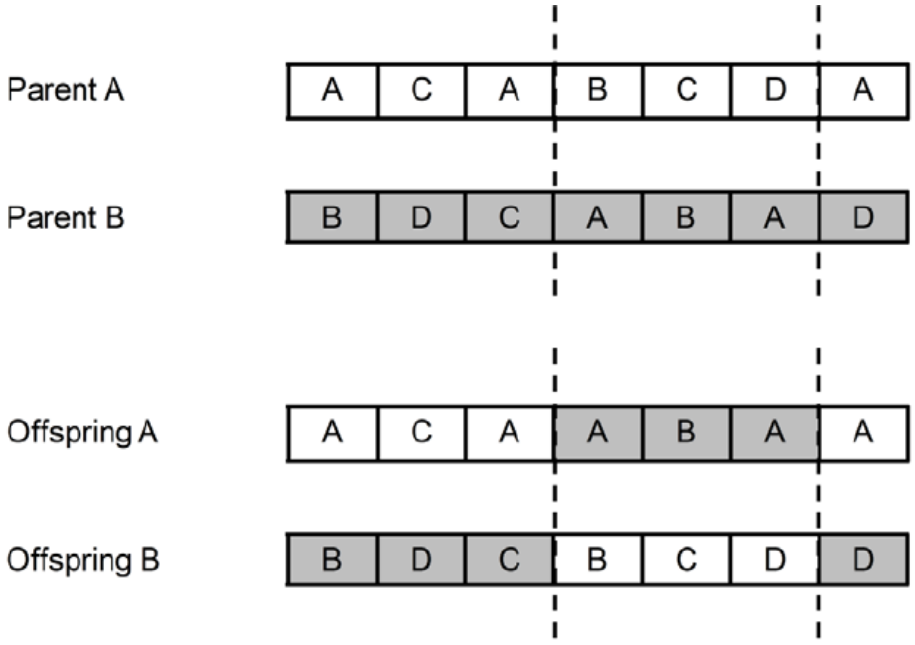


Figura 4.2: Esempio di *Two-Point Crossover*

4.3.3 Operatore di Mutazione

Con una probabilità globale p_m, un individuo viene sottoposto a una mutazione puntuale gene-per-gene volta a introdurre diversità e prevenire la convergenza prematura verso ottimi locali.

Nello specifico, per ciascun microservizio m_i, con probabilità indipendente p_{gene}, l'indice del nodo n_i viene ricampionato uniformemente nell'intervallo $\{1, \ldots, |\mathcal{E}|\}$. Ugualmente, ma in modo indipendente, l'indice della versione v_i viene ricampionato uniformemente in $\{1, \ldots, |\mathcal{V}_{m_i}|\}$ con probabilità p_{gene}.

Poiché ciascun gene viene riassegnato esclusivamente all'interno del proprio specifico intervallo di ammissibilità, l'operatore non può generare valori fuori scala o non validi.

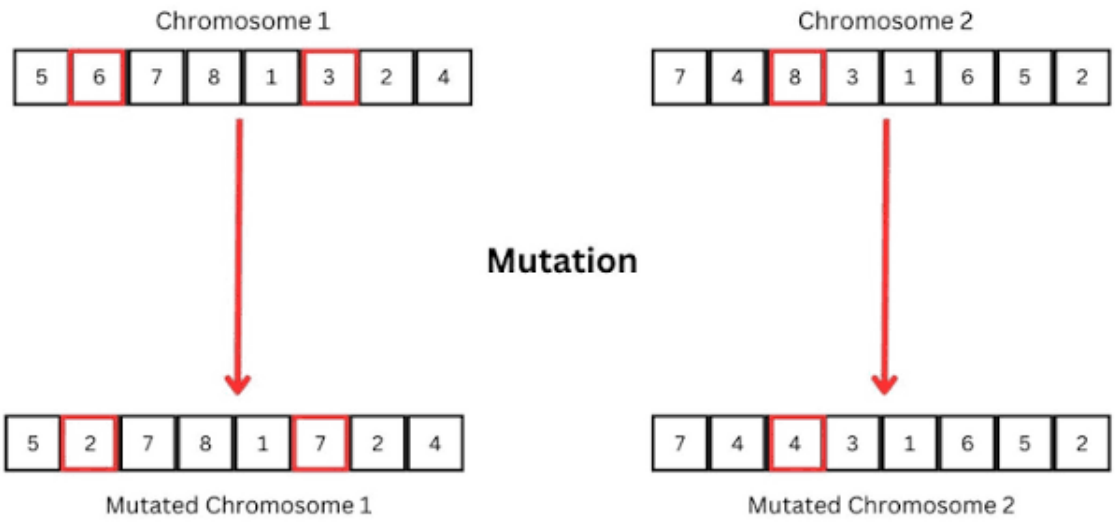


Figura 4.3: Esempio di Mutazione Casuale

4.3.4 Selezione Ambientale

La gestione del passaggio generazionale segue uno schema elitista di tipo $(\mu+\lambda)$, con $\mu = \lambda = N_{pop}$, che prevede le seguenti fasi:

1. A ogni generazione g, gli N_{pop} discendenti prodotti dalle operazioni di crossover e mutazione, denotati con Q_g, vengono aggregati agli N_{pop} genitori della generazione precedente, indicata con P_{g-1}, formando un pool complessivo $R_g = P_{g-1} \cup Q_g$ di cardinalità $2N_{pop}$.

2. L'insieme esteso R_g viene ordinato gerarchicamente mediante la procedura di *non-dominated sorting*, suddividendo e ordinando le soluzioni in fronti paretiani con dominanza decrescente $(F_1, F_2, \dots)$.

3. I migliori individui vengono inseriti nella nuova popolazione P_g fronte dopo fronte fino a raggiungere la cardinalità N_{pop}. Qualora l'ultimo fronte considerato ecceda lo spazio residuo disponibile, le soluzioni appartenenti a quest'ultimo vengono discriminate in base all'operatore di *crowding distance* nello spazio degli obiettivi, privilegiando i punti collocati nelle aree meno dense del fronte per massimizzare la dispersione e l'uniformità della copertura.

Per concludere, l'evoluzione si arresta al raggiungimento di un numero prefissato di generazioni massime G_{max}. Il fronte di Pareto finale restituito corrisponde all'insieme delle soluzioni

non dominate raccolte durante l'intera esecuzione attraverso un archivio globale costantemente aggiornato.

Algorithm 1 Algoritmo Genetico Multi-Obiettivo per *Placement* Multi-Versione

$P_0 \leftarrow$ InitPopulation(N_{pop})
for all $C \in P_0$ **do**
    Repair(C); Evaluate(C)
end for
for $g = 1$ **to** G_{max} **do**
    $Q_g \leftarrow$ Crossover(P_{g-1}, p_c)
    $Q_g \leftarrow$ Mutate(Q_g, p_m)
    for all $C \in Q_g$ **do**
        Repair(C); Evaluate(C)
    end for
    $R_g \leftarrow P_{g-1} \cup Q_g$
    $P_g \leftarrow$ EnvironmentalSelection(R_g, N_{pop})
end for
return non-dominated solutions of $\bigcup_g P_g$

4.4 Gestione dell'Ammissibilità

L'inizializzazione casuale della popolazione e l'azione combinata degli operatori stocastici di crossover e mutazione possono generare individui che non soddisfano l'insieme dei vincoli operativi formulati nella **Sezione 3.5**. Nello specifico, le violazioni possono manifestarsi secondo due modalità distinte:

- **Saturazione computazionale del nodo:** Eccessiva concentrazione di carico su una singola *Road Side Unit*, tale da violare il vincolo di stabilità delle code, definita nell'**Equazione 3.20**.
- **Violazione del vincolo temporale:** Superamento del tempo massimo di risposta ammissibile per un microservizio, fare riferimento all'**Equazione 3.28**.

Per gestire tali condizioni senza compromettere la convergenza dell'algoritmo genetico, viene adottata una strategia ibrida composta da una fase euristica di riparazione del sovraccarico [10], affiancata da un meccanismo di penalizzazione della funzione obiettivo prestazionale per l'ammissibilità residua.

4.4.1 Procedura di Riparazione del Sovraccarico

Prima della fase di valutazione della fitness, ciascun cromosoma C viene sottoposto a una procedura di riparazione deterministica di tipo *greedy* mirata a ricondurre i fattori di utilizzo dei nodi al di sotto della soglia critica.

L'algoritmo di riparazione itera finché sussiste almeno un nodo sovraccarico ($\rho_e \geq 1$) o fino al raggiungimento di un numero massimo di tentativi pari a $|\mathcal{M}|$. Nello specifico, si susseguono le prossime operazioni:

1. Viene individuato il nodo *Edge* e_{over} caratterizzato dal valore massimo di utilizzo $\rho_e \geq 1$.
2. All'interno del sottoinsieme di microservizi mappati su e_{over}, viene selezionato il servizio che contribuisce con la frazione di carico computazionalmente maggiore.
3. Tale microservizio viene riallocato sul nodo *Edge* che in quel momento presenta il minimo fattore di carico complessivo tra tutte le macchine accese.
4. I tassi aggregati λ_e, i tempi medi S_e e i relativi fattori di utilizzo ρ_e vengono aggiornati per tutti i nodi coinvolti.

La procedura si arresta non appena tutti i nodi soddisfano la condizione $\rho_e < 1$, oppure nel momento in cui ulteriori riassegnazioni non producono una riduzione del picco massimo di utilizzo. Poiché questo meccanismo è integrato direttamente all'interno della fase di valutazione, come si vede dall'**Algoritmo 1**, esso risulta trasparente agli operatori di crossover e mutazione, trasformando la quasi totalità degli individui generati in soluzioni ammissibili prima del calcolo effettivo delle metriche.

4.4.2 Gestione dell'Inammissibilità Residua e Vincoli di SLA

In specifiche circostanze, la sola euristica di riparazione non è in grado di ripristinare la piena ammissibilità della soluzione. Questo fenomeno si verifica tipicamente quando il carico aggregato totale Λ supera la capacità di calcolo complessiva dell'infrastruttura, oppure qualora un microservizio violi la soglia T_m^{SLA} pur essendo allocato su un nodo non saturo a causa dell'accumulo di ritardi di trasmissione di rete a lungo raggio (δ_{near}) e dell'impiego di varianti YOLO computazionalmente onerose.

Per questi casi residui, viene introdotta una funzione indicatrice binaria di ammissibilità $\Phi(C) \in \{0, 1\}$:

$$\Phi(C) = \begin{cases} 1 & \text{se } \exists e \in \mathcal{E} : \rho_e \geq 1 \vee \exists\, (m, v, e) : x_{m,v,e} T_{m,v,e} > T_m^{SLA} \\ 0 & \text{altrimenti} \end{cases} \tag{4.10}$$

La gestione delle soluzioni non ammissibili residue avviene modificando la componente prestazionale del vettore di fitness (f_2) mediante l'aggiunta di un termine di penalità costante $\|$. In questo modo si ottiene la seguente equazione:

$$\hat{f_2}(C) = f_2(C) + k \cdot \Phi(C) \tag{4.11}$$

Il parametro $\|$ viene calibrato su un valore sufficientemente elevato in modo tale che qualsiasi individuo non ammissibile risulti rigorosamente dominato, rispetto a f_2, da qualunque soluzione che rispetti tutti i vincoli operativi.

Questo schema garantisce un duplice vantaggio:

- Nella procedura di *non-dominated sorting*, le soluzioni ammissibili ottengono sistematicamente un rango di Pareto superiore rispetto a quelle non ammissibili, impedendo che configurazioni invalide confluiscano nel fronte finale.
- Gli individui marginalmente inammissibili non vengono scartati a priori dalla popolazione, consentendo al materiale genetico potenzialmente favorevole presente negli altri geni di sopravvivere e ricombinarsi nelle generazioni successive, preservando la diversità genetica dell'intero pool evolutivo.

5. Campagna Sperimentale e Analisi dei Risultati

In questo capitolo viene presentata la validazione empirica del modello formale e dell'euristica evolutiva proposti nei capitoli precedenti. L'obiettivo primario della campagna sperimentale è analizzare approfonditamente la capacità dell'algoritmo genetico di analizzare lo spazio dei compromessi tra latenza, consumo energetico e accuratezza dell'output in contesti Edge/Fog realistici.

L'indagine sperimentale è articolata in una serie di scenari incrementali volti a testare la robustezza e la sensibilità del sistema:

- **Caratterizzazione dello scenario reale e profilazione dei microservizi:** Definizione dell'infrastruttura di test basata sulla piattaforma di *Smart Mobility Modena Automotive Smart* Area[1] [16]. In questa fase vengono modellate la distribuzione territoriale dei nodi RSU, le matrici dei ritardi di rete, oltre a essere condotta la profilazione empirica delle prestazioni e dell'accuratezza per diverse varianti YOLO su dataset standard.

- **Configurazione e robustezza del motore evolutivo:** Analisi di sensitività per la scelta dei parametri operativi dell'algoritmo genetico, quali la dimensione della popolazione, probabilità di crossover e mutazione.

- **Esplorazione del Fronte Pareto nominale:** Valutazione della struttura e delle interdipendenze tra le tre funzioni obbiettivo in condizioni operative standard (carico stazionario e uniforme su tutta la rete), analizzando i *trade-off* emergenti tra qualità del servizio e overhead dell'infrastruttura.

- **Impatto della composizione eterogenea dei nodi di calcolo:** Studio sistematico di 11 diverse miscele hardware di RSU, partendo da una rete interamente Standard a una completamente Hi-Perf, al fine di quantificare i benefici prestazionali dell'accelerazione hardware e identificare la soglia di ottimo bilanciamento costo-efficienza.

- **Comportamento del sistema sotto carichi asimmetrici e validazione statistica:** Valutazione della resilienza dell'infrastruttura di fronte a picchi di domanda disomogenei, quantificando

[1] **MASA**.

il degrado dell'Ipervolume, la frammentazione del Fronte, il sovracosto energetico statico di base e la varianza.

5.1 Set-up Applicativo di Riferimento

La validazione empirica del modello formale e dell'euristica genetica proposta è stata condotta all'interno di uno scenario realistico di *Smart Mobility* basato sull'infrastruttura del *Modena Automotive Smart Area*. Il caso d'uso analizzato modella una rete urbana pervasiva di sensori e *Road Side Unit*,[2] fondamentali per la comunicazione V2X, dislocate sul territorio cittadino, adibite al monitoraggio in tempo reale del traffico veicolare mediante compiti di computer vision e *object detection* sfruttando una rete neurale *Ultralytics YOLO26*. La dislocazione geografica delle *RSU* nella città è presentata in **Figura 5.1**.

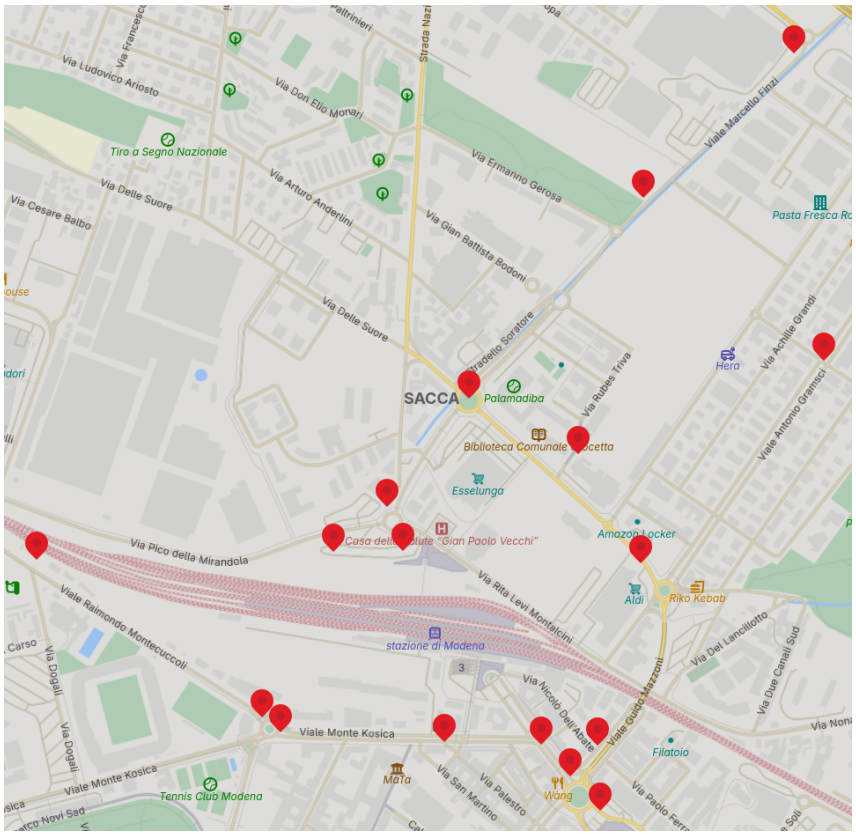


Figura 5.1: Posizione delle RSU

[2] **RSU**.

5.1.1 Modello del Carico Nominale

Nello scenario applicativo di riferimento, condizione nominale stazionaria, le richieste di elaborazione generate dalle telecamere seguono un processo di Poisson con tasso di pari a $\lambda_m = 10.0$ req/s per ciascuna sorgente. Questa frequenza rispecchia il vincolo applicativo stringente tipico dei sistemi di assistenza alla guida e guida autonoma. Tuttavia, questa condizione impone un tempo massimo per l'anello di retroazione e controllo non superiore a 100 ms.

5.1.2 Infrastruttura di Rete e Nodi di Calcolo RSU

L'architettura Edge/Fog considerata sfrutta la collocazione geografica reale delle postazioni RSU abilitate all'interno del *living lab* **MASA**. Per modellare l'eterogeneità computazionale ed energetica tipica delle infrastrutture perimetrali, i nodi di elaborazione sono suddivisi in due classi distinte: nodi *Standard* e nodi ad alte prestazioni, o *Hi-Perf*.

I nodi sono caratterizzati dai seguenti parametri operativi, analizzati con maggiore rigore nella **Tabella 5.1** ed esplorati nel dettaglio nella **Sottosezione 3.1.2**:

- **Potenza statica** (P_e^0): Potenza elettrica di base a macchina accesa ma in stato *idle*.
- **Potenza a pieno carico** (P_e^1): Consumo massimo assorbito in condizioni di saturazione della capacità di calcolo.
- **Fattore di accelerazione** (A_e): Moltiplicatore di *speedup* computazionale che riduce proporzionalmente il tempo di esecuzione dei microservizi ospitati sul nodo.

Tipologie di RSU	P_e^0 [W]	P_e^1 [W]	A_e
Standard	200	400	1.0
Hi-perf	350	1000	1.5

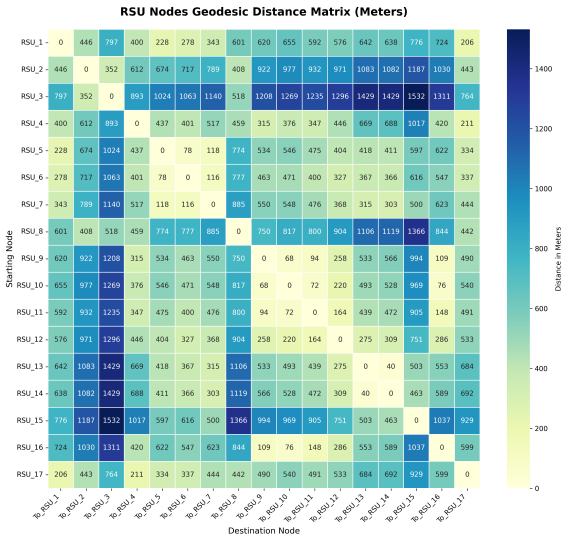
Tabella 5.1: Tipologie di Nodi Edge

Come indicato nella **Sottosezione 3.3.1**, per le interconnessioni di rete tra le sorgenti dati, *smart camera*, e i nodi di elaborazione RSU, è stata adottata una matrice di comunicazione *wireless* a

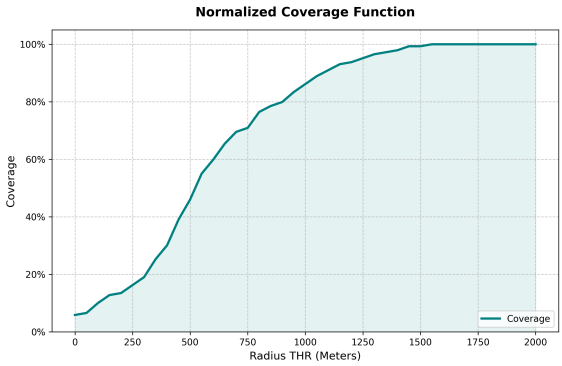
lungo raggio con ritardi binomiali: δ_{near} e δ_{far}. Sulla base del modello teorico di riferimento, metà delle trasmissioni dovrebbero svolgersi con un regime di ritardo basso (δ_{near}), mentre l'altra metà con un ritardo superiore (δ_{far}).

Tuttavia, ci si è scontrati con un limite teorico importante ovvero la mancanza di dati sufficienti per identificare una soglia che permettesse di discriminare una connessione con ritardo δ_{near} o δ_{far}.

Per superare tale limite, è stato necessario attingere ai dati sulle posizioni delle RSU per creare una matrice delle distanze, mostrata in **Figura 5.2a**. Con i risultati ottenuti dalla matrice è stata calcolata la distribuzione cumulativa delle distanze, visibile in **Figura 5.2b**.



(a) Matrice delle Distanze delle RSU



(b) Distribuzione Cumulativa delle Distanze delle RSU

Figura 5.2: Distribuzione delle Distanze tra RSU

Grazie ai valori ottenuti dalla distribuzione cumulativa, è stato possibile osservare un intervallo di distanze da esplorare $[500, 600]$.

Tracciando quindi dei grafici per osservare la variazione del raggio di copertura rispetto al numero di nodi RSU attivi, è stato possibile identificare il valore di soglia pari a 530 metri, come si evince dalla **Figura 5.3**.

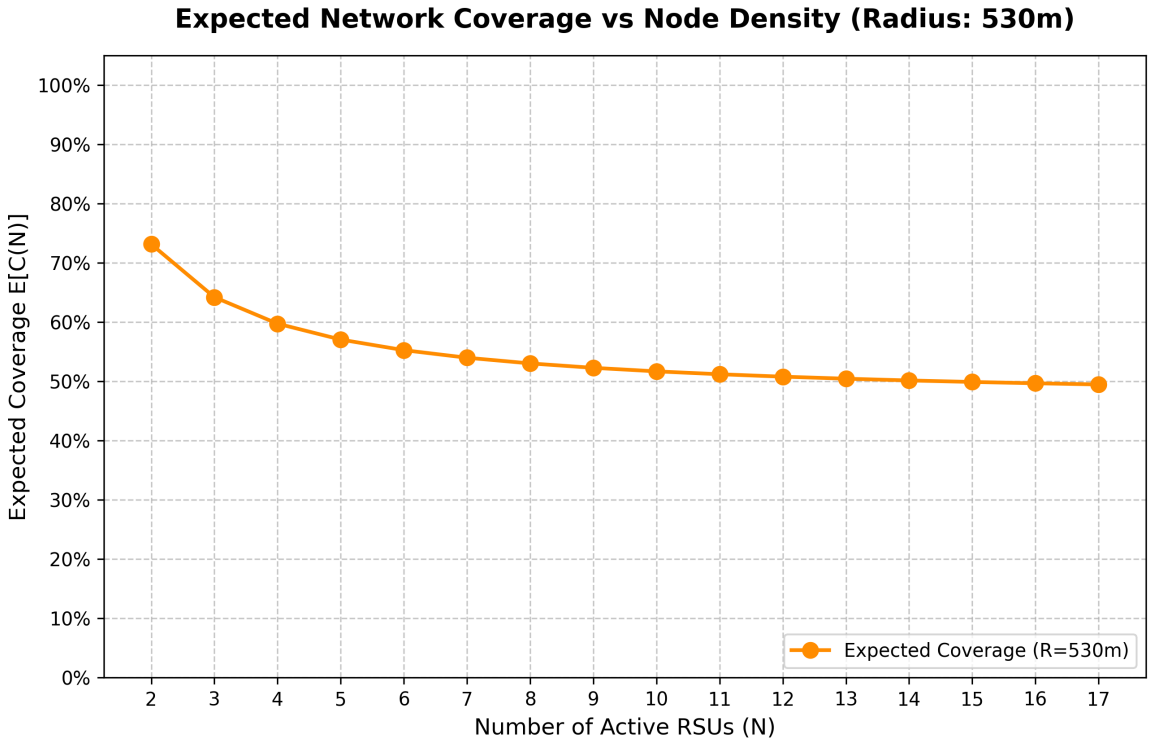


Figura 5.3: Rapporto Raggio vs. Copertura

Tale risultato è stato fondamentale per rendere lo scenario di riferimento quanto più realistico possibile.

5.1.3 Caratterizzazione dei Microservizi Multi-Versione

Ciascun flusso video catturato dalle telecamere richiede l'esecuzione di un microservizio di *object detection* per l'identificazione e la localizzazione dei veicoli. Per abilitare il paradigma multi-versione, sono state considerate le 5 varianti della rete neurale ***Ultralytics* YOLO** presentate nella **Sottosezione 2.2.2**.

I parametri statistici di tempo di servizio medio (S_m^v), deviazione standard (σ_m^v) e i tassi di classificazione ($TP_m^v, FP_m^v, FN_m^v, TN_m^v$) sono stati ricavati profilando sperimentalmente ciascuna variante sul dataset di benchmark standard **Microsoft COCO** [11, 2]. Per isolare le metriche prestazionali a regime dai transienti di inizializzazione della GPU/NPU, i dati relativi alle prime inferenze sono stati scartati tramite una fase di *warm-up*.

Parametri	Versioni				
	N	**S**	**M**	**L**	**XL**
TP_m^v	0.41	0.49	0.53	0.54	0.55
FP_m^v	0.46	0.34	0.29	0.28	0.26
FN_m^v	0.13	0.17	0.19	0.18	0.19
TN_m^v	0.00	0.00	0.00	0.00	0.00
S_m^v [ms]	12.81	14.99	24.76	30.29	51.58
σ_m^v [ms]	1.63	1.78	2.67	3.24	5.07

Tabella 5.2: Parametri dei Microservizi

Nel contesto dell'*object detection* con classi di background esplicite, gli oggetti non rilevati vengono contabilizzati all'interno dei veri positivi della classe di sfondo. Di conseguenza, la frazione TN_n^v assume convenzionalmente valore nullo.

Sulla base dei risultati ottenuti dalla **Tabella 5.2**, è stato possibile osservare la stretta relazione inversa tra l'accuratezza e il tempo di risposta delle versioni del modello YOLO. In particolare, osservando la **Figura 5.4**, si nota la tipica forma iperbolica che caratterizza relazioni di questo tipo.

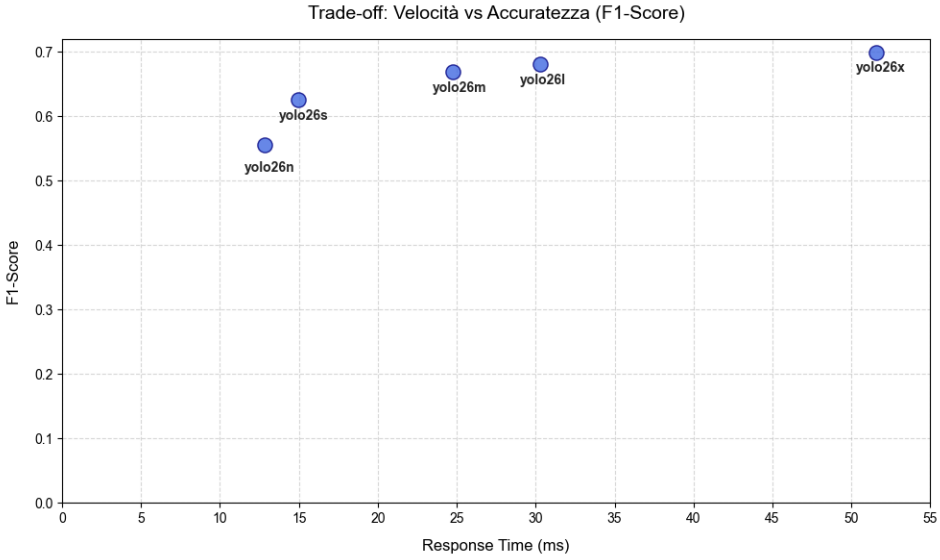


Figura 5.4: F1-Score vs. Latenza

5.2 Calibrazione dei Parametri dell'Algoritmo Genetico

L'efficacia degli algoritmi evolutivi nel risolvere questi problemi di ottimizzazione multi-obiettivo complessi dipende in larga misura dalla corretta sintonizzazione dei loro iperparametri operativi.

Per determinare la configurazione in grado di garantire una convergenza rapida e una copertura uniforme dello spazio di ricerca, è stata condotta una fase preliminare di calibrazione, detta anche *parameter tuning*, impiegando un *framework* di simulazione sviluppato in *Python* ed esteso a partire dalla libreria standard **DEAP** [3].

5.2.1 Spazio di Esplorazione dei Parametri

La fase di taratura ha esplorato sistematicamente le combinazioni dei parametri fondamentali del motore genetico:

- **Dimensione della popolazione (N_{pop}):** Valutata in un intervallo compreso tra 50 e 250 individui. Da questa analisi è emerso che: popolazioni troppo ridotte tendono a soffrire di convergenza prematura verso ottimi locali, mentre popolazioni eccessivamente ampie comportano un onere computazionale elevato senza apportare miglioramenti proporzionali nella qualità del fronte.

- **Probabilità di *crossover* (P_c):** Analizzata nello spazio 0.1 e 0.9. Regola la frequenza con cui le coppie di geni (n_i, v_i) vengono ricombinate tra individui genitori per generare nuove soluzioni.

- **Probabilità di mutazione (P_m):** Esplorata nell'intervallo $0.1 \div 0.9$. Definisce la frequenza con cui i cromosomi subiscono modifiche modifiche puntuali nell'assegnazione dei nodi o delle versioni, favorendo l'esplorazione di regioni non ancora campionate.

5.2.2 Criteri di Valutazione e Configurazione Ottima

Per identificare l'assetto migliore, ogni combinazione di parametri è stata valutata sulla base di metriche quantitative primarie relative al fronte non-dominato ottenuto. Tali indicatori sono:

[3] *Distributed Evolutionary Algorithms in Python*.

- **Ipervolume (HV):** Misura il volume dello spazio dominato dal fronte Pareto rispetto ad un punto di riferimento convenzionale. Un valore elevato dell'ipervolume attesta capacità dell'algoritmo di convergere verso soluzioni che minimizzano contemporaneamente la latenza e i consumi, massimizzando al contempo l'accuratezza del servizio.

- ***Spacing*/Uniformità:** Valuta la dispersione geometrica e l'omogeneità dei punti lungo il fronte identificato. Un valore contenuto di *spacing* indica una distribuzione regolare delle alternative decisionali lungo l'intero spettro dei compromessi, evitando lacune o addensamenti localizzati.

Dai test preliminari, il compromesso ottimale tra qualità della convergenza, omogeneità del fronte e tempo di calcolo è stato identificato nella configurazione con: dimensione della popolazione pari a 200, probabilità di crossover fissata a 0.3 e probabilità di mutazione pari a 0.2.

Sebbene l'euristica sia stata progettata sullo schema elitista di NSGA-II, in fase di calibrazione è stata condotta anche una validazione comparativa con NSGA-III. Il test ha mostrato come il secondo approccio abbia prestazioni superiori in termini di ipervolume [6, 8, 14], risultando pertanto l'algoritmo di selezione adottato nella configurazione finale.

Parametro dell'Algoritmo Genetico	Range Esplorativo	Valore Ottimo Selezionato
Algoritmo di Selezione	-	NSGA-III
Dimensione della Popolazione (N_{pop})	$50 \div 250$	200
Numero di Generazioni (G_{max})	-	300
Probabilità di Crossover (P_c)	$0.1 \div 0.9$	0.3
Probabilità di Mutazione P_m	$0.1 \div 0.9$	0.2
Schema Evolutivo	-	$(\mu + \lambda)$/elitista

Tabella 5.3: Tabella Riassuntiva dei Parametri del Problema

5.2.3 Protocollo di Validazione dei Risultati

Dato che gli algoritmi evolutivi integrano operatori intrinsecamente probabilistici (come la generazione pseudo-casuale della popolazione iniziale $t = 0$, selezione dei genitori taglio del crossover

e attrazione dei geni), i risultati di una singola esecuzione possono dipendere da fluttuazioni stocastiche favorevoli o sfavorevoli.

Per isolare il comportamento oggettivo dell'algoritmo della casualità delle estrazioni e garantire la piena significatività stocastica dei dati, è stato adottato un protocollo di validazione rigoroso basato su *run* indipendenti. Nello specifico, ciascun esperimento è stato ripetuto per 30 volte complete dall'inizio alla fine. Questo valore soddisfa le assunzioni del Teorema del Limite Centrale, consentendo di descrivere le prestazioni del sistema tramite medie e deviazioni standard affidabili

5.3 Analisi del Fronte Pareto sotto Carico Uniforme

In questa sezione viene analizzato il Fronte di Pareto identificato dall'algoritmo evolutivo nello scenario di riferimento nominale, caratterizzato da un'infrastruttura con il 40% di nodi *Hi-perf* e il 60% di nodi *Standard* soggetto ad un carico omogeneo, $\lambda_m = 10.0$ req/s. L'obiettivo primario è caratterizzare la geometria dello spazio dei compromessi generato dalle tre funzioni concorrenti: la minimizzazione del tempo di risposta, la minimizzazione della potenza complessiva assorbita e la massimizzazione dell'accuratezza del servizio (espressa analiticamente come la minimizzazione di $1 - Acc$.

5.3.1 Struttura dello Spazio di Ricerca

La rappresentazione tridimensionale del Fronte di Pareto evidenzia una superficie continua e convessa che racchiude l'insieme delle soluzioni non dominate individuate lungo le generazioni di calcolo.

La conformazione della superficie riflette il conflitto intrinseco tra le metriche di prestazione e qualità. Il raggiungimento di livelli elevati di accuratezza, ovvero valori minimi di $1 - Acc$, impone l'adozione prevalente di modelli computazionalmente complessi come *YOLO Large* o *Extra-Large*. Questo comporta un incremento diretto dei tempi di calcolo locale (S_m^v), traducendosi in una maggiore latenza di risposta e nella necessità di mantenere attivi più nodi Edge con consumi elettrici (P_e) elevati.

Ai vertici estremi del fronte, visibile in **Figura 5.5**, si osservano due regimi operativi opposti:

- **Regime *ultra-low power*/Bassa latenza:** Configurato tramite l'assegnazione quasi esclusiva di varianti YOLO *Nano* e *Small* accoppiate al consolidamento del carico su un numero minimo di nodi. Si raggiungono tempi di risposta inferiori a 50 ms e potenze totali che non superano 2000 W. Tuttavia, tale risultato è ottenuto a scapito di un'accuratezza complessiva sensibilmente ridotta ($1 - Acc > 0.53$).

- **Regime ad alta accuratezza:** Configurato impiegando i modelli YOLO *Extra-Large* distribuiti su tutti i nodi disponibili per prevenire la saturazione delle code ($\rho_e \geq 1$). L'infrastruttura assorbe potenze prossime a 7000 W con tempi di risposta che superano i 200 ms. D'altra parte, questa configurazione garantisce un tasso di errore di classificazione minimo ($1-Acc \approx 0.45$).

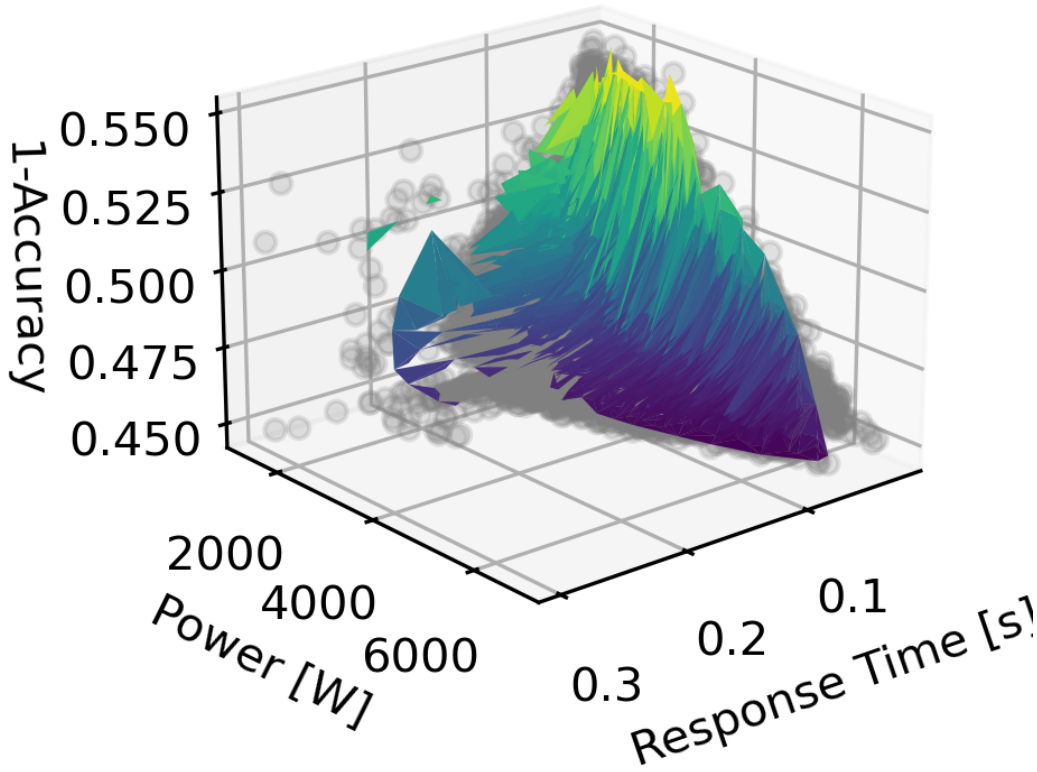


Figura 5.5: Fronte di Pareto Tridimensionale

5.3.2 Analisi delle Proiezioni Bidimensionali

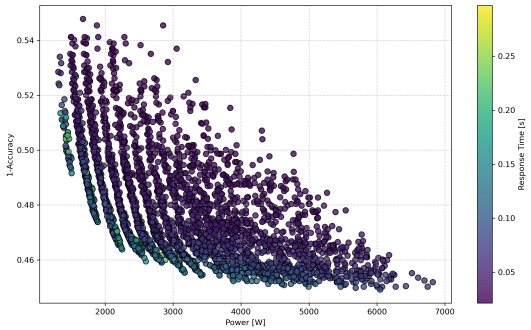
Per comprendere a fondo le correlazioni a coppie tra gli obiettivi, i punti non dominati vengono proiettati su piani bidimensionali codificando la terza metrica mediante una scala cromatica a gradiente. Grazie a tale strategia è possibile analizzare i seguenti compromessi:

- ***Trade-off* Potenza vs. Accuratezza (*Encoding* del Tempo di Risposta):** Proiettando il consumo di potenza sull'asse delle ascisse e l'errore di accuratezza $(1 - Acc)$ sulle ordinate, i punti mostrano una dispersione decrescente, come mostrato in **Figura 5.6a**. All'aumentare dell'accuratezza, la curva si sposta progressivamente verso i livelli di potenza via via più alti. La codifica cromatica mostra come le soluzioni a tempo di risposta rapido (colorazioni chiare) siano confinate verso consumi energetici intermedi ed elevati. Poiché per abbattere contemporaneamente il tempo di attesa in coda W_e ed elaborare modelli pesanti è indispensabile accendere più nodi contemporaneamente e sfruttare i nodi *Hi-perf*, man mano che il tempo di risposta si contrae, il margine per contenere la potenza si assottiglia drasticamente.

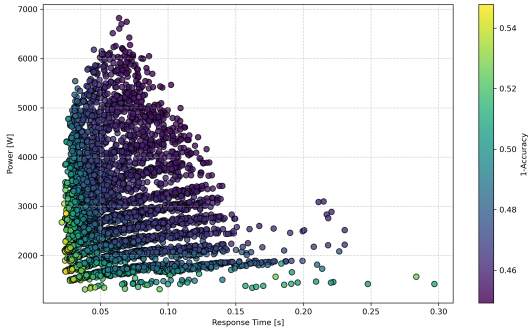
- ***Trade-off* Tempo di Risposta vs. Potenza (*Encoding* dell'Accuratezza):** Nel piano che pone in relazione la latenza di servizio con la potenza assorbita, emerge un classico profilo paretiano iperbolico, visibile in **Figura 5.6b**. La riduzione del tempo di risposta al di sotto di 100 ms determina una ripida ascesa della potenza complessiva. Tale impennata è causata dalla non linearità della formula di Pollaczek-Khinchine per garantire code snelle sotto tassi di arrivo sostenuti. Il fattore di utilizzazione ρ_e deve mantenersi basso, costringendo l'algoritmo genetico a disperdere i microservizi su più nodi e pagando la potenza statica P_e^0 di ciascun server attivato.

- ***Trade-off* Tempo di Risposta vs. Accuratezza (*Encoding* della Potenza):** La proiezione in **Figura 5.6c** evidenzia la relazione diretta tra dimensione del modello neurale e tempi di attraversamento. L'adozione di versioni avanzate di YOLO determina un trascinamento verso destra dei tempi di risposta complessivi a causa sia dell'aumento S_m^v sia dell'incremento della varianza di servizio σ_m^v, che appesantisce la coda *M/G/1*. I punti caratterizzati da consumi di energetici minimi si collocano invariabilmente nelle regioni a tempi di risposta intermedi

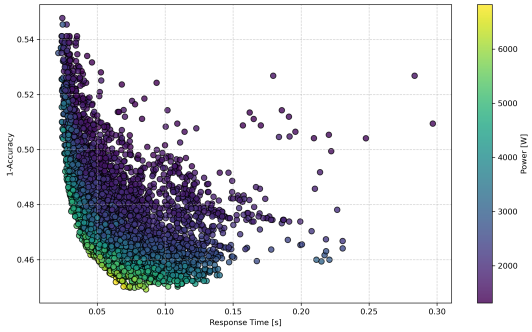
e bassa accuratezza, confermando come il risparmio energetico e l'alta qualità del modello siano due requisiti fortemente contrastanti.



(a) Potenza/Accuratezza



(b) Tempo di Risposta/Potenza



(c) Tempo di Risposta/Accuratezza

Figura 5.6: Rappresentazione Bidimensionale del Fronte di Parteo

5.3.3 Rilevanza per la Pianificazione dell'Infrastruttura

L'esame analitico del fronte Pareto mostra come non esista un'unica soluzione ottimale a priori, bensì una famiglia articolata di allocazioni paretiane tra cui un operatore può selezionare la configurazione target in base ai requisiti contingenti.

In situazioni di traffico ordinario a bassa densità, il sistema può operare nella regione a basso consumo energetico attivando unicamente i nodi strettamente necessari con modelli intermedi.

D'altra parte, in presenza di condizioni critiche, la transizione verso punti a elevata accuratezza del fronte garantisce la massima affidabilità di rilevamento veicolare a fronte di un budget energetico opportunamente dimensionato.

5.4 Analisi del Fronte di Pareto in Scenari Eterogenei

In questa sezione, si estende l'analisi oltre la configurazione nominale di riferimento, introducendo due dimensioni fondamentali di variabilità sperimentale: la composizione eterogenea dell'infrastruttura computazionale e la disomogeneità spaziale della domanda di traffico. L'obiettivo è comprendere come l'accoppiamento tra diverse disponibilità di calcolo locale e differenti profili di carico influenzi la struttura dei *trade-off* paretiani, la fattibilità dei vincoli operativi e la convergenza dell'algoritmo genetico.

5.4.1 Dimensionamento dell'Eterogeneità Hardware

Nel *setup* nominale, la percentuale di nodi *Hi-perf* è mantenuta fissa al 40%. Per determinare l'impatto architetturale dell'accelerazione hardware sui tre obbiettivi (latenza, potenza, accuratezza), viene definita una progressione sistematica su 11 configurazioni di rete, variando il rapporto tra nodi *Hi-perf* e nodi Standard con un passo del 10% come segue:

$$Ratio \in \{(1.0, 0.0), (0.9, 0.1), (0.8, 0.2), \ldots, (0.1, 0.9), (0.0, 1.0)\}$$

Tale scansione spazia da un'infrastruttura omogenea basata su macchine ad elevate prestazioni (100% Hi-perf, $A_e = 1.5$) a una rete interamente costituita da macchine Standard (100% Standard, $A_e = 1.0$).

L'indagine di queste configurazioni mira a chiarire specifici comportamenti del sistema:

- **Flessibilità di allocazione e spazio di ricerca:** All'aumentare della quota di nodi *Hi-perf*, la riduzione dei tempi di servizio medi locali (S_m^v / A_e) allenta il vincolo di stabilità sulle code computazionali ($\rho_e < 1$). Ciò amplia lo spazio delle soluzioni ammissibili verso combinazioni a più elevata accuratezza, come modelli YOLO *Large* ed *Extra-Large*, senza incorrere in penalizzazioni di latenza o violazioni dei requisiti di tempo massimo (T_m^{SLA}).

- **Dinamica dei consumi energetici:** I nodi *Hi-perf* presentano una potenza di *idle*, $P^0 = 350W$, e un assorbimento a saturazione, $P^1 = 1000W$, sensibilmente maggiori rispetto ai nodi *Standard* ($200W$ e $400W$). L'introduzione di percentuali crescenti di nodi potenziati

impone quindi una selezione più accurata dei nodi da mantenere accesi ($y_e = 1$), verificando se il maggior *throughput* locale giustifichi il sovracosto energetico statico e dinamico.

5.4.2 Modellazione del Carico Eterogeneo

Mentre la condizione nominale assume un tasso di arrivo stazionario e omogeneo su tutte le telecamere ($\lambda_m = 10$ req/s), gli scenari reali di mobilità urbano presentano spesso forti disomogeneità territoriali: arterie a scorrimento veloce o incroci semaforici centrali registrano volumi di fotogrammi molto più elevati rispetto a sensori periferici.

Per simulare questo fenomeno garantendo la piena comparabilità con il caso uniforme, è stato implementato il metodo di partizionamento *Random Broken Stick*:

- **Conservazione del carico totale:** Il carico complessivo generato dall'intera rete viene mantenuto costante seguendo la formula:

$$\Lambda = \sum_{m \in \mathcal{M}} \lambda_m = |\mathcal{M}| \cdot \overline{\lambda} = |\mathcal{M}| \cdot 10 \; req/s \tag{5.1}$$

- **Meccanismo di suddivisione:** L'intervallo continuo $[0, \Lambda]$ viene suddiviso casualmente mediante l'estrazione di $(|\mathcal{M}| - 1)$ punti di taglio uniformemente distribuiti. Le lunghezze dei segmenti risultanti definiscono i singoli tassi di arrivo λ_m per ciascuna sorgente.

Questo approccio genera distribuzioni asimmetriche senza alterare la quantità di traffico complessiva che l'infrastruttura Edge deve smaltire. Nello specifico, alcune sorgenti gestiscono tassi di arrivo molto elevati $\lambda_m >> 10 \; req/s$ e altre tassi quasi trascurabili $\lambda_m \to 0$.

5.4.3 Confronto Comparativo e Validazione *Multi-Seed*

Per valutare quantitativamente l'interazione tra eterogeneità infrastrutturale e la disomogeneità della domanda del traffico, i risultati ottenuti sui due regimi di carico sono stati posti a confronto diretto lungo le 11 configurazioni di rete. Per lo scenario a carico variabile, i dati sintetizzano l'aggregazione statistica di 25 esecuzioni pseudo-casuali indipendenti, riportando la media campionaria (μ) e la fascia d'incertezza corrispondente ad una deviazione standard ($\pm 1\sigma$).

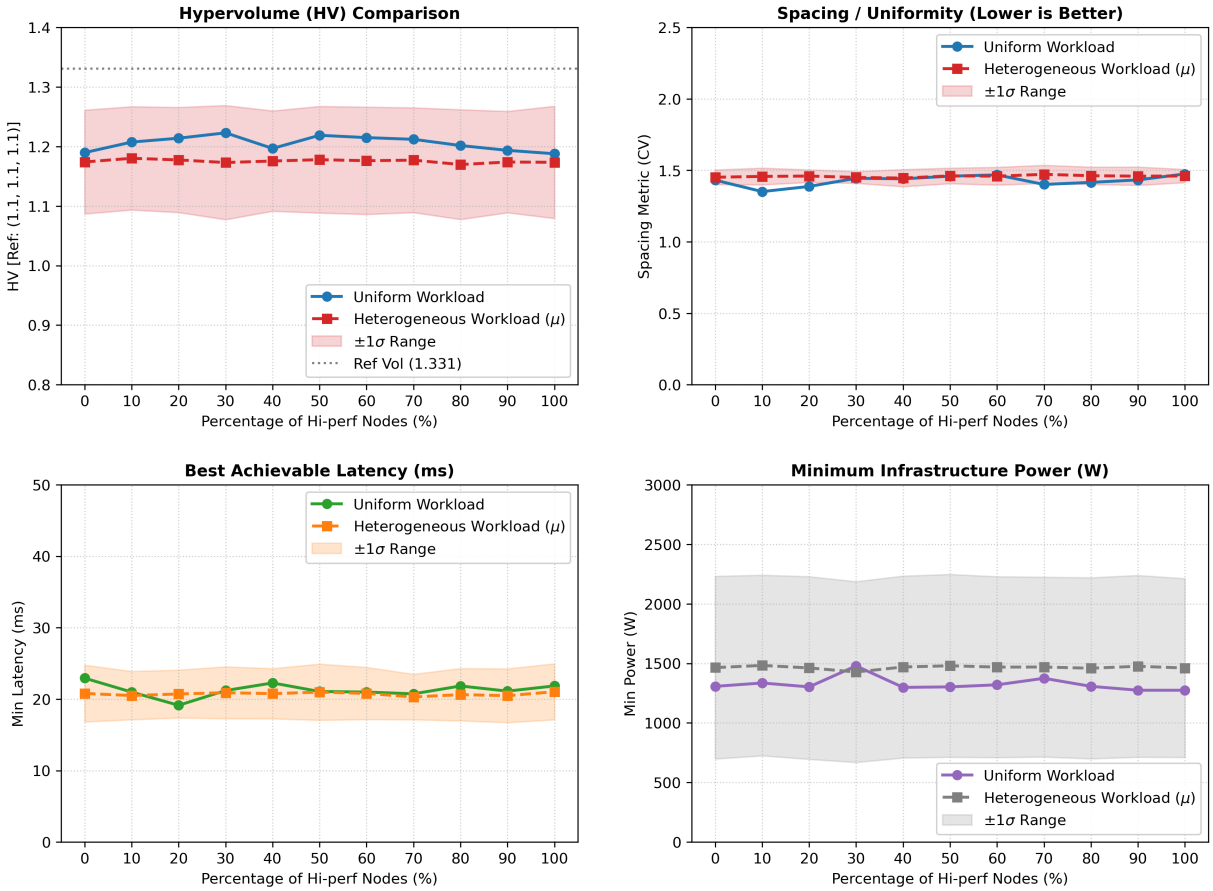


Figura 5.7: Analisi Comparativa delle Metriche di Interesse

Dall'analisi congiunta delle quattro metriche illustrate nella **Figura 5.7** emergono le seguenti evidenze statistiche:

- **Contrazione dell'ipervolume nel regime disomogeneo:** Mentre sotto uniforme l'ipervolume cresce fino a superare il valore di 1.22 grazie alla possibilità di impiegare diffusamente modelli ad alta capacità (YOLO *Large* ed *Extra-Large*), sotto carico asimmetrico la metrica collassa su un plateau medio costante pari a circa 1.1175. Tale contrazione è imputabile alla presenza di sorgenti calde soggette a *burst* di traffico ($\lambda_m \gg 10$req/s), le quali saturano rapidamente le code locali ($\rho_e \to 1$) costringendo l'algoritmo genetico a declassare i microservizi verso versioni più compatte (*Nano* o *Small*) per preservare l'ammissibilità fenomenologica delle soluzioni.

- **Invarianza e robustezza dello *spacing*:** La metrica di *Spacing* presenta un profilo pressoché

identico tra i due scenari, attestandosi stabilmente nell'intorno di $1.45 \div 1.48$. Questo valore riflette la natura intrinsecamente discreta delle scelte decisionali del problema. La perfetta sovrapposizione delle due curve dimostra l'elevata robustezza dell'algoritmo evolutivo, che mantiene la medesima capacità di campionare uniformemente lo spazio dei *trade-off* evitando frammentazioni o aggregazioni localizzate delle soluzioni, anche in presenza di forti asimmetrie territoriali nella domanda di traffico.

- **Asintoto fisico della latenza di servizi:** In entrambi i regimi di carico, la latenza minima raggiungibile si attesta rigidamente attorno a $20 \div 22$ ms, indipendentemente dalla quota di nodi *Hi-perf* installati. Tale soglia riflette il limite fisico invalicabile costituito dalla somma tra il ritardo minimo di propagazione di rete a corto raggio ($\delta_{near} = 2$ ms), il tempo di elaborazione irriducibile della variante YOLO Nano ($S_m^{Nano} \approx 12.81$ ms) e il tempo di attesa minimo non nullo nella coda computazionale *M/G/1*.

- **Sovracosto della *baseline* energetica:** L'assorbimento di potenza minimo evidenzia un divario netto e costante. Nel caso uniforme la dispersione bilanciata del carico permette all'ottimizzatore di attuare politiche di consolidamento spegnendo le macchine non necessarie (~ 1300 W). Nel caso eterogeneo, invece, la presenza di *burst* localizzati impone l'accensione contestuale di più server per evitare la saturazione delle code, fissando il pavimento energetico rigido a 1470 W causato dall'impossibilità di azzerare la potenza di *idle* (P_e^0) dei nodi coinvolti.

Infine, la limitata ampiezza della deviazione standard ($\pm 1\sigma$) lungo tutte le metriche, attesta l'elevata stabilità statistica dell'euristica proposta, dimostrando che l'algoritmo converge verso fronti paretiani qualitativamente e quantitativamente equivalenti a prescindere dalle specifiche fluttuazioni stocastiche generate dai semi pseudo-casuali.

6. Conclusioni e Sviluppi Futuri

Nel presente lavoro è stato affrontato il problema dell'allocazione congiunta di microservizi multi-versione all'interno di infrastrutture di calcolo perimetrali eterogenee. Riconoscendo come le moderne applicazioni distribuite, soprattutto nell'ambito della video analisi e della mobilità intelligente, adottino componenti scalabili disponibili in molteplici varianti neurali caratterizzate da complessità e dimensioni differenti, è stato formulato un problema di ottimizzazione multi-obiettivo.

Il modello matematico ha integrato simultaneamente tre metriche contrastanti:

1. La minimizzazione del tempo medio di risposta dell'infrastruttura. Il risultato è stato ottenuto mediante la modellazione dei servitori di calcolo come code *M/G/1* con tempi di servizio descritti da miscele di distribuzioni e risolti tramite la formulazione di Pollaczeck-Khincine.

2. La massimizzazione dell'accuratezza del servizio. Tale quantità è espressa attraverso la minimizzazione del complemento a una dell'accuratezza, $(1 - Acc)$, calcolata a partire da matrici di confusione aggregate e ponderate sui volumi di traffico effettivi.

3. La minimizzazione della potenza elettrica complessiva assorbita dall'infrastruttura. La formulazione è stata realizzata sfruttando un modello lineare che considera la potenza di base di *idle*, dell'assorbimento dinamico proporzionale al carico di lavoro e dell'eterogeneità hardware dei nodi.

Per esplorare con efficienza l'elevata dimensionalità dello spazio di ricerca e aggirare la complessità computazionale tipica dei problemi *NP-Hard*, è stata proposta un'euristica basata sull'algoritmo genetico elitista NSGA-II. L'architettura del motore evolutivo è stata concepita per rispecchiare fedelmente i vincoli operativi del sistema. La codifica cromosomica di coppie di geni (n_i, v_i) ha garantito a priori il soddisfacimento del vincolo di assegnamento univoco per ciascun microservizio. L'adozione congiunta di una procedura *greedy* di riparazione del sovraccarico locale e di una penalizzazione numerica vincolata per le violazioni di SLA ha consentito di gestire l'ammissibilità delle soluzioni, preservando la diversità genetica della popolazione.

La validazione empirica è stata condotta all'interno di un caso d'uso reale e applicativo incentrato sulla *Modena Automotive Smart Area*, impiegando microservizi basati sulla famiglia di modelli

per l'*object detection* Ultralytics YOLO. I risultati ottenuti confermano la validità dell'approccio proposto:

- **Mappatura dei *trade-off* paretiani:** L'algoritmo genetico ha approssimato una superficie continua di Pareto che declina chiaramente i regimi operativi estremi, offrendo ai pianificatori dell'infrastruttura uno strumento quantitativo per selezionare configurazioni target mirate.
- **Resilienza dell'euristica a carichi disomogenei:** Il confronto condotto sotto carichi stazionari e distribuzioni fortemente asimmetriche ha evidenziato l'elevata stabilità statistica dell'algoritmo. Nonostante si renda necessario l'uso di *plateau* rigido al consumo di minimo di potenza per impedire saturazioni di coda, la metrica di *spacing* si è mantenuta invariata, confermando la capacità dell'algoritmo di preservare una copertura geometrica uniforme dello spazio di ricerca senza generare frammentazioni o addensamenti localizzati.
- **Limiti asintotici dell'infrastruttura:** L'analisi comparativa ha permesso di isolare il limite fisico inferiore della latenza. Il risultato è stato determinato dalla componente non comprimibile del ritardo di trasmissione *wireless* a corto raggio sommata al tempo minimo di inferenza di YOLO Nano e ai ritardi residui accodamento. Ciò dimostra che oltre una certa soglia, l'aggiunta indiscriminata di accelerazione hardware non apporta benefici specifici significativi alla latenza complessiva.

6.1 Sviluppi Futuri

Le evidenze raccolte aprono diverse strade per estendere la ricerca:

- **Orchestrazione dinamica real-time:** Il modello formulato in questo lavoro si colloca a tempo di progettazione, configurandosi come uno strumento di pianificazione statica per scenari di orchestrazione dinamica a tempo di esecuzione. Il sistema è in grado di ricalcolare e riallocare progressivamente microservizi e versioni in risposta a fluttuazioni temporali del traffico veicolare o guasti imprevisti dei nodi, integrando politiche di minimizzazione dell'*overhead* introdotto dalle riconfigurazioni e dalle migrazioni.

- **Introduzione di nuovi obiettivi:** Lo spazio dei criteri decisionali potrà essere ampliato integrando metriche di affidabilità dell'infrastruttura, costi economici di *deployment* e manutenzione.

- ***Reinforcement Learning* e meta-euristiche ibride** Al fine di ridurre il tempo di calcolo necessario all'esplorazione del fronte durante variazioni repentine del carico, si potranno investigare architetture ibride in cui modelli di *Deep Reinforcement Learning* guidino la generazione della popolazione iniziale o l'orientamento degli operatori di mutazione, accelerando ulteriormente la convergenza verso le configurazioni Pareto-ottime.

Bibliografia

[1] Davide Agostini et al. «A Comparison of Static and Dynamic Micro-Service Placement Strategies for Edge Computing». In: *2025 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. 2025, pp. 1–6. DOI: 10.23919/SoftCOM66362.2025.11197363.

[2] Hadeel Muhammad Aljami et al. «Benchmarking YOLOv8 Variants for Object Detection Efficiency on Jetson Orin NX for Edge Computing Applications». In: *Computers* 15.2 (2026). ISSN: 2073-431X. DOI: 10.3390/computers15020074. URL: https://www.mdpi.com/2073-431X/15/2/74.

[3] Hemant Kumar Apat et al. «Fog service placement optimization: A survey of state-of-the-art strategies and techniques». In: *Computers* 14.3 (2025), p. 99.

[4] Claudia Canali e Riccardo Lancellotti. «GASP: Genetic Algorithms for Service Placement in Fog Computing Systems». In: *Algorithms* 12.10 (2019). ISSN: 1999-4893. DOI: 10.3390/a12100201. URL: https://www.mdpi.com/1999-4893/12/10/201.

[5] Claudia Canali et al. «OptiFog: A Framework to Optimize the Placement of Microservices in Fog Scenarios». In: *IEEE Transactions on Network and Service Management* 23 (2026), pp. 1499–1514. DOI: 10.1109/TNSM.2025.3648449.

[6] Kalyanmoy Deb e Himanshu Jain. «An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints». In: *IEEE Transactions on Evolutionary Computation* 18.4 (2014), pp. 577–601. DOI: 10.1109/TEVC.2013.2281535.

[7] Kalyanmoy Deb et al. «A fast and elitist multiobjective genetic algorithm: NSGA-II». In: *IEEE Transactions on Evolutionary Computation* 6.2 (2002), pp. 182–197. DOI: 10.1109/4235.996017.

[8] Renzhong Deng, Weijie Zheng e Benjamin Doerr. «The first theoretical approximation guarantees for the non-dominated sorting genetic algorithm III (NSGA-III)». In: *arXiv preprint arXiv:2504.21552* (2025).

[9] Muhammad Hussain. «YOLOv1 to v8: Unveiling Each Variant–A Comprehensive Review of YOLO». In: *IEEE Access* 12 (2024), pp. 42816–42833. DOI: 10.1109/ACCESS.2024.3378568.

[10] Huiting Li, Yaochu Jin e Ran Cheng. «Adaptive multi-stage evolutionary search for constrained multi-objective optimization». In: *Complex & Intelligent Systems* 10.6 (2024), pp. 7711–7740.

[11] Tsung-Yi Lin et al. «Microsoft coco: Common objects in context». In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[12] Soheil Mahdizadeh et al. «CRAFT: Latency and Cost-Aware Genetic-Based Framework for Node Placement in Edge-Fog Environments». In: *arXiv preprint arXiv:2507.12445* (2025).

[13] Adhitya Nugraha et al. «Service placement in fog computing: A systematic review of issues, techniques and multi-objective optimization approaches». In: *International Journal of Cognitive Computing in Engineering* (2026).

[14] Andre Opris. «Runtime Analyses of NSGA-III on Many-Objective Problems: Provable Exponential Speedup via Stochastic Population Update». In: *arXiv preprint arXiv:2505.01256* (2025).

[15] Joseph Redmon et al. «You Only Look Once: Unified, Real-Time Object Detection». In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[16] Giulia Tagliazucchi et al. «MASA - Modena Automotive Smart Area as Scalable Living Lab. Experiences and Development from the R-Nord District to the Modena Urban Area». In: *Transport Transitions: Advancing Sustainable and Inclusive Mobility*. A cura di Ciaran McNally et al. Cham: Springer Nature Switzerland, 2025, pp. 582–588. ISBN: 978-3-031-85578-8.