La Comunicazione - Note, Recensioni e Notizie n. 65
Anno 2022

GPU computing per l'ottimizzazione della copertura delle reti di broadcasting

GPU computing for coverage optimization of broadcasting networks

Emiliano Pallotti♦, Federica Mangiatordi♦

♦ Fondazione Ugo Bordoni

Sommario

L'evoluzione delle tecnologie di broadcasting e la rapida diffusione di nuovi servizi digitali rende necessario disporre di strumenti di analisi estremamente accurati ed efficienti in grado di valutare le prestazioni di copertura delle reti al variare dei parametri operativi e delle risorse disponibili. L'adozione di piattaforme di calcolo in grado di elaborare rapidamente i dati delle reti SFN (Single Frequency network) su larga scala consente di effettuare un'analisi rapida ed economica di diversi scenari operativi ottimizzando le prestazioni di copertura ottenibili per le possibili pianificazioni implementate.

Con riferimento a tale ambito l'articolo presenta una piattaforma di elaborazione parallela GPU (Graphics Processing Unit) per ottimizzare i ritardi artificiali introdotti nei trasmettitori televisivi DVB-T2 (Digital Video Broadcasting- Terrestrial Version 2) Questo al fine di massimizzare la copertura delle reti SFN con diverse strategie di sincronizzazione secondaria.

L' elevato numero dei parametri presenti nel problema viene affrontato tramite tecniche di ottimizzazione euristica CSA (Corana Simulated Annealling) e GPSO (Generational Particle Swarm Optimization), mentre la complessità temporale del processo di ottimizzazione è contenuto distribuendo l'esecuzione del calcolo della copertura dei numerosi pixel del territorio italiano sulle migliaia di multiprocessori disponibili nelle moderne schede grafiche.

Abstract

The evolution of broadcasting technologies and the rapid spread of new digital services makes it necessary to have highly accurate and efficient analysis tools capable of assessing the coverage performance of networks as the operating parameters and available resources vary.

Furthermore, the adoption of computing platforms capable of rapidly processing the data of large-scale SFN networks allows for a quick and economic analysis of different operating scenarios by optimizing the coverage performance obtainable for the possible plans implemented. Concerning these issues, the article presents a GPU parallel computing platform to optimize the artificial delays introduced in the DVB-T2 television transmitters to maximize SFN networks' coverage with different secondary synchronization strategies. Moreover, CSA (Corana Simulated Annealling) and GPSO (Generational Particle Swarm Optimization) heuristic optimization techniques are used to address the high number of parameters of the underlying mathematical problem. In contrast, distributing the radio coverage calculation of the massive number of Italian territory pixels on the thousands of multiprocessors of the modern graphics cards reduces the optimization process's time complexity.

Keyword
DVB-T2, GPU Computing, Parallel programming, Heuristic optimization

1 - Introduzione

Nei prossimi anni il settore del broadcasting televisivo terrestre subirà profonde trasformazioni dettate dalla necessità di rilasciare le risorse spettrali della banda 700MHz a favore dei servizi di telefonia mobile [1] e dalla prospettiva di offrire all'utenza canali interattivi con programmi ad elevata qualità video.

Questo processo evolutivo richiede la disponibilità di avanzati strumenti di pianificazione delle reti di broadcasting per razionalizzare l'uso delle risorse spettrali in funzione della qualità di servizio e della sostenibilità tecnica valutando l'impatto delle modifiche richieste alla capacità trasmissiva dall'adozione di nuovi standard di codifica dei contenuti multimediali.

L'analisi accurata dei molteplici scenari operativi, prospettabili per le reti SFN pianificate a livello locale e nazionale, necessita di tempi computazionali contenuti per individuare soluzioni efficaci ed economiche, che possano essere eventualmente condivise tra gli operatori del settore come riferimento comune ed indipendente.

Questa prospettiva motiva la scelta di ricorrere ad architetture hardware e software ad elevatissima capacità computazionale in grado di simulare efficientemente scenari radio complessi con un elevato numero di parametri in gioco. Un caso tipico è il posizionamento e il dimensionamento ottimale dei parametri trasmissivi degli impianti d'antenna di una rete SFN per il broadcasting televisivo usando la tecnologia DVB-T2.

Nella stima della massima copertura ottenibile per una rete, un ruolo rilevante è rivestito dall'impiego di opportuni ritardi di trasmissione del segnale OFDM (Orthogonal Frequency-Division Multiplexing). Tale artificio permette di limitare le interferenze distruttive tra i segnali iso-frequenziali a scapito di una maggiore complessità del ricevitore televisivo. Questo specifico problema della pianificazione di una rete SFN è affrontato in questo articolo, che presenta una piattaforma di GPU computing [2], [3] per gestire in modo ottimale i ritardi artificiali in trasmissione utilizzando le tecniche di ottimizzazione euristica GPSO [3] e CSA [4].

Nelle sezioni successive viene fornita una breve panoramica del GPU computing e del paradigma di parallelizzazione CUDA (Compute Unified Device Architecture); poi viene descritta la metodologia di calcolo della copertura di una rete SFN con riferimento al servizio di broadcasting televisivo, e formulato il problema di ottimizzazione dei ritardi; le due sezioni finali descrivono rispettivamente l'architettura logica della piattaforma e presentano il miglioramento delle performance di copertura per due reti SFN regionali di riferimento quando si ottimizza il valore dei ritardi artificiali in trasmissione.

2 - GPU computing

Le GPU (Graphics Processing Unit) sono schede hardware, collegate alle motherboard dei calcolatori e progettate per elaborare rapidamente e in modo efficiente elevate quantità di dati matriciali. A partire dagli ultimi anni 90, queste unità si sono evolute da circuiti dedicati alla manipolazione accelerata della memoria per la creazione di immagini, a matrici di migliaia di coprocessori per il calcolo logico aritmetico (Arithmetic-Logic Units) programmabili. I chip delle GPU attualmente sul mercato (NVidia RTX, Tesla, GTX,ecc) sono realizzati da decine di miliardi di transistor finalizzati a svolgere operazioni computazionali piuttosto che funzioni di memorizzazione cache e controllo di flusso dati. Questa caratteristica rende le GPU estremamente efficienti nel computing

parallelo su grandi volumi di dati con capacità computazionali che, raggiungono le decine di teraflops/s [5]. La disponibilità di migliaia di unità logico aritmetiche e la presenza di decine di GByte di memoria RAM a bassa latenza sulle schede rende le GPU adatte non solo all'elaborazione grafica ma anche per applicazioni di calcolo scientifico, di intelligenza artificiale e di high performance computing (HPC) quali quelle del data science, della robotica, della dinamica molecolare, della finanza computazionale, del mobile e dell'edge computing [6], [7]. Questo ampliamento delle potenzialità di applicabilità delle schede grafiche GPU a molteplici ambiti del mondo industriale, scientifico e ingegneristico è sintetizzato con l'acronimo GPGPU (General-Purpose GPU) Computing.

Nell'ultimo decennio sono stati proposti vari approcci al GPU Computing, facilmente identificabili attraverso le modalità e i framework di implementazione ed esportazione del software da eseguire su GPU, tra questi i principali sono: OpenACC, OpenMP, CUDA e OpenCL [8], [9], [10].

L'OpenACC e l'OpenMP sono standard di direttive di precompilazione e librerie runtime che consentono di sviluppare applicazioni di "parallel computing" per sistemi di elaboratori eterogenei con vari sistemi operativi. L'Open ACC consente di implementare il parallelismo dei dati attraverso codici sequenziali implementati utilizzando i linguaggi di programmazione C, C++ e Fortran e identificando, con specifiche direttive di precompilazione, le porzioni di codice da eseguire su GPU, mentre l'OpenMP permette il multithreading parallelo e concorrente su architetture di processori a memoria condivisa.

Il CUDA e l'OpenCL nascono come estensioni di linguaggi ad alto livello (C, C++, Fortran) attraverso i quali produttori come la NVIDIA e le aziende del Khronos Group (www.khronos.org/opencl) hanno sviluppato librerie, framework e interfacce per supportare il GPGPU computing tramite l'interazione delle GPU con la CPU su strutture dati non strettamente rivolte alla manipolazione grafica. Il modello di programmazione abilitato dal CUDA e dall' OpenCL permette la progettazione e l'implementazione di sistemi di calcolo eterogenei in cui coesistono task demandati ai cores della CPU e funzioni demandate ai cores della GPU. Nello specifico i task sequenziali e di controllo algoritmico vengono controllati ed eseguiti dalla CPU, mentre le funzioni e i processi ripetitivi su ingenti volumi di dati sono parallelizzati in accordo al paradigma SIMT (Single Instruction, Multiple Threads) per eseguirli

sulla GPU. In questo lavoro, per affrontare il problema della scelta ottimale dei ritardi dei trasmettitori televisivi per la massimizzazione della copertura delle reti SFN si è adottato il paradigma di parallel computing proposto da CUDA. In questo lavoro, per affrontare il problema della scelta ottimale dei ritardi dei trasmettitori televisivi per la massimizzazione della copertura delle reti SFN si è adottato il paradigma di parallel computing proposto da CUDA.

3 – Il paradigma di parallelizzazione CUDA

La piattaforma CUDA supporta un modello di programmazione parallela di tipo multi-threads (multi-processo) gerarchico che espone le GPU come macchine virtuali multicore costituite da cluster di unità logico matematiche (ALU o core processor), denominati Streaming Multiprocessor (SM) [11]. Ogni SM esegue più threads in parallelo secondo il paradigma SIMD (Single Instruction on Multiple Data), pertanto più processori eseguono la stessa istruzione su gruppi di dati diversi. Ciascun SM dispone di una memoria cache condivisa di primo e di secondo livello per le istruzioni e i dati e di un'interfaccia ad alta velocità alla memoria globale della GPU (GDDR - Graphics Double Data Rate Synchronous Dynamic Random-Access Memory).

La parallelizzazione del codice sugli SM della GPU presuppone la suddivisione dei threads in blocchi, disposti secondo delle griglie aventi dimensione massima pari a 3.

I threads di un blocco possono utilizzare la memoria condivisa per lo scambio di dati e sincronizzarsi tra loro, mentre i threads appartenenti a blocchi diversi non possono scambiare informazioni. Ogni blocco di threads è assegnato ad un multiprocessore (SM) ed ulteriormente suddiviso in gruppi (warp) di 32 threads per l'esecuzione in parallelo sui cores di un SM. Il numero massimo di warps per SM e il numero di core in un SM dipendono dalla specifica architettura della GPU. Nelle architetture NVIDIA Ampere e Pascal gli SM hanno 128 cores e al più 64 warps concorrenti per SM. Gli schedulatori dei warp (warp scheduler) di ciascun SM gestiscono in modo congiunto l'esecuzione dei threads concorrenti. Pertanto, per sfruttare totalmente il parallelismo intrinseco all'architettura della GPU è opportuno che tutti i threads di un warp eseguano la stessa istruzione su dati diversi. Il mancato soddisfacimento di questa

condizione determina una divergenza tra i threads del warp e degrada le prestazioni in termini di velocità e parallelismo in quanto lo scheduler deve elaborare le istruzioni del warp in modo seriale disattivando di volta in volta parte dei threads per eseguire in parallelo solo i sottoinsiemi di threads omogenei.

Nello sviluppo software, la porzione di codice parallelo da eseguire sui "core" della GPU è specificata nelle "kernel functions". La definizione di ogni kernel richiede "l'istanziamento" di una griglia di blocchi, tutti con lo stesso numero di thread; ogni blocco viene eseguito da un solo multiprocessore, mentre più blocchi possono essere eseguiti in parallelo dallo stesso SM. Il compilatore NVCC (NVIDIA CUDA Compiler) provvede alla separazione e codifica in istruzioni PTX (Parallel Thread eXecution) delle kernel function da inviare alla GPU rispetto alle istruzioni da eseguire su CPU.

In fase di sviluppo software, il programmatore decide la dimensione dei blocchi e delle griglie per ciascuna kernel function, ma lo schedulazione dei threads e la distribuzione dei blocchi sugli SM viene gestita in modo trasparente consentendo di scalare facilmente rispetto alle risorse computazionali della specifica GPU. La scelta della struttura della griglia (dimensionalità, numero complessivo di blocchi e threads) influenza, tuttavia, in modo significativo le massime prestazioni ottenibili in termini di tempi di esecuzione e costituisce un elemento cruciale in fase di progettazione e sviluppo del codice.

4 - Valutazione della copertura

In una rete broadcasting SFN tutti i siti trasmissivi impiegano la stessa frequenza, sfruttando la tecnologia COFDM (Coded Orthogonal Frequency Division Multiplexing), per comporre costruttivamente i segnali iso-frequenziali in ricezione, quando tutti gli echi del segnale raggiungono il ricevitore in un intervallo di tempo inferiore al tempo di guardia [12].

Nella decodifica del segnale, il ricevitore OFDM effettua la sincronizzazione in due fasi: nella prima si sincronizza al tasso di simbolo; nella seconda seleziona la posizione ottimale in cui aprire la finestra FFT (Fast Fourier Transform) per demodulare i diversi simboli trasmessi.

La presenza in ricezione di più segnali iso-frequenziali complica il processo di sincronizzazione secondaria in quanto nella finestra temporale in cui deve essere demodulato il simbolo OFDM

n-esimo è presente una parte della potenza dei simboli precedenti o successivi a quello n-esimo, come rappresentato in figura 1. Questo fenomeno genera l'interferenza inter-simbolo (ISI) e può causare l'impossibilità di demodulare il segnale OFDM.

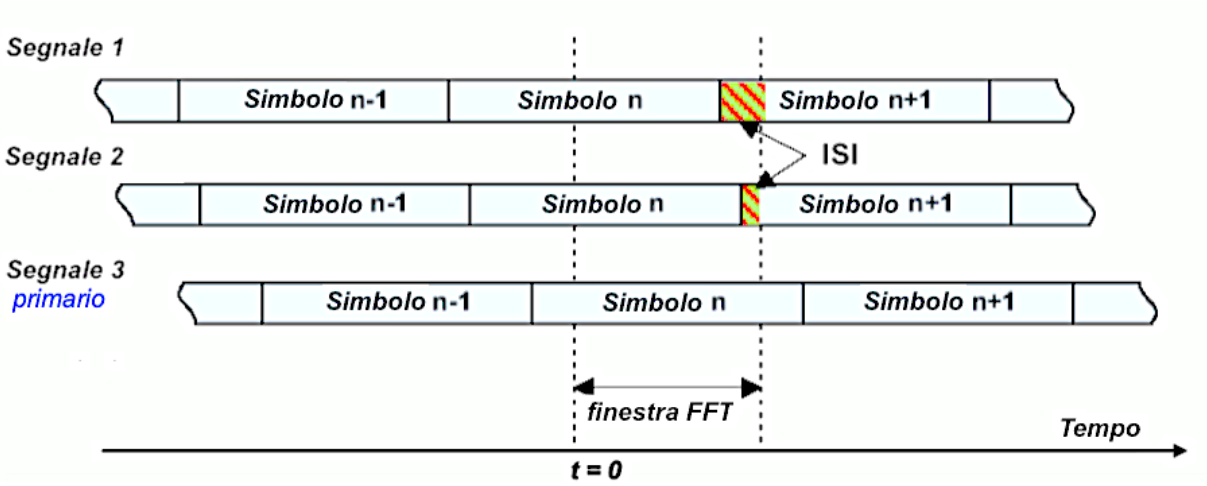

Figura 1 – *Sincronizzazione secondaria [12]*

Una soluzione per tentare di attenuare l'ISI è costituita dalla ripetizione di una parte del simbolo OFDM per un tempo di guardia T_g pari ad una frazione del tempo di simbolo T_u.
L'estensione del simbolo OFDM consente l'eliminazione dell'ISI degli echi con sfasamento inferiore al tempo di guardia e l'incremento della potenza del segnale utile.
Nel caso in cui il ricevitore effettua la stima della risposta del canale per l'equalizzazione con una finestra di durata T_p, i segnali che forniscono un contributo positivo alla potenza utile sono quelli con sfasamento relativo di durata inferiore all'intervallo di equalizzazione T_p.
Con queste premesse, la valutazione delle prestazioni di una rete SFN pianificata per un territorio, si effettua calcolando la percentuale di copertura del servizio erogato tramite le Equazioni (1) di seguito riportata.

Equazione 1 - Funzione calcolo copertura.

$$f_{copertura} = 100 * \frac{\sum_i a_i * Pop_i}{\sum_i Pop_i}$$
$$a_i = \begin{cases} 0 \; se \; (C/(I+N))_i < EPT \\ 1 \; se \; (C/(I+N))_i \geq EPT \end{cases} \quad (1)$$

Questo calcolo presuppone:

- la mappatura del territorio su una griglia di aree quadrate Q_i (di seguito denominate pixel);
- la scelta di un punto di verifica (x_i, y_i) per ciascun pixel;
- la conoscenza del numero di individui residenti in ciascun pixel (Pop_i);
- la stima del rapporto tra la potenza del segnale utile C e la potenza aggregata del segnale interferente e del rumore $(I + N)$ nel punto (x_i, y_i), ovvero della quantità $(C/(I + N))_i$;

Nella (1) il parametro EPT (Effective Protection Threshold) è il valore di soglia minimo che assicura il corretto funzionamento del demodulatore. Questo parametro dipende dalla qualità dello specifico servizio che si intende erogare.

Considerato che:

- le potenze delle repliche del segnale accrescono positivamente la potenza del segnale utile C se giungono in ricezione con ritardi inferiori al tempo di guardia T_g;
- il contributo positivo delle repliche alla potenza del segnale utile diminuisce progressivamente quando il loro ritardo supera T_g e si annulla per sfasamenti superiori a T_p;

le potenze del segnale utile ed interferente nella (2) si possono determinare tramite le seguenti espressioni:

Equazione 2 - Segnale Utile ed Interferente.

$$C = \sum_k w_k P_k \qquad (2)$$
$$I = \sum_k (1 - w_k) P_k$$

dove:

- w_k è il coefficiente di peso della potenza dell'eco k-esimo che giunge sul ricevitore;
- P_k è la potenza dell'eco k-esimo all'ingresso del ricevitore.

Il valore del coefficiente di peso w_k è determinato dall'espressione (3) di seguito specificata:

Equazione 3 - Coefficiente di peso.

$$w_k(t) = \begin{cases} 0 \quad se \quad t \leq T_g - T_p \\ \left(\frac{T_u + t}{T_u}\right)^2 \quad se \quad T_g - T_p < t \leq 0 \\ 1 \quad se \quad 0 < t \leq T_g \\ \left(\frac{T_u + T_g - t}{T_u}\right)^2 \quad se \quad T_g < t \leq T_p \\ 0 \quad se \quad t > T_p \end{cases} \tag{3}$$

in cui:

- t è l'istante di arrivo dell'eco rispetto all'inizio della finestra di equalizzazione adottata dal ricevitore;
- T_u , T_g *e* T_p sono rispettivamente le durate del tempo di simbolo, del tempo di guardia e della finestra di equalizzazione.

Occorre evidenziare che il valore dei pesi w_k dipende strettamente dal posizionamento dell'inizio della finestra FFT rispetto agli istanti di ricezione di ogni replica del segnale, per cui la strategia di sincronizzazione secondaria del ricevitore si riflette strettamente sulle prestazioni della rete SFN in termini di copertura.

4.1 – Criteri di sincronizzazione

La letteratura scientifica [13] propone diversi criteri di sincronizzazione secondaria da utilizzare nel ricevitore quando l'algoritmo dettagliato per il posizionamento della finestra FFT non è prescritto dallo standard, come nel caso del broadcasting DVB-T e DVB-T2.

La mancanza di un criterio privilegiato implica la necessità di considerare diversi metodi nella simulazione delle prestazioni di copertura della rete SFN in quanto gli algoritmi adottati nei ricevitori commerciali non sono resi disponibili dai produttori.

Indicato con τ_k l'istante di arrivo della replica k-esima del segnale OFDM associato a una potenza P_k, i principali criteri di sincronizzazione secondaria del ricevitore sono quelli di seguito specificati:

- *Strongest Signal* - La finestra FFT viene sincronizzata con il segnale più forte che viene ricevuto nel pixel, ovvero l'istante t=0 della finestra di equalizzazione è l'istante di arrivo del segnale con la potenza P_k più elevata. La figura 2 illustra un caso di finestra di equalizzazione sincronizzata con il criterio esposto;
- *First Received Signal* - La finestra FFT è sincronizzata con il segnale che presenta il più piccolo ritardo di propagazione in ricezione;
- *First Signal Above a Threshold level*- La finestra FFT è posizionata in modo che il tempo *t*=0 coincida con l'istante di arrivo τ_k del primo segnale che sia superiore a una soglia N_{th};. questa soglia solitamente è scelta tra 6 e 10 dB al di sotto della potenza del segnale più forte in ricezione. In figura 3 è presentato un caso di finestra di equalizzazione sincronizzata con il criterio esposto.
- *Centre of Gravity* - La finestra FFT è allineata sull'istante t_c calcolato come la media dei tempi di ricezione τ_k pesati con la potenza dei segnali P_k, ovvero $t_c = \frac{\sum_k \tau_k \cdot P_k}{\sum_k P_k}$;
- *Maximum C/I* - la finestra FFT viene posizionata in modo da massimizzare il rapporto tra potenza del segnale utile C e potenza del segnale interferente I in ricezione;
- *Quasi Optimal C/I* - La finestra FFT viene allineata andando prima a valutare il (C/I) utilizzando come "*first received signal"* tutti i segnali superiori ad una determinata soglia; successivamente si individua la configurazione che assicura la demodulazione da parte del ricevitore, ovvero $(C/I) \geq \delta_{min}$;
- *Quasi Optimal C* - la finestra FFT viene sincronizzata in modo da massimizzare la potenza utile C all'interno dell'intervallo di equalizzazione andando a utilizzare progressivamente come *first received signal* tutti i segnali superiori ad una determinata soglia.

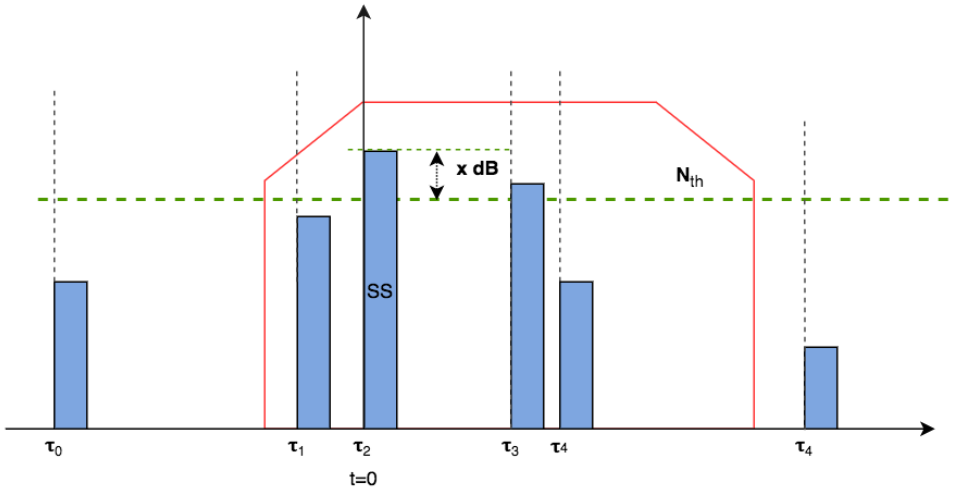

Figura 2 – *Finestra di equalizzazione sincronizzata con il criterio* ***Strongest Signal***

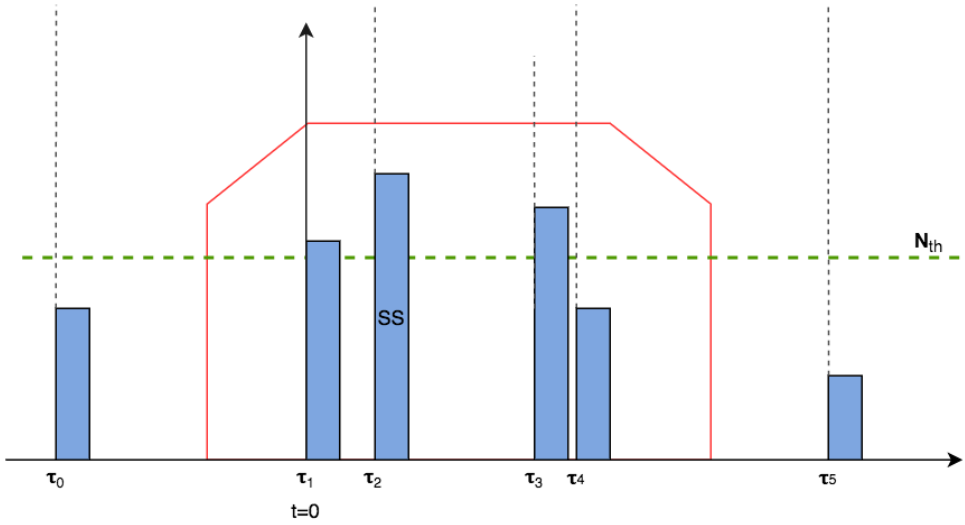

Figura 3 – *Finestra di equalizzazione sincronizzata con il criterio* ***First Signal Above a Threshold level****.*

5 – Problema della ottimizzazione dei ritardi

Nella pianificazione di una rete SFN è possibile ottimizzare la copertura di un servizio di broadcasting per un determinato territorio tramite l'introduzione di ritardi artificiali ϑ_k nella trasmissione dei segnali iso-frequenziali al fine di aumentare i pixel per cui la condizione $\left({}^{C}/_{I}\right) \geq SNR_{min}$ sia soddisfatta. Per una rete SFN con N trasmettitori $\{Tx_k\}$ questo problema è NP complesso [14]. La circostanza per cui esistono diverse combinazioni dei ritardi in grado di assicurare la copertura di un pixel Q_i, e la possibilità che lo "*strongest server*" di Q_i, interferisca distruttivamente con i segnali dominanti nei pixel limitrofi a Q_i, rende complesso formulare a priori una legge per la determinazione dei valori ottimi dei ritardi artificiali ϑ_k degli impianti trasmissivi $\{Tx_k\}$. D'altra parte, l'esplorazione esaustiva di tutti i valori ammissibili per ciascun ritardo ϑ_k non è praticabile, in quanto la complessità computazionale sarebbe esponenziale con il numero dei trasmettitori Queste ragioni motivano il ricorso a tecniche di ottimizzazione euristica per la ricerca della soluzione ottimale del problema di massimizzazione della copertura del servizio tramite l'introduzione di ritardi artificiali in trasmissione per ciascun impianto della SFN.

Assegnato l'insieme $\{Tx_1, \ Tx_2, \ \cdots \ Tx_N\}$ degli impianti trasmissivi della rete SFN e indicato con ϑ_i il ritardo artificiale di T_{x_i}, si può formalizzare il problema di ottimizzazione assumendo come funzione obiettivo (o funzione di fitness [15]) la funzione di copertura nell'espressione (1). Questa funzione deve essere valutata considerando che l'istante di arrivo τ_k del segnale diffuso dal trasmettitore Tx_j nel generico pixel Q_i è la somma del ritardo di propagazione del campo elettromagnetico generato da Tx_j e del ritardo artificiale ϑ_j. Di conseguenza, la copertura risulta funzione del vettore $\boldsymbol{\vartheta}$ con componenti ϑ_i definito dall'espressione (4).

Equazione 4. Espressione del vettore dei ritardi.

$$\boldsymbol{\vartheta} = (\vartheta_1, \quad \vartheta_2, \quad \cdots \quad \vartheta_N)^T \tag{4}$$

Risolvere il problema di ottimizzazione dei ritardi artificiali consiste nel determinare il valore ottimale del vettore $\boldsymbol{\vartheta}$ che massimizza la copertura del servizio, ovvero individuare il valore $\boldsymbol{\vartheta_{opt}}$ che soddisfa l'equazione (5).

Equazione 5 - Modello matematico del problema di ottimizzazione dei ritardi.

$$f_{copertura}(\boldsymbol{\vartheta_{opt}}) = \max_{\boldsymbol{\vartheta_1},..,\boldsymbol{\vartheta_N},} f_{copertura}(\vartheta_1, \vartheta_2, \cdots \vartheta_N) = \max_{\boldsymbol{\vartheta}} f_{copertura}(\boldsymbol{\vartheta}) \quad s.v. \quad \Delta_{L_i} \leq \vartheta_i \leq \Delta_{U_i}(4) \tag{5}$$

Il problema descritto dall'equazione (5) può essere riformulato come la minimizzazione della funzione di costo, costituita dalla percentuale di popolazione non raggiunta dal servizio erogato dalla rete SFN. Questo problema può essere risolto con tecniche di ottimizzazione euristica, dato l'elevato numero di pixel e di siti trasmissivi.

In questo lavoro si è fatto riferimento a due metodi meta-euristici: l'algoritmo GPSO (Generational *Particle Swarm Optimization*) e quello CSA (*Corana Simulated Annealing Optimization)*, brevemente presentati nelle Sezioni 5.1 e 5.2.

5.1 – Generational Particle Swarm Optimization (GPSO)

Il PSO è un algoritmo meta-euristico che emula il comportamento di uno stormo di uccelli nella ricerca del cibo [16,17]. L'esplorazione dello spazio N-dimensionale dei ritardi viene effettuata andando a valutare la funzione di fitness in corrispondenza delle posizioni assunte da M individui dello stormo. La posizione $\boldsymbol{x}$ di un individuo dello stormo rappresenta una ennupla di valori per il vettore dei ritardi artificiali $(\vartheta_1, \vartheta_2, \dots, \vartheta_N)$.

Ad ogni iterazione, per tutti gli M componenti dello stormo, viene valutata la funzione di fitness al fine di individuare la migliore posizione $\boldsymbol{r}_{best}$, definita come quella che massimizza la copertura $f(\boldsymbol{r}_i)$ per $i \in \{1,2,..,M\}$.

La posizione di ciascun individuo dello stormo viene aggiornata alla *i*-esima iterazione tramite una delle seguenti modalità:

Equazione 6 - Espressione di aggiornamento della posizione dell'individuo nella modalità "coefficiente di costrizione".

$$\begin{cases} \mathbf{v}_{i+1} = \omega\left(\mathbf{v}_i + \eta_1\mathbf{r_1}\cdot\left(\mathbf{x}_i - \mathbf{x}_i^{l}\right) + \eta_2\mathbf{r}_2\cdot\left(\mathbf{x}_i - \mathbf{x}^{g}\right)\right) \\ \mathbf{x}_{i+1} = \mathbf{x_i} + \mathbf{v_i} \end{cases} \quad (6)$$

Equazione 7 - Espressione di aggiornamento della posizione dell'individuo nella modalità "pesatura inerziale"

$$\begin{cases} \mathrm{v}_{i+1} = \omega\mathbf{v}_i + \eta_1\mathbf{r_1}\cdot\left(\mathbf{x_i} - \mathbf{x_i^{l}}\right) + \eta_2\mathbf{r_2}\cdot\left(\mathbf{x}_i - \mathbf{x}^{g}\right) \\ \mathbf{x_{i+1}} = \mathbf{x_i} + \mathbf{v_i} \end{cases} \quad (7)$$

Il termine $\mathbf{x^l}$ presente nelle equazioni (6) e (7) è la migliore posizione raggiunta da un individuo dello stormo tra tutte quelle che ha visitato nelle iterazioni precedenti (per individuare il massimo della funzione di fitness ovvero della funzione di copertura).
Il valore del *global best* $\mathbf{x^g}$ è la migliore posizione raggiunta in assoluto dagli individui dello stormo fino a quel momento.
I parametri ω, η_1, η_2 sono dei fattori di inerzia che determinano con quale velocità l'individuo tende a modificare la sua posizione nella direzione del suo *local best* $\mathbf{x^l}$.ovvero del *global best* $\mathbf{x^g}$.
Nel GPSO [18] gli individui dello stormo esplorano lo spazio dei ritardi in accordo alle migliori prestazioni raggiunte da tutti gli individui della generazione precedente (generazione i-esima); conseguentemente si aggiornano prima le velocità di tutti gli individui dello stormo tramite l'equazione (8) e poi si modificano le loro posizioni nello spazio secondo l'equazione (9).

Equazione 8 - Espressione di aggiornamento della velocità dell'individuo m-esimo dello stormo all'*i*-esima generazione nella versione "*canonical*" del GPSO

$$\mathbf{v_{i,m}} = \omega\mathbf{v}_{i-1,m} + \eta_1\mathbf{r_1}\cdot\left(\mathbf{x_{i-1,m}} - \mathbf{x^{l}_{i-1,,m}}\right) + \eta_2\mathbf{r_2}\cdot\left(\mathbf{x}_{i-1,m} - \boldsymbol{x_{i-1}}^{g}\right) \quad m = 1,2,..|M| \quad (8)$$

Equazione 9 - Espressione di aggiornamento della posizione dell'individuo m-esimo dello stormo all'*i*-esima iterazione

$$\mathbf{x}_{\mathrm{i,m}} = \mathbf{x}_{\mathbf{i-1,m}} + \mathbf{v}_{\mathbf{i,m}}$$
$$m = 1,2,..|M| \qquad (9)$$

Nella versione "*canonical*" del GPSO (*pesatura inerziale*) il fattore di aleatorietà ω moltiplica solo la velocità dell'individuo e non influenza i valori randomici (uniformemente distribuiti nell'intervallo (0,1)) - dei vettori $\mathbf{r}1$ e $\mathbf{r}2$ come in (6).

A seconda della modalità di scelta di $r_{1\mathrm{j}}$ e $r_{2\mathrm{j}}$ si distinguono 4 varianti [16]:

- *Variante 1* - quando la componente cognitiva $\mathbf{r}1$=[r_{11},r_{12},...,r_{1n}] è completamente differente dalla componente sociale $\mathbf{r}2$=[r_{21},r_{22},...,r_{2n}] (*canonical GPSO with inertia weight*);
- *Variante 2* - quando i pesi della componente cognitiva $\mathbf{r}1$=[r_{11},r_{12},...,r_{1n}] sono uguali a quelli della componente sociale $\mathbf{r}2$=[r_{11},r_{12},...,r_{1n}] (*canonical PSO with inertia weight and equal random weights of social and cognitive components*);
- *Variante 3* - quando tutti i pesi della componente cognitiva e della componente sociale assumono lo stesso valore per ogni dimensione dello spazio di ricerca ma $\mathbf{r}1$=[r_1,r_1,...,r_1], e $\mathbf{r}2$=[r_2,r_2,...,r_2] sono diversi (*PSO with some random number for all components*);
- *Variante 4* - quando i pesi della componente cognitiva sono uguali a quelli della componente sociale e assumono lo stesso valore per ogni dimensione dello spazio di ricerca, ovvero $\mathbf{r}1$=[r_1,r_1,...,r_1] e $\mathbf{r}2$=[r_1r_2,r_1r_2,...,r_1r_2] (*PSO with some random number for all components and with equal random weights of social and cognitive components*);

Il GPSO con *coefficienti di costrizione* rappresenta la *Variante 5* e prevede la moltiplicazione di $\mathbf{r}1$ e $\mathbf{r}2$ per il coefficiente di inerzia ω. In questo caso i vettori $\mathbf{r}1$ e $\mathbf{r}2$ sono diversi tra loro e con valori delle componenti distinte per ogni dimensione dello spazio di ricerca.

La *Variante 6* (Nayyar, 2018) dell'algoritmo è quella denominata in letteratura *Full Informed Particle Swarm (FIPS)* [19] e prevede, rispetto alla versione *canonical,* la dipendenza del movimento dell'individuo dalle migliori posizioni raggiunte da tutti i suoi vicini.

Indicato con Σ_i l'insieme dei vicini dell'individuo m-esimo dello stormo all'i-esima iterazione, nel FIPS la velocità dell'individuo viene modificata in accordo all'equazione (10), dove $\mathbf{x}_{\mathrm{i}-1}$ è la posizione dell'individuo m-esimo all'iterazione precedente ed $\mathbf{x}_{\mathrm{p}}^{\mathbf{b}}$ la migliore posizione raggiunta dal suo p-esimo vicino.

Equazione 10 - Espressione di aggiornamento della posizione dell'individuo m-esimo dello stormo all'*i*-esima iterazione per la variante FIPS.

$$\mathbf{v}_{\mathbf{i},\mathbf{m}} = \omega \mathbf{v}_{\mathrm{i}-1,\mathrm{m}} + \sum_{p=1}^{|\Sigma_i|} \gamma_p \cdot \frac{\mathbf{x}_{\mathrm{p}}^{\mathbf{b}} - \mathbf{x}_{\mathrm{i}-1}}{|\Sigma_i|}$$
$$\mathbf{x}_{\mathrm{i,m}} = \mathbf{x}_{\mathrm{i}-1,\mathrm{m}} + \mathbf{v}_{\mathbf{i},\mathbf{m}}$$
$$m = 1,2,..|M| \qquad (10)$$

5.2 – Corana Simulated Annealing (CSA)

Il Simulated Annealing (SA) è un algoritmo stocastico di ottimizzazione, proposto da Kirpatrick e Cerny [20], per simulare il processo che porta un materiale dallo stato fluido alla cristallizzazione. Se la riduzione della temperatura è eccessivamente rapida nel processo, gli atomi possono essere intrappolati in uno stato a energia minima locale con conseguente formazione di imperfezioni nella struttura del reticolo cristallino. Una riduzione graduale della temperatura consente, invece, di portare gli atomi da uno stato ad elevata entropia ad uno stato a minima entropia con la loro disposizine secondo una struttura reticolare regolare.

Questo processo fisico è simulato dal metodo SA per trovare la soluzione di problemi di programmazione non lineare con un numero discreto di variabili continue.

L'algoritmo consiste nel:

- Generare una sequenza di punti casuali $x^{(k)}$ nello spazio di ricerca in corrispondenza della migliore posizione corrente $x^{(oc)}$ per la funzione di costo $f(\cdot)$ del problema in esame;
- Valutare la funzione $f(\cdot)$ sui punti ottenuti;

- Accettare il punto $x^{(k)}$ se il valore della funzione di costo è inferiore al suo migliore valore corrente f^{oc}, modificando il valore migliore della funzione di costo in $f^{oc} = f(x^{(k)})$;
- Accettare o rifiutare il nuovo punto $x^{(k)}$, quando $f(x^{(k)}) > f^{oc}$, sulla base del valore della funzione di densità di probabilità della distribuzione di Bolzman-Gibbs calcolato dall'equazione (11) di seguito riportata.

Equazione 11 - Espressione della probabilità secondo la distribuzione di Bolzman-Gibbs

$$p(\Delta f) = e^{-\frac{\Delta f}{T_i}} \qquad \Delta f = f(x^{(k)}) - f(x^{oc}) \tag{11}$$

In particolare, si genera un numero aleatorio z uniformemente distribuito in $z \in [0,1]$ e se $p(\Delta f) > z$ si accetta $x^{(k)}$ come il nuovo punto ottimale (criterio di Metropolis). Per ogni valore di temperatura T_{i} si effettuano al massimo L di tentativi di ricerca di un nuovo punto ottimale, dopodiché si riduce la temperatura;

- Decrementare la temperatura secondo l'equazione (12) quando la variazione del valore della funzione $f(x^{oc})$ è inferiore a uno specificato valore δ nelle ultime J iterazioni consecutive (situazione di equilibrio termico) o quando è stato raggiunto il numero massimo L di tentativi di ricerca di un nuovo punto ottimale a una certa temperatura.

Equazione 12 - Legge di decremento della temperatura

$$T_{\mathrm{i}} = T_{\mathrm{i-1}} * \left(\frac{\mathrm{Tf}}{\mathrm{Ts}}\right)^{\frac{1}{\mathrm{T_{Adj}}}} \tag{12}$$

I parametri T_S, T_f , T_{Adj} rappresentano rispettivamente la temperatura iniziale, la temperatura finale e il numero degli aggiustamenti della temperatura del processo di *simulated annealing*. Questi parametri, definiti in fase di inizializzazione dell'algoritmo, ne condizionano la capacità di convergenza.

Il Corana Simulated Annealing (CSA) è una variante del SA e prevede un'ulteriore equazione per determinare il vettore dei punti casuali $\boldsymbol{x}^{(k)}$ da esplorare nello spazio delle soluzioni.

L'obiettivo è garantire un equilibrio tra il numero di punti $\boldsymbol{x}^{(k)}$ accettati e quelli scartati in base alla funzione di distribuzione di Boltzman. I nuovi punti $\boldsymbol{x}^{(k)}$ sono estratti nell'intorno della migliore posizione corrente $\boldsymbol{x}^{(oc)}$ per la funzione di "costo" utilizzando l'equazione (13) di seguito riportata.

Equazione 13 - Equazione di movimento casuale per determinare le nuove posizioni nello spazio delle soluzioni.

$$\boldsymbol{x}' = \boldsymbol{x}^{oc} + \mathrm{r} * \mathrm{v}_{r,\mathrm{h}} * \mathbf{e}_{\mathbf{h}} \tag{13}$$

dove:

- r è un numero aleatorio con $r \in [-1,1]$
- $\boldsymbol{e}_h$ è il versore della *h*-esima direzione dello spazio di ricerca;
- $v_{r,h}$ è la componente *h*-esima del vettore dei movimenti **v** nella direzione *h* , al passo *r-esimo,* con $r = 1,2,..N_m$

Il vettore di movimento **v** presente nella (12) viene modificato al termine di ogni ciclo relativo ad una certa temperatura T_{m} secondo l'equazione (14).

Equazione 14 - Equazione di aggiornamento del vettore dei movimenti.

$$\begin{cases} v_h' = v_{m_h} * \left(1 + 2 * \dfrac{n_h/N_s - 0.6}{0.4}\right) & se\ \ n_h > 0.6N_s \\ v_h' = \dfrac{\mathrm{v}_{\mathrm{m_h}}}{1 + 2 * \dfrac{0.4 - n_h/N_s}{0.4}} & se\ \ n_h < 0.4N_s \\ v_h' = v_{m_h} & altrimenti \end{cases} \tag{14}$$

dove:

- v_h' è la componente di **v** nella direzione *h*;
- n_{h} è il numero di modifiche effettuate per la componente *h*.

In definitiva, nell'inizializzazione del processo di ottimizzazione CSA bisogna definire i parametri:

- T_S – la temperatura iniziale;
- T_f – la temperatura finale;
- T_{Adj} – il numero di passi di aggiornamento della temperatura;
- N_s – il fattore di accettazione delle modifiche del vettore di movimento **v**;
- N_m – il numero massimo di spostamenti casuali da effettuare

La scelta del valore di questi parametri determina il numero massimo delle valutazioni della funzione di costo $f(\cdot)$ che è pari a $N_S \cdot N_m \cdot T_{Adj} \cdot D$, con $D = N$ dimensione dello spazio delle soluzioni.

L'algoritmo CSA coniuga essenzialmente due comportamenti:

- alle alte temperature, il metodo è simile al "*random search"* con salti notevoli nello spazio di ricerca;
- alle basse temperature, il metodo è simile all'algoritmo di "*steepest descent"* con una capacità di esplorare in dettaglio un'area limitata del dominio delle variabili.

6 – Piattaforma GPU

L'ambiente di sviluppo in cui implementare gli algoritmi euristici per la massimizzazione della copertura delle reti di broadcasting SFN è stata pilotata dalla necessità di avere uno strumento di prototipazione rapido che consentisse di:

- sfruttare contemporaneamente le potenzialità di parallelizzazione offerte dalle CPU multicore e quelle degli streaming multiprocessore clusters della GPU;
- minimizzare i tempi di calcolo della copertura per una fissata combinazione dei ritardi di trasmissione artificiali $(\vartheta_1, \vartheta_2, \dots, \vartheta_N)$ in considerazione delle migliaia di trasmettitori che costituiscono una rete dispiegata sul territorio nazionale e del corrispondente numero elevato di pixel.
- utilizzare una ampia gamma di librerie di analisi e visualizzazione dei dati parziali del processo di ottimizzazione.

Tali esigenze hanno indotto a selezionare come ambiente di implementazione prototipale il *Jupiter Notebook* con linguaggio *Python 3*. La scelta del compilatore *just-in-time Numba* consente di avvalersi delle funzionalità offerte dalle librerie NVIDIA CUDA per il calcolo accelerato della copertura su GPU e per la traduzione delle funzioni Python in codice macchina ottimizzato di tipo C++ LLVM.

Il sistema di calcolo per l'ottimizzazione dei ritardi finalizzata alla massimizzazione della copertura delle reti di broadcasting SFN si compone di tre moduli:

- il *modulo di SetUp* per inizializzare i parametri dell'algoritmo (GPSO o CSA) di ottimizzazione e per specificare i files dei dati dei campi radioelettrici della rete SFN pianificata. Nel caso di SFN per il broadcasting televisivo, per ogni pixel Q_j del territorio, i dati radioelettrici sono i valori di campo elettromagnetico generati da ciascun sito trasmissivo, non superati per il 50% del tempo e il 10% del tempo ;
- Il *modulo di pre-processing* per trasformare i dati in strutture di tipo matriciale (*CUDA Data Structures*) atte ad effettuare il calcolo parallelo e concorrente su GPU delle potenze del segnale utile ed interferente in ogni pixel Q_j usando il metodo k-LMN dell'EBU per comporre i contributi di potenza dei singoli trasmettitori;
- Il *modulo di ottimizzazione* per eseguire l'algoritmo di ottimizzazione euristica che massimizza la copertura del servizio della rete SFN eseguendo su GPU una *kernel function* di calcolo del $\frac{C}{I+N}$ in ogni pixel per ogni istanza del vettore dei ritardi artificiali $\boldsymbol{\vartheta}$.

7 – Risultati sperimentali

Questa sezione presenta i risultati sperimentali ottenuti applicando le strategie di ottimizzazione euristica CSA e GPSO al broadcasting televisivo DVB-T2 (con *Pilot Pattern* PP4, 256 QAM, FEC di 2/3, $T_g = \frac{1}{16}T_u = 224\mu s$, $T_p = 266\mu s$) e quando si assume in ricezione come strategia di sincronizzazione secondaria "*Quasi Optimal C* ".

I test sono stati svolti su delle reti di riferimento, precisamente:

- una rete locale di primo livello [21] che utilizza la frequenza portante di 578 MHz (canale 34 dello spettro UHF) su un'area costituita dalle provincie di Alessandria, Bergamo,

Biella, Brescia, Como, Cremona, Lecco, Lodi, Monza e Brianza, Milano, Novara, Piacenza, Pavia, Sondrio, Varese, Verbano-Cusio-Ossola, Vercelli. Tale rete in seguito sarà denominata con l'acronimo *rete AT03_l1* ed è dotata di 57 siti trasmissivi. Il territorio su cui è pianificata la rete è formato da 6795 pixel di 2km x 2km. La strategia di sincronizzazione secondaria in ricezione è il *" Quasi Optimal C* "mentre l'*Effective Protection Threshold* (EPT) è pari a 20dB.

- una rete locale di primo livello che utilizza la frequenza portante di 634 Mhz (canale 41 dello spettro UHF) per l'area costituita dalle provincie di Arezzo, Firenze, Grosseto, Livorno, Lucca, Massa-Carrara, Pisa, Pistoia, Prato, Siena. Questa rete nel seguito sarà denominata con l'acronimo rete *AT09_l1* e dispone di 54 siti trasmissivi per illuminare 4362 pixel di 2km x 2km. In ricezione si adotta il "*Quasi Optimal C"* come strategia di sincronizzazione e un EPT =22dB.

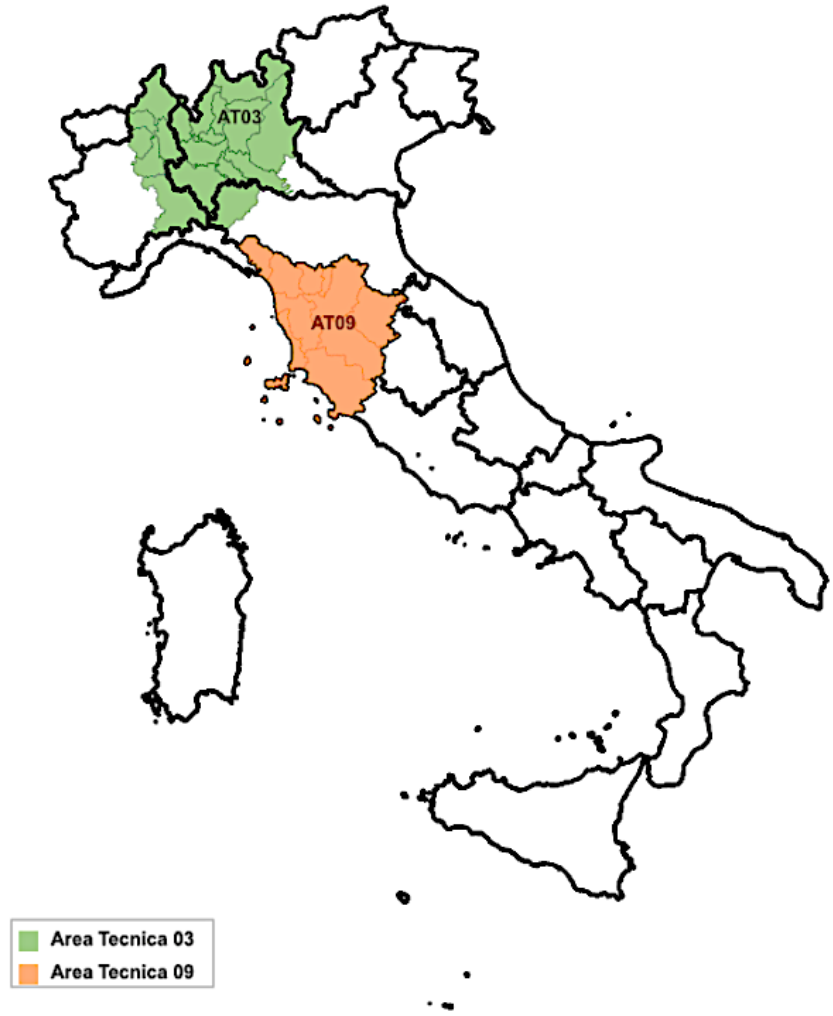

Figura 3 – *Rappresentazione grafica delle Aree servite dalle reti AT03_l1_2 e AT09_l1_1*

Per ciascuna di queste reti i dati forniti in ingresso alla piattaforma di ottimizzazione sono:

- le specifiche tecniche dei siti trasmissivi della rete SFN DVB-T2; gli impianti possono avere polarizzazione verticale, orizzontale o mista e sono identificati da un codice alfanumerico (*Id-impianto*) tale che la coppia (*Id-impianto*, polarizzazione) costituisca un identificativo univoco;
- i dati ISTAT della popolazione residente in ciascun pixel della griglia di mappatura;
- i risultati delle simulazioni radioelettriche per ciascun pixel, ovvero i valori dei campi elettromagnetici (EM) ricevuti al 50% del tempo e al 10% del tempo nel centro di ciascun pixel della griglia di mappatura del territorio su cui si pianifica la rete. Questi campi, irradiati da ciascuno dei trasmettitori DVB-T2 e successivamente discriminati secondo il puntamento dell'antenna ricevente, sono stati ricavati a partire dal modello di propagazione ITU-R P.1812-4 [22], impiegando nel calcolo del profilo altimetrico diversi modelli DEM (Data Elevation Model) che stimano l'elevazione nei punti di verifica a partire dai dati del database NASA SRTM 90 metri v3 [23].

Nello specifico i campi EM ricevuti nel centro dei pixel sono stati calcolati stimando l'elevazione del terreno al centro del pixel di 2km x2km con i seguenti modelli DEM:

- SRTM3 – determina l'elevazione a patire dai punti del SRTM 90 v3 assumendo un passo di campionamento di 250 metri;
- i DEM CSpline, Bilinear, Nearest e Max che adottano gli omonimi interpolatori per calcolare l'elevazione del terreno al centro dei pixel a partire dai valori dei punti del SRTM 90 v3 con passo di campionamento di 250 metri;

La possibilità di utilizzare diversi modelli di interpolazione DEM per valutare il profilo altimetrico del territorio nel computo dei campi EM generati da ciascun sito trasmissivo ha consentito di valutare le prestazioni di copertura ottenibili con l'ottimizzazione dei ritardi in diversi contesti operativi.

I risultati sperimentali ottenuti sono presentati nelle tabelle 1 e 2. L'analisi di dati evidenzia un'equivalenza nella capacità di ottimizzazione dei due algoritmi GPSO e CSA sebbene l'esecuzione delle due tecniche abbia tempi di convergenza diversi. Su un server dotato di una CPU Intel i5-8600K CPU @ 3.60GHz - 6 Cores, 32 Gbyte di RAM e una GPU NVIDIA 1080 Ti, il GPSO, per ottimizzare i ritardi di trasmissione della rete AT03_l1, impiega circa 2h:15m e 1h:50m per la rete AT09_l1, mentre il CSA impiega rispettivamente 4h:31 per la rete AT03_l1 e 3h:28m per la rete AT09_l1.

Tabella 1 - Copertura del servizio rete SFN AT03_l1 con EPT= 20 dB.

DEM	Copertura in assenza di ritardi artificiali	Copertura in presenza di ritardi artificiali forniti da GPSO	Copertura in presenza di ritardi artificiali forniti da CSA
SRTM3	91,69 %	94,75 %	94,80%
CSPLINE	83,98 %	94,58%	94,60 %
BILINEAR	86,33 %	94,30 %	94,31 %
NEAREST	91,52 %	94,71%	94,76%

Tabella 2 - Copertura del servizio rete SFN AT09_l1 con EPT= 22 dB.

DEM	Copertura in assenza di ritardi artificiali	Copertura in presenza di ritardi artificiali forniti da GPSO	Copertura in presenza di ritardi artificiali forniti da CSA
SRTM3	89,63 %	92,50 %	92,40%
CSPLINE	93,77 %	95,99%	96,01 %
BILINEAR	93,46 %	95,51 %	95,52 %
NEAREST	90,20 %	92,96%	93,06%

8 – Conclusioni

L'articolo propone una soluzione per l'ottimizzazione euristica delle coperture di servizio di una rete SFN, con l'adozione di ritardi artificiali nella trasmissione del segnale COFDM basata sull'uso delle GPU. Utilizzando un paradigma di programmazione parallela è possibile sfruttare la capacità computazionale, offerta dagli *array* di processori delle moderne GPU, per elaborare rapidamente i dati radioelettrici di reti SFN su larga scala, valutandone le prestazioni di copertura in tempi inferiori alla decina di millisecondi al variare degli scenari e delle condizioni operative. Conseguentemente è possibile comprimere di 30 volte i tempi di convergenza degli algoritmi euristici di ottimizzazione dei ritardi artificiali e determinare i valori che migliorano la copertura del servizio di una rete SFN a prescindere dalla scelta operata sul DEM per il calcolo dei valori di campo elettromagnetico. I test computazionali effettuati su reti locali di primo livello mostrano un incremento della percentuale di copertura di almeno il 2,2%.

9 – Bibliografia

[1] "Commission implementing decision (EU) 2016/687 of 28 april 2016 on the harmonisation of the 694-790 MHz frequency band for terrestrial systems capable of providing wireless broadband electronic communications services and for flexible national use in the Union," 2012.

[2] GPU parallel computing: Programming language, debugging tools and data structures, 2012. Front. Electr. Electron. Eng. 7, 5–15 (2012).

[3] A. Aldahlawi, Y. Kim and K. K. Kim, "GPU Architecture Optimization For Mobile Computing", ISOCC-2019 International SoC Design Conference, pp. 247-248, 2019.

[4] A. Corana, M. Marchesi, C. Martini, S. Ridella, "Minimizing Multimodal Functions of Continuous Variables with the 'Simulated Annealing' Algorithm", ACM Transactions on Mathematical Software , vol. 13, Issue 3, pp 262-280 1989.

[5] NVIDIA Ampere Architecture In-Depth, https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

[6] J. Zhu, H. Yao, T. Yang, Q. Zhou and K. Li, "A new method to parallel implementation for batching vast small-scale computing tasks based on GPU", 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA), 2016.

[7] G. Barlas, "Multicore and Gpu Programming: An Integrated Approach", Publisher: Morgan Kaufmann, 2014.

[8] https://www.openacc.org

[9] https://www.openmp.org

[10] https://developer.nvidia.com/cuda-zone

[11] J. Cheng, M. Grossman, T. McKercher, "Professional CUDA C Programming", Wiley, 2014.

[12] EBU, "SFN frequency planning and network implementation with regard to t-dab and dvb-t", Geneva (CH), 2013.

[13] R. Brugger, D. Hemingway, "OFDM receivers – Impact on Coverage of Inter-Symbol Interference and FFT window positioning", EBU Technical Review, July 2003

[14] R. S. Mannino, "Time Offset Optimization, in Digital Broadcasting", Discrete Applied Mathematics (Elsevier), vol. 156, Issue 3, pp. 339-351, 2008.

[15] C. Vercellis, "Ottimizzazione: teoria, metodi,applicazioni",McGraw-Hill,2014

[16] R. Poli., J. Kennedy, T. Blackwell, "Particle swarm optimization". Swarm Intelligence 1, 33–57, 2007, https://doi.org/10.1007/s11721-007-0002-0

[17] J. Kennedy, R. Eberhart, "Particle Swarm Optimization" Proc. of IEEE international Conference on Neural Networks, pp. 1942-1949, vol.4 ,1995.

[18] ESA, "Particle Swarm Optimization Generational", https://esa.github.io/pagmo2/docs/cpp/algorithms/pso_gen.html

[19] J. Kennedy, R. Mendes, "Neighborhood topologies in fully informed and best of neighborhood particle swarms", in Proceedings of the IEEE International Workshop on Soft Computing in Industrial Applications, 2003, pp. 45–50.

[20] S. Kirkpatrick, C. D. Gela, M. P. Vecchi, "Optimization by simulated annealing". Science 220, 671-680, 1983.

[21] AGCOM "DELIBERA N. 39/19/CONS - Piano nazionale di assegnazione delle frequenze da destinare al servizio televisivo digitale terrestre (PNAF),feb. 2019"

[22] Recommendation ITU-R P.1812-4. A path-specific propagation prediction method for point-to-area terrestrial services in the VHF and UHF bands, 07/2015

[23] SRTM3 v3 - NASA Shuttle Radar Topography Mission (SRTM) Version 3.0, 2014 https://www2.jpl.nasa.gov/srtm/