Alma Mater Studiorum · Università di Bologna

DIPARTIMENTO DI **INFORMATICA - SCIENZA E INGEGNERIA**
Laurea Magistrale in **Ingegneria Informatica**
Tesi Magistrale in **Intelligent Systems**

Ottimizzazione offline/online in condizioni di incertezza con applicazione su Virtual Power Plant

Relatrice:
Chiar.ma Prof.ssa
MILANO MICHELA

Candidata:
SIMONI CHIARA

Correlatrice:
Dr.ssa
DE FILIPPO ALLEGRA

Anno Accademico 2018-2019

Indice

Introduzione

L'utilizzo efficiente di energia nella produzione industriale ha avuto un grande impatto sui costi operativi e gestionali a causa del prezzo altamente variabile dell'energia. Mentre l'uso dei combustibili fossili per la produzione energetica e la cogenerazione sono previsti in aumento nei prossimi 20 anni [1], diventa necessario individuare metodi più efficaci per gestire le centrali esistenti che si basano sull'utilizzo di combustibili fossili come loro sorgente primaria e introdurre soluzioni alternative.

I sistemi energetici sono, nel corso del tempo, mutati da strutture convenzionali a strutture di generazione decentralizzata. Nei sistemi convenzionali l'energia è trasmessa dalle unità centrali, in cui è generata, ai sistemi di distribuzione che poi la convogliano ai consumatori. Il flusso di energia è uni-direzionale, e la struttura viene definita "top-down".
Gli svantaggi di questa soluzione sono determinati dal fatto che l'energia prodotta deve essere trasportata per lunghe distanze dalla generazione ai punti di consumo, che sono connessi tramite infrastrutture costose e non garantendo l'assenza di perdite di energia.

Si parla dunque di Distributed Generation (DG), ovvero una soluzione distribuita, non più centralizzata attorno ad un unico nucleo, basata su risorse rinnovabili. L'idea di fondo è quella di utilizzare nuclei di dimensioni più ridotte, distribuiti geograficamente, e istallati vicino a dei carichi di potenza.
In alcuni paesi europei, come la Germania, lo sviluppo di DG e RES (Renewable Energy Sources) è stato supportato dal governo attraverso leggi ad-hoc. Provvedimenti come questo favoriscono il supporto a soluzioni innovative analoghe, che stanno cambiando radicalmente la concezione dei sistemi di approvvigionamento nazionali. Vogliono innanzitutto, favorire le energie pulite e alternative effettive al consumo di combustibili fossili.

Tra le soluzioni figura il concetto di VPP, Virtual Power Plant, che consiste nella possibilità di aggregare risorse energetiche di diverso tipo: rinnovabili, batterie, termiche, come fossero una unica unità operativa. Il funzionamento del VPP è gestito dall'EMS, l'Energy Management System, che si occupa di decidere come gestire il flusso energetico all'interno dell'infrastruttura.
Basandosi sul concetto di VPP, questa tesi propone un metodo di ottimizzazione per un modello a componenti; modello che sarà poi da applicare alla fase decisionale dell'EMS per garantire il funzionamento ottimale del VPP.
La prima fase implementa un approccio robusto basato su scenari che cerca di minimizzare il valore della funzione obiettivo per ogni istante temporale. Gli scenari sono lo strumento tramite cui si rappresentano possibili realizzazioni di situazioni in condizioni di incertezza, e che considerano dei valori stimati per gli elementi energetici a cui viene associata l'incertezza stessa. Al termine della fase offline si ottengono dei flussi energetici e in particolare i valori ottimizzati della domanda di carico, che sarà inserita nella computazione della fase successiva, del giorno dopo. La fase online rappresenta un approccio greedy che considera invece non più stime, ma gli effettivi valori delle variabili energetiche e minimizza il valore della funzione obiettivo, ad esempio il costo operazionale.
Una volta elaborato il modello matematico, è stata implementata un'interfaccia web con cui selezionare e comporre diverse configurazioni di VPP, andando a realizzare un insieme di unità da ottimizzare, considerando che ciascun elemento sia parte di una rete e interconnesso agli altri. Tramite l'interfaccia, è possibile dunque personalizzare il processo di ottimizzazione scegliendo i componenti, lo step (offline, online), se e dove applicare l'incertezza (carico, fotovoltaico), quale funzione obiettivo utilizzare. Tali scelte determinano risoluzioni diverse del modello elaborato, e risulta interessante osservare come il comportamento dei VPP si adatta alle configurazioni selezionate e quanta flessibilità si riesce a offrire verso il mercato. Il procedimento complessivo simula la fase decisionale dell'EMS nella gestione del VPP. Si può dunque parlare di aggregatore, ovvero di un sistema che non elabora un modello locale limitato al singolo VPP, ma che considera più unità che interagiscono tra loro, e tramite il processo di ottimizzazione, definisce per ogni istante temporale le variabili di flusso.

La tesi sarà strutturata nel modo seguente: verrà presentato il contesto in cui questa tesi si colloca e lo stato dell'arte dell'ottimizzazione di

modelli energetici distribuiti, mettendo in luce i diversi approcci e analizzando limiti e vantaggi. A seguire sarà presentata la descrizione del modello matematico, esaminando componente per componente i vincoli necessari per il processo di ottimizzazione ed esponendo le motivazioni di alcune scelte implementative. Analogamente sarà presentata l'interfaccia e la sua struttura, i componenti, e anche in questo caso le scelte che sono state effettuate per rendere la soluzione scalabile e fault-tolerant. Una volta delineati i due componenti principali di questa tesi, saranno esposti i casi di studio presi in considerazione, per poi analizzare ciascuno di essi e trarre dunque approfondite conclusioni sullo studio effettuato.

1 Contesto

1.1 Sistemi Energetici Distribuiti

Al giorno d'oggi le unità di generazione distribuita sono installate prevalentemente a basso e medio voltaggio.
Tuttavia ci sono comunque alcuni problemi da affrontare affidandosi a sistemi di produzione energetica rinnovabili: il cambiamento di flusso dell'energia, il bilanciamento della generazione e del consumo e la frequenza di controllo dell'intera struttura.
Un altro aspetto da considerare è un modus operandi molto diffuso di alcune centrali, chiamato "feed it and forget it" [2]: consiste nella produzione di energia rinnovabile e la seguente immissione sulla rete, non in base all'attuale necessità, ma in base alla disponibilità effettiva, dipendente, ad esempio, dalle condizioni meteorologiche.
Questo modo di agire consegue la necessità di dover bilanciare la richiesta con la generazione non costante e imprevedibile della centrale tramite sistemi di stoccaggio dell'energia particolarmente grandi, o con l'utilizzo di centrali convenzionali.
Per evitare di dover ricorrere a tali soluzioni, si cerca di individuare nuove alternative come sistemi di stoccaggio dell'energia (ESS) o programmi di gestione della domanda, i DMS.

1.2 Virtual Power Plant (VPP)

Al centro della generazione distribuita si trova il concetto di VPP, Virtual Power Plant, ovvero un cluster di unità energetiche distribuite chiamate DER, carichi di potenza e sistemi di batterie, aggregati in modo da operare come una singola unità. Gli elementi possono essere controllabili o non controllabili, se si ha controllo o meno sulla loro gestione. Come si può osservare nella Fig. 1.1, le abitazioni sono elementi non controllabili, perché non è possibile regolare la domanda di energia derivante da essi o

altri parametri per la loro gestione.
Il vantaggio di utilizzare una struttura a VPP è quello di poter fornire energia nel momenti di picco della domanda anche se la richiesta arriva con poco preavviso: ciò risulta possibile grazie alle risorse di produzione interna e, se c'è un eccesso di energia prodotto, questo viene immagazzinato nelle batterie e mantenuto finché non viene eventualmente richiesto. In tale maniera è possibile dunque soddisfare anche domande più alte del carico effettivamente disponibile in un determinato momento proprio perché si utilizza l'eccesso avanzato in altri momenti della giornata. Il risultato finale è quello di un'infrastruttura flessibile ed efficiente, la cui ottimizzazione richiede tuttavia complicate operazioni di controllo, comunicazione e gestione.

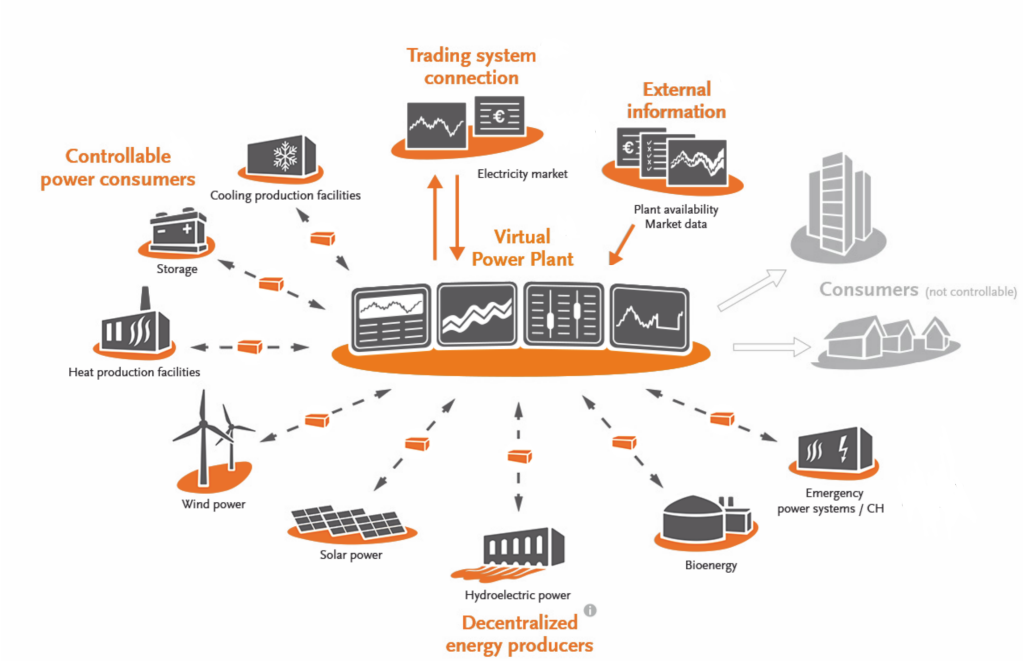

Figura 1.1: Architettura di un VPP - credits: energymeteo.com

Il cuore del VPP è l' Energy Management System (EMS), che coordina il flusso di energia a partire dai generatori, dai carichi e dai sistemi di batteria.
La particolarità di un sistema VPP è che ogni unità è connessa all'EMS, e la comunicazione avviene in entrambe le direzioni, in modo che il VPP non solo riceva informazioni relative allo stato corrente di ogni unità, ma possa anche inviare segnali per la gestione dei componenti.

Gli obiettivi dell'EMS possono variare in base alle esigenze. Ad esempio si può cercare di raggiungere la minimizzazione dei costi di generazione, della produzione di GHG (green house gases) o la massimizzazione dei profitti. L'EMS ha il compito di decidere quanta energia deve essere prodotta, quali generatori contribuiscono alla produzione richiesta, se l'energia in eccesso deve essere venduta o immagazzinata; se i carichi controllabili sono da staccare o se serve acquistare energia dal mercato.

Quando risulta necessario un consumo sia di energia termica sia di energia elettrica, si considerano sistemi combinati chiamati CHP, ovvero "Combined Heat and Power", che a loro volta sono presenti in varie forme e potenza di produzione. CHP è una tecnologia che riduce il consumo totale di carburante e le relative emissioni di gas serra, producendo sia elettricità sia utile energia termica a partire da una singola sorgente energetica [3].

Diventa fondamentale non solo conoscere lo stato di ogni elemento di produzione o consumo, ma anche eseguire predizioni nel caso di unità energetiche rinnovabili come fotovoltaico o eolico. A causa della natura mutevole delle risorse rinnovabili, la previsione della produzione di energia risulta alquanto complessa.
Uno degli aspetti innovativi del VPP è l'apertura al mercato energetico: tramite comunicazione diretta, la produzione in eccesso o altri servizi, possono essere messi in vendita ad utenti e consumatori. Analogamente, in caso di necessità, ad esempio se il sistema non riesce a soddisfare il fabbisogno energetico degli elementi del VPP, si può acquistare energia dalla rete.
I servizi che un VPP può mettere a disposizione possono essere relativi alla rete sottostante, per mantenerne l'equilibrio e la stabilità della rete stessa: tra i servizi offerti ci sono ad esempio la regolazione della frequenza o l'inseguimento del carico (fornendo cioè una quantità di energia necessaria per soddisfare la volubile domanda di carico).

Un approccio interessante che si pone come alternativa al metodo "feed it and forget it" è presentato da [4], metodo in cui i DER sono aggregati in VPP e gestiti in maniera controllabile. Basandosi sul concetto di VPP, i DER possono accedere al mercato energetico per ottimizzare e massimizzare le opportunità di profitto; il sistema usa tutta la capacità energetica disponibile per essere sempre più efficiente. L'approccio presentato esplora ulteriormente il concetto di VPP e di come può par-

tecipare in maniera attiva nella compravendita di energia, e gestire le attività interne, considerando due casi d'uso: il primo in cui si valuta l'errore potenziale derivato dallo squilibrio causato dall'interazione con il mercato, mentre il secondo utilizza un algoritmo di flusso dell'energia per definire il VPP.

L'analisi di Awerbuch [33] mostra ad esempio l'utilità di introdurre il concetto di DER in ambito commerciale o nei mercati elettrici e come possa creare opportunità per decentralizzare il ruolo delle tradizionali infrastrutture di produzione. In [2] invece, si analizza in maniera molto dettagliata la relazione tra la capacità dei sistemi di stoccaggio, la quantità di energia prodotta tramite RES e la quantità di carico di generazione controllabile. Nella trattazione sono esposte diverse tipologie di sistemi per lo stoccaggio di energia e si delinea un modello per l'ottimizzazione della gestione della produzione energetica. Tale modello presenta un problema mixed-integer che va a minimizzare la sommatoria dei costi di produzione e introduce dei metodi di riduzione dei picchi di carico, considerando anche i costi di carica e scarica delle batterie, i costi di investimento e di accensione o spegnimento dei generatori.

Le analisi [5,6] esplorano invece dei modelli non lineari per l'ottimizzazione della gestione del VPP, introducendo il concetto di una rete con cui il sistema interagisce e metodi per lo stoccaggio di energia termica. La funzione dei costi considerata è in questo caso quadratica, e si focalizza sulla gestione desiderata di produzione elettrica e termica, o altre risorse se disponibili, da parte del DSO (Distributed System Operator), che deve sempre soddisfare sia la potenza reattiva sia la potenza attiva.

1.3 Smart Grid e Microgrid

Dal momento che questi servizi sono spesso forniti tramite generatori a combustibili fossili, le infrastrutture che si basano in maggioranza su fonti rinnovabili hanno bisogno di altre forme di generazione o consumo controllabili.
In questo contesto, risulta fondamentale introdurre un ulteriore concetto, quello di Smart Grid. Una Smart Grid è una rete energetica che può integrare in maniera economicamente efficiente il comportamento e le azioni di tutti gli utenti connessi, siano essi generatori, consumatori o utenti che agiscono in entrambi i sensi. La smart grid è una rete moderna che utilizza comunicazioni avanzate e controlla tecnologie per generare e

distribuire elettricità. Questa infrastruttura rappresenta la spina dorsale del VPP e richiede una importante quantità di sistemi di stoccaggio di energia elettrica. I veicoli elettrici e le possibilità di immagazzinare energia che offrono, costituiscono un interessante soluzione tecnica per implementare batterie distribuite e applicare modelli simili al V2G.

L'approccio proposto da Palma-Behnke in [34] considera un sistema di EMS basato sulla strategia Rolling Horizon per microgrid incentrate sull'utilizzo di risorse rinnovabili.
Una microgrid è un gruppo di risorse energetiche e carichi che solitamente opera in connessione e in maniera sincrona ad una rete più ampia, ma può anche essere disconnessa in modalità "isola", e funzionare autonomamente. L'infrastruttura è su piccola scala rispetto agli approcci centralizzati o più evoluti come le smart grid. La strategia Rolling Horizon si affida alla risoluzione di una sequenza di sotto-problemi di lunghezza definita e potrebbe non essere applicabile a problemi in cui le condizioni devono essere soddisfatte lungo un intero orizzonte temporale. L'analisi di Palma-Behnke utilizza una previsione di due giorni per le risorse rinnovabili e per il consumo elettrico basato su reti neurali, in più l'EMS integra uno schema DSM per i consumatori; l'incertezza non è tuttavia considerata esplicitamente.

1.4 Approcci e modelli matematici usati in letteratura

Solitamente un sistema distribuito è basato sul concetto "produzione locale per consumo locale" [8], dovuto al fatto che la produzione locale è su piccola scala, e i carichi che mutano frequentemente nel tempo e la variazione di domanda energetica rendono il bilanciamento della domanda e richiesta una sfida complessa.
Per questa ragione i sistemi di energia distribuiti potrebbero non essere in grado di produrre i potenziali benefici a causa della mancanza di sistemi di configurazione di strategie operative adeguate.

Dunque, quando si modellano differenti realtà fisiche per risolvere un problema comune, deve essere prestata grande attenzione al livello di astrazione del modello. Da un punto di vista fisico, un certo livello di complessità è necessario per produrre risultati attendibili. Tuttavia, è anche necessario mantenere un livello di astrazione tale da poter rappre-

sentare unità per cui i dettagli fisici non sono disponibili o conosciuti, come accade nella maggior parte dei casi.
Dal momento che il VPP non può contrattare per la compravendita di energia, si suppone che le decisioni vengano prese in base alle variabili di costo, paragonate al costo orario assegnato alla rete di distribuzione, per soddisfare sia le richieste di energia elettrica sia termica.

Diversi modelli matematici sono stati proposti per sopperire a questa lacuna, modelli che usano diverse tecniche di programmazione matematiche come mixed-integer programming (MIP) e multi-objective programming (MOP) [9,10]. La maggior parte dei modelli si concentra sui sistemi CHP, che risulta rappresentativo tra i sistemi di generazione distribuiti, mentre alcuni modelli indagano situazioni più complesse con risorse rinnovabili. Un aspetto comune della maggior parte di questi approcci è il fatto di formulare i modelli senza tenere conto dell'incertezza derivata dalle risorse rinnovabili. Tuttavia, se tale aspetto non è indirizzato correttamente, l'attuale realizzazione potrebbe deviare da quella ottima.
Per suddetta ragione, la modellazione dell'incertezza in sistemi energetici distribuiti ha ricevuto negli ultimi tempi grande attenzione.
Sono stati proposti modelli di un metodo di dimensionamento ottimale per sistemi di cogenerazione che prendono in considerazione l'incertezza della domanda (Gamou et al. [11]), oppure sono state condotte analisi sull'incertezza causata dalla perfomance fluttuante di alcuni strumenti [12], costi energetici e possibile declino nell'utilizzo di tali strumenti (Yoshida et al. [13]).

Altre analisi sono state condotte e il quadro finale ha evidenziato come l'incertezza può essere applicata a diversi fattori: nella modellazione dei sistemi energetici distribuiti, nella richiesta ed offerta di energia elettrica, relativamente ai parametri economici come le unità di investimento e il costo dell'energia, o ancora in parametri tecnologici come l'efficienza delle batterie.
In molte delle citate analisi, l'attenzione è concentrata, relativamente alla pianificazione dei sistemi energetici distribuiti, sull'incertezza della domanda energetica.
Tale incertezza deriva dalla descrizione semplificata dei pattern di richiesta energetica, che viene delineata usando profili rappresentativi distribuiti su 24 ore, riducendo quindi la dimensione del problema di ottimizzazione per favorire l'efficienza computazionale. Nonostante tutto, si deve sempre considerare un trade-off tra potenza computazione e l'accu-

ratezza dei risultati del modello: molti dati ad esempio, vengono persi a causa della scelta di giornate rappresentative. Molte informazioni sono così automaticamente escluse, causando il calo dell'accuratezza del modello. Gli studi indipendenti di Mavrotas et al., e Hawkes e Leach [14,15] hanno evidenziato come l'aggregazione dei dati relativi alla richiesta di carico mettano a rischio l'accuratezza dei risultati dell'ottimizzazione, implicando che considerare la volatilità della domanda energetica sia di vitale importanza nella modellazione di sistemi energetici distribuiti.

1.4.1 Unit Commitment

L'obiettivo primario nella gestione di sistemi elettrici, distribuiti e non, è poter soddisfare la domanda di energia in ogni istante temporale indipendentemente dalle condizioni climatiche. Per minimizzare il costo delle operazioni di risorse non controllabili, è essenziale studiare un approccio per selezionare, tramite un processo di ottimizzazione, le unità e i relativi valori di output per mantenere il sistema efficiente e affidabile.
Unit Commitment (UC), uno dei processi decisionali più critici, rappresenta il problema di ottimizzazione che genera l'output di tutti i generatori in modo da minimizzare il costo del carburante a livello dell'intero modello.
Le caratteristiche incluse nei modelli di UC più moderni comprendono limiti (massimi e minimi) di produzione dei generatori, limiti di rampe, vincoli sui tempi di accensione e spegnimento, costi di avvio dipendenti dal tempo, e limiti della capacità di trasmissione.
Durante le operazioni di routine, gli amministratori del sistema gestiscono le risorse di produzione in modo da soddisfare la domanda energetica. Nel caso di cambiamenti significativi tra le condizioni del sistema effettive e quelle stimate, l'amministratore deve poter prendere decisioni correttive come far partire dei generatori o regolare i voltaggi, o ancora abbassare i carichi, per mantenere la sicurezza del sistema. I cambiamenti improvvisi possono essere causati dall'incertezza delle stime del carico o a causa di trasmissioni inaspettate e interruzioni di produzione.

Tradizionalmente, la formulazione deterministica di UC è la soluzione in cui il carico della rete è modellato usando una singola previsione per ogni risorsa rinnovabile, e l'incertezza associata è gestita usando delle regole ad-hoc. In pratica, questo approccio è semplice da implementare, ma la gestione corretta dell'incertezza dipende dalla modellazione delle regole. Ad esempio, dedicare delle risorse di produzione in più è economicamente inefficiente, mentre il sistema soffre di capacità limitata nel

caso in cui ci sia una differenza sostanziale tra il carico effettivo e quello stimato.

Un approccio più rigoroso, è incorporare l'incertezza nel modello stesso, come descritto da Kristina Jurkovic in [16].
Stochastic UC è un approccio basato su scenari probabilistici. Un insieme finito di scenari viene definito e a ciascuno di essi viene associato un peso, in proporzione alla probabilità che si verifichi tale situazione. Questo approccio è formulato come un problema a due stadi che determina la produzione per minimizzare il costo stimato su tutti gli scenari, in base alle loro probabilità. Le decisioni di *commitment* (decisioni che specificano come dev'essere modellato lo scenario) sono uniche su tutti gli scenari, mentre quelle di *dispatch* (ovvero quale generatore accendere, ad esempio), dipendono dallo scenario specifico.
Più alto è il numero di scenari, più il costo computazionale diventa rilevante. Un altro fattore che influisce pesantemente sul costo è l'orizzonte temporale. Quindi diventa necessario introdurre delle tecniche per ridurre il numero di scenari, eliminando quelli con probabilità molto basse e aggregando quelli simili, definiti tali in base a particolari metriche (ad esempio, la probabilità o il costo risultante).

In [17] gli autori hanno analizzato diverse tecniche per la riduzione del numero di scenari per Stocastic UC. In particolare, K-means, è un metodo di clustering (raggruppamento), che viene usato per partizionare un determinato insieme di scenari in un numero specificato di cluster. Il problema più incalzante che sorge è quello della selezione del numero di scenari: da un lato, un numero troppo basso riduce l'accuratezza delle soluzioni e aumenta il costo, dall'altro lato, un numero troppo alto diventa computazionalmente proibitivo.

In [18] invece, gli autori considerano un algoritmo per includere la variabilità della generazione rinnovabile. Si assume che quest'ultima sia soggetta ad una distribuzione normale N (μ, σ^2) con valore stimato μ e una percentuale di μ come sua variabilità (σ). Viene utilizzata la simulazione di Monte Carlo, tecnica usata per comprendere i rischi e l'incertezza legati a modelli predittivi, per generare un vasto numero di scenari soggetti a distribuzione normale.
La probabilità assegnata a ciascuno scenario è uno, diviso per il numero totale di quelli considerati. Anche in questo caso vengono utilizzate tecniche per la riduzione del numero degli scenari. L'algoritmo viene

dunque formulato come un problema di ottimizzazione con la funzione obiettivo composta dai costi del carburante e i costi legati all'accensione e allo spegnimento delle unità di produzione lungo l'orizzonte temporale. Il problema è un ampio programma mixed-integer non-linear che offre come soluzioni le indificazioni di commitment e dispatch che minimizzano il costo operativo.

Diversi modelli sono stati proposti anche per l'approccio *Robust UC*. In [19] viene descritto un modello a due stadi ad esempio: il primo stadio trova i valori ottimi delle variabili decisionali, mentre il secondo genera il peggior caso possibile, considerando il costo operazionale e usando determinati valori delle variabili di commitment trovati nella prima fase.
L'incertezza viene qui limitata da bounds sul carico della rete in ogni intervallo temporale. Il modello genera dunque la soluzione ottima realizzabile per tutti gli scenari entro i limiti specificati, minimizzando il costo più alto su tutti gli scenari considerati.
Il modello presentato invece in [20] è un robust UC a due stadi sviluppato in modo da ottenere la programmazione del giorno prima dei generatori. L'incertezza è modellata in modo da catturare la mutevole natura della risorsa rinnovabile senza una descrizione esplicita della funzione di distribuzione.

Un'ulteriore alternativa è data da *Interval UC*, ovvero la possibilità di produrre una sequenza di produzione che minimizza il costo di della più probabile previsione del carico della rete, garantendo la fattibilità in tutto l'insieme determinato dall'incertezza, limitato dai bounds superiore e inferiore, come nel caso di Robust UC. Risulta un metodo più efficiente rispetto a Stochastic UC in quanto il modello può essere descritto usando tre scenari (upper bound, lower bound, e caso centrale), e può anche essere descritto come un problema a due stadi, in cui la soluzione è trovata nella prima fase e poi testata in quella successiva.

Infine sono introdotte anche soluzioni ibridi che sfruttano i vantaggi di un modello ed eliminano gli svantaggi presentati brevemente nel paragrafo precedente. Un esempio sono modelli che unificano Stochastic e Robust UC come [21].

Riassumendo, UC è una componente fondamentale della gestione dei sistemi energetici, ancor più in sistemi in cui bisogna considerare un livello di incertezza. Tramite stime del giorno prima riferite al mercato

energetico, l'amministratore di sistema deve poter derivare una sequenza di produzione per le centrali (i.e VPP) per poter garantire la affidabilità del sistema al minor prezzo possibile.
In letteratura i due principali approcci che emergono sono Stochastic UC e Robust UC. Mentre quest'ultimo si basa sulla formulazione del caso peggiore, il fulcro della modellazione è l'affidabilità del sistema. Lo svantaggio principale è quello che tende a considerare delle programmazioni troppo conservative. Nel caso di Stochastic UC invece, si cercando delle programmazioni più efficienti dal punto di vista economico usando delle stime e degli scenari, ma entrambi gli approcci sono computazionalmente proibitivi o non riescono a garantire l'affidabilità del sistema. I modelli ibridi cercano quindi di sopperire alle lacune dei due appena menzionati.

1.4.2 Approcci matematici stocastici a due stadi

In letteratura si trovano diversi esempi di approcci e modelli matematici a due stadi stocastici, e alcuni di questi saranno esaminati prima di discutere sullo stato dell'arte dell'ottimizzazione offline-online.
Wang et al [36] propongono come soluzione per un problema di UC associato a risorse eoliche un metodo di ottimizzazione stocastico a due stadi che utilizza nella prima fase le stime dei valori dell'eolico per generare lo schedule di produzione, la pianificazione degli startup e shutdown un giorno in anticipo, e che adotta nella seconda fase un'analisi dinamica dello scenario per valutare le operazioni da eseguire. In particolare, sottolinea l'utilità del'utilizzo di *Stochastic Unit Commitment* (SUC) come ausilio per gestire l'incertezza in problemi di ottimizzazione.
L'approccio proposto si concentra su alcuni punti cardine. Innanzitutto il modello a due stadi elaborato mira alla minimizzazione del costo operazionale e cerca di coordinare nella maniera più efficiente la programmazione day-ahead con quella real-time; il modello è stato elaborato per infrastrutture con un'alta penetrazione di sistemi di produzione eolica; la generazione degli scenari basati su incertezza statistica e variabilità considera la continuità tra molteplici periodi temporali.
In letteratura emergono alcuni metodi per la generazione di scenari in situazioni di incertezza, come Latin Hypercube Sampling (LHS) e Monte Carlo Sampling, metodi che si focalizzano sulla previsione dell'incertezza assumendo che le variabili siano tra loro indipendenti e la continuità tra i periodi temporali è ignorata. Questa assunzione rischia di diventare un impedimento nel caso in cui sia invece necessario considerare una situazione di continuità temporale, ed è per questo che l'approccio di Wang et al. considera una tecnologia di generazione dinamica degli scenari basata

su una funzione di distribuzione empirica e cumulativa per caratterizzare l'incertezza della risorsa eolica, descritto in dettaglio in [37].

Maggiormente concentrato sull'interazione con la microgrid, è l'approccio descritto in [38], che propone una nuova metodologia per considerare in maniera dettagliata le necessità in termini di carico degli utenti ed evitare dunque di incorrere in picchi inaspettati che la rete non è capace di supportare. Il metodo da loro proposto considera: un modello stocastico bottom-up per la generazione di profili di domanda di carico, che serviranno in un secondo momento per la generazione di potenziali scenari di richiesta; un modello di ottimizzazione a due stadi stocastico specifico per la microgrid, che consente di dimesionare in maniera robusta la rete tenendo conto di tutti gli scenari basati sulla probabilità che avvengano. Entrambi i modelli sono open-source, per essere disponibili e utilizzabili e adattabili ad ulteriori situazioni.

Infine, un approccio per l'ottimizzazione del design di sistemi di energia distribuiti, è quello proposto da Zhou et al.[8]. Estendendo un loro precedente lavoro, considerano un modello di ottimizzazione a due stadi basato su una strategia di scomposizione in stage, considerando sia l'incertezza relativa alla domanda, sia l'incertezza delle risorse disponibili. Nella prima fase vengono utilizzati degli algoritmi genetici per effettuare una ricerca sui valori delle variabili decisionali, mentre il metodo Monte Carlo viene adattato alla seconda fase per gestire l'incertezza legata a domanda e risorse. La novità delle loro considerazioni risiede nell'aggiungere al sistema distribuito estremamente complesso con molteplici risorse energetiche già proposto nel loro precedente lavoro, un modello stocastico in cui si considerano simultaneamente le incertezze di domanda e risorse disponibili.

1.4.3 Ottimizzazione Offline-Online

Per la risoluzione di problemi in condizioni di incertezza sono stati proposti sia modelli offline "strategici", che possono essere risolti senza vincoli stretti, sia problemi online "operativi", in cui le decisioni sono vincolate da limiti temporali molto più stringenti.
Si usano dunque metodi di ottimizzazione stocastica per risolvere problemi in condizioni di incertezza per la fase offline [22], e efficienti euristiche non anticipative sono state considate invece per la fase online.
Gli algoritmi online permettono di sfruttare al meglio le informazioni aggiuntive mentre l'incertezza si rivela gradualmente. Sfruttando questo

concetto, molti lavori si sono concentrati sulla risoluzione di problemi online usando tecniche originariamente introdotte per la programmazione stocastica, ad esempio Sample Average Aproximation (SAA) [23].
SAA è un metodo per risolvere problemi di ottimizzazione stocastica usando la simulazione di Monte Carlo. La funzione obiettivo stimata del problema viene approssimata facendo una media stimata, derivata da un campionamento casuale. Da qui risulta un problema SAA che viene risolto tramite tecniche di ottimizzazione deterministiche. Il processo viene solitamente ripetuto su diversi campionamenti, in modo da ottenere possibili soluzioni e la stima dei loro intervalli di ottimalità.
Applicando tale concetto alla situazione in esame, il campionamento viene effettuato sulle variabili casuali usate per modellare l'incertezza, e ottenere delle possibili realizzazioni, chiamate *scenari*.
Risolvendo i problemi tramite tecniche di ottimizzazione deterministiche costruiti su multipli scenari e calcolando delle medie, è possibile rendere l'algoritmo online in una qualche maniera anticipativa [24, 25].

L'approccio presentato in questa tesi riprende quello presentato da DeFilippo et al [35]: per l'offline si utilizza un approccio robusto per gestire l'incertezza (dunque usando dei range in cui sono definite le variabili "incerte") e supportando metodi di DMS, utilizzando i valori shiftati ottimizzati nella fase successiva, online, in cui si calcolano gli effettivi flussi energetici. L'online è rappresentato da un algoritmo greedy, che con l'ausilio dell'ottimizzazione della fase precedente, riesce a prendere decisioni maggiormente mirate e curate.
La metodologia descritta è stata poi ampliata e perfezionata in successivi lavori, in cui l'euristica greedy viene sostituita con un metodo anticipativa [24,25].

2 Il modello

Il modello matematico proposto in questa tesi è un problema Mixed Integer Linear Programming (MILP).
Un MIP o MILP è un problema in cui alcune delle variabili decisionali sono vincolare ad assumere valori interi.
La forma canonica di un problema MILP è del tipo:

$$\min c^T x$$

$$Ax = b$$

$$x \geq 0$$

$$x_i \in Z \forall i \in I$$

Dove $\min c^T x$ è la funzione obiettivo che va a minimizzare il costo associato alla variabile x, mentre le altre equazioni sono i vincoli del modello matematico (A è la matrice dei valori di x).

Il modello è a componenti, identificati con il termine POD (Point Of Delivery), rappresentati idealmente tramite l'infrastruttura del VPP. I POD possono essere di produzione o consumo in base agli elementi costitutivi: sistemi di stoccaggio, chp, fotovoltaico o carico. Viene inoltre considerata una componente legata alla rete esterna, per la compravendita dell'energia.

L'ottimizzazione avviene in due fasi: si utilizza un approccio stocastico che calcola i valori dello shifting del carico, assieme ad un'euristica online, che determina i flussi delle risorse energetiche.
L'incertezza del modello viene associata alle componenti del fotovoltaico, in quanto risorsa rinnovabile soggetta a condizioni meteorologiche, e alla domanda del carico, in quanto mutevole deve essere considerata incerta per evitare che il comportamento del VPP devi in maniera rilevante da quello ottimo. L'introduzione dell'incertezza associata a tali componenti è fondamentale per garantire il funzionamento corretto e affidabile del

sistema.
Nella fase *offline* si utilizzano delle stime effettuate sui componenti a cui è associata l'incertezza per ottenere uno schedule di tutti gli elementi di tutti i POD, avendo dunque per ogni istante di tempo considerato un valore associato alle variabili decisionali presenti nel problema. In particolare, tra i valori di output così ottenuti, si ricava la quantità di carico shiftato, ovvero spostato rispetto alla pianificazione originale in un altro istante temporale, magari in modo da ottenere profitti economici. Suddetti valori saranno necessari per la computazione del giorno successivo, la fase *online*.
Nella fase online si utilizzano i valori ottimizzati del giorno prima per individuare i valori delle variabili decisionali in relazione alla funzione obiettivo scelta, ad esempio quella per minimizzare i costi di produzione. Si suppone che i controlli sulle variabili vengano effettuati a intervalli temporali di 15 minuti: l'ottimizzazione considera, dalle 00 alle 24 di ogni giorno, un totale di 96 istanti.
Tale scelta è dettata dalla volontà di bilanciare al meglio la complessità computazionale e il livello di dettaglio dei risultati. Viene in ogni caso data la possibilità di regolare la durata degli step temporali tramite una funzione di scaling dei file di ingresso, in modo da garantire il corretto funzionamento dell'intero script, uno per ogni fase dell'ottimizzazione.

2.1 Fase Offline

Ogni elemento è identificato nel modello tramite equazioni matematiche e vincoli per individuare ad ogni istante di tempo t, qual è il valore di un determinato componente in un determinato POD.
Viene considerato, in tutta la fase offline, un singolo modello, in cui si itera sulla lista dei componenti e sugli istanti temporali.
Come già menzionato, carico e fotovoltaico sono i due elementi a cui è stata associata l'incertezza: è possibile considerarla su un singolo elemento o su entrambi. Infatti, a causa della natura mutevole dell'elemento naturale sfruttato dagli impianti fotovoltaici, i valori presi in ingresso, sono solo una stima, ma bisogna considerare che si discostino dall'effettiva disponibilità. Nel caso del carico invece, si stima quanta energia venga richiesta, ma anche questa domanda è soggetta a variabilità.

Gli scenari permettono di ipotizzare l'esistenza di alcune situazioni base, in cui la richiesta di carico o fotovoltaico sia in eccesso o in difetto rispetto alle previsioni. L'insieme degli scenari S, è quindi in tal maniera

costruito:

$$S = \{s_1, s_2, s_3, s_4\}$$

dove s_1, ad esempio, è il caso in cui il reale valore delle variabili è in eccesso rispetto alle previsioni, per cui $P_{PV}(t) = P_{PV}(t) + \delta_{PV}(t)$ e $P_{Load}(t) = P_{Load}(t) + \delta_{Load}(t)$, dove $\delta_{PV}(t)$ e $\delta_{Load}(t)$ sono le quantità di energia in più rispetto alla stima; o sia s_2 il caso in cui la potenza del fotovoltaico è in eccesso e quella del carico in difetto, quindi $P_{PV}(t) = P_{PV}(t) + \delta_{PV}(t)$ e $P_{Load}(t) = P_{Load}(t) – \delta_{Load}(t)$.
Analogamente si costruiscono i rimanenti scenari.
Nel caso in cui l'incertezza sia applicata solo ad uno dei due elementi, il numero totale di scenari si dimezza, e alla stessa maniera, se l'incertezza non viene considerata, rimane un singolo scenario base in cui il valore effettivo delle variabili sarà il valore predetto per carico e fotovoltaico.
Sono due dunque, le macro-situazioni da considerare: un caso deterministico, in cui l'incertezza è del tutto inesistente, e uno stocastico, in cui invece tale componente è inclusa.
Si assume che l'errore per la previsione della domanda di carico possa essere considerata come una variabile indipendente casuale, che ci permette di definire l'incertezza usando intervalli di confidenza. In particolare, si assume che l'errore segua una distribuzione Normale $\mathrm{N}(\theta, \sigma^2)$, e che la varianza per ogni istante temporale sia tale che il 95% dell'intervallo di confidenza corrisponda al 20% del carico stimato.
Formalmente, significa che $1.96\sigma = 0.2P_{Load}(t)$, mentre in pratica si traduce nel fatto che il parametro δ_{Load} usato per ottenere gli scenari sia uguale a $0.2P_{Load}(t)$.
Relativamente all'errore della stima del fotovoltaico, viene considerata come previsione la media oraria globale dell'irradiazione solare proposta da [26] basata sui dati registrati ed illustrati in [27].
Quindi, si assume che l'errore in ogni istante di tempo sia modellabile come una variabile casuale indipendente e che sia distribuito con distribuzione normale in modo da avere il 95% dell'intervallo di confidenza corrispondere al $\pm$ 10% del valore stimato. In pratica, il valore del parametro δ_{PV} per ogni t sarà uguale a $0.1P_{PV}(t)$.

Tutti i problemi sono stati modellati come Mixed Integer Linear Programming, e il linguaggio usato per creare il modello (sia per la fase offline, sia per la fase online) è Python, con l'ausilio di Pyomo, un linguaggio open-source per la modellazione dell'ottimizzazione, che mette a disposizione molti strumenti per creare i vincoli ed elaborare le diverse fasi del processo.

La scelta del linguaggio di programmazione da utilizzare è ricaduta su Python per la vasta disponibilità di librerie per la gestione dei dati tramite strutture apposite come *NumPy* per la manipolazione degli array, *pandas* per la manipolazione ad alto livello di dati strutturali e *Matplotlib* per la produzione di rappresentazioni grafiche di dati.
Pyomo risulta ulteriormente utile, in quanto permette di scegliere il solver da utilizzare per la risoluzione del modello.
Un *solver* è uno strumento che supporta il decision-making allocando e distribuendo le risorse di un sistema: incorporano dei potenti algoritmi per la risoluzione di modelli di programmazione matematica, constrint programming e modelli constraint-based scheduling. Esempi conosciuti di solver sono CPLEX, glpk, ipopt, e Gurobi.
Nel caso in questione, è stato scelto Gurobi, un solver molto diffuso per linear programming, quadratic programming, quadratically constrained programming, mixed integer linear programming, mixed-integer quadratic programming, and mixed-integer quadratically constrained programming.

Di seguito saranno descritti nel dettaglio le variabili e i vincoli dello step offline. Per ogni POD, vengono considerati gli elementi inclusi, e ad essi sono associate delle variabili e dei vincoli. Per ogni elemento presente in un POD, viene aggiunta l'equazione relativa all'elemento costitutivo: semplicemente, se l'elemento non è presente, non si considera il vincolo legato ad esso.

2.1.1 Sistemi di storage

Molti studi (menzionati nel capitolo precedente) sono stati condotti sull'utilizzo di batterie come sistemi di stoccaggio, e sul loro utilizzo in ambito energetico: il loro scopo principale è quello di coprire il fabbisogno dell'unità quando la produzione interna non riesce a soddisfarla.
Il modello prevede di poter utilizzare la batteria in maniera bidirezionale: sia per attingere energia e dunque scaricarla, sia per immagazzinare l'eccesso di energia e quindi caricare la batteria.
Non è stato considerato alcun caso particolare di sistemi di storage, anche se in generale l'efficienza di carica e scarica viene considerata del 90%, che è comune tra le batterie.
In questo lavoro è stato considerato sufficiente tenere in considerazione la quantità di energia immagazzinata per ogni istante di tempo, assumendo che ogni t sia sufficientemente lungo per evitare lo stress della batteria e dei livelli di degrado della batteria stessa, che potrebbero influenzarne

l'andamento. Considerazioni più elaborate sui sistemi di batteria possono essere consultate in [28].
L'equazione per ottenere la carica all'istante di t dell'elemento di stoccaggio è la seguente:

$$charge^s(t) = charge^s(t-1) - \eta * P^s_{Sin}(t) + \eta * P^s_{Sout}(t) \quad \forall t \in T \qquad (1)$$

Dove $charge^s(t)$ è la variabile che indica lo stato di caricamento della batteria all'istante t e nello scenario s, e si ricava dalla quantità di carica all'istante temporale precedente a cui si va eventualmente a sottrarre la quantità di energia convogliata verso l'esterno, per sopperire alla richiesta di energia o ad aggiungere, invece, la quantità di energia ricevuta dagli atri elementi del POD. Il valore dell'efficienza di carica o scarica della batteria è η, considerato, come detto in precedenza, del 90%.
In aggiunta sono state considerate alcune equazioni basilari per il corretto funzionamento della batteria: in particolare si assume che in ogni istante iniziale t=0, la batteria sia già parzialmente carica, e che la quantità di carica che è possibile emettere o introdurre sia soggetta a vincoli di massimo o minimo.

2.1.2 Rete esterna e Mercato

Viene assunto che il POD possa interagire con una rete esterna. Anche in questo caso, viene considerata solo la quantità di energia comprata e venduta per ogni istante di tempo: non sono stati contemplati ulteriori vincoli riguardanti la trasmissione dell'energia lungo le linee o ulteriori dettagli dell'infrastruttura di connessione, per mantenere il modello ad un livello di astrazione tale da essere adattabile a diversi casi d'uso.

Il valore che indica l'utilizzo della rete all'istante t è $P_{Grid}(t)$, identificato dalla somma dei due elementi P_{Gin} e P_{Gout}, che rappresentano rispettivamente il valore dell'energia in ingresso dalla rete, comprata dal mercato, e il valore dell'energia in uscita verso la rete, ovvero venduta ad utenti o clienti. Entrambi questi valori sono sottoposti a dei vincoli dipendenti dall'infrastruttura della rete: vengono assunti dei limiti dalla letteratura esistente [29], basati su dati reali per la capacità in entrata e uscita.

$$min_{P_{Gin}} \leq P^s_{Gin}(t) \leq max_{P_{Gin}} \quad \forall t \in T \qquad (2)$$

$$min_{P_{Gout}} \leq P^s_{Gout}(t) \leq max_{P_{Gout}} \quad \forall t \in T \qquad (3)$$

2.1.3 Demand Side Management

La domanda di energia viene modellata tramite il componente del carico. Nella fase offline vengono presi in ingresso, per ogni POD, dei valori stimati di carico, ovvero della domanda prevista, a cui andrà aggiunta la quantità "spostata", di richiesta calcolata in questa fase, chiamata *shifted load* e indicata tramite la variabile S_{Load}, definita per ogni istante temporale t e ogni scenario s.

$$\tilde{P}^{s}_{Load}(t) = P^{s}_{Load}(t) + S_{Load}(t) \quad \forall t \in T, \forall s \in S \qquad (4)$$

$\tilde{P}_{Load}$ in particolare è la variabile di bilanciamento della potenza, e assume il valore della quantità di carico definitiva, ottenuta sommando la stima e il valore ottimizzato di carico. Si assume che la quantità di carico possa essere redistribuita, mantenendo costante il consumo totale finale, in base alle esigenze, per esempio economiche, e soprattutto in base alla disponibilità della produzione interna.
Questa procedura prende il nome di *Load Shifting*, ed è molto diffusa in ambito industriale e commerciale. L'idea è quella di spostare delle quantità di carico, dai picchi ad esempio, e distribuirle in istanti temporali diversi da quelli inizialmente programmati. Ogni centrale industriale ha una quantità di margine per accettare la presenza di malfunzionamenti o manutenzione. Con la giusta tecnologia tale margine può anche essere utilizzato per controllare il load shifting e viene definito come *energy flexibility*. Introdurre il concetto di load shifting nel modello energetico significa poter bilanciare i servizi offerti dal VPP/POD, avere energia già pronta nei casi di picco, o produrla quando conviene in termini di costo. Leggermente diverso è invece il concetto di *Peak Shaving*, con cui si rappresenta il caso in cui la produzione di energia viene diminuita (processo anche denominato "Load Shedding") in maniera rapida e per un breve intervallo di tempo per evitare dei picchi di consumo. Ciò diventa possibile riducendo momentaneamente la produzione, attivando un generatore incorporato nel sistema o affidandosi a dei sistemi di batterie. La differenza tra i due metodi, illustrata nella Fig. 2.1 consiste nel mantenere costante la quantità di energia prodotta nel caso del Load Shifting, oppure diminuirla, come nel caso del Load Shaving.

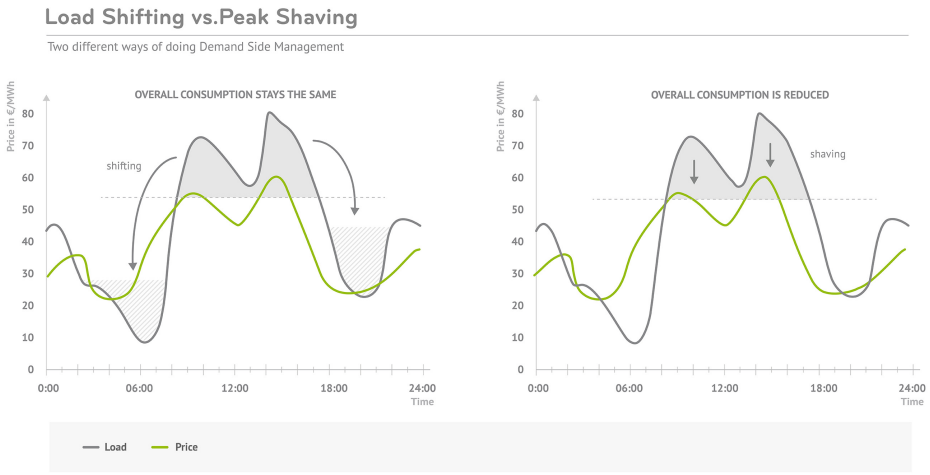

Figura 2.1: Load Shifting vs Peak Shaving - Credits to next-kraftwerke.com

Il valore del carico shiftato $S_{Load}(t)$, potrà assumere valori reali entro un range determinato dagli stessi valori stimati del carico, ovvero del 10% rispetto ai valori di $P_{Load}(t)$. Formalmente, si assume che il consumo di energia possa essere ridotto o aumentato in ogni istante temporale t del 10% del carico stimato, per ogni istante di tempo e per ogni scenario.

$$-(P^s_{Load}(t) + 10\%) \leq S_{Load}(t) \leq P^s_{Load}(t) + 10\% \qquad (5)$$

Rimanendo fedeli al concetto di Load Shifting, si deve imporre che il consumo di energia lungo l'orizzonte temporale T rimanga invariato, ovvero che in qualsiasi modo il carico previsto venga spostato lungo l'asse temporale in base alle necessità, la quantità totale non cambi.

$$\sum_{t \in T} S_{Load} = 0 \qquad (6)$$

Una volta terminato la fase di ottimizzazione offline saranno ottenuti come risultato i valori di $S_{Load}(t)$, ovvero il carico spostato, per ogni istante di tempo e associati ad ogni POD, valori che saranno mandati in input alla fase online.

2.1.4 Sistemi CHP

Combined Heat and Power è un'efficiente tecnologia che genera elettricità e cattura il calore che sarebbe altrimenti disperso, in modo da fornire utile energia termica, come vapore o acqua calda. Viene usato come mezzo di riscaldamento, in ambito domestico o industriale. I sistemi CHP

sono diffusi dunque, sia in strutture individuali come case o condomini, oppure per la produzione energetica. Infatti, si utilizzano quando esiste la necessità di usufruire sia energia elettrica sia energia termica.
Un sistema CHP tipicamente consiste in un generatore elettrico come una turbina a gas, turbina a vapore, motore a combustione. In aggiunta è installato uno scambiatore termico che cattura l'energia termica in eccesso o il gas esausto dal generatore per convertirlo in acqua calda o vapore. Ci sono due grandi categorie di CHP: la prima viene definita "topping cycle", sistemi in cui si genera prima energia elettrica e poi viene usato il vapore rimasto in un processo alternato, la seconda invece prende il nome di "bottoming cycle", e il procedimento è l'esatto opposto del primo. Si produce prima il vapore e poi elettricità, sono sistemi usati nelle vetrerie o nelle acciaierie, che necessitano di fornaci a temperature particolarmente elevate. Un altra tipologia di sistema CHP utilizza il gas naturale per la produzione di energia tramite celle a gas, senza combustione o bruciando il gas. Le celle tuttavia, producono sia calore sia elettricità, e nonostante siano un sistema relativamente giovane, ci si aspetta che la loro adozione cresca molto rapidamente.

Nel caso in esame sono considerati due diversi sistemi CHP, uno del primo tipo (topping cycle), in particolare la configurazione denominata Micro-CHP, il cui voltaggio di produzione è sufficiente per edifici individuali o palazzi. La produzione tipica ammonta a circa 1kW di elettricità una volta entrato a regime: il totale di elettricità prodotta annualmente dipende dalla quantità di tempo in cui il sistema è in funzione.
L'altro sistema di CHP considerato, è quello a gas naturale. Il costo ridotto del gas naturale permette di ottenere un guadagno in termini economici, in quanto risulta più conveniente rispetto a dover comprare energia dalla rete ed è anche un'ottima scelta in termini di ecologia, riducendo il consumo di diossido di zolfo (SO2) e ossidi di azoto (NOx). La quantità di energia prodotta dipende dal sistema specifico, ma si aggira attorno alla decina di kW.
Inoltre, viene assunto che il valore definitivo della potenza del CHP possa avere della flessibilità. In particolare tale flessibilità è modellata tramite la variabile a valori reali S_{CHP}, che andrà sommata alla potenza iniziale del CHP, denominata P_{CHP}, definita per ogni istante di tempo t e per ogni scenario *s*. Il ragionamento è analogo a quello effettuato nel caso del Load Shifting, anche se viene introdotta un'ulteriore condizione: si suppone che i bound di flessibilità del CHP siano variabili nell'arco di una specifica finestra temporale.

Associando una flessibilità al CHP è possibile forzarne l'utilizzo in certe finestre temporali o al contrario forzarne lo spegnimento in altre. Sarà il modello a trovare il valore ottimale per ogni finestra temporale in base alle necessità del sistema. I bound per ogni finestra temporale sono stati calcolati in riferimento ai dati illustrati in [27], basati sul reale utilizzo di un sistema micro-CHP.

Il valore definitivo della produzione è determinato da $\tilde{P}_{CHP}(t)$, somma dei due elementi, potenza e potenza shiftata, che definisce l'andamento del CHP stesso.

$$\tilde{P}^s_{CHP}(t) = P^s_{CHP}(t) + S_{CHP}(t) \quad \forall t \in T, \forall s \in S \qquad (7)$$

$\tilde{P}_{CHP}(t)$ è soggetta ad un vincolo dettato dalla tecnologia stessa, definita da un limite superiore e un limite inferiore, i cui valori sono stati definiti per il sistema CHP a diesel e per il sistema CHP [1] a gas naturale[2] .

$$min_{\tilde{P}_{CHP}} \leq \tilde{P}^s_{CHP}(t) \leq max_{\tilde{P}_{CHP}} \quad \forall t \in T \qquad (8)$$

L'insieme dei vincoli appena illustrati, deve essere specificato in maniera separata per ogni tipologia di CHP considerata, in quanto essendo configurazioni diverse, hanno limiti tecnologici diversi.

2.1.5 Bilanciamento del carico

Per ogni istante di tempo considerato bisogna imporre che tutti i flussi di energia, in entrata e in uscita siano bilanciati, ovvero che la produzione interna sia uguale alla richiesta di carico $P^s_{Load}(t)$, in tutti gli istanti di tempo e in tutti gli scenari possibili.
Si considera la produzione energetica delle risorse interne, i sistemi di batteria, e l'energia comprata dal mercato; l'energia venduta e quella direzionata verso il sistema di stoccaggio devono invece essere sottratte alla quantità di bilanciamento.
In particolare:

$$\tilde{P}^s_{Load}(t) = \tilde{P}^s_{CHP}(t) + P^s_{Gin}(t) - P^s_{Gout}(t) + P^s_{Sin}(t) - P^s_{Sout}(t) + P^s_{PV}(t) \qquad (9)$$

[1] CHP Project Profiles Database:
https://betterbuildingssolutioncenter.energy.gov/chp/chp-project-profiles-database

[2] Energy Prices:
https://www.eea.europa.eu/data-and-maps/indicators/en31-energy-prices/en31-energy-prices

2.1.6 Funzioni obiettivo

Sono stati considerati tre casi, nel tentativo di valutare con quale configurazione fosse possibile ottenere il funzionamento più efficiente delle unità. Nel primo caso si va ad utilizzare come funzione obiettivo, la minimizzazione dei costi, ovvero, si cerca il valore minimo del costo operazionale z del VPP lungo l'orizzonte temporale T. La funzione obiettivo è costruita nella seguente maniera:

$$z = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} c_{Gin}(t) P^s_{Gin}(t) + c_{CHP} \tilde{P}^s_{CHP}(t) + c_{Sout} P^s_{Sout}(t) - c_{Gout}(t) P^s_{Gout}(t) \qquad (10.a)$$

Dove S è il numero di scenari complessivi, $c_{grid}(t)$ è il costo orario associato all'energia del Mercato, derivato da [31]. Lo stesso prezzo è stato assunto per le variabili c_{Gout}, c_{Gin} e c_{Sout}. Per l'elemento CHP è stato considerato il prezzo del diesel o il prezzo del gas naturale, assunti costanti per ogni istante di tempo.

Il secondo caso utilizza invece una funzione obiettivo che mira alla minimizzazione dell'utilizzo della rete, ovvero del valore P_{Grid}, dato dalla somma dei valori P_{Gin} e P_{Gout}. In questa maniera il VPP sarà forzato ad usare al massimo le risorse disponibili al suo interno prima di eventualmente andare a comprare o vendere energia sul Mercato.

$$z = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} c_{Gin}(t) P^s_{Gin}(t) + c_{Gout}(t) P^s_{Gout}(t) \qquad (10.b)$$

Il terzo ed ultimo caso, prevede di minimizzare la distanza da un profilo di carico desiderato, in cui si calcola il valore minimo della varianza relativa al profilo di carico ottenuto tramite (4-7) e il profilo piatto definito a priori.
La varianza misura per definizione la distanza di una variabile rispetto ad un valore stimato, ed è individuata dalla media della differenza al quadrato tra un individuo e il valore atteso.

$$z = \frac{1}{|S|T} \sum_{s \in S} \sum_{t \in T} (\tilde{P}^s_{Load}(t) - P_{DesiredLoad}(t))^2 \qquad (10.c)$$

Generazione del profilo desiderato

Un profilo piatto è un profilo di generazione energetica che per definizione tende ad avere un andamento quasi lineare e costante, senza picchi di energia o sbalzi che ne modificano improvvisamente la direzione. Si può ipotizzare uno scenario in cui questo profilo sia calcolato su indicazione di operatori o gestori della rete energetica, come ad esempio Terna.

Terna è uno dei principali operatori europei di reti per la trasmissione dell'energia e gestisce la rete di trasmissione italiana in alta tensione. Un modello che ottimizza i flussi energetici in base ad un profilo desiderato, definito dall'operatore esterno, potrebbe rappresentare una valida offerta per la gestione della produzione di energia.
Per la creazione del profilo desiderato, definito come "profilo piatto" è stato scritto uno script ad hoc: tale script è invocato solo nel caso in cui sia scelta la funzione obiettivo che minimizza la distanza dal profilo desiderato. Prende in ingresso i valori del carico stimato e, utilizzando dei valori di soglia e di tolleranza prestabiliti, in maniera simile al processo di Load Shifting, va a spostare le quantità di energia in modo da livellare i picchi. In questo caso solo il procedimento è analogo, il ragionamento dietro al metodo è completamente diverso: l'obiettivo finale di questa fase è quello di ottenere un profilo con il minor numero di variazioni possibili, per questo definito piatto. Si suppone che i due parametri per la generazione del profilo vengano forniti dall'esterno, presumibilmente da un potenziale cliente o dall'amministratore del sistema.
Di seguito un esempio di come viene generato un profilo piatto, a partire da un profilo di carico (valori misurati ogni 5 minuti).

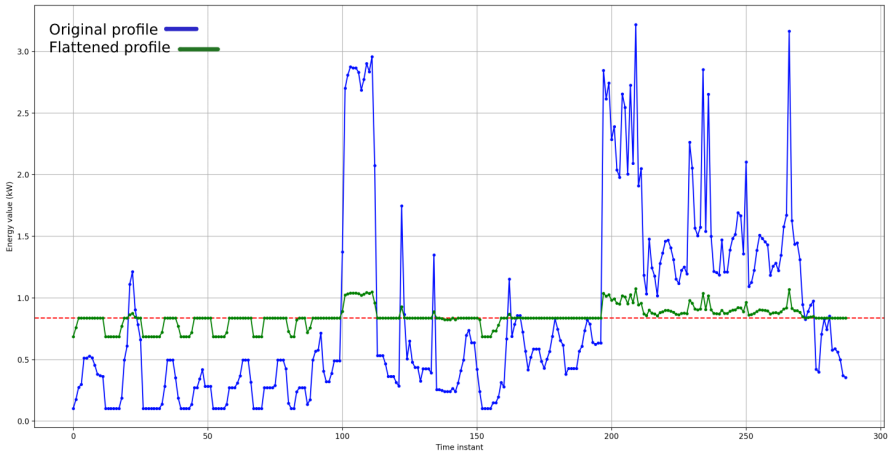

Figura 2.2: Processo di generazione del profilo piatto

La redistribuzione avviene usando degli *slice* in cui viene divisa la quantità di carico in eccesso, definita come la quantità di energia al di sopra della soglia, indicata con il colore rosso nell'immagine. Gli slice sono stati realizzati sufficientemente piccoli in modo da avere, eventualmente, rimanenze talmente ridotte da essere trascurabili.
La scelta della soglia risulta fondamentale per garantire il funzionamento

ottimale del processo: se tale soglia dovesse essere troppo elevata, le aree "da riempire" diventerebbero immediatamente piene lasciando così una quantità non trascurabile di energia ancora da redistribuire. Al contrario, se la soglia dovesse essere troppo bassa, il profilo non risulterebbe sufficientemente piatto.
Il valore di tolleranza è invece stato introdotto per evitare di ottenere una livellazione netta dei picchi, ottenendo così una curva realisticamente caratterizzata da variazioni di carico.

2.2 Fase Online

La fase online è la fase in cui vengono determinati i valori reali per le variabili dei flussi energetici, assumendo che i valori degli shift di carico e CHP siano stati realizzati utilizzando la fase offline del modello. La fase online è una versione ridotta del problema MILP utilizzato nella prima fase, ottenuta andando ad inserire i valori di Sload(t) e Schp(t) ricavati dall'ottimizzazione del giorno precedente e ottenendo la realizzazione dell'incertezza effettiva.

$$min_{\tilde{P}_{CHP}} \leq \tilde{P}_{CHP}(t) \leq max_{\tilde{P}_{CHP}} \quad \forall t \in T \tag{11}$$

$$min_{P_{Gin}} \leq P_{Gin}(t) \leq max_{P_{Gin}} \quad \forall t \in T \tag{12}$$

$$min_{P_{Gout}} \leq P_{Gout}(t) \leq max_{P_{Gout}} \quad \forall t \in T \tag{13}$$

$$charge(t) = charge(t-1) - \eta * P_{Sin}(t) + \eta * P_{Sout}(t) \quad \forall t \in T \tag{14}$$

$$\tilde{P}_{Load}(t) = \tilde{P}_{CHP}(t) + P_{Gin}(t) - P_{Gout}(t) + P_{Sin}(t) - P_{Sout}(t) + P_{PV}(t) \tag{15}$$

Essendo stata considerato il caso limite in cui non venga effettuata la fase di ottimizzazione offline, i valori degli shift, poiché non calcolati, non vengono inseriti, e saranno utilizzati solamente i valori effettivi di carico e potenza del CHP. Considerando un singolo scenario, che corrisponde alla reale situazione tra quelle descritte considerando l'incertezza.
In questa fase non viene più considerato un modello che elabora tutti gli istanti temporali allo stesso tempo poichè disponibili (vengono usate delle stime), ma un modello ad ogni istante temporale. Per ogni t sono ricavati i valori delle variabili decisionali e memorizzati in apposite strutture temporanee per essere facilmente utilizzabili.

2.2.1 Funzioni obiettivo

Le scelte disponibili per la funzione obiettivo sono limitate alla minimizzazione dei costi e alla minimizzazione dell'utilizzo della rete. Il profilo desiderato risulta utile unicamente nello step offline per ottenere gli shift del Load. Nella fase online i valori di carico non sono più delle variabili ma dei valori definiti dalla somma del valore stimato della potenza $P_{Load}(t)$ e del valore shiftato $S_{Load}(t)$. Analogamente, il valore della potenza del CHP sarà determinato dalla somma del valore della potenza $P_{CHP}(t)$, che è nuovamente una variabile decisionale, e dal valore shiftato $S_{CHP}(t)$.

$$z = \sum_{t \in T} c_{Gin}(t) P_{Gin}(t) + c_{CHP} \tilde{P}_{CHP}(t) + c_{Sout} P_{Sout}(t) - c_{Gout}(t) P_{Gout}(t) \tag{16.a}$$

$$z = \sum_{t \in T} c_{Gin}(t) P^{s}_{Gin}(t) + c_{Gout}(t) P^{s}_{Gout}(t) \tag{16.b}$$

Dove S è il numero di scenari complessivi, $c_{grid}(t)$ è il costo orario associato all'energia del Mercato[3]. Lo stesso prezzo è stato assunto per le variabili c_{Gout}, c_{Gin} e c_{Sout}. Per l'elemento CHP è stato considerato il prezzo del diesel o il prezzo del gas naturale, assunti costanti per ogni istante di tempo.

[3] http://www.mercatoelettrico.org/En/Default.aspx

3 Interfaccia Web

L'interfaccia web è stata realizzata seguendo un pattern MVC (Model-View-Controller) con l'ausilio di Spring[1].
Lo scopo della realizzazione dell'interfaccia era quello di avere uno strumento per poter gestire la creazione degli elementi base del modello, i POD. In particolare, vengono fornite diverse possibilità di personalizzazione, a partire dalle configurazioni dei POD, della gestione dell'incertezza e degli step di ottimizzazione, alle funzioni obiettivo da utilizzare.
Una volta completate le scelte necessarie, l'interfaccia gestisce in maniera autonoma l'invocazione degli script con i parametri selezionati e rimanda in output il risultato della computazione. In questo capitolo saranno descritti alcuni elementi fondamentali per la comprensione dell'interfaccia e della sua realizzazione, poi ne sarà descritta in maniera approfondita l'architettura.
Lo strumento per realizzare l'interfaccia è Spring, un framework estremamente duttile e versatile, molto diffuso per la realizzazione di infrastrutture e applicazioni Java.
Di seguito, una breve sezione introduttiva su Spring e la descrizione di come è stato utilizzato per il caso in questione assieme ad altri strumenti ausiliari.

3.1 Strumenti utilizzati

3.1.1 Spring Framework

Spring è una piattaforma che fornisce supporto per un'infrastruttura a 360° per la creazione e lo sviluppo di applicazioni Java.
Visto come una valida, e ormai ben solida alternativa al modello basato su Enterprise Javabeans, questo framework supera l'appesantimento derivato dall'utilizzo forzato di interfacce EJB di tipo *home* e *remote*,

[1]Spring official website: https://spring.io/

troppo invasive nel codice scritto, grazie a nuovi ed innovativi modelli di programmazione, come l'Aspect Oriented Programming (AOP) e l'Inversion of Control (IoC).
Spring è inoltre definito come container *leggero*: grazie alla sua struttura estremamente modulare è possibile utilizzarlo nella sua interezza o solo in parte, senza stravolgere l'architettura del progetto; questa peculiarità permette una facile integrazione anche con altri framework già esistenti come Hibernate.

Il punto di partenza sono i POJO (Plain Old Java Object), da cui si costruisce la applicazione e a cui si possono associare servizi enterprise in maniera non invasiva. Con applicazione Java si intendono sia applicazioni embedded sia applicazioni server-side di livello enterprise, identificando un insieme di oggetti che collaborano tramite una rete di dipendenze per far funzionare correttamente l'infrastruttura.
I moduli che compongono l'architettura di Spring si possono suddividere in due macro-categorie:

- Il *modulo Core* fornisce le funzionalità fondamentali del framework. Tale modulo si basa sulle seguenti funzionalità e pattern: Dependency Injection, Aspect-Oriented Programming, la gestione delle transazioni, la struttura di applicazioni Web, l'accesso ai dati, la messaggistica, i test e altro.

- Altri moduli facoltativi : dalla configurazione alla sicurezza, dalle web app ai Big Data.

Nonostante la piattaforma Java consenta un ampio range di funzionalità di sviluppo, non supporta la possibilità di organizzare i blocchi dell'architettura in un insieme coerente, lasciando l'incombenza agli sviluppatori e ai programmatori. Per comporre le classi e le istanze degli oggetti si possono utilizzare dei pattern come Factory, Builder o Decorator: pattern che rimangono tuttavia delle pratiche da implementare a mano nell'applicazione. In Spring, l'*Inversion of Control* (IoC) si occupa esattamente di tale questione, offrendo la possibilità di creare disparati componenti e comporli in un'applicazione completamente funzionante e pronta da usare. I pattern vengono codificati come degli oggetti di prima classe che possono essere integrati direttamente nell'architettura.
Formalmente, l'Inversion of Control è un principio architetturale nato alla fine degli anni Ottanta, basato sul concetto di invertire il controllo del flusso di sistema rispetto alla programmazione tradizionale, in cui lo sviluppatore definisce la logica del flusso di controllo, specificando le

operazioni di creazione, inizializzazione degli oggetti ed invocazione dei metodi. Nell' Inversion of Control si inverte il control flow, facendo in modo che non sia più lo sviluppatore a doversi preoccupare di questi aspetti, ma il framework.

In particolare si usa il termine *Dependency Injection* (DI) con cui Spring implementa la IoC. La DI prevede che tutti gli oggetti all'interno di una applicazione accettino le dipendenze, ovvero gli oggetti di cui hanno bisogno, tramite costruttore o metodi setter. Non sono quindi gli stessi oggetti a creare le proprie dipendenze, ma sono esse vengono iniettate dall'esterno (Hollywood Principle).
L'operazione di "iniezione" appena descritta viene eseguita in maniera automatica da Spring, che crea le dipendenze necessarie alle classi dell'applicazione, sia per componenti del framework sia per oggetti definiti dallo sviluppatore.
L'ecosistema all'interno del quale le applicazioni Spring vivono, viene definito *IoC Container*. Lo IoC container si occupa di istanziare gli oggetti (beans) dichiarati nel progetto e di reperire e iniettare tutte le dipendenze ad essi associate. Tali dipendenze possono essere componenti del framework o altri bean dichiarati nel contesto applicativo.
L'altro concetto su cui si basa Spring è l'*Aspect Oriented Programming* (AOP), ovvero un approccio inventato da alcuni software engineers con cui, se possibile, si cerca di separare la logica di business e la logica di *cross-cutting concern* (problematiche trasversali). L'obiettivo è quello di migliorare sia la pulizia del codice sia la sua riutilizzabilità, ed è messo in pratica tramite il modulo AOP del framework.

3.1.2 Altri Strumenti

Di seguito gli strumenti principali utilizzati per la realizzazione dell'interfaccia web. Il sistema operativo sottostante è Linux 16.04.

- IntelliJ Idea [v. Ultimate] - un IDE (integrated development environment) per lo sviluppo software. Creato e diffuso da JetBrains (precedentemente chiamato IntelliJ) e disponibile sia in versione community sia in versione proprietaria.

- Wildfly [v. 18.0.1 Final] - in precedenza noto come JBoss, è un application server al momento sviluppato da Red Hat. Scritto in Java e supportato dalla Java Platform Enterprise Edition, è anche supportato da altre piattaforme. Wildfly è open-source e gratuito soggetto alla licenza LGPL v2.1.

- MySQL [v. 8.0.18] - sistema open source per la gestione del Database.
- Anaconda [v. 2019.10] - Distribuzione dei linguaggi di programmazione Python e R gratuita e open-source per scientific computing che mira alla semplificazione della gestione dei packages e del deployment. Utilizzato per creare un environment, un ambiente in cui eseguire gli script di pyomo contenenti i modelli matematici.
- Gurobi Solver [v. 8.1] - Ottimizzatore commerciale per linear programming (LP), quadratic programming (QP), quadratically constrained programming (QCP), mixed integer linear programming (MILP), mixed-integer quadratic programming (MIQP), and mixed-integer quadratically constrained programming (MIQCP); è stata utilizzata la licenza gratuita in dotazione agli studenti UniBo.

Per il corretto funzionamento dell'interfaccia, e dunque della corretta invocazione ed esecuzione dei modelli matematici, è necessario eseguire alcune operazioni fondamentali. Se non fosse già stato fatto in precedenza, assicurarsi che MySQL sia installato e funzionante sul sistema: sarà inoltre necessario creare un Database specifico per il progetto, da indicare poi nel file di configurazione della persistenza di Spring (persistence-mysql.xml).

1. Aprire Anaconda, e in particolare l'environment in cui è stato configurato Pyomo e le variabili di ambiente di Gurobi,
2. Assicurarsi che il file .war in cui risiedono i dettagli del progetto sviluppato in Spring si trovi nella cartella di esecuzione dell'application server (Wildfly),
3. Lanciare l'application server (Wildfly) e attendere la compilazione del progetto.

3.1.3 Pattern MVC

Il pattern MVC è uno schema di design per software usato comunemente per sviluppare UI: al fine di separare le modalità interne con cui sono elaborati i dati con le modalità esterne con cui tali dati sono esposti e mostrati agli utenti.
Brevemente, lo schema è formato da tre componenti:

- Model - Il componente centrale dello schema che costituisce la struttura dinamica dei dati dell'applicazione, indipendente dalla UI. Gestisce direttamente i dati, la logica e le regole dell'applicazione.
- View - Comprende qualsiasi forma di rappresentazione dei dati. Molteplici viste della stessa informazione sono concesse, ad esempio un grafico a barre per il Management, e una vista tabulare per il Settore Commerciale.
- Controller - Accetta gli input dell'utente e li trasmette al Model o alla View.

L'importanza di questo pattern non risiede unicamente nella suddivisione che realizza tra i componenti, ma anche nel fatto che definisce le interazioni tra gli stessi. Il Model è responsabile di gestire i dati dell'applicazione: riceve gli input dell'utente dal Controller.
La View rappresenta il modello in formato specifico; il Controller risponde alle azioni dell'utente e le esegue sugli oggetti della struttura.
Da notare che sia lo strato View che lo strato Controller dipendono direttamente dal Model, il quale non dipende dagli altri. Questo è uno dei fattori più importanti di questa architettura, poiché permette al modello di essere implementato e testato indipendentemente dallo strato di visualizzazione. Analogamente ad altri pattern, MVC è la concreta soluzione ad un problema, mentre si adatta al sistema in cui viene sviluppato.

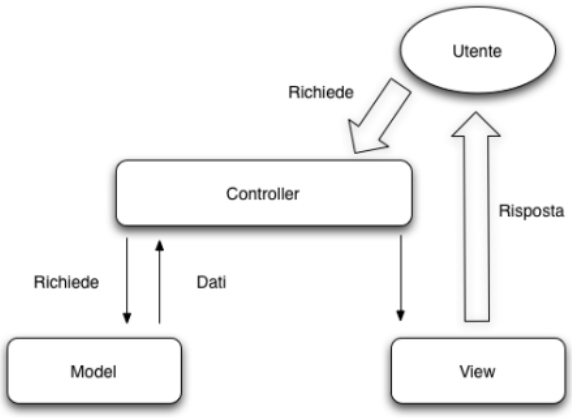

Figura 3.1: Schema MVC

3.2 Architettura

Nelle sezioni successive verrà analizzata in dettaglio l'interfaccia web, spiegandone le funzionalità e le scelte implementative, con l'ausilio di diagrammi e schemi UML, specifici per la visualizzazione del design di un sistema software.

3.2.1 User Case Diagram

Si assume che ci sia una unica tipologia di utente che può interagire con il sistema, non ci sono differenziazioni di categoria. In particolare, gli unici fruitori dell'interfaccia saranno gli amministratori del sistema, che dunque non hanno necessità di registrarsi o identificarsi. Di seguito, il diagramma del Caso d'Uso per gli attori che interagiscono con l'applicazione.

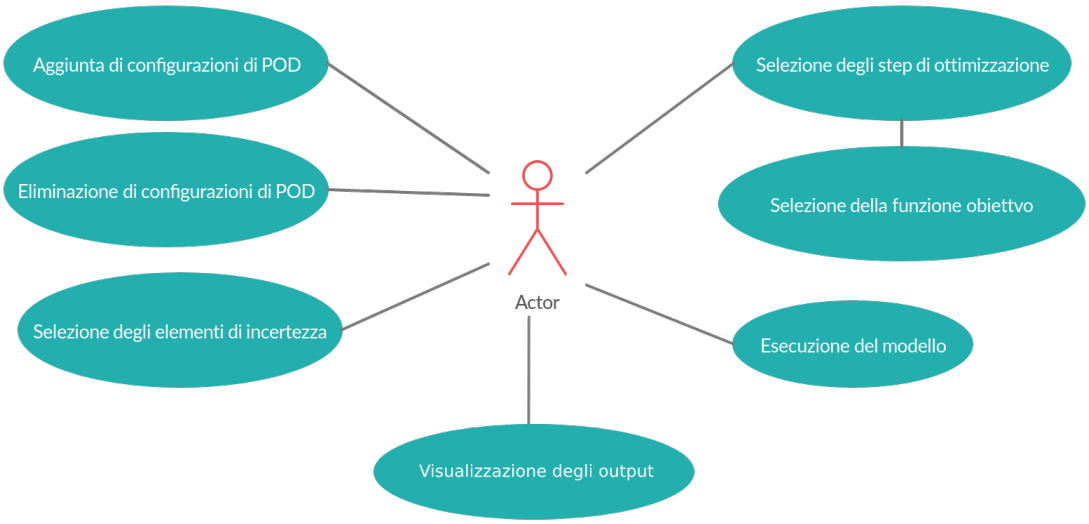

Figura 3.2: User Case Diagram

Come si deduce dallo schema, le azioni a disposizione dell'utente sono relative all'elaborazione del modello da compilare ed eseguire, in particolare l'attore in questione può: gestire le configurazioni dei POD, decidere dove applicare l'incertezza, quali step di ottimizzazione eseguire e scegliere le relative funzioni obiettivo.

Aggiunta di configurazioni di POD

Attori coinvolti: Utente del sistema.
Pre-Condizioni: Nessuna.
Post-Condizioni: Aggiunta nel Database delle configurazioni selezionate, visualizzazione a schermo delle configurazioni disponibili.
Flusso Principale:

1. L'utente seleziona tramite il menù a tendina gli elementi da inserire nel POD (a partire da una lista di elementi disponibili),
2. Seleziona il fattore di replicazione della configurazione,
3. Aggiunta dei componenti selezionati,
4. Redirect alla home page e visualizzazione dei POD disponibili e delle loro configurazioni.

Flussi Alternativi: Nessuno, l'utente può aggiungere quanti componenti desidera, replicati una singola volta o più volte.

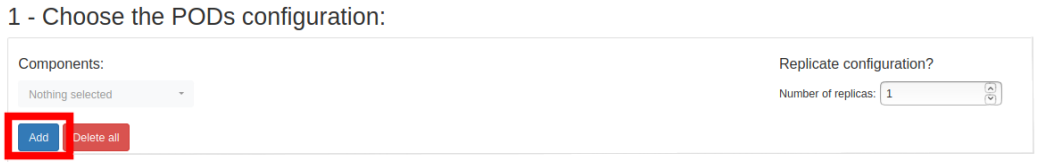

Figura 3.3: Aggiunta di configurazioni di POD

Eliminazione di configurazioni di POD

Attori coinvolti: Utente del sistema.
Pre-Condizioni: Devono esserci delle configurazioni da eliminare.
Post-Condizioni: Rimozione dal Database delle configurazioni selezionate, visualizzazione a schermo delle configurazioni disponibili.
Flusso Principale:

1. L'utente elimina la configurazione cliccando il bottone "Delete",
2. Rimozione dei componenti selezionati,
3. Redirect alla home page e visualizzazione dei POD disponibili e delle loro configurazioni.

Flussi Alternativi: Si può decidere di eliminare tutte le configurazioni disponibili utilizzando il bottone "Delete all".

Selezione degli elementi di incertezza

Attori coinvolti: Utente del sistema.
Pre-Condizioni: Devono esserci delle configurazioni di POD disponibili.
Post-Condizioni: I flag relativi agli elementi selezionati saranno attivi. Possono essere selezionati uno, due o nessun elemento.
Flusso Principale:

1. L'utente seleziona tramite il checkbox a quali elementi applicare l'incertezza.

Flussi Alternativi: Nessuno.

Selezione degli step di ottimizzazione

Attori coinvolti: Utente del sistema.
Pre-Condizioni: Devono esserci delle configurazioni di POD disponibili.
Post-Condizioni: I flag relativi agli elementi selezionati saranno attivi. Possono essere selezionati uno o entrambi gli elementi.
Flusso Principale:

1. L'utente seleziona tramite il checkbox quali step di ottimizzazione eseguire.

Flussi Alternativi: Se non viene selezionato alcuno step di ottimizzazione, nel momento in cui si cerca di eseguire il modello, verrà visualizzato un messaggio di errore.

Selezione delle funzioni obiettivo

Attori coinvolti: Utente del sistema.
Pre-Condizioni: Deve essere selezionato il relativo step di ottimizzazione.
Post-Condizioni: I flag relativi agli elementi selezionati saranno attivi. Può essere selezionato un singolo elemento.
Flusso Principale:

1. L'utente seleziona tramite il radiobox quale funzione obiettivo associare allo step di ottimizzazione.

Flussi Alternativi: Se non viene selezionato alcuna funzione obiettivo, nel momento in cui si cerca di eseguire il modello, verrà visualizzato un messaggio di errore.

Visualizzazione degli output

Attori coinvolti: Utente del sistema
Pre-Condizioni: Deve essere eseguito il modello matematico e le immagini di output disponibili.
Post-Condizioni: Nessuna.
Flusso Principale:

1. L'utente esegue il modello tramite il bottone "Compute".
2. In base alle scelte eseguite sarà eseguito il modello matematico relativo.
3. Vengono elaborate le immagini di output.
4. Redirect nella pagina di visualizzazione degli output.

Flussi Alternativi: Se le immagini non sono disponibili non saranno visualizzate a schermo.

Figura 3.4: Visualizzazione degli output

Realizzazioni

Nelle immagini successive si possono osservare le realizzazioni di ciascun caso d'uso nell'interfaccia finale e in particolare, la scelta della funzione obiettivo per entrambe le fasi dell'ottimizzazione. Se una delle due fasi non è selezionata, le opzioni relative rimangono nascoste per una maggiore leggibilità.

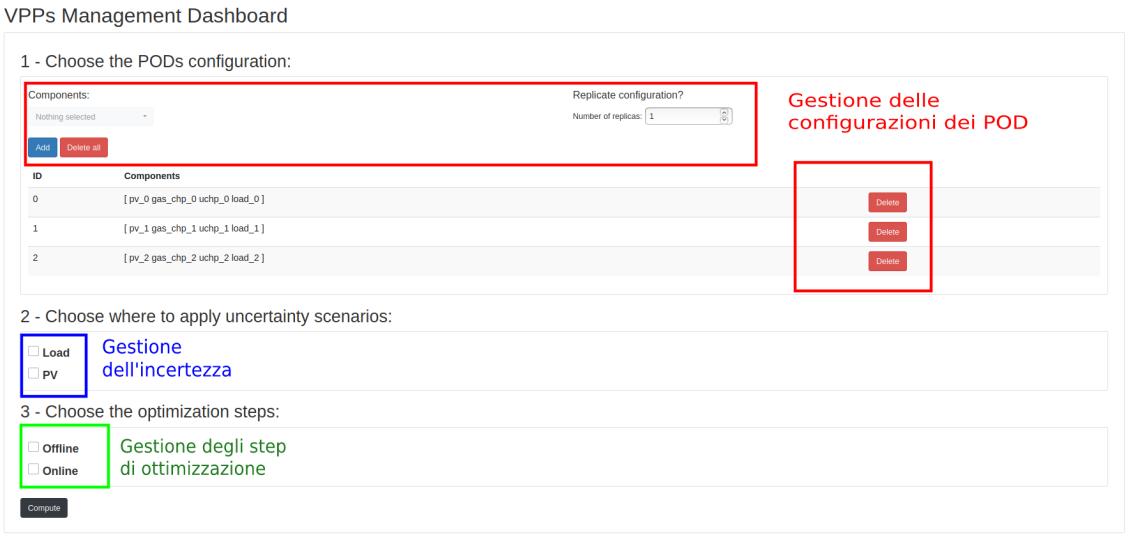

Figura 3.5: Realizzazione dell'Interfaccia Web

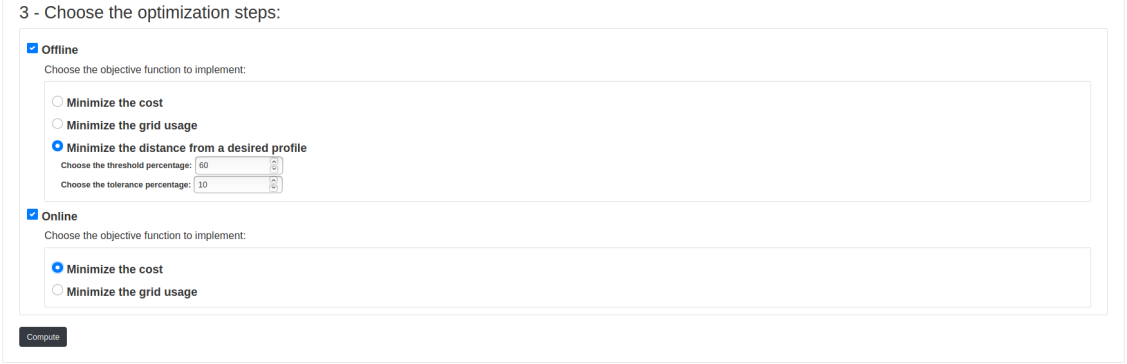

Figura 3.6: Scelta degli step di ottimizzazione

Per quanto riguarda i grafici di output invece, è stata necessaria un'ulteriore lavorazione sui dati disponibili. Tramite la libreria `Matplotlib` ci sono molti strumenti disponibili, ma differenziare un grafico in cui sono

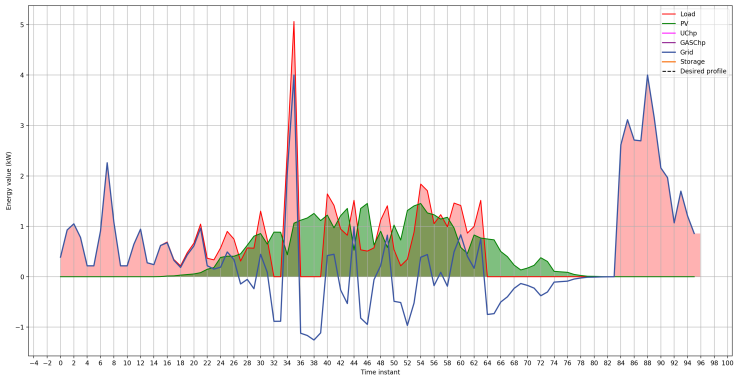

Figura 3.7: Esempio di grafico generato in output

presenti così tante curve ha richiesto ulteriori considerazioni. In particolare, per ottenere una maggiore definizione nel rendering delle curve e delle aree evidenziate si è introdotto il concetto di *interpolazione lineare*, realizzato tramite la funzione `interp` della libreria `Numpy`.
Il metodo dell'interpolazione lineare è un metodo numerico per trovare le radici di una funzione e necessita della stima iniziale di un intervallo (a,b) entro cui debba esser compresa la radice, tale che $f(a) * f(b) < 0$.
Dunque, data una sequenza di n numeri reali distinti x_k per $k = 1, ..., k = n$ chiamati nodi e per ogni x_k sia dato un secondo numero reale y_k. L'interpolazione si propone di cercare una funzione di variabile reale $f(x)$ di una certa famiglia di funzioni di variabile reale tale che sia

$$f(x_k) = y_k \text{ per } k = 1, \ldots, n$$

Una coppia (x_k, y_k) viene chiamato *punto dato* ed f viene detta interpolante per i punti dati.

Di seguito un esempio di grafico le cui curve sono definite tramite interpolazione. In ordine, vengono eseguite le seguenti azioni:

1. Creazione del quadro dell'immagine
2. Definizione della curva
3. Interpolazione su range (0,96) con step 0.1, con coordinate x determinate dagli istanti temporali e le coordinate y determinate dalla curva *tildelist*.

4. Colorazione dell'area compresa tra la curva e l'asse delle x

```
resultimg, result = plt.subplots(figsize=(20, 10))
    images, = result.plot(tildelist, linestyle='-', color='red')
    tilde = np.interp(np.arange(0.0, 96.0, 0.1), fixed_time_list,
    tildelist)
    result.fill_between(np.arange(0.0, 96.0, 0.1), tilde, 0, facecolor='
    red', alpha=0.3)
```

Listing 3.1: Utilizzo della funzione Numpy.interp

3.2.2 Flow Chart

Viene di seguito riportato il Diagramma di Flusso che rispecchia il flusso di azioni per l'esecuzione di un modello correttamente configurato, eseguendo tutte le scelte disponibili.
Non sono stati riportati gli Activity Diagram relativi all'interazione utente-sistema, rappresentanti i casi d'uso elencati nella sezione precedente, in quanto non particolarmente significativi per illustrare il funzionamento del modello. Da considerare, nello svolgimento delle attività descritte nel Diagramma, che è obbligatorio selezionare almeno uno dei due step di ottimizzazione, altrimenti un messaggio di errore sarà visualizzato al tentativo di esecuzione del modello.

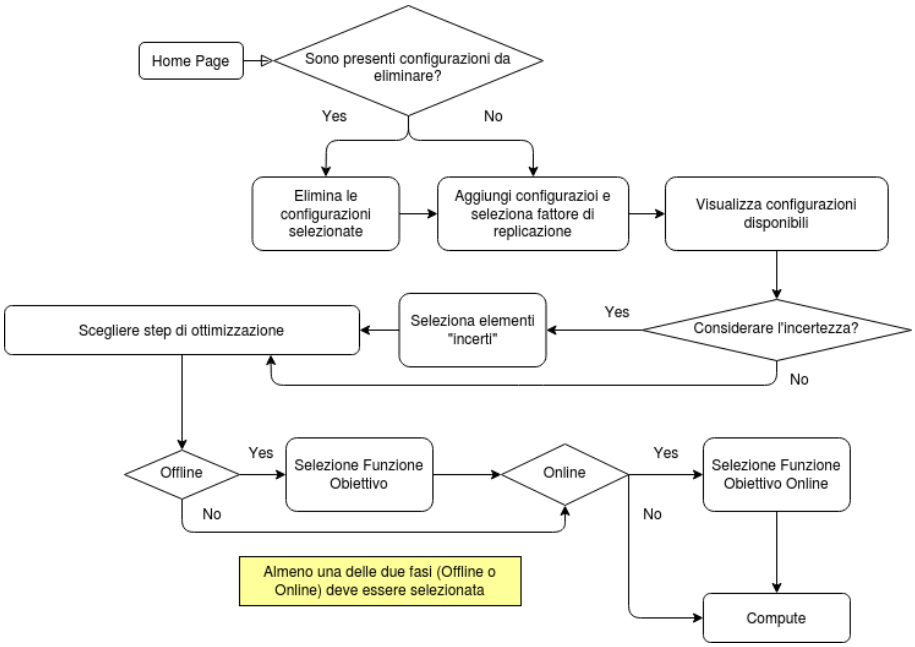

Figura 3.8: Dragramma di Flusso

3.2.3 Sequence Diagram

Ci sono quattro livelli dell'applicazione, che rispecchiano il pattern MVC: Web Application (View), Controller (Controller), a cui vanno aggiunte le classi di Service e Dao, e la componente di Model, che viene invece mappata tramite la classe Component. La classe Component descrive gli oggetti su cui sono basati gli elementi del sistema che si identificano con i POD, e i loro attributi.
Ci sono due ragioni per cui si utilizzano le due classi Service e Dao. La prima è innanzitutto per mantenere le due entità logicamente separate: il Dao (Data Access Object) di solito esiste unicamente per creare una connessione con il database, il Service contiene invece tutti i metodi aggiuntivi che potrebbero servire in termini di logica.
La seconda ragione per cui queste due strutture sono tenute separate è la sicurezza: avendo un livello di servizio che non ha relazioni con il database, è più difficile poter accedere al database stesso. Per questi motivi è sempre consigliato (e buona pratica di programmazione), inserire nella classe Dao i metodi per aggiungere, eliminare o modificare gli oggetti del database e una classe Service per tutto il resto. Questo approccio modulare consente anche di sostituire facilmente il codice nel caso in cui il database subisse modifiche. In Spring la sicurezza è idealmente applicata a livello del livello Service.

Di seguito, viene rappresentato il Diagramma delle Sequenze relativo alle classi appena menzionate. In particolare, sarà possibile osservare le invocazioni e il comportamento dell'applicazione nei casi in cui l'utente voglia aggiungere o eliminare delle configurazioni, ed eseguire infine il modello.
Il caso di eliminazione di tutti gli elementi non è stato rappresentato in quanto analogo a quello di eliminazione singola: l'unica differenza è che nel primo caso non è necessario effettuare il match tra l'id del componente da eliminare e quello presente nel database, poiché tutti quelli presenti vengono iterativamente eliminati.

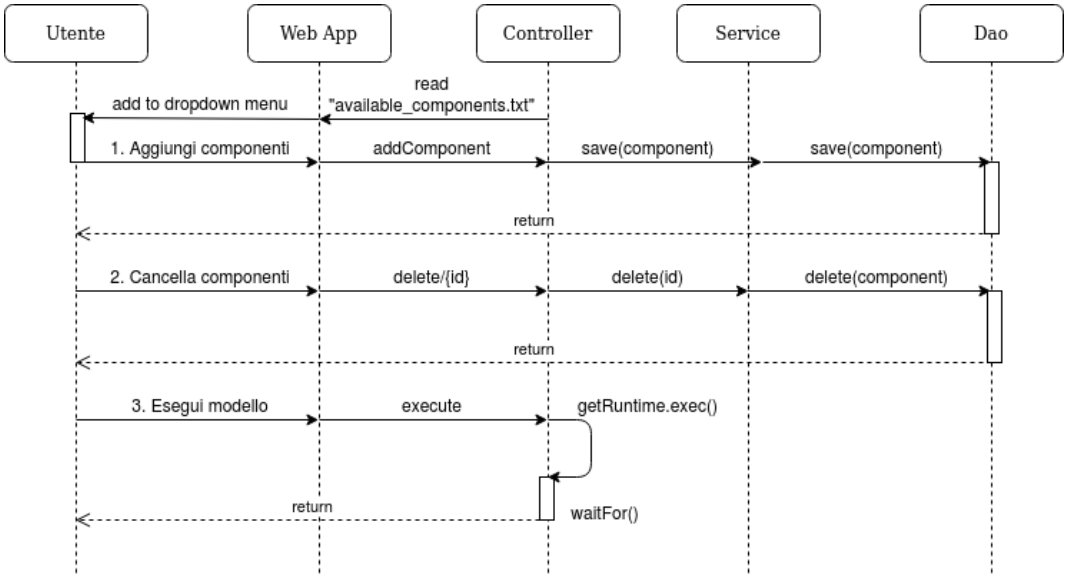

Figura 3.9: Sequence Diagram

3.2.4 Modello di Dominio

Un modello di Dominio generalmente viene implementato come un modello ad oggetti su un livello che utilizza dei sotto-livelli per la persistenza e "pubblica" le informazioni ad un livello più alto per guadagnare l'accesso ai dati e al comportamento del modello.
In linguaggio UML, si rappresenta tramite il diagramma delle Classi, in particolare si divide il progetto in tre categorie: Boundary Class, Control Class, Entity Class. Le tre categorie rappresentano tre prospettive chiave e, per quanto specifiche dei linguaggi, dei framework e delle euristiche utilizzate, il comportamento del sistema può sempre essere associato ad esse. Il pattern EBC è simile al pattern MVC ma non è solamente appropriato per gestire le interfacce utente, consente al Controller di assumere un ruolo leggermente diverso. In figura, un esempio di pattern EBC.
Applicando questo schema, l'approccio risulta robusto, e identifica gli elementi, il comportamento e le relazioni necessarie a supportare l'intero sistema. Ci sono quattro regole per la comunicazione:

- Gli utenti (definiti attori in linguaggio UML) possono solo comunicare con oggetti Boundary,
- Gli oggetti Boundary possono comunicare solo con le classi Controller e con gli utenti,
- Gli oggetti Entity possono comunicare solo con le classi Controller,

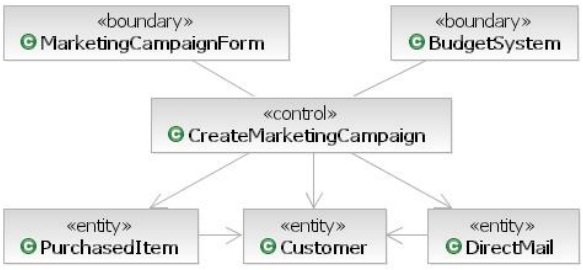

Figura 3.10: Esempio di Diagramma EBC - credits: utm.mx

- Le classi Controller possono comunicare agli oggetti Boundary e Entity e agli altri Controller, ma non con gli utenti.

Riassumento, vengono rappresentate nella seguente tabella le interazioni concesse, segnalate con il segno "X".

	Entity	Boundary	Control
Entity	X		X
Boundary			X
Control	X	X	X

Entity

Un oggetto *Entity* è un elemento passivo, long-lived, responsabile di contenere informazioni importanti per il sistema, in particolare, nel caso di Spring, può rappresentare un oggetto nel database. Un esempio di Entity per un'applicazione customer service è una CustomerEntity che gestisce tutte le informazioni relative ad un cliente. L'elemento di software design che deriva da questa entità dovrebbe fornire dunque una comprensiva panoramica del cliente, il comportamento per gestirne e validarne le informazioni e le azioni nel sistema (ad esempio se tale cliente può accedere ad un determinato prodotto). A livello di codice, l'identificazione delle Entity può essere realizzata in diversi metodi e a diversi livelli di astrazione.

Nell'applicazione web realizzata, l'unico oggetto Entity necessario per definire l'intera struttura è l'oggetto denominato *Component*. Un "componente" è idealmente realizzato basandosi sul concetto di VPP e realizza un elemento POD: gli attributi identificativi necessari per distinguere i componenti tra loro sono l'ID, indice univocamente generato dal Database, l'indice del POD, identificato da un valore intero, anch'esso univoco sull'intero insieme dei POD, e la lista di risorse energetiche che lo compongono. Gli unici metodi definiti a livello di entità sono i metodi getter

e setter per modificare o ricavare informazioni sugli oggetti.

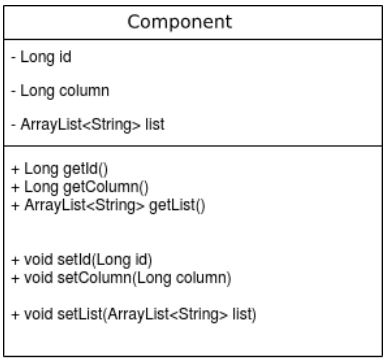

Figura 3.11: Schema UML della classe Componente

Nel caso specifico di Spring, l'oggetto Entity è un oggetto associato alla persistenza, che deve permanere nel sistema anche oltre la fine di una sessione. Dal momento che sono oggetti long-lived, il costo di caricamento dei dati dal database viene pagato solo la prima volta, mentre da quel momento in poi, nel sistema ci sarà un oggetto Java pronto e popolato, garantendo una grande efficienza in lettura e in scrittura. Il secondo vantaggio nell'utilizzo degli Entity è la capacità di fronteggiare i fallimenti: se succede qualcosa prima che finisca la transazione non è necessario fare dei costosi rollback, in quanto l'interazione con il database non è ancora avvenuta.
A livello di codice la classe Component viene marcata usando delle *Annotations*, uno spazio di programmazione dichiarativo in cui aggiungere informazioni e integrarle con il programma stesso, senza utilizzare file XML esterni (come avveniva in passato). Le annotazioni possono essere lette dal compiler o utilizzando un parser, ed è possibile specificare se le informazioni offerte sono disponibili solo a tempo di compilazione o fino al runtime. Le annotazioni utilizzate sono @Entity e @Table: la prima specifica che l'oggetto in questione è un oggetto Entity che deve essere persistente, mentre la seconda viene utilizzata da Hibernate per specificare dove andare a mappare tale oggetto.
Si ricorda che Hibernate fornisce un servizio di Object-relational mapping (ORM), ovvero gestisce la persistenza dei dati sul database attraverso la rappresentazione e il mantenimento su database relazionale di un sistema di oggetti Java.
L'utilizzo delle annotazioni appena descritte realizza una corrispondenza tra l'applicazione e il database: per gli oggetti Component del sistema, ci

sarà una tabella nel database che contiene tutte le informazioni relative ad essi, chiamata "Components".

```
@Entity
@Table(name = "COMPONENTS")
public class Component implements Serializable { ... }
```

Listing 3.2: Annotazioni della classe Component

Control

Un elemento appartenente alla classe Control gestisce il flusso di interazioni dello scenario, ad esempio il comportamento end-to-end. In generale, le regole di comportamento e business relative alle informazioni pertinenti allo scenario dovrebbero essere associate alle entità; gli elementi di controllo dovrebbero invece essere responsabili solo del flusso delle azioni e delle interazioni dell'applicazione.
Facendo riferimento alla Figura 3.5, CreateMarketingCampaign è un esempio di elemento di controllo per un'applicazione mirata al Customer Service: deve essere responsivo a certi elementi della classe Boundary (introdotti nella prossima sotto-sezione) e collaborativi con elementi della classe Entity.
Per identificare gli elementi di controllo si può innanzitutto eseguire un'analisi su quali azioni siano da effettuare e quali comportamenti sia necessario garantire in modo che l'applicazione operi in maniera corretta e sicura. Importante rimane cercare di riutilizzare i metodi e le interazioni già presenti nel sistema. Nel caso in questione, le classi per la realizzazione della classi Control sono quelle denominate come Service: in particolare, ad ogni Service è associata una classe astratta e la relativa implementazione specifica per l'applicazione realizzata. Come descritto già in precedenza, per garantire un maggiore livello di sicurezza, oltre alle classi Service sono state utilizzate le classi Dao.

Fulcro della classe Control, è il Controller. La classe Controller è l'intermediario tra il livello direttamente accessibile all'utente, ovvero la View e il resto del sistema: si occupa di gestire la corretta esecuzione delle azioni e il loro smistamento ai livelli inferiori. In particolare, è in questa classe in cui sono definite tutti i metodi invocabili nelle JSP, appartenenti alla classe Boundary. Di seguito lo schema che illustra tramite il linguaggio UML la composizione delle classi appartenenti e le loro relazioni. Si noti la gerarchia: la classe Dao mette a disposizione i metodi che accedono al database (e ne gestiscono gli oggetti) alla classe Service, che a sua volta è messa al servizio della classe Controller, che invoca i metodi corrispon-

denti alle azioni eseguite dall'utente.

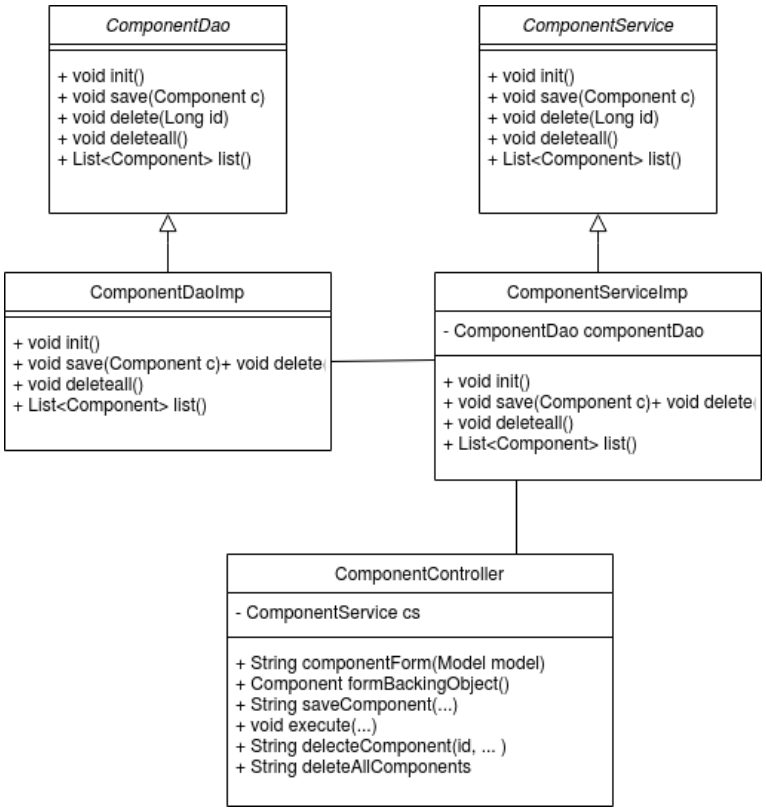

Figura 3.12: Schema delle classi Control

Nella classe Controller è inoltre presente un metodo di supporto estremamente importante: *formBackingObject()*. Quando viene caricata una pagina web, questa necessita talvolta di alcune informazioni per essere visualizzata correttamente. Alcune di queste informazioni sono read-only, e non sono parte della transazione corrente (ad esempio i campi di un menù a tendina). Altre informazioni amcora, possono essere utili per leggere e scrivere i contenuti di una form, ad esempio. Tutti questi dati vengono incapsulati in oggetti che devono essere disponibili quando la pagina li richiede: possono esistere quanti oggetti siano necessari, e sono restituiti usando il metodo formBackingObject().
Utilizzato assieme alle Annotations, questo metodo rende Spring uno strumento per la programmazione estremamente flessibile e versatile. In aggiunta, tutti i metodi della classe ComponentServiceImp vengono marcati utilizzando la annotation @Transactional. In generale, utilizzare questa annotazione, significa che Spring crea dinamicamente dei proxy

per le classi che la dichiarano, proxy praticamente invisibili a runtime che consentono al framework di iniettare i comportamenti prima, dopo o attorno ai metodi nell'oggetto marcato. La gestione delle transazioni è solo un esempio di come i comportamenti possono essere iniettati. I proxy implementano la stessa interfaccia delle classi che vengono annotate e quando gli utenti invocano i metodi dell'oggetto, le chiamate sono intercettate e i comportamenti iniettati usando il meccanismo appena descritto. Tuttavia, questo procedimento è valido solo nel caso in cui le chiamate siano effettuate dall'esterno. Se i metodi dovessero essere invocati internamente all'oggetto, il meccanismo dei proxy viene bypassato.

Per quanto riguarda il Controller invece, si utilizza la annotazione @Controller, che segnala a Spring quale classe viene selezionata per tale ruolo, ed è usata in associazione a @RequestMapping, usata per i metodi di gestione delle richieste. Questi ultimi metodi sono fondamentali per indicare a Spring come mappare le richieste dei metodi che avvengono nel contesto della JSP. Come si nota dallo schema infatti, i metodi sono quelli che elaborano le azioni dell'utente tramite dei metodi ben specifici. L'annotation @RequestMapping(method = RequestMethod.POST) o @RequestMapping(method = RequestMethod.GET) è la forma più generale delle derivate @PostMapping o @GetMapping e il loro uso è equivalente: contiene l'indicazione di come gestire le richieste nel momento in cui il metodo viene invocato.

3.2.5 Boundary

Un elemento Boundary, come suggerisce il nome, si trova alla periferia del sistema, ma ancora al suo interno. Gli oggetti appartenenti a questa categoria, denominati "front end", accettano gli input dall'esterno, mentre altri elementi, denominati "back end", gestiscono le comunicazioni per supportare gli elementi al di fuori del sistema. In riferimento alla Figura 3.5, due esempi di elementi Boundary per una applicazione customer service sono il front end MarketingCampaignForm e il back end BugdetSystem element. The MarketingCampaignForm gestisce lo scambio di informazioni tra utente e sistema e BudgetSystem può gestire invece lo scambio di informazioni tra il sistema e altri sistemi esterni che gestiscono i budget.

Nel caso in esame, all'insieme di classi Boundary appartengono le Java Server Pages (JSP) realizzate:

- home.jsp
- execute.jsp

Le JSP sono una tecnologia server-side che permette la creazione di metodi per la costruzione di applicazioni dinamiche, indipendenti dalla piattaforma e web-based. Hanno accesso all'intera famiglia delle Java API, compresa JDBC per accedere ai database.
Come già menzionato all'inizio di questo capitolo, non c'è una distinzione in base agli utenti, in quanto si considera una singola categoria che può accedere agli elementi del sistema.
La prima JSP realizza la home page dell'applicazione web, a cui l'utente si collega e tramite cui ha la possibilità di elaborare il modello in base alle preferenze, mentre la seconda è la pagina di output a cui si viene ridirezionati una volta che il modello è stato eseguito e i risultati prodotti. Ulteriori dettagli su come il processo di elaborazione del modello avvenga, verranno forniti nella sezione successiva.

3.2.6 Integrazione Modello-Interfaccia

Per rendere funzionante l'intero sistema modello-interfaccia è stato necessario introdurre un meccanismo che permettesse di rendere automatica l'intera sequenza di passaggi una volta ottenuti i parametri di configurazione. Complessivamente ogni step del modello è stato implementato in

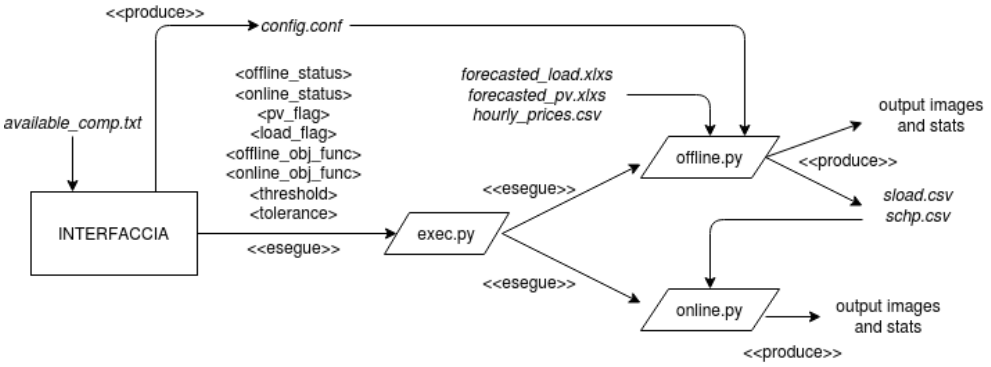

Figura 3.13: Sistema di invocazione degli script

uno script specifico in Python, ma serviva innanzitutto un intermediario tra l'interfaccia e gli script stessi. Il sistema complessivo è composto da quattro componenti: interfaccia, uno script di esecuzione denominato `exec.py`, uno script per la fase offline `offline.py` e uno per la fase online `online.py`. Gli script delle due fasi possono essere utilizzati o meno

in base al valore del flag associato al loro stato.
Il metodo per azionare l'invocazione degli script del modello matematico descritto nel capitolo precedente è il metodo *execute()* che si trova nella classe ComponentController.
Viene di seguito riportato, con alcune parti scritte in pseudo-codice a scopo riassuntivo, il metodo execute().

```
@RequestMapping(value="/execute", method = RequestMethod.POST)
    public void execute(HttpServletRequest request) throws IOException{

        //redirezionamento dello stderr su file

        // apri il file "config.cong", se non esiste crealo
        // inizializza una lista vuota
        // prendi in ingresso le liste dei componenti,
        // attributo di ogni elemento POD
        // inserisci tutte le configurazioni nel file

        try {
            process = Runtime.getRuntime().exec("python exec.py"
                        + " " + request.getParameter("opt1") + " " +
    request.getParameter("opt2")
                        + " " + request.getParameter("uc1") + " " +
    request.getParameter("uc2")
                        + " " + request.getParameter("objOff")
                        + " " + request.getParameter("objOn")
                        + " " + request.getParameter("threshold") + " "
    + request.getParameter("tolerance"));
            process.waitFor();

        //gestione delle eccezioni
        // redirect alla pagina di output
    }
```

Listing 3.3: Metodo execute()

L'importanza di questo metodo consiste nei diversi accorgimenti presi per la successiva esecuzione del modello matematico. In primo luogo c'era la necessità di compilare una lista ordinata di POD con tutte le informazioni necessarie da mandare in input agli script del modello, contenuta dal file `config.conf`. All'interno del file, ogni riga è compilata nel seguente formato:

$$pod :< id >:< index >: [< listofelements >]$$

In tale maniera è possibile, all'interno degli script del modello, accedere alla lista di elementi e andare ad inserire nelle diverse fasi dell'ottimizzazione tutte le equazioni relative ad ogni elemento, associate all'indice del POD.
Gli attributi `<id>` e `<index>` sono necessari nel momento in cui si vanno a considerare i file di input del modello, quali i file xlxs in cui sono contenuti i valori delle stime di carico e fotovoltaico, per andare a recuperare

i profili corrispondenti agli indici. La struttura associata ad ogni POD viene poi ampliata andando ad inserire un ulteriore campo: per ogni valore stimato (nel caso in esame solo fotovoltaico e carico sono associati ad una stima), viene inserito il rispettivo profilo sotto forma di Dataframe, struttura tipica di Python usata per memorizzare dati. L'idea è quella di avere dunque un oggetto in cui contenere tutte le informazioni utili relative ad un POD.

Per l'invocazione si è scelto di ricorrere al metodo `Runtime.exec()` messo a disposizione dal package `java.lang`. Uno dei metodi per ispezionare l'ambiente di esecuzione corrente di Java è utilizzare il metodo statico `getRuntime()` ed è l'unico modo per ottenere un riferimento all'oggetto Runtime. Usando il riferimento, si possono eseguire dei programmi esterni usando il metodo `exec()` della classe Runtime. In particolare viene utilizzata una versione overloaded in cui al metodo si passa come parametro la stringa contenete il comando che esegue l'invocazione al programma desiderato.
Nell'utilizzo di questo metodo si nascondono sono spesso molte insidie, tra cui eccezioni particolarmente difficili da risolvere, come la *IllegalThreadStateException*. Il problema in questo caso è ottenere delle informazioni sui codici di uscita del programma, e ci sono diversi metodi per farlo. Il più efficace è sicuramente il metodo `waitFor()` che deve essere però usato con cautela, soprattutto se è possibile incorrere in deadlock causati da programmi esterni. Un altro accorgimento fondamentale, è quello di utilizzare dei metodi (come è stato fatto nell'applicazione) per gestire gli stream di input, errore e di standard output, in quanto alcune piattaforme potrebbero fornire un buffer di dimensione limitata per gli stream, causando l'interminabile attesa del completamento dell'invocazione. Rimane buona pratica eseguire dei controlli sui codici di uscita dei metodi e gestire le eccezioni in cui si potrebbe incorrere nella maniera più esaustiva possibile.

Una volta compilato il file di configurazione si può procedere a invocare lo script denominato `exec.py`: l'utilizzo di un file intermediario tra interfaccia e gli script specifici per ogni fase dell'ottimizzazione si è rivelato indispensabile per gestire la concorrenza dei processi.
Multiprocessing, in generale, si riferisce all'abilità si un sistema di supportare più di un processo allo stesso tempo: le applicazioni in un sistema multiprocessing sono suddivisi in routines più piccole che vengono eseguite indipendentemente. Il sistema operativo alloca i thread al processore

migliorando le performance del sistema.
In particolare: ad ogni fase dell'ottimizzazione viene associato un processo, in realtà creato solamente se il valore del flag corrispondente, settato tramite l'interfaccia, è a 1.
Tuttavia, nel caso in cui entrambi i processi siano creati, bisogna considerare la sequenzialità della loro esecuzione: lo step offline deve essere eseguito per primo, e solo quando sarà terminato e avrà generato in uscita i valori shiftati (memorizzati in file esterni) potrà proseguire l'esecuzione dello step online.
Di seguito, un estratto dello script `exec.py`.

```
[...]
# import delle librerie necessarie
# redirezione dello stderr e controllo sul numero degli argomenti

        if sys.argv[1] == 'on':
            p1 = multiprocessing.Process(target=offline.run_offline,
    args=(15, sys.argv[3], sys.argv[4], sys.argv[5], sys.argv[7], sys.
    argv[8]))
            p1.start()
            p1.join()
        if sys.argv[2] == 'on':
            p2 = multiprocessing.Process(target=online.run_online, args
    =(15, sys.argv[6], sys.argv[1]))
            p2.start()
            p2.join()
```

Listing 3.4: Codice di exec.py

Per una maggiore comprensione, si consideri che gli argomenti sono passati all'invocazione dello script nel seguente ordine:

1. Stato della fase offline,
2. Stato della fase online,
3. Flag dell'incertezza sul fotovoltaico,
4. Flag dell'incertezza sul carico,
5. Funzione obiettivo della fase offline,
6. Funzione obiettivo della fase online,
7. Valore di soglia (solo per la funzione obiettivo 10.c descritta nel paragrafo 2.2.1),
8. Valore di tolleranza (solo per la funzione obiettivo 10.c descritta nel paragrafo 2.2.1).

Nel caso in cui alcuni flag non fossero attivi, ad esempio quelli dell'incertezza su fotovoltaico e carico, il valore passato sarà `null`, esattamente come il valore di soglia o tolleranza saranno pari a 0 in tutti i casi in cui non si stia considerando la funzione obiettivo che minimizza la distanza da un profilo desiderato.
Conoscere lo stato delle fasi offline e online è indispensabile per una corretta esecuzione del modello, soprattutto nel caso in cui la fase offline non venisse considerata. Passando il valore del flag è possibile indicare, nell'invocazione dello script della fase online, che manca la prima parte di ottimizzazione, e quindi i file in cui sono contenuti i valori degli shift, sono inesistenti.
In tale maniera è possibile evitare l'evenienza in cui dei file possano essere rimasti da iterazioni precedenti, e quindi erroneamente considerati.
Nell'eventualità in cui l'invocazione venisse eseguita, per qualche ragione, in maniera non corretta, ci sono ulteriori meccanismi di controllo negli script di entrambe le fasi, e viene dunque eseguito il settaggio dei parametri appena elencati a valori standard. Questa possibilità è in ogni caso scongiurata utilizzando l'invocazione tramite interfaccia web.

4 Risultati sperimentali

Lo scopo dell'elaborazione del modello matematico proposto è trovare un compromesso tra astrazione dei componenti e complessità computazionale per riuscire poi a rappresentare nella maniera più efficace possibile un sistema a componenti distribuiti.
Uno degli aspetti che maggiormente sono risultati rilevanti nell'approccio proposto è la flessibilità, per poter considerare situazioni in cui la domanda di carico da parte dei clienti sia variabile, e offrire la possibilità di un margine di elasticità, permettendo uno spostamento del carico ad intervalli di 15 minuti, al mercato e alla rete lungo l'asse temporale.
La flessibilità è identificata dalla quantità compresa tra la stima del carico ottenuta da dataset pubblici e l'energia prodotta dalle risorse interne disponibili, considerando anche la possibilità di associare la flessibilità a sistemi CHP, in modo da poter soddisfare al meglio la richiesta in base alla disponibilità energetica corrente del sistema. Tramite i vari componenti considerati nello studio, è possibile creare delle configurazioni relative ad ogni singolo POD, e quindi creare poi un insieme di elementi che si coordinano tra loro cercando di raggiungere l'obiettivo comune di minimizzazione prefissato.
Va considerata inoltre la situazione in cui questa flessibilità sia soggetta all'incertezza delle risorse rinnovabili presenti nel sistema.

Sono stati considerati tre modelli con tre diverse funzioni obiettivo:

- Minimizzazione i costi operativi

$$z = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} c_{Gin}(t) P^s_{Gin}(t) + c_{CHP} \tilde{P}^s_{CHP}(t) + c_{Sout} P^s_{Sout}(t) - c_{Gout}(t) P^s_{Gout}(t)$$

- Minimizzazione l'uso della rete

$$z = \frac{1}{|S|} \sum_{s \in S} \sum_{t \in T} c_{Gin}(t) P^s_{Gin}(t) + c_{Gout}(t) P^s_{Gout}(t)$$

- Minimizzazione della distanza da un profilo desiderato

$$z = \frac{1}{|S|T} \sum_{s \in S} \sum_{t \in T} (\tilde{P}^{s}_{Load}(t) \text{ - } P_{DesiredLoad}(t))^2$$

L'Energy Management System deve essere in grado di soddisfare la richiesta di carico per ogni istante di tempo t.
L'analisi, effettuata in prima istanza su singolo POD, sarà suddivisa in alcune macro-categorie.
Innanzitutto due categorie sono determinate dalle fasi di ottimizzazione offline e online: per ciascuna di esse saranno poi esaminati i risultati in base alle funzioni obiettivo disponibili e in base alle configurazioni di POD adottate. A titolo esemplificativo, è stata analizzata la diversa risposta del modello in una delle configurazioni proposte applicando l'incertezza su uno o entrambi i componenti di carico e fotovoltaico.
In seguito, l'analisi procede in modo analogo considerando il caso in cui il numero di POD sia molto più elevato: in questa maniera è possibile simulare una situazione di aggregazione tra diversi nuclei e testare allo stesso tempo la scalabilità della soluzione proposta.
Lo schema ad albero in figura mostra la struttura dell'analisi eseguita. In questo capitolo non sono riportati tutti i casi presi in esame durante l'analisi (come raffigurato nell'albero), ma solo quelli significativi o particolarmente degni di nota.

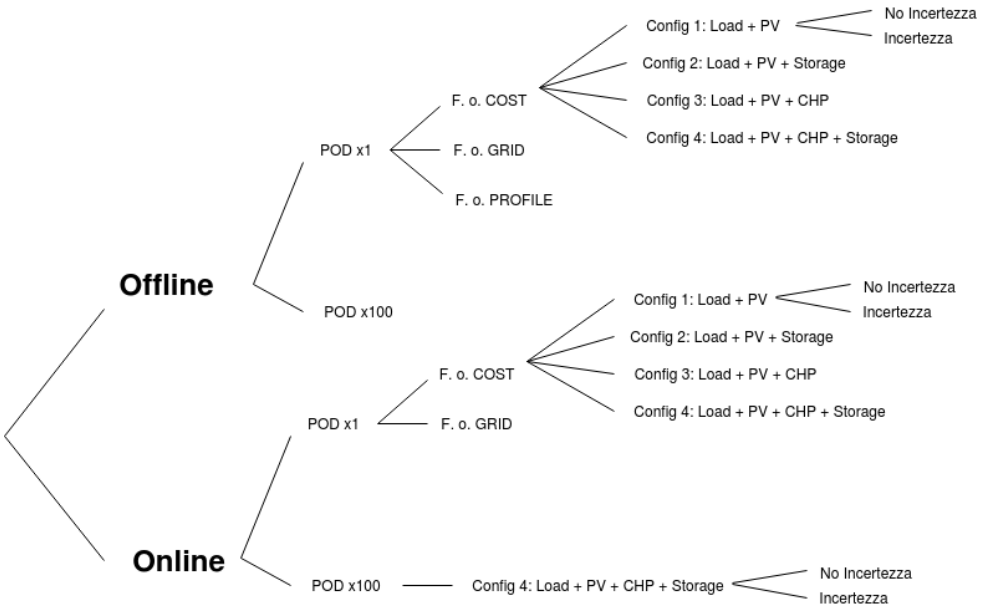

Figura 4.1: Struttura dell'analisi sperimentale

4.1 Descrizione Dataset

I test sono effettuati a partire da istanze del dataset pubblico in [27]. Nel dataset sono presenti 100 profili giornalieri i cui valori sono ricavati ad intervalli di 5 minuti, dalle 00:00 alle 23:00. Dal momento che il modello in esame considera time step di 15 minuti, i profili sono stati aggregati tramite una funzione di scaling che prende come parametro il valore dell'intervallo di tempo desiderato (dunque modulabile anche per altre risoluzioni) e che genera in output i file contenenti i profili agli intervalli definiti.
Questa fase di pre-processing dei dati è stata effettuata sia per i valori di fotovoltaico sia per i valori di carico. Analizzando i profili è possibile notare che la produzione fotovoltaica sia maggiormente concentrata nelle ore centrali della giornata. Dall stesso dataset sono stati ricavati anche i valori massimi e minimi della produzione di un sistema microCHP: si può osservare che ha un andamento a gradino, ovvero mantiene la stessa produzione per un determinato intervallo di tempo.
Per quanto riguarda la modellazione delle finestre temporali a cui associare i valori di flessibilità per la richiesta di potenza di CHP, si sono considerati intervalli di tempo di 4 ore. Questo valore è risultato il miglior compromesso: finestre troppo piccole determinano una variazione frequente del valore della potenza del CHP, mentre finestre troppo estese non risultano sufficientemente elastiche e dunque poco significative per la considerazione della flessibilità.
Infine, i valori orari dei prezzi, come descritto nel Capitolo 2, sono stati ottenuti sulla base dei dati del Gestore dei Mercati Energetici e calcolati in euro/kWh, mentre il prezzo del diesel e del gas naturale[1] sono stati considerati costanti per tutto l'orizzonte temporale delle 24 ore (96 intervalli totali di 15 minuti).

4.2 Caso Singolo POD: fase Offline

Il punto di partenza dell'analisi è stato considerare un singolo POD, modellato sulla struttura del VPP. Dunque abbiamo diversi componenti al suo interno, risorse energetiche di produzione di diverso tipo, che tramite le configurazioni proposte devono soddisfare la domanda di carico.
La fase al momento considerata è quella Offline, in cui l'obiettivo dell'approccio robusto è quello di modellare l'incertezza del carico tramite l'utilizzo degli scenari (se l'incertezza viene effettivamente considerata),

[1] https://ec.europa.eu/eurostat

e ottenere i valori shiftati di carico e di potenza del CHP.
Gli scenari rappresentano delle possibili realizzazioni dei valori delle variabili "incerte", in particolare, considerando il caso in cui l'incertezza sia applicata al carico, si considerano due scenari, uno in cui il valore della potenza associata al carico sia in eccesso rispetto alla stima, un altro in cui sia invece in difetto. Formalmente, se $P_{Load}(t)$ è la potenza del carico all'istante t e si ipotizza di essere in una condizione di incertezza, saranno elaborati due scenari $s_1, s_2 \in S$ tali per cui $P^{s_1}_{Load}(t) = P_{Load}(t) + \delta_{Load}(t)$; $P^{s_2}_{Load}(t) = P_{Load}(t) - \delta_{Load}(t)$.
Si ricorda che tramite il calcolo degli shift si consente alle richieste dei clienti di avere un margine di flessibilità. In particolare, si redistribuisce l'energia lungo l'orizzonte temporale, mantenendo invariata la quantità complessiva di energia, per ridurre la presenza di picchi minimizzando i costi complessivi di gestione del sistema.

4.2.1 Funzione obiettivo di minimizzazione dei costi

La funzione obiettivo, descritta nella Sezione 2.1.6 considera tutti i contributi energetici in entrata e uscita e per ogni istante temporale cerca il minor costo operazionale. Le diverse configurazioni proposte hanno in comune la componente legata alla rete esterna, mentre si differenziano le une dalle altre per gli elementi di produzione considerati. Rendere la rete esterna un componente fisso all'interno della struttura del POD si è rivelato un fattore determinante in molti casi d'uso considerati in questa sezione, come sarà mostrato in seguito.

Configurazione 1: PV

La prima configurazione considera una situazione base in cui il POD contiene un singolo elemento di produzione, un impianto di fotovoltaico, e un elemento di consumo, il carico. Dal momento che la fase in esame è quella offline, i valori considerati per carico e fotovoltaico sono delle stime prese da dataset pubblici presenti in [27].
A partire da questi valori l'approccio robusto calcola per ogni istante di tempo t i valori shiftati di carico in uscita e quelli dei flussi energetici per ogni variabile decisionale presente nel sistema, in questo caso solo la domanda complessiva di carico composta dalla quantità di carico stimata e quella shiftata.
Si rimanda a quanto detto nel Capitolo 2: per ogni configurazione sono stati considerati due casi, il primo caso in cui l'incertezza venga considerata, e quindi vengano introdotti nel modello i rispettivi scenari, e il

caso in cui l'incertezza non venga invece considerata.
Di seguito vengono raffigurati i seguenti casi: in ordine, la situazione in cui l'incertezza non sia applicata, la situazione in cui venga invece applicata prima alla singola variabile di carico, poi alla variabile del fotovoltaico, infine il caso in cui l'incertezza sia applicata ad entrambe le variabili.
Si ricorda che gli scenari sono associati solamente alla variabile su cui viene considerata l'incertezza: se ad esempio il carico viene considerato una variabile "incerta" e la potenza del fotovoltaico no (situazione non possibile ma utile a finalità di testing), ci saranno unicamente due scenari in cui si considerano le possibile realizzazioni della variabile del carico, mentre alla potenza del fotovoltaico sarà associato un singolo valore, ovvero la stima.

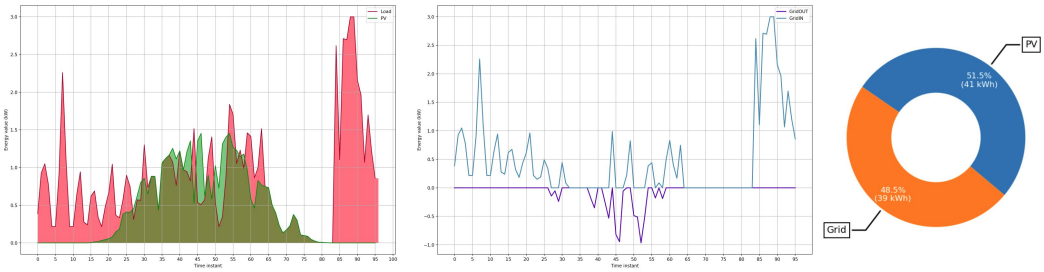

Figura 4.2: Incertezza non applicata

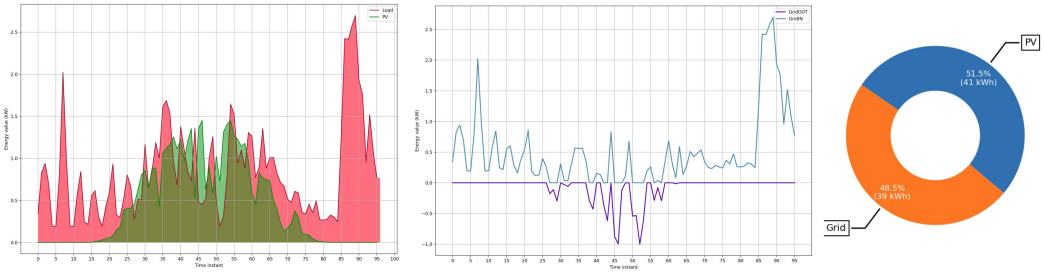

Figura 4.3: Incertezza applicata unicamente al carico

Per ogni configurazione si utilizzano complessivamente tre figure per la descrizione delle variabili in gioco: la prima figura contiene i soli contributi di carico e fotovoltaico, a confronto, mentre la seconda immagine contiene tutti gli altri contributi a seconda della configurazione.
Accanto a questi grafici viene riportato un diagramma a ciambella con le percentuali e i valori in termini di kWh dei diversi contributi per una

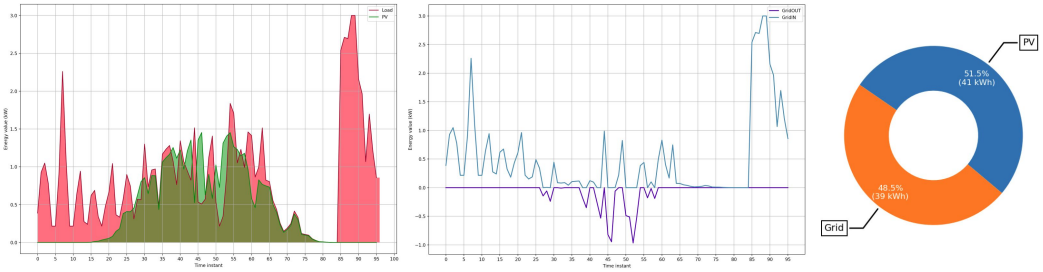

Figura 4.4: Incertezza applicata unicamente al fotovoltaico

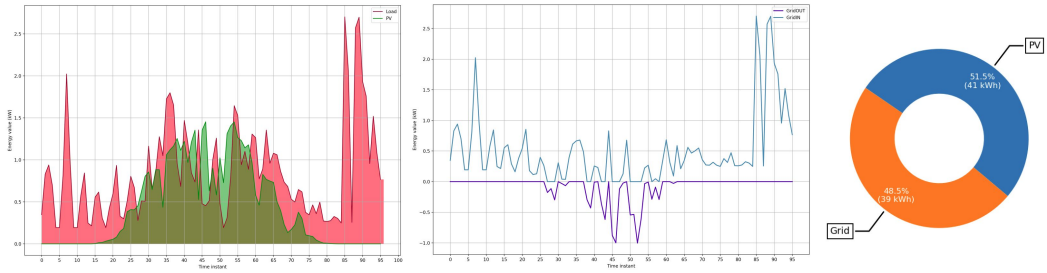

Figura 4.5: Incertezza applicata a carico e fotovoltaico

consultazione più immediata. Per ottenere i valori mostrati nel grafico si sommano i contributi totali delle iterazioni: per una corretta lettura e interpretazione si consiglia di associare sempre ai grafici a ciambella le altre immagini prodotte. L'interpretazione basata sulle singole percentuali e i contributi complessivi potrebbe essere fuorviante.

In ordine: in figura 4.2 è possibile notare il caso in cui l'incertezza non è applicata a nessun componente: le curve delineate in rosso (carico) e in verde (fotovoltaico), ricalcano perfettamente le stime prese in ingresso dal dataset.
Il comportamento del modello è quello di comprare energia dalla rete nelle prime e nelle ultime ore del giorno quando non è in grado di soddisfare la domanda utilizzando le risorse interne (il solo fotovoltaico), e andare a vendere invece il surplus quando ne rimane in eccesso nelle ore centrali della giornata.
La situazione cambia leggermente nel caso in cui l'incertezza sia applicata solamente al carico: come si può notare in figura 4.3, la richiesta da parte dei clienti, rappresentata dall'area colorata di rosso, aumenta. Di conseguenza anche l'energia richiesta dalla rete aumenta.
In generale le differenze tra le immagini 4.2 - 4.4 sono poco evidenti:

situazione comprensibile considerando che i grafici si ottengono calcolando il contributo complessivo su tutti gli scenari e facendone una media, anche osservando i grafici a ciambella di tutte e quattro le figure si può notare che i contributi variano di poco nelle situazioni presentate.
Per le configurazioni seguenti saranno presentati solamente i grafici in cui l'incertezza è considerata su entrambi i componenti o su nessuno dei due.

Configurazione 2: PV + CHP

Di seguito, la configurazione in cui viene in aggiunta considerato l'elemento di produzione CHP.
Nel modello preso in esame vengono considerati contemporaneamente due sistemi CHP differenti, uno a combustione diesel, denominato microCHP, e un altro a gas naturale. La differenza principale tra i due sistemi è innanzitutto il costo molto più ridotto del gas naturale, che è un'ottima scelta anche in termini ecologici poiché riduce il consumo di diossido di zolfo e ossidi d'azoto, ma anche per ragioni di potenza. In generale i sistemi microCHP, come quello considerato (uCHP), sono sistemi di produzione su piccola scala, utilizzabili in ambito domestico, ad esempio.

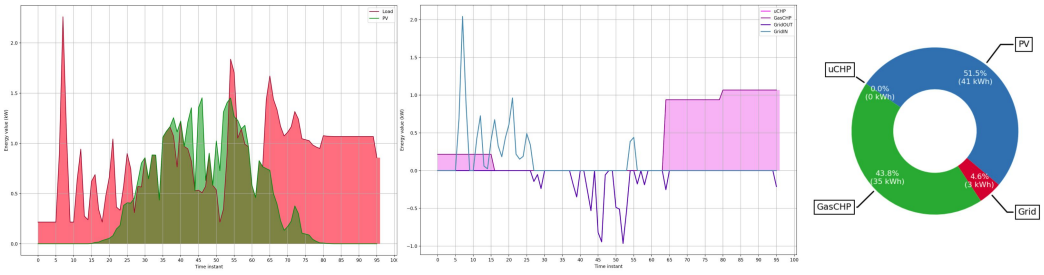

Figura 4.6: Incertezza non applicata

In figura 4.6 è possibile notare l'effetto del processo di *Load Shifting*: in particolare nelle prime e nelle ultime ore della giornata la curva del carico si adatta alla curva del sistema CHP. Questa è la situazione in cui l'incertezza non è considerata: il modello preferisce soddisfare la domanda di carico utilizzando il sistema CHP a gas naturale per una semplice ragione economica, è più conveniente.

In figura 4.7 si nota invece che il contributo è completamente assente: la ragione di ciò è sempre la funzione obiettivo. Si possono notare dei

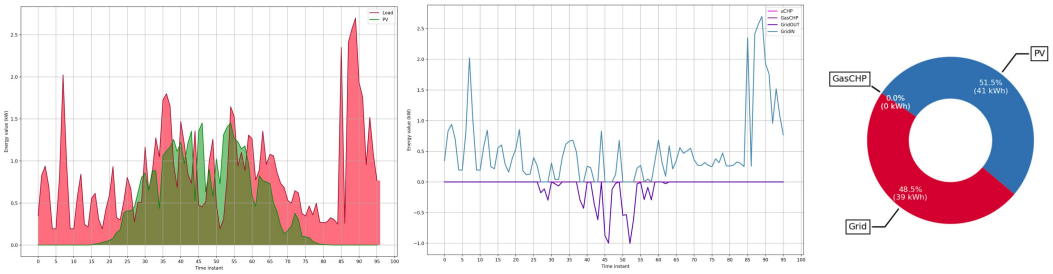

Figura 4.7: Incertezza applicata

contributi della rete in entrata (quindi energia che viene comprata dalla rete) principalmente nelle prime e nelle ultime ore della giornata. Solitamente il prezzo dell'energia è più conveniente. Dal momento che in una situazione di incertezza si considerano degli scenari che introducono delle variazioni dei valori delle variabili "incerte", è possibile che per il modello fosse più conveniente comportarsi nella maniera descritta in figura.
In aggiunta si ricorda che è stato introdotto un processo il cui concetto alla base riprende quello del Load Shifting, anche per i sistemi CHP. Sono state modellate delle finestre temporali di durata di 4 ore (in modo da non avere intervalli troppo piccoli e specifici nè troppo ampi e poco rappresentativi) in cui al valore della potenza del sistema CHP viene concessa una flessibilità, modellata tramite degli shift.
I vincoli per ciascuna finestra sono stati modellati in supporto all'eventuale mancanza di produzione interna da parte del fotovoltaico, e ridotti dunque quando si suppone di averne in abbondanza. Considerando due sistemi CHP, a entrambi dei quali è stato associato il concetto di flessibilità, è stato possibile far in modo che i vincoli nelle finestre temporali si bilanciassero in modo da non rimanere mai completamente sprovvisti di produzione interna (se i sistemi CHP sono presenti nella configurazione). Per ulteriori dettagli sui vincoli considerati nel modello si rimanda alla Sezione 2.1.4.

Configurazione 3: PV + Storage

Nella terza configurazione viene invece introdotto un sistema di storage: la funzione è duplice, nel caso in cui la produzione interna non sia soddisfatta dal solo fotovoltaico è possibile ricorrere al sistema di storage; se invece c'è un eccesso di energia prodotta in un determinato istante di tempo si può scegliere di immagazzinare l'energia per usufruirne dunque in un secondo momento.
Analogamente al contributo della rete, il contributo dello storage è stato modellato usando una singola curva che rappresenta la sua potenza in uscita, dunque immessa sulla rete P_{Sin}, a cui va sottratto il contributo in entrata, dunque in uscita dalla rete, P_{Sout}.
In questo lavoro è stato considerato sufficiente valutare per ogni istante di tempo la quantità di energia immagazzinata nella batteria, determinata dalla quantità di carica dell'istante precedente e da eventuali quantità di energia sottratte o aggiunte nell'istante temporale corrente. Si assume che ogni t sia sufficientemente lungo per evitare lo stress della batteria e dei livelli di degrado della stessa e che il sistema di storage sia inizialmente carico, anche se non completamente, e che esistano limiti per l'estrazione e l'inserimento di energia in esso. Per ulteriori dettagli sui vincoli considerati nel modello si rimanda alla Sezione 2.1.1.

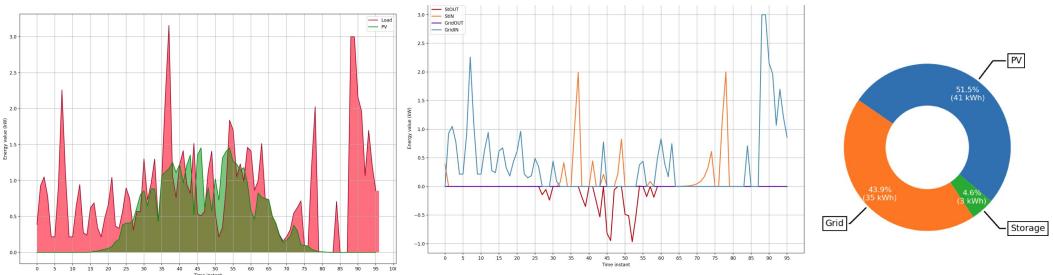

Figura 4.8: Incertezza non applicata

La differenza fondamentale tra le figure 4.8 e 4.9, rispettivamente in cui l'incertezza non è ed è considerata, è l'utilizzo dello storage, in particolare la quantità di energia immagazzinata che si prende da esso per essere immessa sulla rete (curva arancione). Come si può osservare, in figura 4.8 ci sono dei picchi molto più alti, perché si usa solo lo storage per bilanciare il carico, mentre in figura 4.9 si utilizza anche un piccolo contributo della rete (curva azzurra).
In entrambi i casi tuttavia si usa il surplus di fotovoltaico per caricare in parte lo storage. Complessivamente però, i contributi nella situazione in cui l'incertezza viene considerata e nella situazione in cui non viene

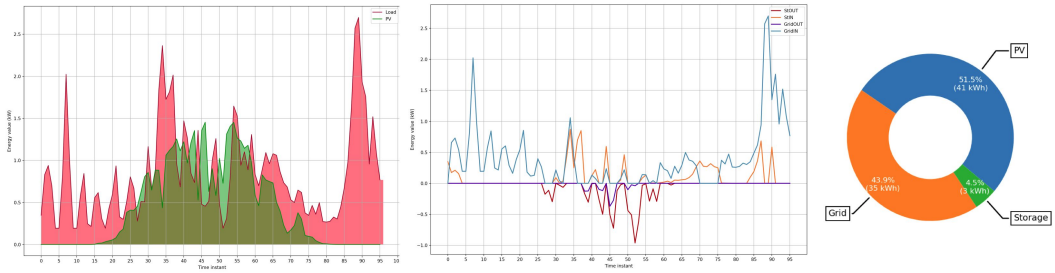

Figura 4.9: Incertezza applicata

considerata, sono paragonabili.
Si ricorda che per l'immissione e l'emissione di energia sulla rete ci sono dei vincoli ben precisi specificati nella sezione 2.1.1.

Configurazione 4: PV + CHP + Storage

Infine viene presentata la situazione in cui tutte le componenti si trovino all'interno del POD.
La situazione è analoga a quella presentata in figure 4.6-4.7. In figura 4.10 (caso deterministico), nelle ultime ore del giorno, il modello decide di utilizzare in associazione alla potenza dello storage la potenza del sistema CHP (ancora una volta quello a gas naturale perché più economico). Interessante notare come nelle ore centrali si affidi principalmente al contributo della produzione di fotovoltaico.
La situazione viene invece totalmente stravolta nel caso in cui si considera l'incertezza: variando la curva del carico varia anche la gestione dei contributi. Ancora una volta il sistema CHP viene accantonato, in favore dello storage e della rete esterna.
Si ricorda che tutte le scelte che il modello prende sono volte al soddisfacimento della funzione obiettivo.

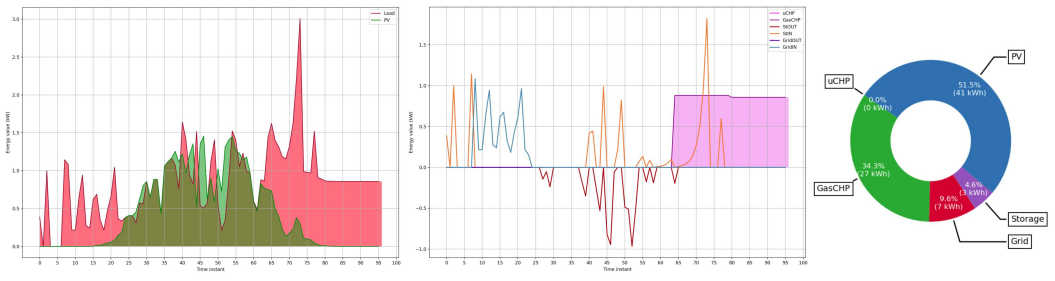

Figura 4.10: Incertezza non applicata

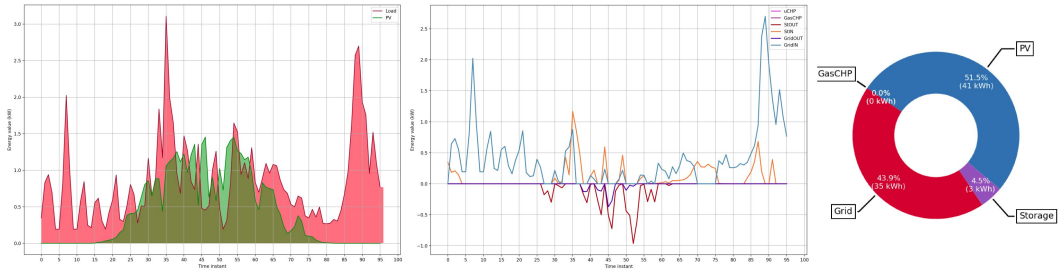

Figura 4.11: Incertezza applicata

4.2.2 Funzione obiettivo per minimizzare l'uso della rete

Un secondo approccio per effettuare la prima fase di ottimizzazione offline è stato considerare una funzione obiettivo che va a minimizzare l'utilizzo della rete. Tale funzione è stata introdotta principalmente per sopperire al comportamento (menzionato nella sezione precedente) dell'EMS nella minimizzazione dei costi operazionali.
Lo scopo è infatti quello di andare a ridurre il più possibile l'interazione del POD con la rete esterna, minimizzando il costo associato alla potenza della rete P_{Grid}, composta dalla somma della potenza in entrata P_{Gin} e quello in uscita P_{Gout}, per ogni istante di tempo t. Analogamente alla sezione precedente, verranno di seguito presentate le diverse configurazioni del POD e il relativo comportamento del modello.

Configurazione 1: PV

La prima configurazione ad essere considerata è quella in cui gli unici elementi presenti all'interno del POD sono la produzione del fotovoltaico e la richiesta di carico. Sono di seguito presentate le situazioni senza e con la considerazione sull'incertezza.

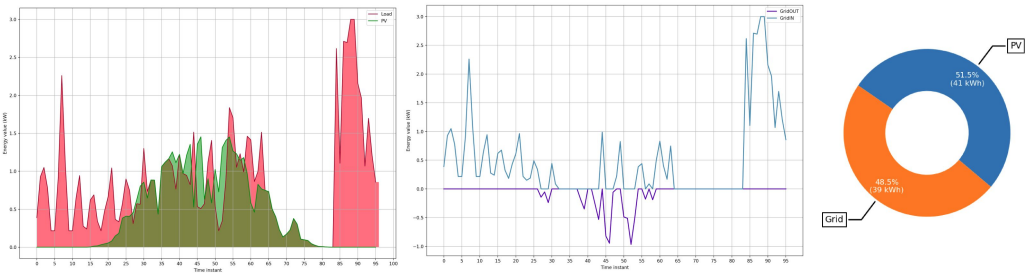

Figura 4.12: Incertezza non applicata

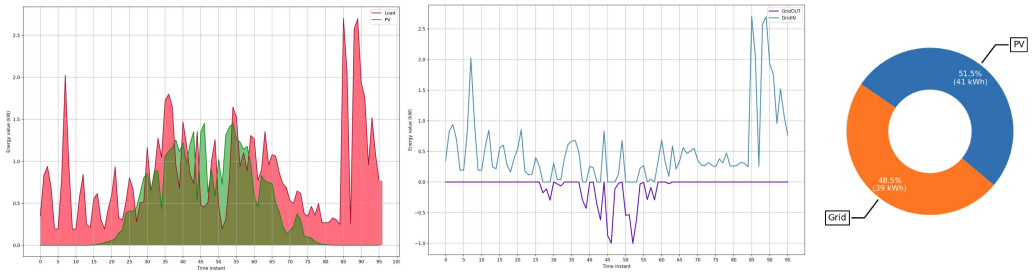

Figura 4.13: Incertezza applicata

I due grafici (figure 4.12 e 4.13) ricordano molto le figure 4.2 e 4.5, ovvero il caso base per la funzione di minimizzazione dei costi. La lieve differenza, poco percepibile per una semplice questione di valori matematici, poiché i contributi disponibili per soddisfare il carico sono gli stessi, risiede proprio nelle curve della rete. In particolare, proprio per specifica della funzione obiettivo, la volontà è quella di limitare il più possibile l'utilizzo della rete esterna in favore degli elementi di produzione interna del POD.

Configurazione 2: PV + CHP

La seconda configurazione considera in aggiunta alla produzione di fotovoltaico la produzione dei sistemi CHP. Ancora una volta sono considerati entrambi i sistemi presentati nel Capitolo 2, dedicato alla costruzione del modello matematico. In figura 4.14 e 4.15 si nota in maniera più eviden-

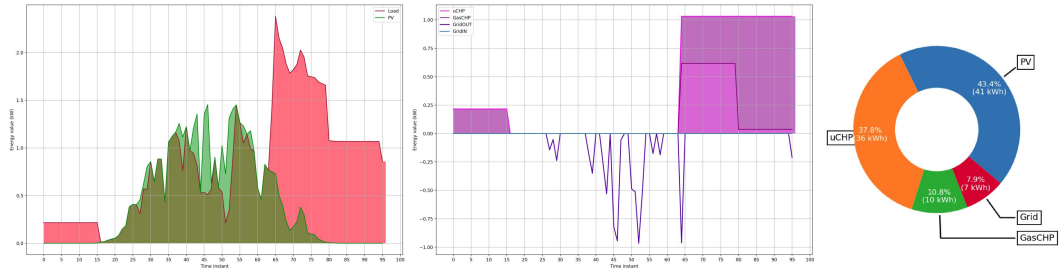

Figura 4.14: Incertezza non applicata

te non solo l'utilizzo dei sistemi CHP, ma anche la presenza del processo di Shifting, sempre riferito a tale componente.
Come descritto in precedenza, nella modellazione dei vincoli relativi alle finestre temporali per la flessibilità associata ai sistemi CHP, si è cercato di dare maggiore supporto ai momenti della giornata in cui il contributo

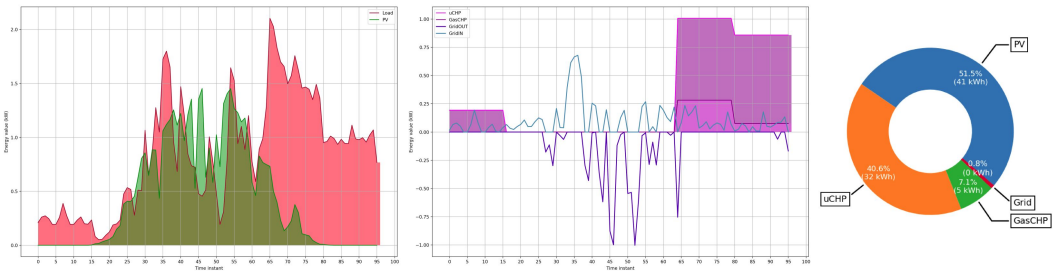

Figura 4.15: Incertezza applicata

di altri elementi di produzione, quali il fotovoltaico siano minori.
In entrambe le figure infatti, si nota come i contributi di entrambi i CHP siano presenti nelle prime e nelle ultime ore del giorno. La differenza tra i due grafici è ancora una volta la seguente: nel caso deterministico (figura 4.14) si utilizzano abbondantemente entrambi i sistemi CHP e l'eccesso della produzione di fotovoltaico viene invece venduta sulla rete esterna. Nel caso invece in cui si introduce l'incertezza (e aumenta la richiesta di carico) è necessario distribuire lungo l'orizzonte temporale le risorse in maniera ben diversa. Maggiore è infatti il contributo della rete, e del CHP a gas naturale rispetto al sistema a diesel.

Configurazione 3: PV + Storage

Di seguito le realizzazioni per le configurazioni con la produzione di fotovoltaico e la disponibilità di un sistema di storage.

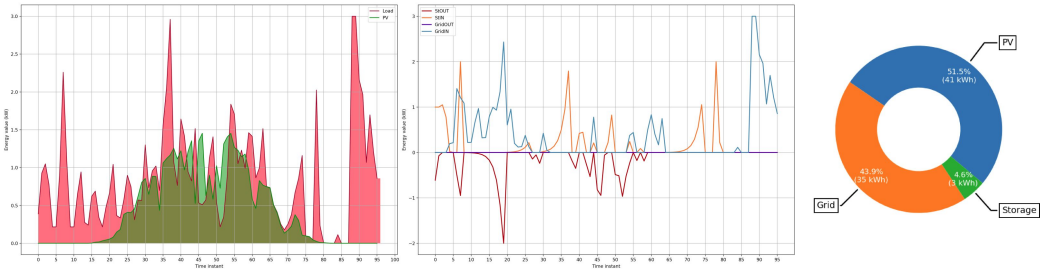

Figura 4.16: Incertezza non applicata

Tra le figure 4.16 e 4.17 si può osservare un comportamento paragonabile, anche i contributi segnalati dal grafico a ciambella sono molto simili. Cambia solo il comportamento della rete in entrata uscita: si ricorda che l'obiettivo finale è quello di utilizzarla il meno possibile.

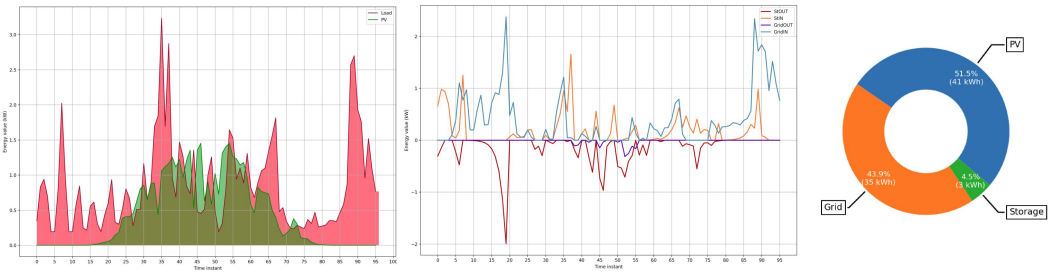

Figura 4.17: Incertezza applicata

Configurazione 4: PV + CHP + Storage

Ultimo caso considerato per la seconda funzione obiettivo introdotta nella modellazione è quello in cui il POD contiene tutti gli elementi di produzione e consumo disponibili.

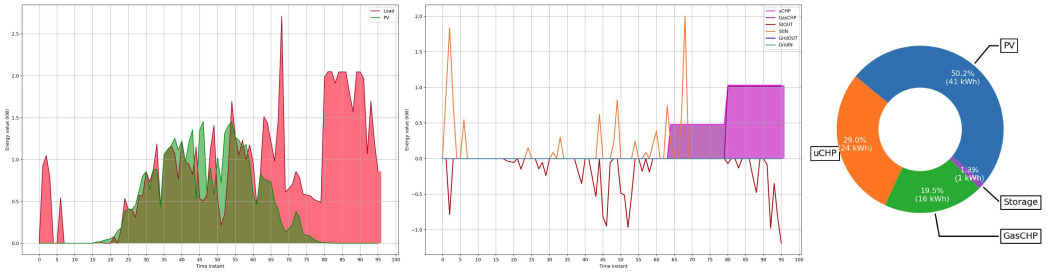

Figura 4.18: Incertezza non applicata

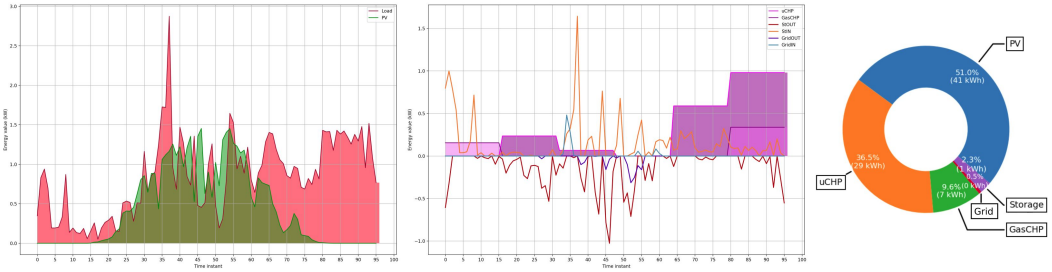

Figura 4.19: Incertezza applicata

I grafici in figure 4.18 e 4.19 mostrano grandi differenze nell'utilizzo dei sistemi CHP. Come già menzionato in altri casi, a causa dell'incertezza è possibile che i valori di carico e fotovoltaico subiscano delle variazioni, e che di conseguenza il modello si trovi a dover prendere delle decisioni

di scheduling dei processi produttivi differenti.
In particolare si può notare che in figura 4.19 il carico è stato distribuito, grazie al processo di Load Shifting lungo l'asse temporale in maniera più efficiente rispetto a quella mostrata in figura 4.18. Ciò consente dunque al modello di poter utilizzare al meglio i propri componenti interni: i sistemi CHP grazie alla flessibilità assegnata loro nelle finestre temporali possono intervenire nel momento il cui il fotovoltaico non è in grado di produrre a sufficienza (nelle prime e nelle ultime ore del giorno), in associazione al sistema di storage, che ha la duplice funzione di immagazzinare energia per poi inserirla sulla rete in un secondo momento, o usufruire della sua carica interna per bilanciare il carico.
Si ricorda che i costi associati alla potenza della rete e alla potenza dello storage sono riferiti agli stessi valori presi da un dataset pubblico (descritto nell'introduzione al capitolo), dunque la scelta di quale contributo usare dipende interamente dalla funzione obiettivo.
In questo caso dal momento che si mira alla minimizzazione dell'uso della rete esterna, è chiaro che la priorità tra i due contributi sia dato allo storage.

4.2.3 Funzione obiettivo per minimizzare la distanza da un profilo desiderato

L'ultima funzione obiettivo introdotta, unicamente nella fase offline dal momento che è l'unica fase in cui si considera il processo di shifting del carico e del CHP (nella fase online si utilizzano i valori di shift prodotti al termine di questa fase di ottimizzazione). In particolare si assume che la richiesta del profilo desiderato, identificato con un profilo piatto generato tramite script, giunga da un operatore energetico esterno, che fornisce i dettagli energetici della struttura e la soglia con cui livellare i picchi del profilo di carico.
In breve, il procedimento utilizzato per generare il profilo piatto considera uno o più profili (in base al numero dei POD dell'infrastruttura), i cui picchi vengono livellati in base ad un livello di soglia e uno di tolleranza. In particolare la soglia si riferisce alla percentuale di picco da rimuovere e "appiattire", percentuale che sarà poi distribuita in altre parti della curva in maniera graduale, in modo da ottenere un profilo tendenzialmente piatto.
Il livello di tolleranza è invece stato introdotto per evitare di ottenere una linea che rimane costante lungo l'orizzonte temporale, situazione poco realistica in ambito di produzione energetica.

Il livello di soglia considerata per tutti i profili presentati è del 60%, con una tolleranza del 10%.
Si ricorda che la scelta del valore della tolleranza è vitale: se la soglia dovesse essere troppo bassa (dunque i picchi vengono abbassati di poco) il profilo desiderato risulterà poco piatto; al contrario se la soglia è troppo alta si potrebbero avere dei problemi nella generazione del profilo e ottenere un risultato poco rappresentativo.

Nelle sezioni successive sarà presentato come caso di studio per ogni funzione obiettivo, la sola situazione in condizioni di incertezza, in quanto la situazione opposta non è particolarmente rappresentativa e degna di nota.

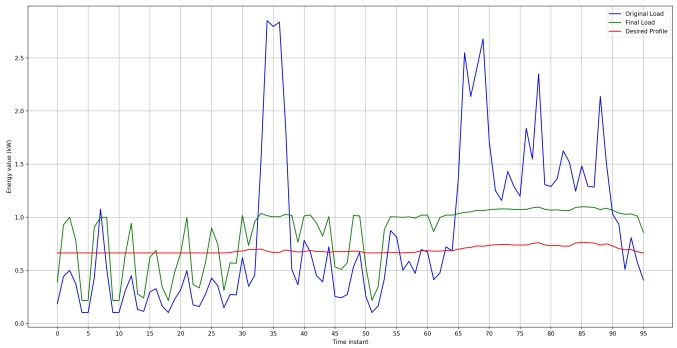

Figura 4.20: Confronto tra le curve di profilo

Infine per calcolare la distanza tra il profilo ottenuto dal processo di shifting del carico e il profilo desiderato si è deciso di utilizzare la funzione di varianza, che misura per definizione la distanza di una variabile rispetto ad un valore stimato, ed è individuata dalla media della differenza al quadrato tra un individuo e il valore atteso. Come risultato si otterrà dunque una valutazione della quantità di dispersione rispetto al valore attteso, più è piccola, più ci si avvicina.
In figura 4.20, un esempio di output della funzione di generazione del profilo desiderato: sono indicati il profilo iniziale stimato, quello piatto ottenuto tramite il processo appena descritto, e il carico finale al termine della fase di ottimizzazione offline.

Configurazione 1: PV

Si noti innanzitutto come i grafici si allontanino molto da quelli visti in precedenza. In particolare la curva del carico: questo è ovviamente dovuto alla funzione obiettivo usata.
Per ogni configurazione saranno visualizzati gli stessi output rispetto alle configurazioni viste in precedenza, in aggiunta verrà considerato un grafico in cui osservare le variazioni del grafico iniziale, finale e del profilo desiderato.

Figura 4.21: Incertezza applicata

In figura 4.21 viene rappresentato il caso in cui nel POD sia contenuto solo l'elemento di produzione del fotovoltaico.
Il motivo per cui la potenza rete in entrata e quello della rete in uscita si assomigliano così tanto è dovuta alla modellazione matematica della fase di ottimizzazione.
Si ricordi infatti l'equazione del bilanciamento del carico, nel paragrafo 2.1.5. Il modello impone che per ogni istante di tempo la somma dei contributi del sistema sia uguale alla quantità di carico finale, denominata $\tilde{P}_{Load}$. In questo caso si considerano solamente tre contributi in totale, fotovoltaico, rete in uscita e rete in entrata, per questa ragione gli andamenti delle due curve vengono così similari.
Come sarà evidente nelle seguenti configurazioni, le analisi effettuate tra-

mite la funzione obiettivo per la minimizzazione della distanza da un profilo desiderato, hanno messo in luce la necessità di effettuare ulteriori considerazioni sui componenti utilizzati, in particolare sulla gestione della rete.

Configurazione 2: PV + CHP

La configurazione che prevede come elemento di produzione i sistemi CHP mette in luce un comportamento poco desiderabile nella realtà: i sistemi CHP vengono quasi sempre mantenuti accessi.
Potrebbe essere necessario effettuare ulteriori considerazioni per rendere più efficiente l'utilizzo di questa risorsa, nel caso in cui l'obiettivo dell'EMS sia l'avvicinamento ad un profilo desiderato, oppure si dovrebbero rendere ulteriormente più stringenti i vincoli nelle finestre temporali definite. In queste circostanze si è preferito mantenerli uguali per tutti i test in modo da avere un coerente metodo di paragone.
L'ulteriore ragione per cui potrebbero essere richieste aggiuntive considerazioni è l'utilizzo rilevante della rete esterna.

L'andamento mostrato in figura 4.22 è analogo a quello visto nella configurazione precedente, con la differenza che il contributo della rete in entrata, quindi l'energia acquistata dall'esterno, è visibilmente minore.
Si ricordi che nel caso in cui gli scenari siano considerati (come quello in esame), il valore della funzione obiettivo finale risulta una media tra tutti gli scenari, che nel caso invece della situazione deterministica è uno solo. Ulteriori considerazioni saranno effettuate nel paragrafo successivo in termini di costo e tempo di esecuzione.

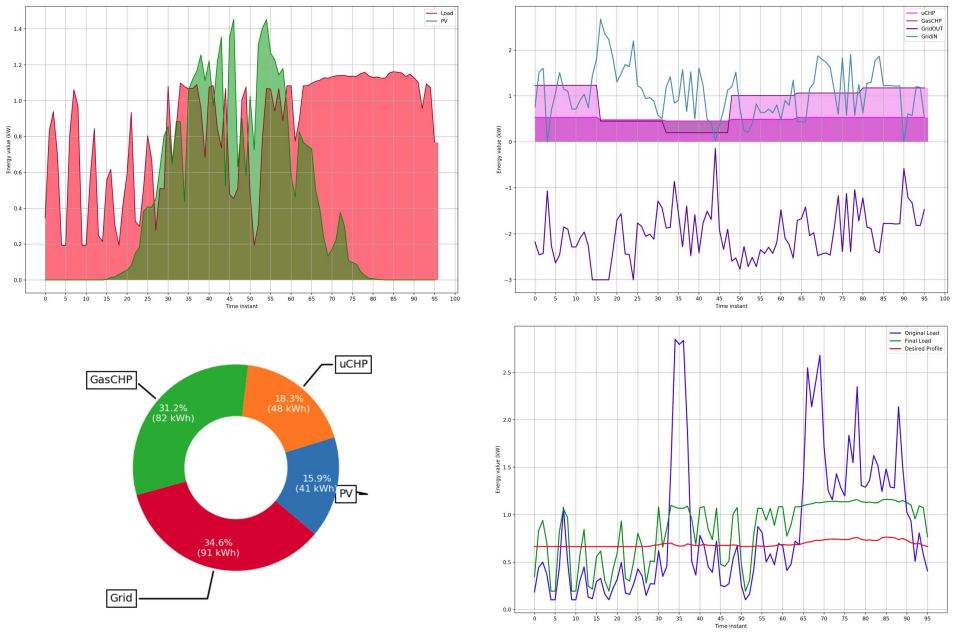

Figura 4.22: Incertezza applicata

Configurazione 3: PV + Storage

Analogamente al caso appena descritto, anche per la configurazione in cui si associa al POD la produzione del fotovoltaico e un sistema di storage è probabile che ulteriori considerazioni siano necessarie.
Si è assunto che la carica e la scarica della batteria avvengano in istanti di tempo sufficientemente distanti da non influire in maniera rilevante sulla degradazione del sistema, ma nonostante ciò il comportamento oscillatorio indicato dal grafico è tutt'altro che ideale. In corrispondenza, anche la curva del contributo legato alla rete esterna segue quello del sistema di storage.
Analogamente non sono state fatte considerazioni approfondite sulla rete esterna, per mantenere un adeguato livello di astrazione, ma realisticamente non è un comportamento auspicabile per via della dispersione di energia lungo le linee e per i tempi richiesti per la trasmissione.
In Figura 4.23 si può notare come il modello consideri di comprare e vendere energia in istanti temporali molto ravvicinati, anche se non strettamente necessario per il soddisfacimento della richiesta di carico. Ancora una volta la ragione di questo comportamento risiede nella funzione obiettivo: il valore finale è una misura della dispersione dei valori del carico finale rispetto a quello desiderato, dunque da un punto di vista

matematico, farà di tutto pur di raggiungere il suo scopo.

Figura 4.23: Incertezza applicata

Configurazione 4: PV + CHP + Storage

Infine, nel caso in cui tutti i contributi disponibili siano considerati, si può notare in figura 4.24 come i problemi emersi nelle configurazioni precedenti siano di nuovo presenti, a partire dalle rilevanti oscillazioni delle curve legate alla rete esterna.
La tendenza in questo caso del modello, il cui comportamento non si differenzia molto nei due casi considerati, è quella di usare al massimo le risorse interne e vendere per quanto possibile sulla rete l'eccesso ricavato in tal maniera.

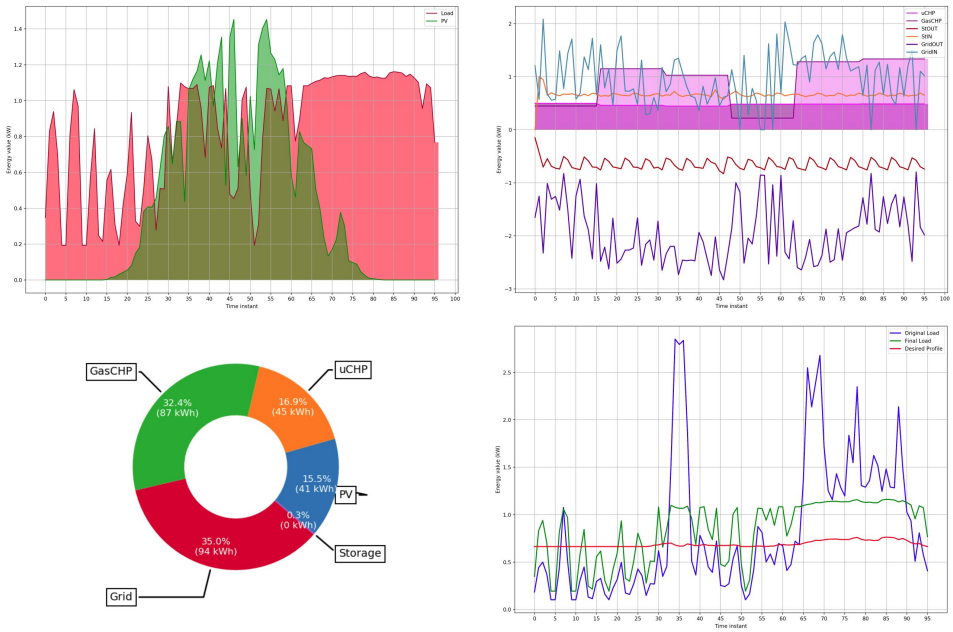

Figura 4.24: Incertezza applicata

4.2.4 Confronto tra le configurazioni

Riassumendo, l'approccio robusto della fase offline ha l'obiettivo di minimizzare il valore della funzione obiettivo e ricavare i valori ottimi degli shift del carico e dei sistemi CHP che poi andranno usati in ingresso nella successiva fase online. Lo step di ottimizzazione è stato implementato utilizzando tre funzioni obiettivo:

- minimizzazione del costo operativo (cost)
- minimizzazione dell'uso della rete esterna e del Mercato (grid)
- minimizzazione della distanza da un profilo desiderato (profile)

Ciascun approccio ha evidenziato alcune affinità ma anche la necessità di ulteriori considerazioni.
Nella Tabella seguente si possono osservare i valori delle prime due funzioni considerate, in cui sono indicati nella casella "NI" i valori della funzione obiettivo nel caso deterministico, e nella casella "I" corrispondente, quelli ottenuti dall'utilizzo degli scenari.
Per una questione di leggibilità il numero di decimali dopo la virgola è stato ridotto a tre. L'unità di misura dei prezzi è euro/kWh.Si ricorda

che l'ottimizzazione offline viene effettuata sull'intero orizzonte temporale (96 time-step).

	Conf 1		Conf 2		Conf 3		Conf 4	
Valore F.O.	NI	I	NI	I	NI	I	NI	I
Cost	2.432	3.794	1.926	3.256	2.152	3.795	1.704	3.256
Grid	2.431	3.794	1.574	1.945	0.372	2.201	0.000	0.307
Profile	0.095	0.164	0.094	0.163	0.096	0.164	0.094	0.164

In primo luogo la funzione di minimizzazione dei costi ha evidenziato un comportamento del modello incentrato unicamente sull'aspetto economico, nel tentativo di ottenere la minore spesa possibile, come è comprensibile che sia. Tuttavia questo approccio ha lo svantaggio di andare a considerare, talvolta in maniera esagerata, il contributo della rete esterna, ottenendo in conclusione delle performance non particolarmente brillanti. Come si può osservare nella Tabella infatti, la funzione per la minimizzazione dell'utilizzo della rete esterna ottiene dei risultati nettamente migliori. Addirittura, nella quarta configurazione considerata, seppur in condizioni di non incertezza, ottiene un risultato ideale: riuscire a bilanciare la produzione con la richiesta.

Per quanto riguarda invece la terza funzione obiettivo, quella per la minimizzazione della distanza da un profilo desiderato, il valore ottenuto consegna un'indicazione della dispersione del risultato ottenuto in confronto al profilo desiderato, per definizione di varianza, ovvero la funziona matematica usata per definire questo approccio. Quindi, minore è il valore migliore è il risultato.
Nonostante i risultati in termini di funzione obiettivo siano promettenti, ci sono sicuramente ulteriori considerazioni da effettuare per la gestione delle risorse interne: per soddisfare il profilo di carico desiderato il modello tende a utilizzare in maniera eccessiva gli elementi di produzione, ancora una volta non badando che l'energia sia strettamente richiesta ed eventualmente vendendola sulla rete. Questo comportamento si spiega tramite l'idea di fondo che risiede dietro a questo approccio: l'interesse non è tanto come utilizzare le risorse interne (che rimangono comunque vincolate esattamente come per gli altri due approcci) ma di quanto ci si avvicina all'obiettivo. Proprio per questa ragione si può dichiarare che quest'ultimo caso di studio necessita di ulteriori perfezionamenti.
In generale l'approccio più efficace per la fase offline è l'utilizzo della funzione obiettivo che va a minimizzare l'uso della rete. Questo implica

che venga data la priorità alle risorse interne per il soddisfacimento del consumo.

4.3 Caso Singolo POD: fase Online

La fase online è stata realizzata tramite un algoritmo greedy in cui vengono determinati i valori reali per le variabili decisionali dei flussi energetici, assumendo che i valori degli shift di carico e dei sistemi CHP siano stati realizzati tramite la fase offline del modello.
A differenza della fase offline, le funzioni obiettivo considerabili nello step in questione, sono quella per la minimizzazione del costo operativo e quella per la minimizzazione dell'utilizzo della rete.
Il motivo è il fatto che il profilo desiderato viene utilizzato per calcolare gli shift del carico che in questa fase non sono più variabili da determinare ma valori già ben definiti. La ragione per cui la fase online non sfrutta il concetto degli scenari è dovuto al fatto che si assume che l'ottimizzazione avvenga il giorno stesso, quindi utilizzando i valori per le sorgenti energetiche a cui era stata associata l'incertezza, che si concretizzano nell'effettiva realizzazione di uno degli scenari considerati invece nella fase del giorno precedente (offline).

Poiché gli unici valori a disposizione sono gli stessi utilizzati per la fase offline, è stata aggiunta una quantità di rumore ai dataset descritti nel Paragrafo 4.1 tramite la funzione `random.normal` della libreria `Numpy` che genera valori a partire da una distribuzione Normale (Gaussiana) con media μ=0 e deviazione standard $sigma$=0.1. In aggiunta vengono azzerati gli eventuali valori che a causa del rumore diventano negativi.

```
1 pv_profiles_df = pv_profiles_df + np.random.normal(0, 0.1, [
      pv_profiles_df.shape[0],pv_profiles_df.shape[1]])
2 pv_profiles_df = pv_profiles_df.clip(lower=0)
```

Analogamente alla sezione dedicata alla fase offline, anche per la fase online saranno esaminati due casi, il primo associato a valori generati da un'ottimizzazione senza incertezza, il secondo invece associato a valori determinati in condizioni di incertezza.

4.3.1 Funzione obiettivo per minimizzare il costo operativo

Configurazione 1: PV

Si suppone dunque che i valori ottimizzati degli shift siano già stati inseriti nelle equazioni della fase online, che sono una versione ridotta del problema MILP usato nella fase offline.
L'andamento sia della curva del carico sia della curva del fotovoltaico ricordano quelle già viste nella fase offline, dal momento che la quantità di rumore considerata non vuole andare a stravolgere i valori da considerare, simulando una situazione verosimilmente realistica.
Il comportamento del modello mostrato in figura 4.25 e 4.26, è esattamente quello che ci si aspetta: ricorre all'acquisto di energia dalla rete esterna nel momento in cui non riesce a soddisfare il carico tramite la produzione interna, quindi nelle prime e nelle ultime ore della giornata, mentre nelle ore centrali utilizza la produzione interna e se rimane un eccesso di energia, questa viene venduta sul Mercato. Analoga la situazione mostrata in figura 4.25, entrambe sono corrispondenti a quelle presentate per la fase offline.

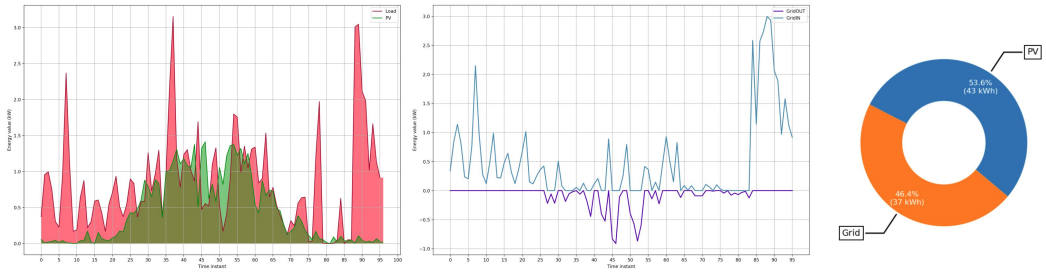

Figura 4.25: Incertezza non applicata nella fase offline

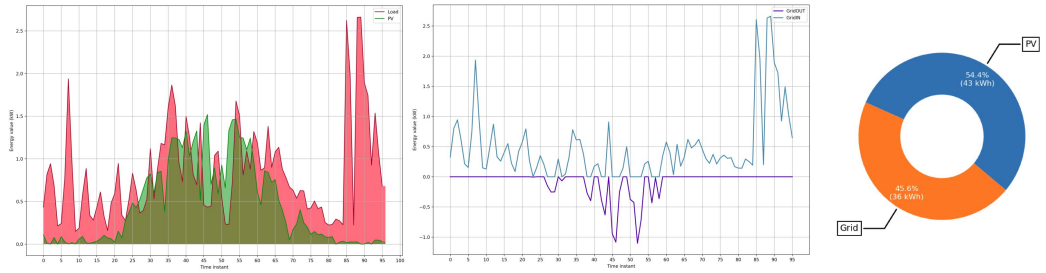

Figura 4.26: Incertezza applicata nella fase offline

Configurazione 2: PV + CHP

Anche utilizzando questa configurazione il comportamento del modello è prevedibile: come mostra la figura 4.26 e 4.27, nelle ore iniziali e finali della giornata utilizza la rete per soddisfare il fabbisogno dettato dal carico, mentre nelle ore centrali si affida al fotovoltaico. In questa configurazione tuttavia sono presenti anche i sistemi CHP.
Vengono usati per la maggior parte come supporto per il fotovoltaico nelle prime e ultime ore della giornata. Si noti il rilevante contributo della rete in uscita (curva blu) che rappresenta la quantità di energia venduta sulla rete grazie al surplus energetico generato dalla produzione fotovoltaica.
La funzione obiettivo considerata è quella della minimizzazione dei costi operativi: se l'EMS ritiene più vantaggioso produrre un surplus, anche se non strettamente necessario, per andare a bilanciare il costo dell'energia comprata nei periodi di scarsa produzione, lo farà.
La differenza tra le due figure deriva dai valori delle rispettive fasi offline. Si ricorda infatti come in generale il modello cerca, in condizioni di incertezza, di distribuire meglio i contributi lungo l'asse temporale e dare maggiore spazio alla rete, dal momento che risulta più conveniente.

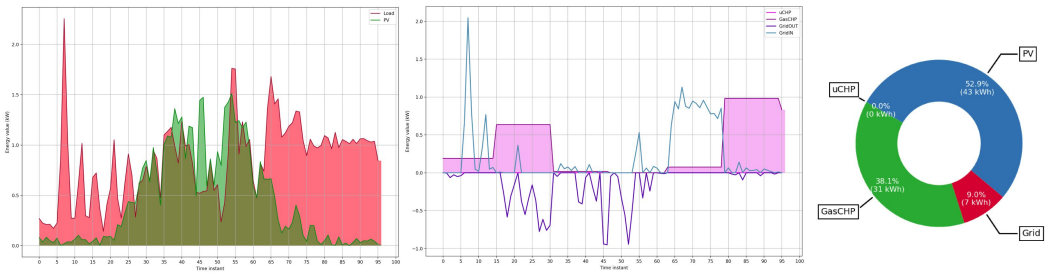

Figura 4.27: Incertezza non applicata nella fase offline

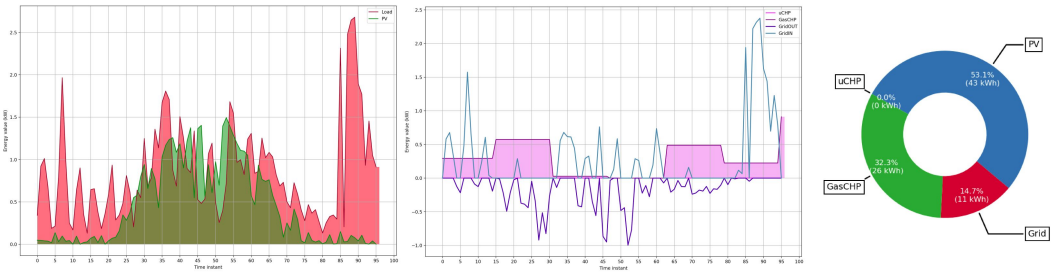

Figura 4.28: Incertezza applicata nella fase offline

Configurazione 3: PV + Storage

In figura 4.28 e 4.29 si possono riscontrare le stesse similitudini già osservate in figure 4.16 e 4.17, ovvero nella fase di ottimizzazione offline. Anche in quel caso gli andamenti delle curve dei componenti erano estremamente simili, questo spiega il perché i due grafici in output siano nuovamente paragonabili.
Interessante la scelta del modello di utilizzare la potenza immagazzinata nello storage solamente nelle prime ore del giorno, mentre nel resto della giornata preferisce affidarsi alla rete esterna per comprare o vendere energia. Si ricorda che al sistema di storage è associato un fattore di efficienza η, che come in questo caso, influisce nelle decisioni del modello.

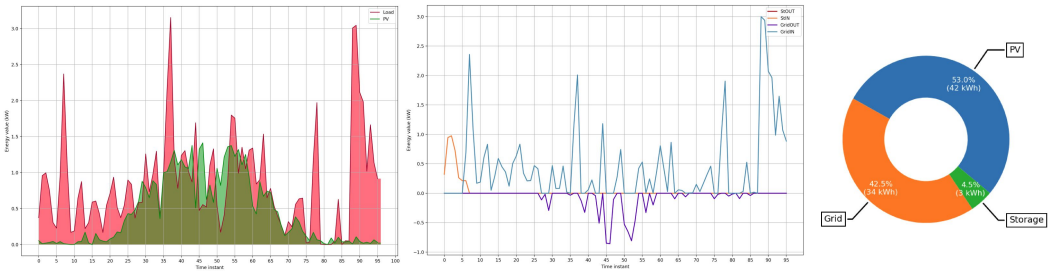

Figura 4.29: Incertezza non applicata nella fase offline

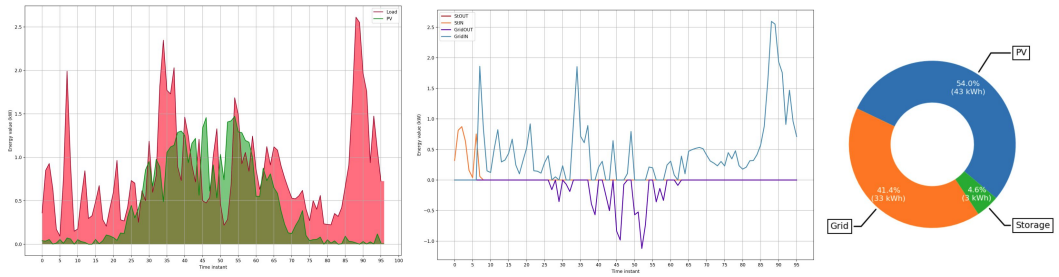

Figura 4.30: Incertezza applicata nella fase offline

Configurazione 4: PV + CHP + Storage

Ultimo caso in esame relativo alla funzione obiettivo per minimizzare i costi, mostrato in figura 4.30 e 4.31. In entrambi i casi si noti come il comportamento nei confronti dello storage è analogo a quello descritto nella configurazione precedente.
Per quanto riguarda invece l'andamento della curva dei sistemi CHP si può risalire all'ottimizzazione effettuata nella fase offline.
Ancora una volta nel caso in cui la fase offline non considera l'incertezza(4.9) i contributi del sistema CHP a gas naturale (usato perchè più conveniente) risultino rilevanti soprattutto nelle ultime ore della giornata, mentre nel caso con incertezza (4.10) viene ridimensionato e bilanciato con il contributo della griglia.

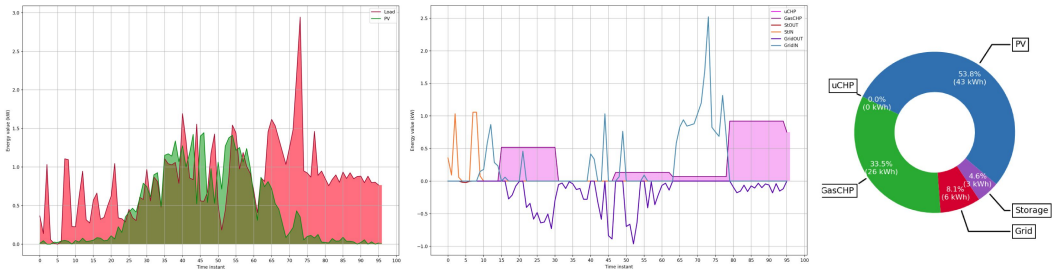

Figura 4.31: Incertezza non applicata nella fase offline

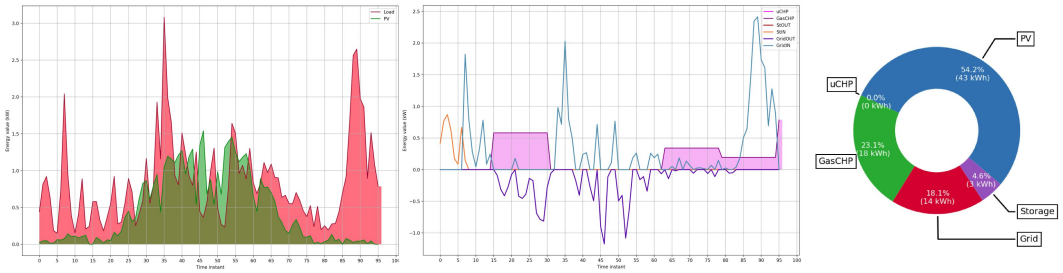

Figura 4.32: Incertezza applicata nella fase offline

4.3.2 Funzione obiettivo per minimizzare l'uso della rete esterna

Configurazione 1: PV

Dal momento che per questa e le successive configurazioni viene considerata la funzione obiettivo di minimizzazione dell'uso della rete esterna e del Mercato, risulta evidente come il modello si comporti di conseguenza.
Tuttavia, in questa prima configurazione (figura 4.33, 4.34), analogamente ad altri casi visti in questo capitolo con la stessa configurazione, il modello sarà costretto ad acquistare energia dal mercato in mancanza di produzione interna, in particolare nelle prime e nelle ultime ore del giorno, mentre nelle ore centrali potrà vendere l'eccesso eventualmente prodotto a partire dal fotovoltaico.

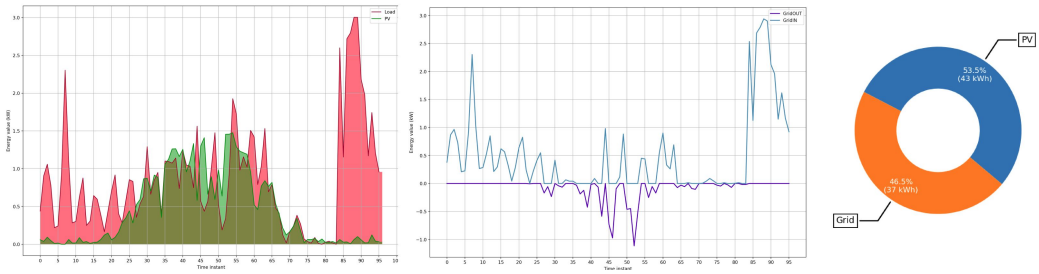

Figura 4.33: Incertezza non applicata nella fase offline

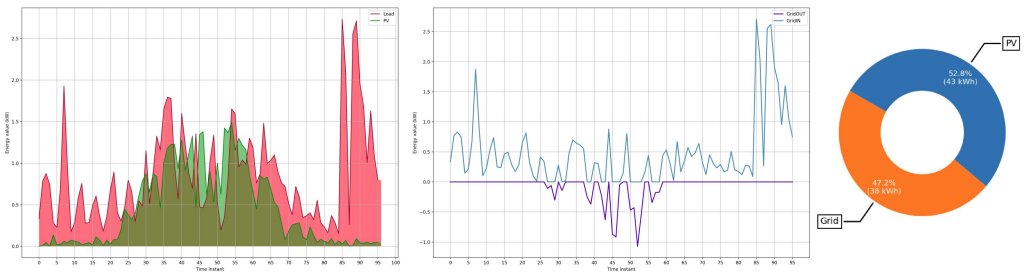

Figura 4.34: Incertezza applicata nella fase offline

Configurazione 2: PV + CHP

Nella seconda configurazione analizzata, in cui viene aggiunta alla produzione interna una componente originata dai sistemi CHP, si può notare un comportamento peculiare, mostrato in figura 4.35 e 4.36.
Nelle prime ore del giorno, dove la produzione fotovoltaica è scarsa, il modello utilizza il sistema CHP per soddisfare la domanda dei clienti. Nella parte centrale del grafico è evidente che ci sia un eccesso di produzione da parte del fotovoltaico, che viene prontamente venduta sulla rete. Tuttavia nella seconda metà della giornata c'è un notevole picco di carico, che viene, senza altre alternative, soddisfatto tramite l'acquisto di energia dalla rete, acquisto che cerca di essere bilanciato tramite la vendita di energia nella finestra temporale immediatamente successiva prodotta tramite il sistema CHP.
Come si può notare, anche in questo caso è prediletto l'utilizzo del sistema CHP a gas naturale (per ragioni economiche).
Interessante osservare come il modello cerchi di bilanciare, situazione messa ben in evidenza in questo esempio, una situazione di carico improvviso e non previsto: si ricorda che tramite gli shift si cerca di evitare situazioni simili, ma non è possibile prevedere con certezza l'andamento della curva del carico un giorno in anticipo. Vengono infatti utilizzate delle stime, che hanno in ogni caso un margine di errore.

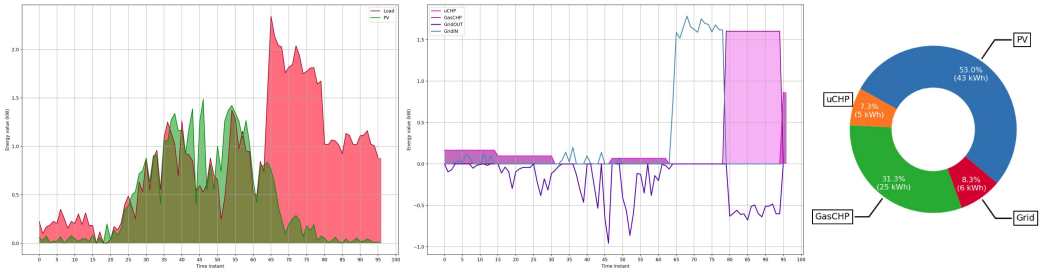

Figura 4.35: Incertezza non applicata nella fase offline

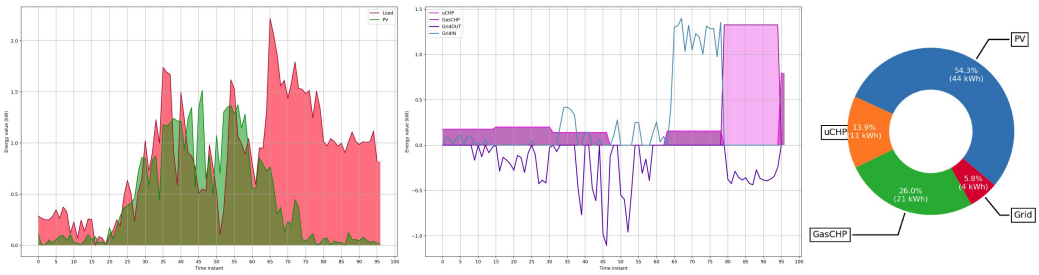

Figura 4.36: Incertezza applicata nella fase offline

Configurazione 3: PV + Storage

Risulta, a primo impatto, sorprendente il comportamento del modello nella configurazione rappresentata in figura 4.37 e 4.38. Il sistema di storage, anche se presente, viene utilizzato in minima parte.
Come per altre situazioni, la motivazione di ciò risiede nella risoluzione del modello matematico da parte del solver. L'utilizzo della rete risulta ad ogni modo ridotto, viene utilizzato unicamente nelle ultime ore del giorno per soddisfare i picchi di richiesta di carico o in generale quando il fotovoltaico non è in grado di soddisfare completamente la domanda.

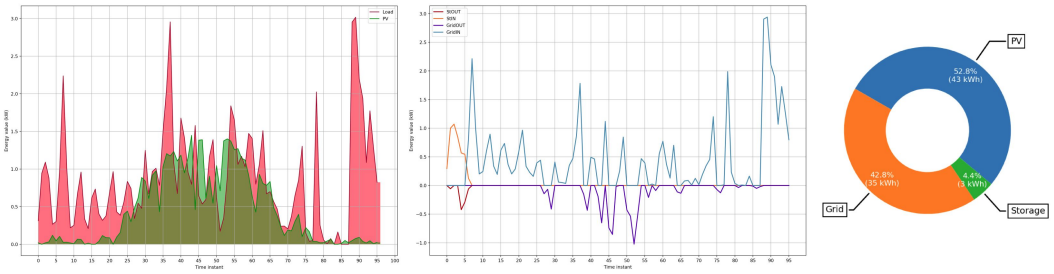

Figura 4.37: Incertezza non applicata nella fase offline

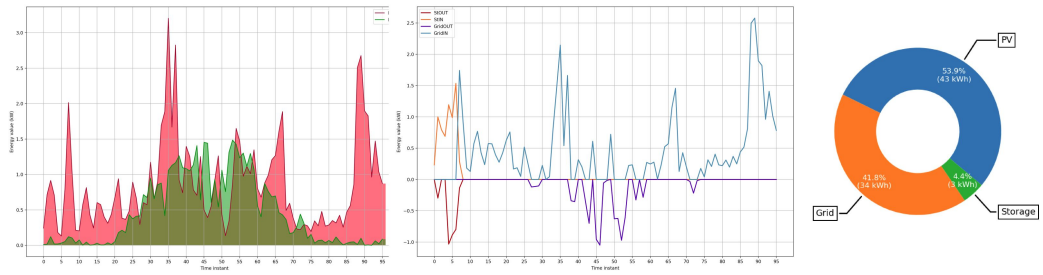

Figura 4.38: Incertezza applicata nella fase offline

Configurazione 4: PV + CHP + Storage

Nell'ultima configurazione considerata (figura 4.39, 4.40), il comportamento del modello risulta lineare e non particolarmente degno di nota. Ancora una volta, solo in caso di necessità (ovvero quando il modello non riesce a soddisfare la richiesta di carico tramite la produzione interna) ricorre all'uso della rete esterna.

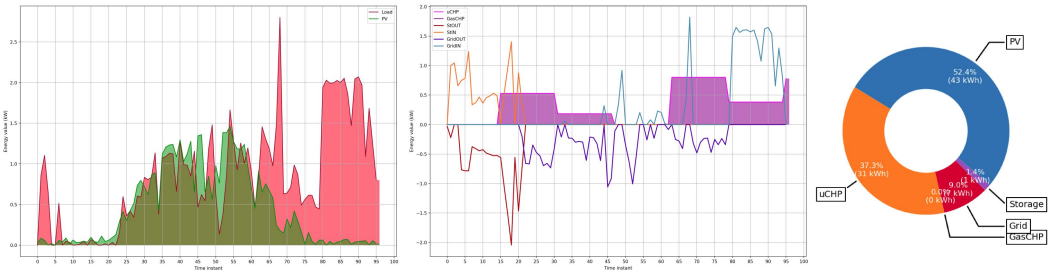

Figura 4.39: Incertezza non applicata nella fase offline

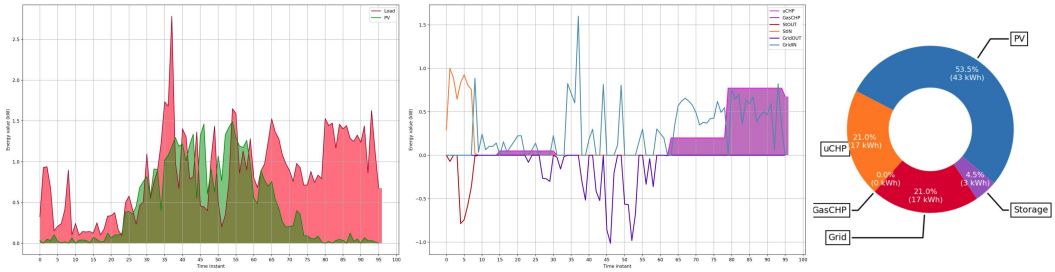

Figura 4.40: Incertezza applicata nella fase offline

Configurazioni particolari

In conclusione a questa sezione, verranno presentati due casi particolari. Il primo, in cui l'ottimizzazione offline viene eseguita utilizzando la funzione obiettivo per la minimizzazione della distanza da un profilo desiderato. Questa scelta ha ovviamente delle ricadute anche sulla fase online, poiché i valori shiftati di carico e di potenza del CHP sono trasmessi alla fase successiva. Viene riportato dunque in figura, un esempio di tale realizzazione usando la configurazione con tutti gli elementi disponibili all'interno del POD.

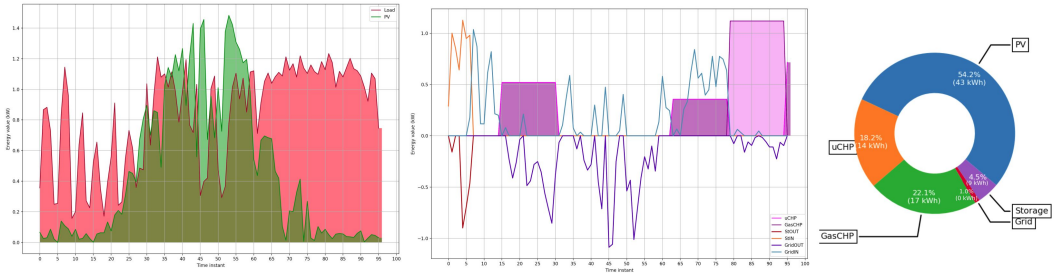

Figura 4.41: Fase online in seguito a fase offline con funzione *profile*

Si riconosce in figura 4.41, l'andamento della curva del carico vista in precedenza nella fase offline con la relativa funzione obiettivo per la minimizzazione della distanza dal profilo desiderato. In generale il comportamento del modello è analogo a quello descritto in altre configurazioni.

L'ultimo caso invece da evidenziare, in figura 4.42, è quello in cui la fase online venga eseguita senza la fase offline. Da un punto di vista pratico, questo significa che gli shift non sono stati generati, e dunque non sono immessi come input nello script online.py.
A livello matematico, ciò si concretizza in un modello che avrà come unica variabile relativa al carico la potenza identificata con P_{Load}, che coincide con $\tilde{P}_{Load}$ dal momento che la variabile degli shift S_{Load} non esiste; e analogamente la potenza dei sistemi CHP sarà determinata unicamente dal valore di P_{CHP}, che coincide con $\tilde{P}_{CHP}$.
Le performance di questa fase risulteranno nettamente peggiori rispetto all'ottimizzazione eseguita utilizzando entrambe le fasi.
Di seguito, viene mostrato un esempio di output di ottimizzazione online senza la fase offline.

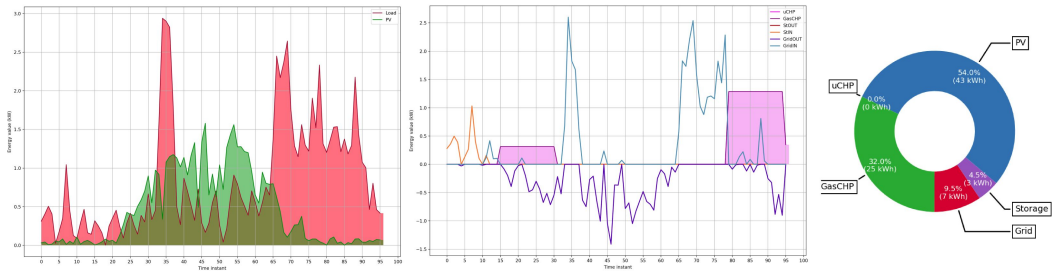

Figura 4.42: Fase online in seguito a fase offline con funzione *profile*

Confronto tra le configurazioni

Viene di seguito riportata una tabella riassuntiva con i valori delle funzioni obiettivo considerate in questa sezione: in particolare, poiché per costruzione, viene generato un modello ad ogni istante di tempo, e dunque un valore della funzione per ogni t, si è deciso di riportare i valori di *Min*, *Avg*, *Max*, *Tot* per l'insieme di iterazioni, per avere un modo di paragonare i processi di elaborazione della fase nella loro complessità.
Analogamente al confronto effettuato nella sezione dedicata alla fase offline, sono stati riportati i valori delle funzioni obiettivo separati in base alla condizione di incertezza della fase precedente. In particolare in corrispondenza di "NI" si troveranno i valori ottenuti da un'ottimizzazione complessiva realizzata tramite offline (senza incertezza) + online, mentre in corrispondenza di "I" sono inseriti i valori realizzati tramite un'ottimizzazione offline (con incertezza) + online.
Come si è mostrato in questa sezione, la considerazione o meno dell'incertezza nella fase offline influisce anche sulla fase online successiva. Per questa ragione è stato ritenuto coerente rappresentare entrambe le situazioni nella tabella, in analogia all'intera sezione.

		Min		Max		Avg		Tot	
		NI	I	NI	I	NI	I	NI	I
Conf 1	Cost	0.001	0.001	0.573	0.323	0.066	0.068	6.351	6.566
	Grid	0.001	0.001	0.510	0.327	0.077	0.048	7.404	4.634
Conf 2	Cost	0.000	0.000	0.309	0.632	0.054	0.085	5.163	8.143
	Grid	0.000	0.000	0.531	0.226	0.068	0.057	6.487	5.478
Conf 3	Cost	0.001	0.002	0.583	0.419	0.056	0.072	5.420	6.870
	Grid	0.000	0.000	0.423	0.371	0.056	0.058	5.381	5.545
Conf 4	Cost	0.000	0.000	0.310	0.341	0.058	0.060	5.594	5.743
	Grid	0.000	0.000	0.316	0.301	0.058	0.041	5.562	4.692

		Min	Max	Avg	Tot
Conf 1	Cost	0.000	0.532	0.078	7.486
	Grid	0.000	0.397	0.088	8.479
Conf 2	Cost	0.000	0.419	0.091	8.762
	Grid	0.000	0.391	0.079	7.589
Conf 3	Cost	0.001	0.526	0.085	8.161
	Grid	0.000	0.457	0.065	6.282
Conf 4	Cost	0.000	0.396	0.087	8.314
	Grid	0.000	0.379	0.067	6.414

Separatamente viene invece rappresentata la tabella con i valori delle configurazioni testate e nella particolare situazione in cui la fase online sia eseguita senza la fase offline (analogamente a come mostrato in figura 4.38). Mancando la fase offline, non c'è distinzione tra situazioni con e senza incertezza.
Si noti come i risultati peggiorino non considerando la prima fase e il processo di Shifting associato ai sistemi CHP e in particolare al carico.

4.4 Caso 100 POD

4.4.1 Configurazioni

Il secondo macro-caso di studio vede l'utilizzo di diverse configurazioni distribuite su un numero maggiore di POD. Nel tentativo di testare la scalabilità della soluzione proposta ma cercando di utilizzare delle configurazioni che rispettano determinate proporzioni tra il numero dei componenti impiegati, la seguente scelta è quella che si è rivelata vincente.
Avendo a disposizione un elevato numero di POD è stato necessario il testing di diverse combinazioni e diverse proporzioni tra i vari elementi, ricordando tuttavia che la rete esterna rimane una costante per ogni singolo componente considerato.
Dunque, assumendo di poter ritrovare tali conformazioni anche nella realtà, l'insieme dei POD considerati è composto dai seguenti elementi:

	Elementi	Fattore di replicazione
Config 1	PV, Storage	20
Config 2	PV, Load	20
Config 3	UCHP, Load	30
Config 4	Gas CHP, UCHP, Storage	15
Config 5	Gas CHP, Storage, Load	15
Config 6	Gas CHP	20

L'insieme degli elementi descritti da queste configurazioni considera quindi un totale di 40 PV, 50 sistemi di storage, 65 carichi, 50 sistemi di CHP a gas naturale e 45 sistemi di microCHP.
Un dettaglio molto rilevante per comprendere i contributi dei sistemi CHP mostrati nei grafici successivi, è quello del valore della loro produzione. I sistemi considerati in questo modello producono piccole quantità di energia, su piccola scala, poiché si è dapprima pensato ad un contesto in cui siano sufficienti delle produzioni su scala domestica. Tuttavia potrebbe essere necessario, con l'aumentare del numero di POD, anche considerare dei sistemi CHP più potenti.

In quest'ultima sezione verranno rappresentati i grafici di output rispettivamente della fase offline e della fase online associati alle funzioni obiettivo disponibili e alle configurazioni appena descritte. Solamente i casi in

condizione di incertezza saranno considerati, poiché costituiscono il principale motivo di interesse. Dunque sarà fatto un confronto complessivo in termini di tempo di esecuzione dell'intero modello (offline + online).

4.4.2 Funzione obiettivo per minimizzare il costo operativo

Il comportamento rappresentato in figura 4.43 è paragonabile a quello visto nelle configurazioni su singolo POD. In particolare, la tendenza del modello è quella di andare a comprare e vendere in istanti temporali ravvicinati grandi quantità di energia dalla rete, anche se non strettamente necessarie per il soddisfacimento del fabbisogno di carico.

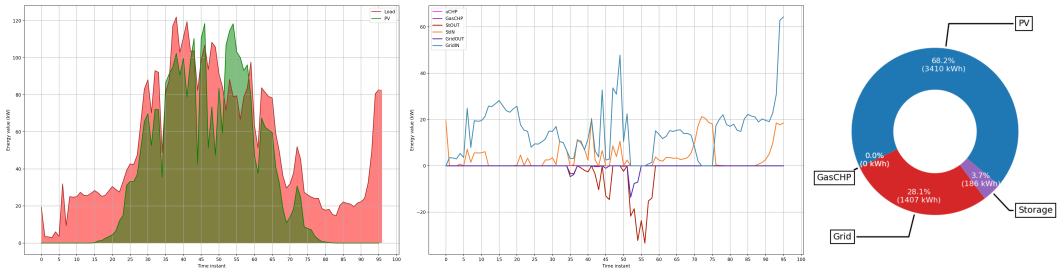

Figura 4.43: Offline - funzione dei costi

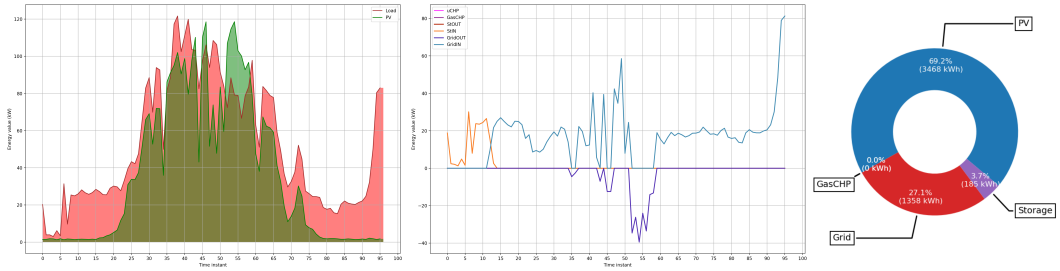

Figura 4.44: Online - funzione dei costi

Risulta particolarmente evidente, nel passaggio da offline (4.43) a offline (4.44), la differenza nella gestione dei contributi. Nonostante complessivamente il risultato sia il medesimo nei grafici a ciambella, nella fase online si può chiaramente vedere come l'uso dello storage sia ridotto notevolmente. Questo ne consegue ovviamente una riduzione in termini di costo, che è esattamente lo scopo di utilizzare due fasi nel processo di ottimizzazione (e nell'uso della specifica funzione).

4.4.3 Funzione obiettivo per minimizzare l'uso della rete esterna

Nel caso in cui la funzione obiettivo mirasse alla minimizzazione della rete esterna, si nota una maggiore varietà dei contributi e una riduzione invece di quello relativo alla rete esterna.
In maniera analoga ai casi esaminati per configurazione a singolo POD, in figura 4.45 e 4.46 si può osservare che nel passaggio da fase offline a fase online ci sia un miglior bilanciamento dei vari contributi. Si ricorda che in tutti i casi considerati in questa sezione è stata introdotta la componente di incertezza sulla domanda di carico e fotovoltaico e che nella fase offline viene eseguito un processo di shifting del carico e della potenza dei CHP, ovvero una redistribuzione lungo l'asse temporale, dove possibile, di quantità di energia in momenti della giornata in cui diventa economicamente più conveniente o in cui la richiesta è tipicamente più bassa.

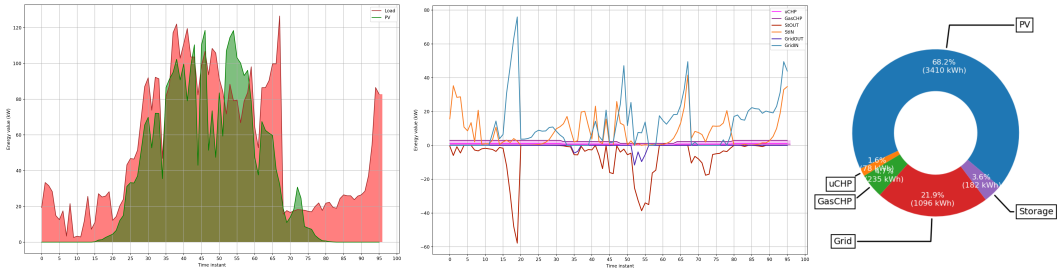

Figura 4.45: Offline - funzione dell'uso della rete

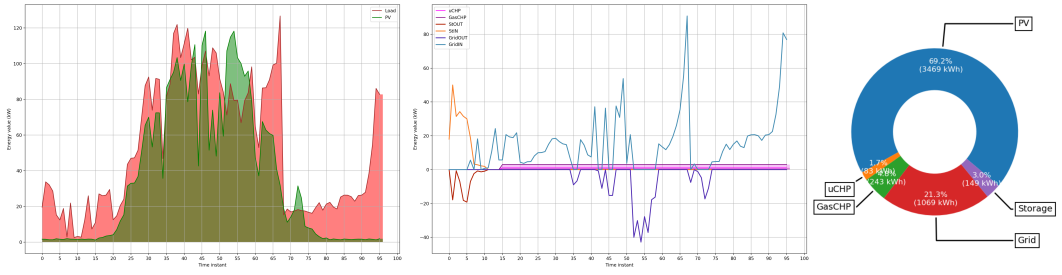

Figura 4.46: Online - funzione dell'uso della rete

4.4.4 Funzione obiettivo per minimizzare la distanza da un profilo desiderato

A differenza della situazione esaminata per un singolo POD, si può notare che la curva del profilo desiderato è in questo caso (figura 4.47) molto più alta rispetto all'effettiva realizzazione della curva di carico. Questo è dovuto sicuramente alla differente scala su cui è stato applicato il modello. Si ricorda che il processo di generazione del profilo piatto è in proporzione al numero di componenti considerati, quindi è naturale che il grafico risulti in questa forma.
Tuttavia si può riconoscere lo stesso andamento della curva: il modello cerca il più possibile di adattarsi e seguire il profilo piatto gestendo le risorse interne di conseguenza.
Come già visto in configurazioni precedenti il contributo relativo al sistema di storage presenta un andamento lievemente oscillatorio e costante nel tempo, sempre riconducibile alla funzione obiettivo scelta. Il contributo della griglia, nonostante si cerchi di utilizzarla il meno possibile, rimane importante. Come precisato nell'introduzione a questa sezione, i sistemi CHP sono poco potenti, e questo ne consegue che in una panoramica come quella analizzata, e soprattutto poiché non presenti in tutti i POD, il loro contributo sia estremamente ridotto.

Figura 4.47: Offline - funzione del profilo

4.4.5 Casi Particolari

Analogamente alla configurazione con singolo POD, anche in questa sezione è stato considerato il caso in cui l'ottimizzazione online avvenga senza la fase offline. Si può osservare dal grafico in figura 4.48 che l'andamento dello shift, a differenza dei grafici precedenti, non è stato ottimizzato: dunque, la richiesta non è più concentrata nelle ore centrali dove c'è abbondanza di produzione fotovoltaica ma semplicemente varia in base alla reale richiesta.

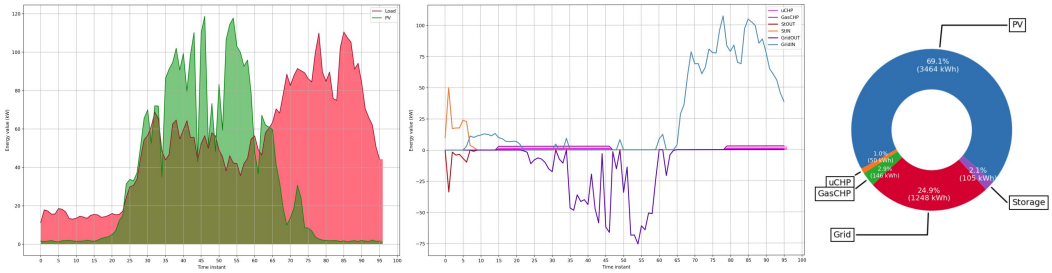

Figura 4.48: Fase online in seguito a fase offline con funzione *profile*

4.4.6 Tempo di esecuzione

In conclusione a questa sezione, vengono presentate alcune considerazioni sul tempo di esecuzione del modello.
La piattaforma usata per l'esecuzione dell'interfaccia e degli script è Ubuntu 18.04 (Bionic) a 64bit su pc portatile con 8 GB di RAM.
Sono state effettuate diverse prove per testare la scalabilità della soluzione proposta e questo è il risultato (Figura 4.49)
Il contributo maggiore per quanto riguarda il tempo di esecuzione totale è sicuramente dato dal modello online: in base al numero di POD, vengono creati altrettanti modelli. Nel caso della fase offline ciò che determina per la maggior parte il tempo di esecuzione è la considerazione dell'incertezza: si ricorda che è possibile associarla a due elementi in tutto il modello, fotovoltaico e carico, ottenendo un numero complessivo di quattro scenari, in ciascuno dei quali deve essere simulato il modello per la fase offline.
La fase di testing si è dovuta interrompere ad un numero massimo di POD pari a 500, a causa della ridotta potenza computazionale dello strumento a disposizione, ma con un supporto adeguato è possibile affermare che il modello sia scalabile e utilizzabile anche con configurazioni molto più

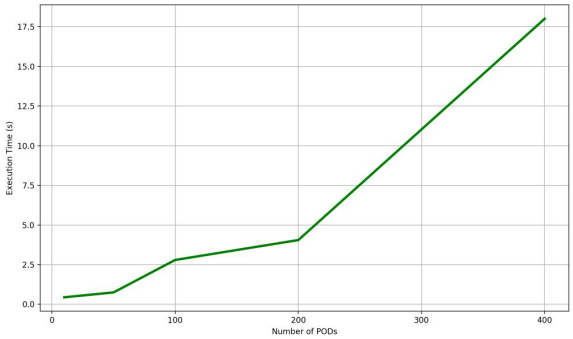

Figura 4.49: Andamento dei tempi di esecuzione in funzione del numero dei POD

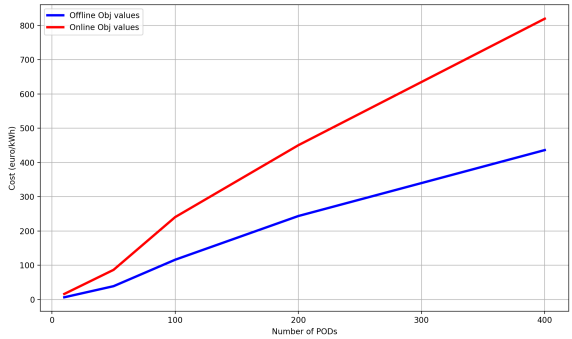

Figura 4.50: Andamento dei valori della funzione obiettivo *cost* in funzione del numero dei POD

numerose.
In associazione al grafico dei tempi di esecuzione è stato aggiunto un grafico per mostrare i valori delle funzioni obiettivo di online e offline (4.50) al variare del numero dei POD.
In figura sono indicate con la linea blu i valori della funzione obiettivo nella fase offline, mentre con la linea rossa sono indicati i valori della funzione obiettivo nella fase online. L'aumento visibile all'aumentare del numero di POD è coerente con la scelta della funzione obiettivo utilizzata: in maniera simbolica, si è usata la funzione obiettivo per la minimizzazione dei costi in condizioni di incertezza.

5 Conclusioni

L'obiettivo di questa tesi è stata l'ottimizzazione di un modello a componenti per un sistema energetico.
Innanzitutto è stato necessario elaborare un modello di base, in condizioni deterministiche, che fosse in grado di considerare un VPP in cui possono essere presenti contributi provenienti da diverse sorgenti energetiche. Si può dunque assumere uno scenario in cui il modello proposto venga implementato tramite l'Energy Management System di un VPP e in base al risultato, questo comunichi ai diversi componenti del VPP lo schedule di produzione. Nell'ambito dell'elaborazione di sistemi energetici distribuiti, come menzionato nell'Introduzione, è molto importante valutare il livello di astrazione con cui andare a descrivere ogni componente, in modo da poter rappresentare con sufficiente accuratezza gli elementi presenti, ma allo stesso tempo mantenere una necessaria astrazione, per evitare che il risultato sia specifico solo per il caso in questione.
Una volta definite le equazioni matematiche, si è aggiunto un livello ulteriore di complessità: la considerazione di condizioni di incertezza. Elaborare un modello in condizioni di incertezza significa tenere conto di stime e predizioni inserendo un certo margine di errore legato proprio alle componenti soggette ad incertezza. Nel caso in esame, i valori a cui si è assegnata tale condizione sono la domanda di carico dei clienti e il flusso energetico del fotovoltaico.
In letteratura è alquanto comune considerare la variabilità di componenti legate a risorse rinnovabili, dal momento che la loro produzione dipende da fattori che non sono completamente stimabili.
Un metodo efficiente per gestire l'incertezza è organizzare l'ottimizzazione in due stadi. Ci sono molte analisi e proposte nell'ambito delle risorse distribuite, ma quello che si è preferito seguire e che sembrava più adatto alla situazione da considerare è quello proposto in [35]. Uno degli aspetti più interessanti della soluzione evidenziata è quella di prevedere un livello di flessibilità per la richiesta di carico dei clienti e per la rete esterna. La flessibilità viene concessa tramite l'utilizzo di un processo chiamato *Load*

Shifting, che permette di redistribuire una certa quantità di energia lungo l'orizzonte temporale considerato, mantenendo invariato il consumo totale giornaliero. Tale considerazioni sulla flessibilità sono state eseguite anche in questa tesi, con l'associazione aggiunta di flessibilità ai sistemi CHP.
L'approccio infine elaborato è composto dunque da due fasi e copre la durata di due giornate. La prima fase, del giorno prima (offline) in cui si considerano le stime dei valori "incerti" e si cerca di ottimizzare la deviazione della domanda di carico in riferimento ad esse. In aggiunta è possibile associare un livello di flessibilità modellato su finestre temporali anche ai sistemi CHP.
Il modello è tuttavia eseguito in condizioni di incertezza: per ogni possibile realizzazione si implementa uno scenario in cui si eseguono le computazioni matematiche per determinare i flussi energetici.
Una volta ottenuti i valori degli shift, è possibile inserirli nella seconda fase, del giorno stesso (online), in cui ci si aspetta di ricevere gli effettivi valori delle variabili a cui è stata associata l'incertezza per ogni istante di tempo considerato, e in base ad esso determinare il reale flusso energetico all'interno del VPP.
In una fase iniziale di questo lavoro si era considerata una unica funzione obiettivo per la minimizzazione dei costi di produzione, funzione solitamente considerata per l'ottimizzazione di sistemi energetici distribuiti (in alternativa ad una funzione di massimizzazione dei profitti). Questo approccio non ha però generato i risultati sperati, in quanto il modello dimostrava una tendenza a usare in maniera talvolta eccessiva la rete esterna, poiché risultava conveniente in termini economici. Per questo motivo sono stati considerati due ulteriori metodi: uno per ridurre l'interazione con la griglia, e uno per introdurre come obiettivo un profilo di carico elaborato a partire dalle richieste di un operatore energetico esterno. Il secondo metodo considerato ha generato risultati migliori, non andando tuttavia ad eliminare completamente l'uso della griglia esterna, ma moderandolo e considerandolo solo nel caso in cui fosse strettamente necessario.
Infine l'ultimo approccio ha evidenziato alcuni comportamenti a prima vista anomali: le curve di utilizzo dello storage e della rete esterna, presentano talvolta un andamento oscillatorio che nella realtà non è molto auspicabile.
Sono stati esaminati il maggior numero di casi realizzabili possibili, usando diverse configurazioni dei POD e variando il numero di componenti considerati. In questa maniera si è potuto delineare al meglio il compor-

tamento del modello nelle differenti situazioni analizzando la risposta dal punto di vista matematico, di output in termini di funzione obiettivo e tempi di esecuzione.
Per una semplificata gestione del modello è stata implementata un'interfaccia web, scalabile e fault-tolerant, che è facilmente adattabile in caso di cambiamenti della struttura sottostante. La realizzazione è stata guidata dai principi della modellazione software e della buona programmazione.
In conclusione, visti i risultati di questa tesi, si può dire che il sistema proposto è inseribile in un contesto di gestione di unità energetiche distribuite in condizioni di incertezza e che pone le basi per nuovi miglioramenti e sviluppi. In futuro potrebbe essere necessario effettuare ulteriori considerazioni per quanto riguarda il sistema di storage, in particolare sui fattori di degradazione successivi alla carica e scarica delle batterie considerati. In aggiunta, potrebbero essere richieste anche considerazioni sul livello di astrazione della rete esterna, in particolare tempi di trasmissione e perdite lungo le linee.
Si ricorda infine che la tematica dell'ottimizzazione di sistemi energetici distribuiti è estremamente seguita e studiata dalla comunità scientifica, ed è possibile che nuovi sviluppi e approcci emergano in futuro. In particolare, bisogna considerare il fatto che questi modelli si basano su infrastrutture come il VPP e smart grid che sono ancora in fase di sperimentazione e sviluppo e che in pochi paesi sono una concreta realtà.

Bibliografia

[1] Siirola JJ, Edgar TF. *Process energy systems: control, economic, and sustainability objectives.* Comput Chem Eng 2012;47:134e44.

[2] P. Lombardi, T. Skolnikova, Z. Styczynski, N. Voropai. *Virtual power plant management considering energy storage systems.*

[3] Verbruggen A, Dewallef P, Quoilin S, Wiggin M. *Unveiling the mystery of combined heat & power (cogeneration).* Energy 2013;61:575e82. project/5e44727a4caca10001046dfd.

[4] D. Pudjianto, C. Ramsay and G. Strbac. *Virtual power plant and system integration of distributed energy resources.*

[5] Roberto Caldon, Andrea Rossi Patria and Roberto Turri. *Optimization algorithm for a Virtual Power Plant Operation.*

[6] Roberto Caldon, Andrea Rossi Patria and Roberto Turri. *Optimal Control of a Distribution System with a Virtual Power Plant.*

[7] M. Musio, P. Lombardi, A. Damiano. *Vehicles to Grid (V2G) concept applied to a Virtual Power Plant Structure.* The XIX International Conference on Electrical Machines - ICEM.

[8] Zhe Zhuo, Jianyun Zhang, Pei Liu, Zheng Li, Michael C. Georgiadis, Efstratios N. Pistikopoulos. *A two-stage stochastic programming model for the optimal design of distributed energy systems.*

[9] Ren H, Gao W. *A MILP model for integrated plan and evaluation of distributed energy systems.* Appl Energy 2010;87:1001–14.

[10] Liu P, Pistikopoulos EN, Li Z. *Energy systems engineering: methodologies and applications.* Front Energy Power Eng China 2010;4:131–42.

[11] Gamou S., Yokoyama R., Ito K. *Optimal unit sizing of cogeneration systems in consideration of uncertain demands as continuous random variables.*

[12] Yoshida S., Ito K., Yokoyama R. *Sensitivity analysis in structure optimization of energy supply systems for a hospital.*

[13] Li C., Shi Y., Huang X. *Sensitivity analysis of energy demands on performance of CCHP systems.*

[14] Mavrotas G, Florios K, Vlachou D. *Energy planning of a hospital using mathematical programming and Monte Carlo simulation for dealing with uncertainty in the economic parameters.* Energy Convers Manage 2010;51: 722–31.

[15] Hawkes A, Leach M. *Impacts of temporal precision in optimisation modelling of micro-combined heat and power.* Energy 2005;30:1759–79.

[16] Kristina Jurkovic, Hrvoje PandZic and Igor Kuzle. *Review on Unit Commitment under Uncertainty Approaches.*

[17] Y. Dvorkin, Y. Wang, H. PandZi6, D. Kirschen. *Comparison of scenario reduction techniques for the stochastic unit commitment.* Proc. of IEEE PES General Meeting, Conference & Exposition 2014., National Harbor, Maryland, July 2014, pp. 1-5

[18] J. Wang, M. Shahidehpour, and Z. Li. *Security-constrained unit commitment with volatile wind power generation.* iEEE Trans. Power Syst., vol. 23, no. 3, pp. 1319-1327, Aug. 2008.

[19] D. Bertsimas, E. Litvinov, X. A. Sun, Z. Jinye, and T. Tongxin. *Adaptive robust optimization for the security constrained unit commitment problem.* IEEE Trans. Power Syst., vol. 28, no. 1 , pp. 52-63, Feb. 2013

[20] R. Jiang, M. Zhang, G. Li, and Y. Guan. *Two-stage network constrained robust unit commitment problem.* Eur. J Opel'. Res., vol. 234, no.3, pp. 751-762, May 2014.

[21] C. Zhao and Y. Guan. *Unified stochastic and robust unit commitment.* iEEE Trans. Power Syst., vol. 28, no. 3, pp. 3353-33761, Aug. 2013.

[22] Alexander Shapiro and Andy Philpott. *A tutorial on stochastic programming.* Manuscript. Available at www2. isye. gatech.edu/ashapiro/publications.html, 17, 2007.

[23] Alexander Shapiro. *Sample average approximation.* In Encyclopedia of Operations Research and Management Science, pages 1350–1355. Springer, 2013.

[24] Allegra De Filippo, Michele Lombardi and Michela Milano. *Methods for Off-line/On-line Optimization under Uncertainty*

[25] Allegra De Filippo, Michele Lombardi and Michela Milano. *Off-line and on-line optimization under uncertainty: a case study on energy management*

[26] S. Kaplanis and E. Kaplani. *A model to predict expected mean and stochastic hourly global solar radiation i(h;nj) values.*. Renewable Energy, 32(8):1414 – 1425,2007.

[27] A. Espinosa and L. Ochoa. *Dissemination document low voltage networks models and low carbon technology profiles.* Technical report, University of Manchester, June 2015.

[28] C. Bordin, H. O. Anuta, A. Crossland, I. L. Gutierrez, C. J. Dent, and D. Vigo.*A linear programming approach for battery degradation analysis and optimization in offgrid power systems with solar energy integration.* Renewable Energy, 101:417 – 430, 2017.

[29] Davide Aloini, Emanuele Crisostomi, Marco Raugi and Rocco Rizzo. *Optimal Power Scheduling in a Virtual Power Plant*

[30] Alejandro N. Espinosa and Luis N. Ochoa. *Dissemination document low voltage networks models and low carbon technology profiles.* Technical report, University of Manchester, June 2015.

[31] CHP Project Profiles Database - https://betterbuildingssolutioncenter.energy.gov/chp/chp-project-profiles-database

[32] Energy Prices - https://www.eea.europa.eu/data-and-maps/indicators/en31-energy-prices/en31-energy-prices

[33] S. Awerbuch and A. Preston. *The virtual utility: Accounting, technology & competitive aspects of the emerging industry, volume 26.* Springer Science & Business Media, 2012.

[34] Palma-Behnke R., Benavides C., Aranda E., Llanos J., and Sez D. *Energy management system for a renewable based microgrid with a demand side management mechanism.* In 2011 IEEE Symposium on Computational Intelligence Applications In Smart Grid (CIASG), pages 1–8, April 2011.

[35] Allegra De Filippo, Michele Lombardi, Michela Milano, Alberto Borghetti. *Robust Optimization for Virtual Power Plants.* November 2017, Conference of the Italian Association for Artificial Intelligence

[36] Xiaofei Wang, Zhijian Hu, Menglin Zhang, Mengyue Hu. *Two-stage stochastic optimization for unit commitmentconsidering wind power based on scenario analysis*

[37] X. Ma, Y. Sun and H. Fang, *Scenario generation of windpower based on statistical uncertainty and variability.* IEEE Transactions on Sustainable Energy, vol. 4, no. 4, 2013, pp.894-904