UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Engineering Sciences

Optimization and Control of Energy Storage in A Smart Grid

by

Lu Wang

Thesis for the degree of Doctor of Philosophy

June 2017

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Doctor of Philosophy

OPTIMIZATION AND CONTROL OF ENERGY STORAGE IN A SMART GRID

by Lu Wang

Environmental issues such as global warming, limited storage of fossil fuels and concerns about cost and energy efficiency are driving the development of the future smart grid. To reduce carbon emissions, it is expected that there will be a large-scale increase in the penetration of renewable generators (RGs), electric vehicles (EVs) and electrical heating systems. This will require new control approaches to ensure the balance of generation and consumption and the stability of the power grid. Energy storage can be used to support grid operations by controlling frequency and voltage, and alleviating thermal overload. This thesis makes three novel contributions to the field: optimal battery sizing; optimal dispatch of vehicle-to-grid batteries; and optimal coordination of EV batteries and RGs. Appropriate sizing of the energy storage is very important when using it to support the power system. In this thesis, an approach has been proposed to determine the capacity of a battery storage providing support during N-1 contingencies to relieve transmission line thermal overload. In addition, as the increasing use of EV is an inevitable trend in the future smart grid, the system's peak demand may increase significantly due to EV charging, causing serious overloading of some power system facilities such as transformers and cables in the grid if an effective EV battery dispatch strategy is not used. Therefore, this report presents a dispatch strategy for EV batteries based on the Analytic Hierarchy Process taking into account both vehicle users' and power system requirements and priorities, as well as the constraints of the battery system. However, using renewable power to charge EVs is the prerequisite of realizing clean transport. EVs can store the extra renewable power and feed it into the grid when needed via vehicle-to-grid operations to increase the utilization and integration of RGs in the power grid. Thus, the optimal dispatch of EVs and RGs to realize the synergy between them will be one of the key challenges. Two optimal agent-based coordinated dispatch strategies are developed in this thesis, respectively using dynamic programming and the A* search procedure (comparisons between these two algorithms are made and discussed), for the synergistic integration of EVs and RGs, so that the benefits of both EV users and power grid are maximized. Each of the proposed approaches was tested on an IEEE Reliability Test System or a modified UK generic distribution system (UKGDS) using MATLAB. The simulation results demonstrate the feasibility and efficacy of the proposed approaches.

Contents

List of Figures

List of Tables

List of Abbreviations

ACEA	European Automobile Manufacturers' Association
AHP	Analytic Hierarchy Process
ASSP	Adjusted System Selling Price
ASBP	Adjusted System Buying Price
BEV	Battery Electric Vehicles
CDF	Cumulative Distribution Function
CHP	Combined Heat and Power
CI	Consistency Index
CR	Consistency Ratio
CSCOPF	Corrective Security-Constrained Optimal Power Flow
DCLF	DC Load Flow
DMOCOP	Distributed Multi-Objective Constraint Optimization Problem
DCOP	Distributed Constraint Optimization Problem
DG	Distributed Generator
DNO	Distribution Network Operator
DPDOD	Dynamic Programming Decentralized Optimal Dispatch
EAFO	European Alternative Fuels Observatory
EEA	European Environment Agency
EPRI	Electric Power Research Institute
ESS	Energy Storage System
EV	Electric Vehicle
EVA	EV Aggregator
EVI	Electric Vehicle Initiative
G2V	Grid-to-Vehicle
h2h	Home-to-Home
ICT	Information and Communication Technology
IEA	International Energy Agency
IEEE	Institute of Electrical and Electronics Engineers
LODF	Line Outage Distribution Factors
LTE	Long Term Emergency
MES	Multi-Energy System
MCDM	Multi-Criteria Decision-Making

MDP	Markov Decision Process
MILP	Mixed-Integer Linear Programming
PCC	Point of Common Coupling
$PfPc$	Power Flow and the associated Penalty Cost
PFC	Primary Frequency Control
PFR	Primary Frequency Regulation
PHS	Pumped Hydro Storage
PHEV	Plug-in Hybrid Electric Vehicles
PTDF	Power Transfer Distribution Factors
RES	Renewable Energy Sources
RG	Renewable Generator
RTS	Reliability Test System
SBP	System Buying Price
SOC	State Of Charge
SSP	System Selling Price
STE	Short Term Emergency
UKGDS	UK Generic Distribution System
V2G	Vehicle-to-Grid
VPP	Virtual Power Plant
WTG	Wind Turbine Generator

Nomenclature

$adj(v)$	nodes that are directly connected to node v via distribution cables
A_g	A pairwise comparison matrix for grid concerns
A_f	area of vehicle's face
A_{NC}	A pairwise comparison matrix for N-1 contingency's criteria
bp	electricity buying price
bp_{max}	the top electricity buying price
bp_{min}	the minimum electricity buying price
$chi(v)$	children nodes of node v
C_a	available capacity of a battery discharging constantly at current i_d
C_{ep}	penalty cost of dispatching EVs in a costly way
C_D	air resistance coefficient
C_{ij}	thermal capacity of the distribution cable between node v_i and v_j
C_p	Peukert Capacity
C_r	rolling resistance coefficient
C_{RG}	penalty cost of wasting renewable power
C_{SOC}	penalty cost of insufficient SOC
CI_i	the carbon intensity of the DG at node v_i
$CE(f_{i\hat{i}})$	accumulated carbon emissions of the dispatch actions of DGs at node v_i and all its children that result in the power flow value $f_{i\hat{i}}$
d_{max}	the maximum system demand during a day
d_{min}	the minimum system demand during a day
D^s	distance of a single h2h trip
D^{d1}	distance of the first trip of double h2h trips
D^{d2}	distance of the second trip of double h2h trips
E_w	energy of propulsion
E_e	electric energy consumed by an EV
ev_i	ith EV
f_{ij}	the power flow from node v_i to v_j
Fp	propulsion force of a vehicle
g_i	ith RG
hbp	high electricity buying price
HD	the high-level system demand

i_d	the discharging current
I_n	nominal discharging current
k	the Peukert Coefficient
LD	the low-level system demand
$LinkToPfDstate$	Dispatch states at node v_i and all its children that result in the corresponding $PfPc$ message formed at node v_i
$load_{ave}$	the average of daily fixed load demand
$load_{ev}^{c}$	EVs' total charging load at a node
$load_{ev}^{d}$	EVs' total discharging power at a node
$load_{fix}^{i}$	non-controllable load at node v_i
$load_h$	high fixed load threshold
$load_i$	load at node v_i
$load_l$	low fixed load threshold
$load_{min}$	minimum fixed load demand
$load_{max}$	maximum fixed load demand
$load_{tot}$	the total load at a node
lsp	low electricity selling price
MD	the mid-level system demand
p_i	power output of ith RG/DG
$pec(f_{i\hat{i}})$	total penalty cost of the dispatch actions at node v_i and all its children that result in the power flow value $f_{i\hat{i}}$
P_{CG}^{c}	the priorities of charging in terms of the cost to grid
P_{CG}^{d}	the priorities of discharging in terms of the cost to grid
P_{EP}^{c}	the priorities of charging in terms of electricity price
P_{EP}^{d}	the priorities of discharging in terms of electricity price
P_{LD}^{c}	the priorities of charging in terms of load levelling
P_{LD}^{d}	the priorities of discharging in terms of load levelling
P_i^{max}	maximum power output of ith RG
P_{RG}	the actual amount of renewable power that is used
P_{LPoC}^{c}	the priorities of charging in terms of potential consequence of load levelling
P_{LPoC}^{d}	the priorities of discharging in terms of potential consequence of load levelling
P_{SOC}^{c}	the priorities of charging with respect to SOC
P_{SOC}^{d}	the priorities of discharging with respect to SOC
P_{PoC}	the priority of charging/discharging in terms of potential consequence
P_{sen}	the priority of charging/discharging an EV at a specific bus with respect to sensitivity
P_{sev}	the priority of charging/discharging with respect to overloading severity

PC_a	the pairwise comparison matrix for the AHP model of agents that have both EVs and RGs
PC_b	the pairwise comparison matrix for the AHP model of agents that have only EVs
$PfPc$	a Power Flow and the associated Penalty Cost message
Poc_C	the potential consequence of contingency C
Poc_{max}	the severest potential consequence
$\widetilde{P_{RG}}$	maximum available renewable power
$Pstate(S_j)$	an array of the power outputs of DGs at v_i and its children nodes that results in the carbon emission described by the function CE of S_j
Q	a state queue
Q_{comb}	a combined state queue
$\overrightarrow{Q}$	a new state message array
RI	a given random consistency index
S_C^j	The sensitivity of load flow through a branch overloaded by an N-1 contingency C to changes of power injection at a particular bus j
S_{comb}	a combined state
S_{max}	the upper limit of SOC
S_{min}	the lower limit of SOC
S_n	represents the per-unit capacity going to be consumed on the next journey
Sev_C	the severity of overloading caused by contingency C
Sev_{max}	the severity of the severest overload caused by the severest contingency
$Snew$	new state
SOC_n	the SOC at the beginning of the next time interval
SOC_p	the expected SOC at the end of the current time interval
SOC_{pf}	the desired SOC of the EV battery before departure
sp	electricity selling price
Sp_j	jth message in $\overrightarrow{Q}$
sp_{max}	the top electricity selling price
sp_{min}	the minimum electricity selling price
t_{ij}	the distribution cable between node v_i and v_j
T_a^s	arrival time of a single h2h trip
T_a^{d1}	arrival time of the first trip of double h2h trips
T_a^{d2}	arrival time of the second trip of double h2h trips
T_d^s	departure time of a single h2h trip
T_d^{d1}	departure time of the first trip of double h2h trips
T_d^{d2}	departure time of the second trip of double h2h trips

T_p	available preparation time before departure in multiples of defined dispatch time interval
T_r	rated discharge time
$Toparent_{i\rightarrow\hat{i}}$	an array of $PfPc$ messages sent from node v_i's agent to its parent agent
U	utility function
$U_{\sum}$	the sum of (accumulated) utilities
v_i	node i of the distribution network
$\widehat{v_i}$	node i of the distribution network
X^*	An solution of the optimal dispatch problem
η	the efficiency of the electrical engine
δ_i	dispatch mode of the ith EV
ΔF_b	the change of power flow through the branch b
ΔP_j	the change in power injection at bus j
μ	mean value
ρ	the linear correlation matrix between standard normal variables
ρ_s	rank correlation — Spearman's ρ
ρ_a	air density
σ	standard deviation
Φ	the standard normal CDF

Acknowledgements

Great thanks to my supervisors Prof. Suleiman Sharkh and Dr. Andy Chipperfield who helped me a lot during my Ph.D. studies, encouraged and inspired me with their optimism, persistence, diligence and passion about their work and academic research. I also want to thank my parents who give me much love and support me all the way along. Thanks to Prof. Mike Russell for sponsoring the research work. Thanks also to my previous examiner Prof. Alex Rogers for giving me valuable advice on my work and all those who have helped me in whatever aspect, my collegues and friends. I really appreciate your help and support.

Academic Thesis: Declaration Of Authorship

I, Lu Wang,

declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

Optimization and Control of Energy Storage in A Smart Grid

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;

2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;

3. Where I have consulted the published work of others, this is always clearly attributed;

4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;

5. I have acknowledged all main sources of help;

6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;

7. Parts of this work have been published as:

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Determination of battery capacity for alleviation of thermal overload", *2014 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 12–15 Oct 2014.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Decentralized Coordination of Distributed Generators in a Distribution Network Using A*", *2016 IEEE International Conference on Power Systems Technology (POWERCON)*, 28 Sep – 1 Oct 2016.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "A*-Based Optimal Coordination of Vehicle-to-Grid Batteries and Renewable Generators in A Distribution Network", *The 26th IEEE International Symposium on Industrial Electronics*, 19-21 June 2017.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, Andrew Cruden, "Dispatch of Vehicle-to-Grid Battery Storage Using an Analytic Hierarchy Process", *IEEE transactions on Vehicular Technology*, 2016.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Optimal Coordination of Vehicle-to-Grid Batteries and Renewable Generators in A Distribution System", *Energy*, vol. 113, pp. 1250 – 1264, 2016.

Signed:

Date:

Chapter 1

Introduction

The Paris Agreement clearly sets the target of global temperature increase by 2050 to be well below 2°C. In this decarbonisation scenario, a power network needs to evolve into a smart grid, as illustrated in Figure 1.1, in order to safely incorporate different kinds of new elements that are essential to abate carbon emission, such as clean energy sources and electric vehicles. This has impacts on all the key components of the power grid, i.e., generators, transmission and distribution systems as well as end users [1], and requires a completely new approach to the control of the power grid including its frequency, voltage and current.

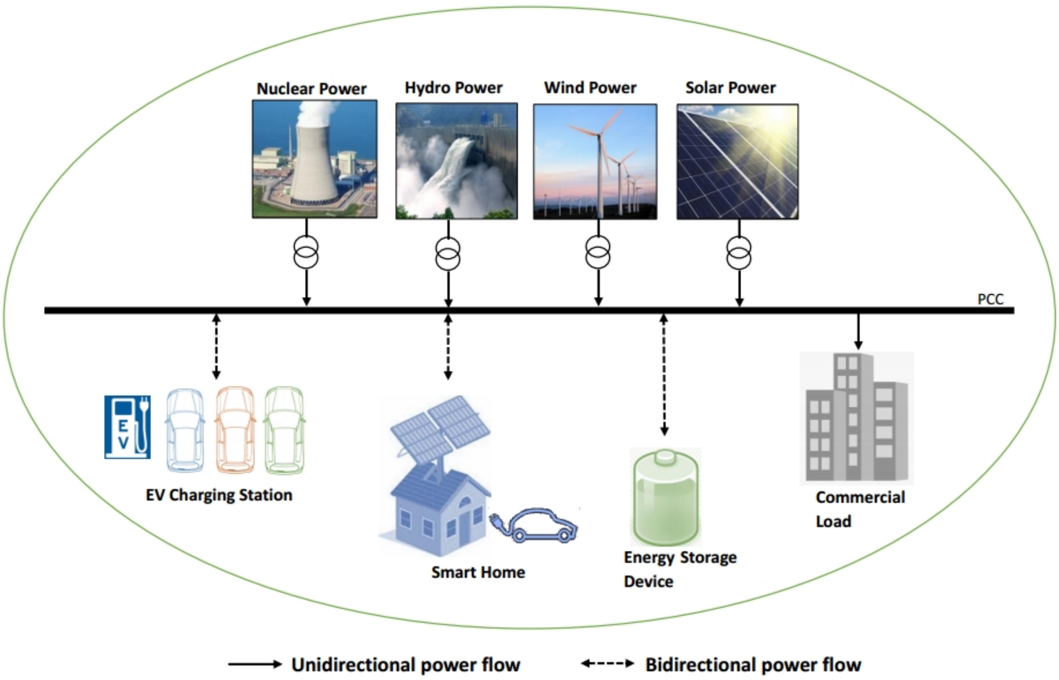


FIGURE 1.1: Future smart grid

Current policy aims to replace fossil-fired generators with low or zero carbon emission generators using renewable and nuclear energy [1, 2, 3]. As more intermittent renewable generators (RGs) are connected, such as wind and solar power, they are most likely to have significant impacts on the transmission and distribution systems, challenging the stability of power systems, due to the uncertainty and volatility of their power flow. The challenges with RG integration include [4]:

1. Power intermittency: RG power output largely depends on the weather and natural environment, which fluctuates considerably. This could result in an imbalance between supply and demand. As economic dispatch requires the generator's power output to change with the load, RG outputs should ideally be dispatchable and regulated as desired to match the load demand, which is difficult in practice unless energy storage is used.

2. Ramp rates: As the weather conditions like wind speed can change rapidly, the fast ramps of RG power can be another issue that affecting the system's stability. Thus, these ramps ought to be limited to save the cost of ancillary services and lessen the impact on system stability. Alternatively, energy storage could be used to limit the ramping rate.

3. Power output curtailment:large-scale RGs may overload the transmission lines or distribution cables in a constrained network. RGs' power may need to be curtailed to avoid congestion unless energy storage is used.

In this framework with large-scale integration of RGs, energy storage systems (ESSs) can help the existing components of power networks to improve their performance by providing voltage and frequency regulation, transmission congestion relief, deferral of system upgrade and so forth [5]. Table 1.1 summaries the key power system support roles that the applications of ESS can provide [6, 7].

As traffic contributes about 26% of the global CO_2 emissions [8], the potential of electric vehicles (EV) to replace fuel-driven ones is very clear and thus the increasing integration of EVs into grids is inevitable. That makes it possible to aggregate a great number of EV batteries as a VPP to participate in the ancillary service market, as discussed in Table 1.1. Although uncontrolled charging of a large number of EVs will significantly increase demand on the distribution system and give rise to power network stability issues, as discussed in [9, 10, 11, 12], appropriate dispatch of this aforementioned EV VPP can not only provide operational support for the power system, but also effectively reduce the total carbon emissions and reap actual environmental benefits in a power system with the integration of RGs [8].

In addition, in order to realize effective control, a smart grid requires a mass of information and data to be collected, exchanged and manipulated. Hence, new techniques like smart meters can be utilized for accurate measurements and data collection [13, 14, 15]. Advanced information and communication technology (ICT) is also very necessary for the future smart grid, and existing issues like communication interference and delays etc. should be addressed [13, 16]. Moreover, cyber security of the communication network is

TABLE 1.1: Energy Storage System Applications/Services

Roles of ESS	Applications of ESS in Power Systems
Generation-side roles	1. Time shift: Store surplus energy during low-demand periods and dispatch energy to the grid during high-demand periods. 2. Output smoothing: ESS is fast regulated to absorb extra energy during RG output spikes while release energy when at the power output falls.
Grid-side roles	1. Energy arbitrage/load levelling: Store cheap off-peak energy and sell it at higher price. 2. Frequency regulation: Provide both primary and secondary frequency control using droop control and/or according to the command of Automatic Generation Control. 3. Voltage control: Provide required reactive power to stabilize the local voltage level via the full-scale converter. 4. Reserve: Because of the forecast error of RG outputs, ESSs can be utilized as additional reserves for emergency support in covering the mismatch between supply and demand. 5. Emergency power supply/black start: ESS could restart from a shut-down condition and be used to energize the power system under emergent situations like catastrophic failure. 6. Transmission utilization efficiency: The utilization of ESS can increase the transmission capacity thus defer the transmission system upgrades and save investment costs.
Demand-side roles	1. Vehicle-to-grid: Use of electric vehicle batteries that are aggregated and considered as a Virtual Power Plant (VPP) to provide ancillary services, such as energy time shift, reserve and frequency regulation. 2. ESS at demand side can also be used to improve the power quality and reduce the demand charges by storing cheap electricity for end users to consume at peak periods.

another issue that needs to be considered, in order to ensure the security and stability of the communication network and the power grid by protecting them from cyber attacks [16, 17]. It is expected in the future that smart meters will be utilized to collect private information, such as energy consumption in an individual household, for the smart grid operation. Thus, end-users' privacy and the relevant policy need to be considered as well. Pricing strategy is another issue in the future smart grid. End-users will be able to generate and store electricity at home, trade with the grid and manage the energy usage based on the electricity price (i.e. demand response), therefore the pricing strategy will be very important and will have significant impact on the consumers' behaviours and grid operation [18, 19].

In this broad context of smart grid research, this thesis focuses mainly on the following 3 topics: Application of batteries for overload alleviation (battery sizing problem), EVs' smart dispatch and the optimal coordination of EVs and RGs when both of them are integrated into the grid.

1.1 Aims and Objectives

In summary, this thesis is concerned with the optimization and control of some aspects of smart grid, such as the dispatch and sizing problems of energy storage, the control of EVs' charging, V2G strategy, integration of intermittent RGs into the grid and the dispatch of EVs in coordination with RGs. Therefore, the aims and objectives are as follows:

- Investigate the sizing of energy storage in the use of alleviating thermal overload caused by N-1 contingencies[1].
- Investigate dispatch of EVs using the V2G concept considering the requirements of both EV users and grid.
- Investigate dispatch of EVs in coordination with renewable generators (RGs) to realize the synergy between them and satisfy the requirements of EV users and power grid.
- Include the consideration of intrinsic uncertainty of renewable power and EV travelling patterns in the coordinated dispatch of EVs and RGs.

1.2 Contributions

The contributions of this thesis are listed as follows:

- A battery-capacity-determination approach is proposed taking into account the different characteristics of contingencies using an Analytic Hierarchy Process (AHP) and tested on an IEEE benchmark system showing improved capacity determination properties.
- A novel decentralized dispatch strategy is developed for EV batteries, such that they can be dispatched to save cost while ensuring reliable driving experience with sufficient SOC left in the battery and help to support the grid for load levelling or under N-1 contingency. As a decentralized approach, each EV determines the dispatch action by itself depending on its own situation and the information gathered from the power grid.
- Novel agent-based optimal dispatch algorithms are developed for EVs in coordination with RGs, so that the stability of a power network is ensured and several objectives are achieved, such as cost saving for EV users while ensuring sufficient electricity for the driving activities, reduction of wasted renewable power and provision of grid's operational support. They utilize dynamic programming and A* search procedure to solve the optimal dispatch problem, which is formulated as a distributed multi-objective constraint optimisation problem, in a decentralized way. Their robustness to the uncertainty of renewable power and EV travel patterns is also verified by the satisfactory simulation results that utilized simulated uncertain RG outputs and EV travel patterns.

[1]N-1 contingency: Only one element in a power network is in outage.

1.3 Publications Arising From This Research

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Determination of battery capacity for alleviation of thermal overload", *2014 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 12–15 Oct 2014.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Decentralized Coordination of Distributed Generators in a Distribution Network Using A*", *2016 IEEE International Conference on Power Systems Technology (POWERCON)*, 28 Sep – 1 Oct 2016.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "A*-Based Optimal Coordination of Vehicle-to-Grid Batteries and Renewable Generators in A Distribution Network", *The 26th IEEE International Symposium on Industrial Electronics*, 19-21 June 2017.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, Andrew Cruden, "Dispatch of Vehicle-to-Grid Battery Storage Using an Analytic Hierarchy Process", *IEEE transactions on Vehicular Technology*, 2016.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Optimal Coordination of Vehicle-to-Grid Batteries and Renewable Generators in A Distribution System", *Energy*, vol. 113, pp. 1250 – 1264, 2016.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "A*-Based Decentralized Dispatch of Distributed Generators in a Distribution Network", to be submitted.

Lu Wang, Suleiman Sharkh, Andy Chipperfield, "Optimal Decentralized Coordination of Electric Vehicles and Renewable Generators in A Distribution Network Using A* Search", *Energy Conversion and Management*, under review.

1.4 Thesis Outline

This thesis is organized as follows:

The background of the work presented in this thesis is covered in Chapter 2, where some key concepts that were leveraged in the following chapters are also introduced.

In Chapter 3 the work explores an AHP-based battery storage sizing approach for the relief of thermal overload caused by the N-1 contingency, taking into account the different characteristics of the possible N-1 contingencies, as mentioned earlier.

A novel AHP-based dispatch strategy of EV batteries is developed in Chapter 4, where requirements of both grid and EV users were taken into consideration, as discussed earlier. Comparisons with the rule-based approach proposed in [20] are made to demonstrate the potential benefits of this AHP-based dispatch strategy over previous approaches.

When RGs are integrated into the power grid to provide clean electricity to EVs and when a better performance of EVs is expected, the dispatch strategy needs to be improved to realize coordination amongst EVs and between EVs and renewable generator. Thus, a novel agent-based coordinated dispatch strategy is developed for EVs and renewable generators in Chapter 5 using dynamic programming approach.

The optimal coordination of EVs and RGs is investigated further using a different methodology to solve the optimization problem, A* search. In order for readers to easier and better understand this algorithm, its concept and application in optimal dispatch of distributed generators (DGs) in a distribution system are described in detail in Chapter 6, where comparisons with the dynamic programming approach used in Chapter 5 are also presented. Then, the A* search is applied to the optimal coordinated dispatch of EVs and RGs. In this work, the uncertainty of renewable power and EV driving activities are simulated using their associated stochastic models developed from Gaussian Copulas.

Furthermore, the proposed approaches in Chapters 3 and 4 were both tested on a modified IEEE Reliability Test System, while the proposed strategy in Chapter 5 and 6 was tested on a modified UK generic radial distribution network. All tests were implemented using MATLAB. The simulation results presented in the corresponding chapters demonstrate the feasibility and efficacy of proposed approaches. Finally, the overall conclusions and possible future studies are presented in Chapter 7.

Chapter 2

Background

In the Introduction (Chapter 1), a big picture of a future smart grid is described, while in this chapter more background information is provided on the main topics discussed in the thesis. In order for the readers to better understand the work presented in the following chapters, the key concepts that were utilized are also described.

2.1 Main Topics of the Research

As mentioned in the Introduction, three main topics are discussed in this thesis. The following provides an introduction of their background, including literature reviews and the motivation for the research.

2.1.1 Application and Sizing of Energy Storage for Thermal Overload Alleviation

Energy storage will perform a vital role in the smart grid with the integration of RGs as mentioned above. This technology has been proved to have the capability of supporting and improving power system performance and providing significant benefits in a variety of ways [21, 22] such as voltage and frequency control [23, 24], smoothing out the power output of wind farms [25] and achieving economic benefits at the demand side [26]. As discussed above, the control of electric current loading is necessary in the smart grid. The application of energy storage in a transmission system can help to control the current to be within its upper limits by alleviating thermal overloads that contingencies can cause. Thermal overload relief as a potentially important application of energy storage has not been investigated in depth; this has been demonstrated in a survey carried out by the Electric Power Research Institute (EPRI) [21, 27].

Three categories of thermal ratings are defined as in [22] for each transmission line: Normal (continuous), Long Term Emergency (LTE) and Short Term Emergency (STE) ratings. Thermal overload occurs when the power flow through the transmission line exceeds the corresponding continuous thermal rating. Traditionally, preventive control is applied to avoid the occurrence of thermal overload that violates the LTE rating, meaning that spare capacity is provided and the load flow over transmission lines is controlled to be well below their normal ratings so that if a contingency occurs they are not severely overloaded until the pre-defined permanent measures, such as generation re-dispatch, become effective and the overload is reduced. However, this method is conservative and costly.

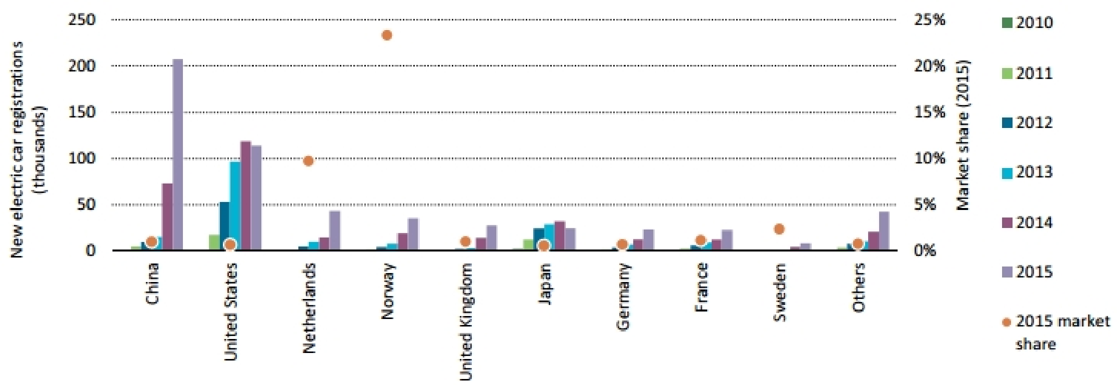
The use of fast energy storage changes the system operating paradigm from preventive to corrective. A set of fast energy storage systems can be dispatched immediately after the contingencies to bring the load flow within LTE ratings for at least 15–20 minutes, providing enough time for the permanent measures to be fully deployed. This results in the increase of transmission capability of certain corridors [22, 28], deferral of the system upgrade, and can in some systems result in significant savings of generation costs [29]. Because the energy storage device used for corrective actions in thermal overload relief must have high power, long discharge duration and fast response, batteries, as one of the most cost-effective energy storage technologies available [30], appear to be the most attractive technology.

In order for the battery storage to operate as expected, appropriate sizing of it is necessary. Many publications have explored approaches to determining optimal battery capacity, for example in wind farm applications for the purpose of smoothing power output fluctuations [31, 32, 33]. However, many papers have discussed the application of batteries in thermal overload relief without a deep investigation into the determination of battery capacity, including the assignment of charging and discharging capacities, for alleviating the thermal overload caused by contingencies [22, 29]. In [29], a set of batteries are defined to have capacities that are much larger than those actually needed and assumed to be able to be either charged or discharged any required amount of electric energy under any possible condition without determining the available charging and discharging capacities of the batteries in advance, which is too costly and not feasible. In [22], the battery size is simply determined for the severest condition. This, however, might not be satisfactory because the severest condition might happen rarely and might not be the one that requires the largest capacity or causes severe cascaded blackouts if it is not coped with in time. In other words, the severity of contingency cannot be the only factor considered when sizing the battery and factors such as probability of occurrence and potential consequences should be taken into account as well.

In response to these shortcomings, Chapter 3 proposes a novel approach to determining the battery capacity while taking into account all these different characteristics of contingencies.

2.1.2 The Integration of EVs into the Smart Grid and Appropriate Dispatch of Vehicle-to-Grid Batteries

The development and usage of EVs are inevitable as global decarbonization progresses: in [34] the real CO_2 emissions of a plug-in hybrid electric vehicles were analysed and CO_2 emission can be significantly reduced with a high-efficiency internal combustion engine; in [35] the potential of plug-in electric vehicles to substantially decrease CO_2 emissions was demonstrated; In recent years, the energy density of lithium batteries has increased such that the range of EVs will be surpassing 300km soon. Battery cost has reduced by a factor of 4 since 2008 [36, 37]. The development of wireless charging technologies, which embeds power track under roads that can charge the passing EVs while they are in motion, allows smaller and lighter batteries to be used and better battery life [38, 39, 40]. The progress of these EV-related technologies makes EVs more affordable, practical and thus more attractive to the public. The Lithium-ion automotive battery business is forecast to expand from \$1.4 billion in 2012 to \$8.5 billion in 2020 [41]. In the year 2015, the total number of EVs (including battery electric vehicles (BEV) and plug-in hybrid electric vehicles (PHEV)) on the road globally reached 1.26 million [37]. Figure 2.1[1] [37] shows the EV sales increase during the past 6 years in certain countries that have large EV markets and their EV market share in 2015. China and USA are the two largest EV markets. In Europe, nearly 200,000 EVs were sold in 2015, roughly double the figure for 2014 [42]. However, the Electric Vehicle Initiative (EVI) set the target of 20 million EVs in use by 2020, and International Energy Agency (IEA) outlines an even more ambitious project for globally deploying 150 million EVs by 2030 in order to meet the objective of limiting global average temperature increase well below 2 °C which was set in the Paris Agreement in 2015 [37]. Therefore, in order to meet these targets, a considerable growth of EV stock and deployment is required and expected.



Sources: IEA analysis based on EVI country submissions, complemented by EAFO (2016), IHS Polk (2014), MarkLines (2016), ACEA (2016a) and EEA (2015).

FIGURE 2.1: EV sales and market share in certain countries

[1] ©OECD/IEA, 2016, Global EV Outlook 2016 Beyond One Million Electric Cars, IEA Publishing. Licence: www.iea.org/t&c

The potentials of EVs and the impacts of EV migration on grids, especially distribution networks, have been investigated by many cities and countries around the world such as Canada [43], Australia [44], Macau [45], Italy [46], Ireland [47], Switzerland [48] and UK [49, 50]. The large-scale penetration of EVs will challenge the power system as additional load due to the substantial need for battery charging, but it will help to increase the penetration of intermittent RGs by absorbing the extra electricity generated from RGs, as discussed in [51, 52, 53]. When parked and connected to the grid, EVs are capable of feeding stored electricity back into the grid allowing the provision of grid operational support and the improvement of power system performance such as load levelling [54], spinning reserves [55], voltage regulation [56, 57], and reduction of the total electric power production cost [58, 59]. This is the key point of the Vehicle-to-Grid (V2G) concept, which requires each EV to have a bidirectional grid-connected battery charger with metering and communication capabilities [60, 61].

With an appropriate dispatch strategy based on the V2G concept, EVs can help to balance the system and perform the many functions of battery storage discussed earlier, because of the significant storage/generation capacity they can offer to the grid. Recent research has investigated and developed some dispatch strategies for EV batteries when EVs are parked and plugged into the grid. An algorithm is proposed in [62] for the optimal charging of EVs formulated as a stochastic dynamic programming problem taking into account the intrinsic uncertainty. The algorithm considers electricity procurement cost with an added inconvenience cost penalty term set by the EV owner to control the availability of the EV. However, the algorithm does not take into account investment or maintenance cost, nor does it take into account the geographical location of the vehicle and the associated local network constraints. Yang and He et al. [63] propose a dispatch strategy of EVs based on particle swarm optimization taking into account the EV's SOC and the operation cost of the grid without considering the cost to EV users. A charging optimization approach is proposed in [64] to maximize the total electric energy that all EVs absorb from the grid while avoiding violations of the limits of voltage and components' loading, but vehicle-to-grid power flow is not considered. In [65], fuzzy logic control is used to dispatch EVs participating in the V2G scheme, so that power grid's load levelling and voltage stability are achieved. Both the state of charge of the vehicles' batteries and the voltage at the connection node are used by the controllers to determine the dispatch action. However, the cost to EV users is not considered. Lopes et al. [61] propose a smart centralized EV charging approach to maximize EV integration capacity of the grid (i.e., the total amount of EVs that can be safely integrated into a grid). Their proposed EV grid interface control strategy allows the provision of local frequency control in an isolated system. However, the benefit for EV users is not discussed in details. Risk-aware day-ahead scheduling and real-time dispatch algorithms are developed for EV charging by Yang et al. [66] to minimize the overall charging costs, but V2G operation is not considered. Damavandi et al. [67] propose an energy hub model of a multi-energy system (MES) to dispatch an aggregation of EVs in parking lots in both

G2V and V2G modes and consider EVs' participation in the reserve market. However, this model mainly focuses on the MES operator's benefits without considering the interests of EV owners. A collaboration of demand response strategies based on dynamic pricing and peak power limiting is investigated in [15] for home-based EV dispatch using mixed-integer linear programming (MILP), which is applied to the day-ahead scheduling and the energy requirement of EV's driving activities is not considered. MILP is also used by Hua et al. [68] to optimize EV charging scheduling in an on-line way for both cost saving and grid stability. However, the possibility of selling energy back to the grid to increase the economic benefits is neglected in [68]. A rule-based EV battery dispatch strategy consisting of three rule sets is demonstrated by Ma et al. [20], where the battery characteristics, SOC and electricity buying/selling prices are considered when determining the dispatch action (i.e. charge/discharge) and the rate of dispatch (i.e. charge/discharge current), but grid operational support is not considered. In all the literatures mentioned above, a charging/discharging dispatch strategy for EV batteries to consider EV users interests and requirements by saving costs to the users while ensuring enough SOC remained within EVs and help to support the power grid operations at a reasonable reward from the grid has yet to be developed.

Chapter 4 of this thesis will then develop a novel dispatch strategy for EV batteries based on the Analytic Hierarchy Process (AHP) taking into account the relative importance of the different criteria such as cost, battery state of charge (SOC), power system contingency and load levelling.

2.1.3 Penetration of RGs into the Smart Grid and Coordinated Dispatch of EVs and RGs

EVs cannot effectively reduce carbon emissions, compared to fuel-driven vehicles, if entirely charged from coal-fired power plants instead of clean energy sources like renewable ones, which has been revealed in [8, 69]. In [70, 71], the studies also demonstrate the economic and environmental benefits of EV and RG systems. Therefore, RGs should be erected in the electricity network and provide clean electric energy for EVs' charging for the practical environmental benefits. In fact, the growth of RGs connected in low-voltage networks in the last decade has been sizeable [72, 73]. In countries such as Spain and Denmark, 21% and 42% of their national electricity supply already come from wind power, respectively [74, 75]. European Commission also set the target of increasing the share of renewable energy to at least 20% by 2020 [76].

However, the dispatch of EVs in coordination with renewable generators hasn't been investigated widely in depth in literature. As most RGs connected in the power network are small sized and geographically distributed as DGs, individual RGs are paltry compared with the metrics of the network, not to mention a single EV. Therefore, it is necessary to group them, providing the visibility needed for smart control of these systems [72]. Based on this idea, the concept of virtual power plants (VPPs) was developed: a model aggregating and managing a bunch of distributed electric power generation and load as if it is a single entity to the system operator and energy and reserve markets, as illustrated in Figure 2.2 (adapted from [72]) and in [77, 78, 79]. Figure 2.3[2] presents the control and communication within a VPP framework. There are three different types of control schemes in VPPs depending on how the decision making and data exchange are achieved: centralized, hierarchical or distributed [80, 81]. Agent-based VPPs consisting of wind generators and EVs are utilized in [82] to solve the issue of intermittent wind power and maximize the profits of selling electricity in the market. A case study [83] concerning the energy resource management of VPP, incorporating RGs, EVs, energy storage systems and demand response, in a distribution network in north Portugal indicates that this combination can achieve higher profits and lower CO_2 emissions.

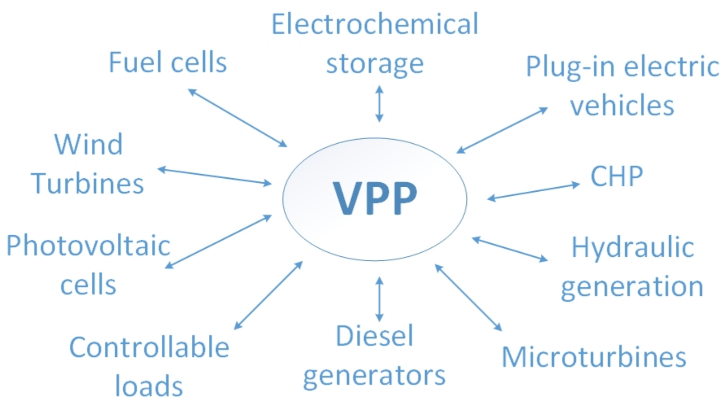


FIGURE 2.2: A VPP: a cluster of distributed electric power generation and demand with control functions

Many studies also investigated various control schemes for the coordination of EVs and RGs. A conceptual framework of wind-EV coordination is developed by Li et al. [84], which includes three-level hierarchy, to minimize the total grid operational cost including emission cost. However, this algorithm is used for day-ahead scheduling rather than real-time dispatch and requires centralized management. An optimal scheduling of EV charging is discussed by Zhang et al. [85] to minimize the mean waiting time of EVs at a renewable-power-aided charging station under cost constraint. However, vehicle-to-grid operation was not considered. Gao et al. [86] develop a hierarchical control structure for the dispatch of V2G batteries in the presence of RGs, in order to minimize operating cost. However, the proposed framework is applied to day-ahead scheduling of EV charging and discharging power instead of real-time operation. A control algorithm is proposed in [51] where wind power is used to minimize the generation cost; EVs are

[2]Reprinted from Renewable and Sustainable Energy Reviews, Vol 34, Francis Mwasilu, Jackson John Justo, Eun-Kyung Kim, Ton Duc Do, Jin-Woo Jung, Electric vehicles and smart grid interaction: A review on vehicle to grid and renewable energy sources integration, Page 507, Copyright (2014), with permission from Elsevier.

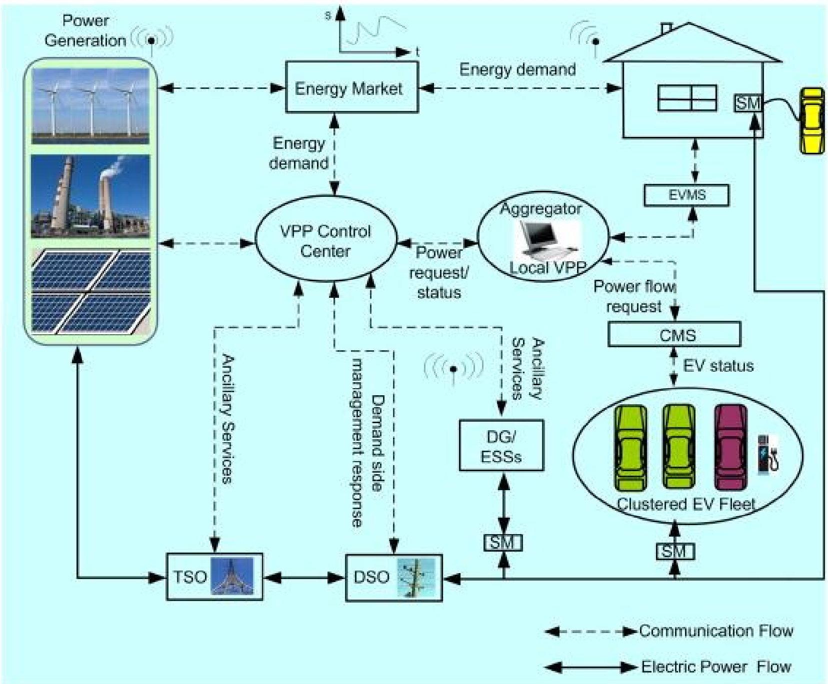


FIGURE 2.3: Control and Information Exchange within A VPP Framework

dispatched to smooth out fluctuations in wind power and improve the system's frequency regulation. However, the algorithm doesn't take into account the geographical location of the vehicle and associated local network constraints. Evangelos et al. [87] propose a distributed price-based algorithm so that EVs are coordinated with RGs and system's load levelling is achieved. However, network constraints are not considered in the model. It is a day-ahead scheduling approach instead of real-time dispatch, and the algorithm is not fully decentralized as in this model an agent requires the data from the whole network, e.g. total load demand, total DG power and SOC of EV batteries etc. In all the aforementioned literature, a comprehensive coordinated dispatch strategy for both EV and renewable generator, which takes into account the concerns and requirements of both EV users and the grid, including cost, sufficient SOC for the next journey, improved utilization of renewable energy and load levelling, was not developed.

Most of the publications mentioned above focus on the centralized dispatch of a power network, which requires an awareness of the whole network: in [88] the energy management of a stand-alone microgrid with RGs and energy storage is investigated; Mohan et al. [89] propose an optimal energy and reserve management approach of a microgrid using a two-stage stochastic model; a day-ahead scheduling approach for RG management in a distribution system is developed in [90]; the work [91], which is based on the VPP concept, proposes a two-stage optimal dispatch of DGs in a distribution network; in [92] an EV charging approach for coordination in both temporal and spacial dimensions is developed. That could be problematic when dealing with a large network incorporating a large number of stochastic and uncertain elements, such as EVs and RGs, whose behaviours (e.g. EVs' charging or driving activities, renewable power output) are likely to change at any time. Hence the volume of information and frequency that the network's operator needs to gather prior to conducting any optimal dispatch strategy or active network management will be considerable in order to make an appropriate dispatch decision based on accurate information about the network. Moreover, the number of constraints grows exponentially with network size [93], and therefore, in the end, it will become nearly impossible to solve an optimal dispatch problem in a centralized way using available computers. A decentralized approach can address the aforementioned shortcomings of centralized dispatch, which shares the computational and information gathering effort amongst several agents, thus solve the dispatch problem more efficiently. Such a system also has the advantage over a centralized approach in terms of expandability. It can be readily extended when the network expands or changes by including additional agents leaving the rest of the agents largely unchanged, as discussed in [94].

Distributed dispatch approaches have been developed by many researchers. Karfopoulos et al. [87] propose a partially distributed coordination scheme for EVs and RGs to increase their penetration into the network and minimize the network load variance. Zhang et al. [95] decompose the optimization problem using a dual concept to distribute the computational effort amongst several local controllers, but the capacity constraints of the network were not taken into account. In [96], a multi-agent system is leveraged to develop a hierarchical decentralized control scheme for DGs to ensure the security of energy supply, but it is not an optimal dispatch approach. Ren et al. [97] also use a multi-agent system to solve the optimal dispatch problem with a recursive strategy in a decentralized manner. However, as the authors point out in [97], the number of recursion will increase exponentially with the increasing penetration of DGs. There is therefore a clear gap in the research with regards to the application of decentralized system to solving the optimal coordinated dispatch of RGs and EV batteries, including V2G.

In Chapters 5–6 such a coordinated dispatch strategy will be discussed and proposed for the EVs and RGs, taking into account both grid's and EV users' concerns and their priorities.

2.2 Benchmark Papers

Mainly two papers were utilized as benchmarks in this thesis for comparisons:

1. "Modeling the Benefits of Vehicle-to-Grid Technology to a Power System" [20]. Ma et al. propose a rule-based approach to dispatch EVs considering EVs' state of charge (SOC), the electricity prices and different time periods during a day. The rules are basically the if-then sets, which determine the dispatch actions of EVs with regards to the current conditions, which are illustrated in Figure 2.4. So basically, the EVs can

only be charged if the control signal is off, otherwise, the EV will be discharged when its SOC is high and charged when SOC is low. The charging/discharging current is determined by the electricity price and depends on the time period of a day. However, the provision of power system operation supports is not considered in this approach.

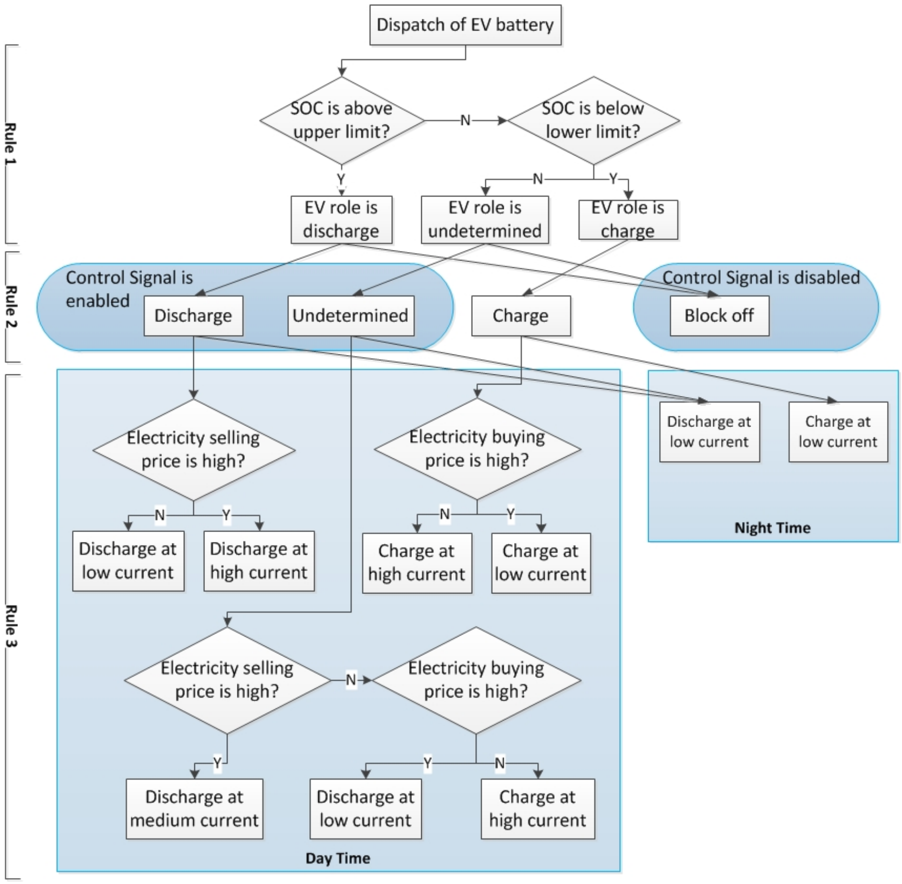


Figure 2.4: Rule-based Dispatch Strategy for an EV Battery

2. "Optimal Decentralised Dispatch of Embedded Generation in the Smart Grid" [98]. Miller et al. propose a distributed constraint optimization problem (DCOP) for DG dispatch which is solved with a max-sum algorithm and dynamic programming, respectively, to minimize the carbon emissions of the network. The computational complexity of the proposed algorithms is also discussed. Based on this paper, DCOP is developed to incorporate multiple objectives for the optimal coordination of EVs and RGs in this thesis, and dynamic programming is applied to solving this more complicated optimal dispatch problem. The details are demonstrated in Chapter 5, therefore for the consistency and better presentation of the thesis, the algorithm is not presented here. The dynamic programming algorithm proposed in this paper [98] is also used as a benchmark in comparison with a proposed A*-based algorithm in this thesis in terms of computation and communication amounts and their inherent properties.

2.3 Essential Concepts

In this section, several key concepts that were utilized in this thesis are introduced, including some relevant state-of-art methodologies.

2.3.1 Decision Making Given Various Criteria

Decision making is a very important part of scientific, economic and social activities. Many studies have investigated and utilized decision making methods such as Best-Worst Method [99], multiplicative and fuzzy preference relations [100], group decision making based on linguistic preference relations [101], fuzzy logic [102], reinforcement learning-based dynamic decision making [103], and learning automata-based decision making [104]. Amongst them, fuzzy logic has attracted a lot of attentions in recent years and is increasingly frequently used in various areas such as investment [105], cloud service [106], power engineering [107], and other engineering applications [108]. Fuzzy logic is a methodology based on natural language. Its primary mechanism is built on qualitative description used in daily language. Unlike conventional control techniques, fuzzy logic highly tolerates imprecision and build it into the process, thereby lowering the cost of solution [109]. It maps an input space to an output space, both of which are defined by fuzzy sets (sets that are without concrete boundaries and contain elements with a partial degree of membership), according to a list of if-then statements, i.e., rules [110]. A graphical example is shown in Figure 2.5. Fuzzy logic can also be used in combination with other algorithms, such as neuro-computing [111, 112, 113] and genetic algorithm [114, 115], so that complex nonlinear systems can be modelled in a much easier way.

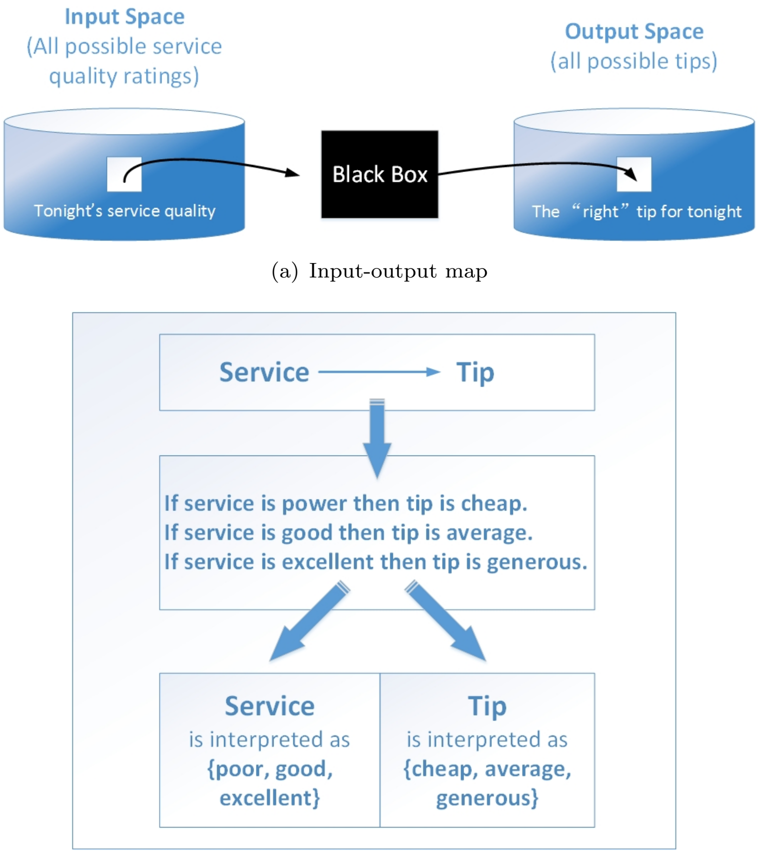
Another well-proven multi-criteria decision-making (MCDM) method is the analytic hierarchy process (AHP) developed by Thomas L. Saaty [116]. It has been widely used in various areas including power systems [117, 118], due to the following characteristics:

1. AHP is convenient and effective in solving complex MCDM problems, where the criteria are described in different dimensions and can be conflictive with each other.

2. AHP can process qualitative information that is very hard to be presented by exact numbers. Thus, instead of directly quantifying those qualitative data, AHP uses pairwise comparison to determine the relative importance between different factors.

In [117] AHP was used to weigh different objective functions and the resulting weighted combination of these functions became the final objective function for combined active and reactive dispatch. In [118] the locations of VAR sources were selected and ranked taking into account the different benefit-to-cost ratios and other relevant factors, which are weighted using AHP according to their relative importance in VAR placement.

The steps of AHP algorithm are as follows:



(a) Input-output map

(b) Fuzzy inference system — What Happens in the Black Box in (a)

FIGURE 2.5: An Example of Fuzzy Logic — Tipping Problem: Given the Service Rating, Determine the Apt Tip

Step 1: A hierarchy model, e.g., Figure 2.6, is established based on the analysis of the problem.

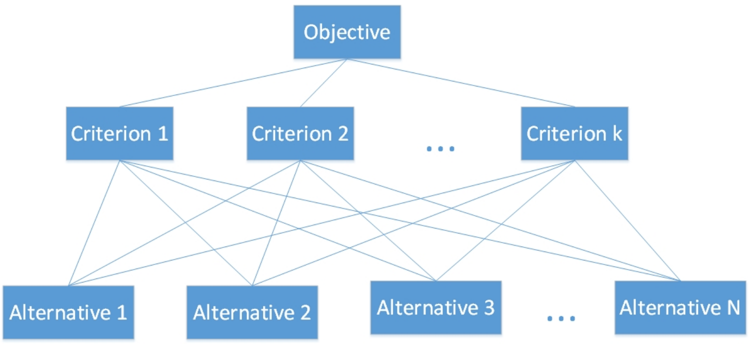


FIGURE 2.6: A 3-Level AHP Hierarchy model

Step 2: Pairwise comparison matrices are formed at each level of the hierarchy model. Each element $A(i, j)$ in the pairwise comparison matrix A depicts the expert's judgement of relative importance between the pair of ith and jth factors using the ratio scale method [116], [118]. It is obvious that $A(i, j)$ is the reciprocal of $A(j, i)$.

Step 3: For each pairwise comparison matrix, the maximum eigenvalue and the corresponding eigenvector are calculated. The normalization of the calculated eigenvector gives the normalized principle eigenvector with elements equal to the priority scales (weightings) of the factors.

Step 4: The consistency of judgements is checked. Consistency Index (CI) and Consistency Ratio (CR) are measured for each pairwise comparison matrix, which are presented in (2.1) and (2.2).

$$CI = \frac{\lambda_{max} - n}{n - 1} \tag{2.1}$$

where λ_{max} and n are the maximum eigenvalue and the dimension of the corresponding pairwise comparison matrix, respectively.

$$CR = \frac{CI}{RI} \tag{2.2}$$

where RI is a given random consistency index. If CR is no larger than 10%, the inconsistency of judgements is acceptable. Otherwise, the judgements need to be revised.

In comparison with fuzzy logic, these two methods share some similarities such as they both can interpret qualitative information described in nature language, can be built upon the experts' experience, have high tolerance of imprecision and can be used with other algorithms to realize more powerful functions as a whole. However, AHP plays a better role in dealing with trade-off and criteria that are supposed to be treated with different priorities. In the cases studied in this thesis, AHP turns out to be easier to use and more flexible in terms of it can be readily adapted to be combined with other techniques that are investigated or leveraged in these works, therefore AHP is more appropriate in the scope of this thesis.

When sizing the battery storage for thermal overload alleviation in Chapter 3, different scenarios require different battery capacities, either charge or discharge, which implies that every possible contingency needs to be weighted based on their relative importance. When dispatching the EV batteries in Chapter 4, focusing on the different requirements of users and the grid will inevitably result in different dispatch patterns of EV batteries, therefore every requirement needs to be weighted according to its priorities in determining the dispatch mode of EV batteries. In Chapter 5 and 6, the dispatch of EVs and RGs within a distribution network is expected to best achieve several objectives with respect to the different requirements of EV users and power grid. Hence, a multi-objective function needs to be formulated for this optimal dispatch problem, which contains multiple defined objectives of dispatch while taking into account their priorities in determining the dispatch mode of EV batteries and RGs. As discussed earlier, the AHP, as a tool that can be used to determine the weighting factors (priorities) of different criteria and alternatives, is thus used in these works.

2.3.2 Modelling and Simulation of Uncertainties

Uncertainty, such as intermittent renewable power and EVs' travel patterns, is considered in this thesis. This is an important issue in the future smart grid operation and control and has attracted many researchers' attentions [119, 120, 121]. Iversen et al. [62] use a Markov decision process (MDP) to model the random driving patterns of the EVs and then propose a charging policy based on that. In [85], renewable energy, EV arrivals and electricity price are respectively modelled as a finite-state ergodic Markov process, based on which an optimization problem is constructed to figure out the optimal EV charging sequence and renewable energy assignment for the minimum queuing time and a reasonable cost. MDP is also used in [122] to formulate the matching problem of uncertain wind power and EV charging demand. Furthermore, EV uncertain mobility could cause mismatch risk if its driving activity varies from the day-ahead forecast, which is discussed and solved by Yang et al. [66] using stochastic linear programming. In [123], Wang et al. handle the uncertainties of photovoltaic and wind power by analysing their mathematical expectation, second-order original moments and variances, according to their assumed probability distribution, and then incorporate those analytic uncertainties into the stochastic optimization problem.

In this thesis, two different ways of dealing with uncertainty are utilized in simulations of EVs and/or RGs dispatch.

In Chapters 4 and 5, Monte Carlo simulation is used, given the probability distribution of parked EVs throughout a day, to randomly select which EVs depart and how much electric energy they have when they finish a journey and park. This is a simple way to model and simulate uncertainties, which however is not good enough to reflect the case in reality.

In most practical cases, random variables are not completely independent — they could have complicated relationships with each other, and these variables could come from different marginal distributions individually. Therefore, copulas, which represents the dependence structure between variables, are used to generate data from multivariate distributions in order to reflect the traits of input data [124].

Copulas are basically multivariate distribution functions whose 1-D probability distributions are uniform on the interval [0, 1], which join univariate marginal distributions to form multivariate distribution functions [125]. Specifically, according to Sklar's theorem [126], the continuous random variables $X_1, \ldots, X_k$ with cumulative distribution functions (CDFs) $F_{X_1}, \ldots, F_{X_k}$, are joined by copula C if their joint distribution can be written as

$$F_{X_1,\ldots,X_k}(X_1, \ldots, X_k) = C(F_{X_1}(x_1), \ldots, F_{X_k}(x_k)). \tag{2.3}$$

As CDFs F_{X_i} transform random variables X_i into uniform variables U_i, $F_{X_i}(x_i) = u_i$ for $i = 1, \ldots, k$ where $u_1 \ldots u_k$ are respectively realizations of uniform variables $U_1, \ldots, U_k$.

Therefore, (2.3) can be rewritten as

$$C(u_1, \ldots, u_k) = F_{X_1,\ldots,X_k}(F_{X_1}^{-1}(u_1), \ldots, F_{X_k}^{-1}(u_k)) \tag{2.4}$$

Gaussian copulas are a frequently utilized family of copulas, which are constructed from the multivariate Gaussian distribution, which can be defined as

$$C_\rho(u_1, \ldots, u_k) = \Phi_\rho(\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_k)) \tag{2.5}$$

where Φ denotes the standard normal CDF and ρ the linear correlation matrix between standard normal variables $\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_k)$. The details are given as follows:

Step 1: Calculate the rank correlation, such as Spearman's ρ (ρ_s), between random variables $X_1, \ldots, X_k$. Then calculate the linear correlation ρ between the corresponding normal variables $Y_{X_1} = \Phi^{-1}(F_{X_1}(X_1)), \ldots, Y_{X_k} = \Phi^{-1}(F_{X_k}(X_k))$, using

$$\rho = 2sin(\frac{\pi}{6}\rho_s), \tag{2.6}$$

where $F_{X_1}, \ldots, F_{X_k}$ are respectively, CDFs of continuous random variables $X_1, \ldots, X_k$. Φ^{-1} is the inverse standard normal CDF.

Step 2: Simulate $y_{x_1}, \ldots, y_{x_k}$ from multivariate standard normal distribution with correlation ρ (e.g. use mvnrnd command in MATLAB).

Step 3: Transform the simulated values $y_{x_1}, \ldots, y_{x_k}$ back to the original domain via standard normal CDF Φ and inverse CDF of each random variable: $x_1 = F_{X_1}^{-1}(\Phi(y_{x_1})), \ldots, x_k = F_{X_k}^{-1}(\Phi(y_{x_k}))$.

This method was used in Chapter 6 to model and simulate the stochastic travel patterns of EVs and intermittent wind power. Readers can refer to Chapter 6 for more details.

2.4 Conclusions

In this chapter, the backgrounds of three main topics studied in this thesis are presented, which give readers a better understanding of the context of researches in this thesis. Moreover, in order for the consistency of presentation in the following chapters, the key concepts that are utilized several times in the works are introduced in this chapter, together with the description of certain essential methodologies that are relevant.

Chapter 3

Determination of Battery Capacity for Thermal Overload Alleviation

The application of batteries in a transmission system can help alleviate thermal overload and increase the transmission capability of critical corridors. The purpose of this chapter is to propose an approach for determining the capacity of a battery providing support during N-1 contingencies to relieve transmission line thermal overload, with the probability, severity and potential consequences of the contingencies taken into account. The Analytic Hierarchy Process (AHP) is used to take into account the relative importance of the different characteristics of the contingencies when determining their relative significance in battery capacity determination. The proposed approach was tested on the IEEE Reliability Test System.

3.1 Introduction

Energy storage has proved to have the capability of supporting and improving power system performance and providing significant benefits in a variety of ways. This chapter focuses on its application in a transmission system, especially the alleviation of thermal overload, in order to improve the system's reliability and defer the system upgrades. Thermal overload relief as a potential important application of energy storage has not been investigated in depth, as illustrated in a survey carried out by Electric Power Research Institute (EPRI) [22]. However, there are some successful previous projects which installed energy storage assets at low-voltage transmission and distribution levels to realize several reliability objectives [127]:

Wisconsin Public Service (WPS) — In July 2000, WPS and American Superconductor installed a Distributed-Superconducting Magnetic Energy Storage System (D-SMES) to relieve the transmission congestion on the Rhinelander loop — a 115 kV, 200 mile loop in northern Wisconsin that is constrained by stability issues. An alternative to the D-SMES solution would have been a transmission system's upgrade, which were estimated to cost $35-46M, take 10 years to complete, and intrude on a bald eagle habitat. Thus storage provided the very short duration needed at roughly one tenth the cost and a faster, less intrusive installation.

Presidio, Texas — A 4 MW, 32 MWh Sodium-Sulfur (NaS) battery was installed to support power quality on a 100 km, radial 69 kV transmission line that feeds the border town of Presidio. Due to its fast response, the battery is utilized for voltage regulation and to improve power quality. It can also be used to supply power for 8 hours during an outage, giving grid operators sufficient time to import power from Mexico without power interruption. The battery system is part of a more complete $45M transmission upgrade to the region, approved by ERCOT and the Texas PUC. Thus, the cost of the battery will be distributed across all rate payers in Texas.

In order for the battery storage to operate as expected, appropriate sizing of it is necessary. The simplest way is to determine the battery capacity by the largest energy amount required by the possible contingencies for the corresponding corrective actions. Although this approach will result in a battery capacity that has the best performance (i.e., 100% N-1 contingencies' overload can be alleviated thoroughly during a defined period), it could be not cost-effective and could cause redundant capacity in the battery. Because, the contingency that requires the greatest amount of energy might very rarely occurs and/or cause only mild overload. Another simple approach to sizing the battery is then based on the severest condition [22]. This, however, might not be adequate because the severest condition might happen rarely and might not be the one that requires the largest capacity or causes severe cascaded blackouts if it is not coped with in time. In other words, the severity of contingency cannot be the only factor considered when sizing the battery and factors such as probability of occurrence and potential consequences should be taken into account as well.

In this chapter, a battery-capacity-determination approach is designed and proposed using an AHP-based method to determine the capacity of a battery providing support during N-1 contingencies (i.e., only one component of transmission system is in outage) to relieve the thermal overload, with the probability, severity and potential consequences of the contingencies taken into account. The proposed approach was tested on an IEEE Reliability Test System [128] under both N-1 and N-2 contingencies (i.e., two components of transmission system are in outage). The comparison with the battery sizing approach in [22] has been made to demonstrate the potential benefits of the proposed AHP-based methods in this work. However, it is important to note that the installation of battery storage is not for the contingencies and overload relief only. It can also be used in other power system applications and provide ancillary services such as frequency and voltage regulations, reserve, load levelling and interruption backup etc. Here is an evaluation of battery capacity in terms of what is required from its important function of overload alleviation under contingencies.

This work has been presented in IEEE PES Innovative Smart Grid Technology Europe 2014 [129].

3.2 Methodology

The objective of the proposed battery-capacity-determination approach is to define the capacity of the battery providing support during N-1 contingencies to relieve transmission line thermal overload, with the probability, severity and potential consequences of the contingencies taken into account. The power system can normally be separated into several areas, according to the transmission voltages or geographic locations, which are connected by several transmission corridors. In this work, only one battery is placed in each area of the power system and dispatched for every contingency that causes thermal overload beyond LTE ratings in that area. In each area, the battery is located at the bus that is connected to the most overloaded line. In the IEEE Reliability Test System under study, this was found to be the bus with the highest sensitivity factor (i.e. the ratio between the resulted change of the line flow and the corresponding change of power injection to the bus) as defined in [22]. By performing power flow analysis, extra power that needs to be absorbed or injected at the battery-siting bus, to bring the load flow of affected transmission lines within LTE ratings under every possible N-1 contingency, can be derived, and it would be equal to the corresponding required battery size for a particular contingency. It is obvious that the battery sizes required for different N-1 contingencies may differ, therefore, each contingency must be weighted as discussed earlier. It is important to note that only those N-1 contingencies that cause thermal overload and require corrective actions of the battery are considered and weighted. Moreover, as the annual outage rates of these contingencies are less than 1 [128], it is reasonable to assume that in this work the battery will not be frequently dispatched, unless it is heavily used for other ancillary services; a balance needs to be struck between meeting contingency needs and providing ancillary services, which maybe necessary to make battery storage economically viable.

3.2.1 Hierarchy Model for Weightings of N-1 Contingencies

In order to weigh each contingency, a hierarchy model is structured based on AHP (The details of AHP are presented in Chapter 2), as shown in Figure 3.1, where four levels are included. The top level is the objective, i.e. to weigh every possible N-1 contingency that requires battery corrective action. The second level is the criteria of weighing contingencies: probability of the occurrence (Pro), severity of overload (Sev) and the potential consequences (PotCon) if the overload cannot be relieved to be within LTE ratings during the first 15 minutes and causes damage to the overloaded line. The probability of occurrence can be determined by using the available data such as the permanent outage rate of every component in a transmission system. The severity of overload is quantified as a percentage of the LTE rating, as follows:

$$Overload\ Severity = \frac{Load\ Flow - LTE}{LTE} \times 100\%. \tag{3.1}$$

It is important to note that only the overload that violates the LTE rating and requires the battery corrective action is considered and measured. The third level forms the sub-criteria for evaluating potential consequences if the contingency is not alleviated within the first 15 minutes. As a result, the overloaded line may be tripped out, which would overload other lines and could result in cascaded blackouts. To quantify the consequences, two criteria are considered: 1) the number of overloaded lines, 2) the average of the overload, expressed as percentages of LTE ratings and calculated using (3.1).

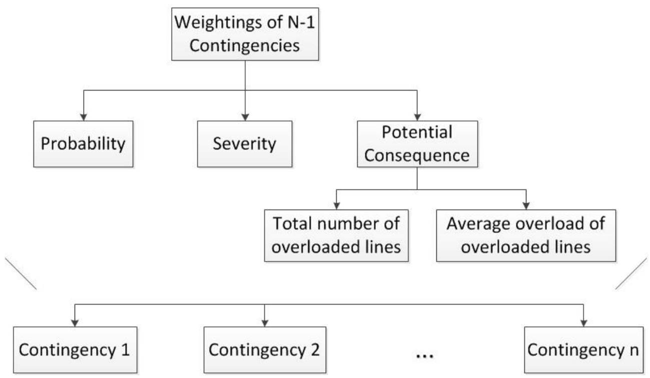


FIGURE 3.1: Hierarchy model for weightings of N-1 contingencies

For each contingency, the second level criteria of probability, severity and potential consequence can be determined from available data or load flow analysis. These can then be normalized and expressed as percentages. The third level sub-criteria of the number of overloaded lines and the average overload are given equal weighting of 50% each.

The relative importance (weighting) of probability, severity and potential consequence is determined using AHP by forming a pairwise comparison matrix PC. Among the three characteristics of contingency, probability is considered as most important, because the battery is expected to cope with as many contingencies as possible. Potential consequence is considered as relatively more important than the severity of contingency, because a contingency that does not cause severe overload but has a severe potential consequence is surely more important and needs more attentions than one that causes severe overload but no severe potential consequence. Because all these three characteristics are important in battery capacity determination, small ratio scales are selected to present the relative importance between each pair of them (i.e. probability is 3 times as important as severity and twice as important as potential consequence, and potential consequence is twice as important as severity.). Therefore, the resulted pairwise comparison matrix PC is shown as follows:

$$PC = \begin{array}{c} \\ Pro \\ Sev \\ PotCon \end{array} \begin{array}{c} \begin{array}{ccc} Pro & Sev & PotCon \end{array} \\ \begin{pmatrix} 1 & 3 & 2 \\ \frac{1}{3} & 1 & \frac{1}{2} \\ \frac{1}{2} & 2 & 1 \end{pmatrix} \end{array}. \tag{3.2}$$

By calculating the normalized principle eigenvector of PC, the resulting priority scales are obtained as 53.96%, 16.34% and 29.7% for probability, severity and potential consequence, respectively.

Finally, multiplying each weighting of the contingency by the priority of its sub-criterion (if applicable) and criterion and summing up the resulting weightings for each contingency gives the final composite weighting of each contingency.

3.2.2 Battery Capacity Determination

Battery corrective actions have two categories: charge and discharge, and the battery capacity needs to be large enough to cope with both conditions. Therefore, all the N-1 contingencies requiring battery corrective actions are separated into two groups, contingencies that require charging actions and ones requiring discharging actions. Then the composite weightings of contingencies are adjusted through normalization so that the sum of adjusted weightings of contingencies in each group is equal to 1, as follows:

$$w_a^i = w_{pa}^i / \sum_{i=1}^{j} w_{pa}^i, \quad i = 1,\ 2 \ldots j \tag{3.3}$$

$$w_a^i = w_{pa}^i / \sum_{i=j+1}^{n} w_{pa}^i, \quad i = j+1,\ j+2 \ldots n \tag{3.4}$$

where contingency 1 to j require charging actions while contingency $j+1$ to n discharging actions of the battery. w_a^i is the adjusted composite weighting of ith contingency. w_{pa}^i is the pre-adjustment composite weighting of ith contingency.

Assuming that the pre-defined permanent measures take 20 minutes to be effective, the corrective actions of a battery should be sustained for at least 20 minutes. Furthermore, the round-trip efficiency of the battery is assumed to be 80%. The required charging or discharging capacity for every corrective action is calculated as the product of the required power evaluated and the duration of $\frac{1}{3}$ hours (20 minutes) divided by the efficiency of 80%. Therefore, the charging C_c and discharging C_d capacities are derived by calculating the weighted average in each group of contingencies, as follows:

$$C_c = w_a^1 E_c^1 + w_a^2 E_c^2 + \cdots + w_a^j E_c^j \tag{3.5}$$

$$C_d = w_a^{j+1} E_d^{j+1} + w_a^{j+2} E_d^{j+2} + \cdots + w_a^n E_d^n \tag{3.6}$$

where E_c^i is the charging capacity required for ith contingency while E_d^i the discharging capacity required.

One approach to calculating the required battery capacity C is to sum up C_c and C_d, as

$$C = C_c + C_d. \tag{3.7}$$

Equation (3.7) allows the battery to cope with contingencies that require either charging or discharging actions.

However, charging and discharging actions are not considered as equally important from the perspective of alleviating the thermal overload caused by N-1 contingencies, because the total pre-adjustment composite weightings of contingencies in charge and discharge groups are different. Another approach would be to add weightings to the calculated charging C_c and discharging C_d capacities before summing them up, as follows:

$$C = w_c C_c + w_d C_d \tag{3.8}$$

where the two weightings w_c and w_d are determined by the relative importance between two groups of contingencies, as

$$w_c : w_d = \sum_{i=1}^{j} w_{pa}^i : \sum_{i=j+1}^{n} w_{pa}^i, \tag{3.9}$$

and the sum of w_c and w_d is equal to 2 allowing the battery to cope with contingencies that require either charge or discharge actions.

This approach actually changes the proportions of charging and discharging capacities within the battery. Both of these two methods are tested on the IEEE Reliability Test System.

3.3 Test Results

The proposed two approaches based on (3.7) and (3.8) were tested on the IEEE Reliability Test System with 38 branches, 24 buses and 2 areas as shown in Figure 3.2

[128]. Area I consists of transmission lines with voltage level of 230kV and transformers, while area II transmission lines with voltage level of 138kV. For the purpose of study and convenience, the system is stressed by decreasing the three thermal ratings of the transmission lines by a factor of 1.46, so that several N-1 contingencies would cause significant overload with only one line loaded beyond its LTE rating in each case. The power flow analysis under every N-1 contingency was performed in MATLAB using MATPOWER 4.1 [130], which gave two congestion points (viz. bus 6 and 16), with one in each area of the system, and a battery was sited at each point. Furthermore, there are 11 N-1 contingencies causing thermal overload violating LTE ratings, of which two require the corrective actions of battery at bus 6 and the rest requires the battery at bus 16. For the battery at bus 6, the two relevant contingencies are the single outage of the transmission lines from bus 2 to 6 and 6 to 10 respectively and both of them require discharging actions. Taking into account the battery efficiency of 80%, a capacity of 5.4 MWh is enough for the battery at bus 6 to handle both of these two contingencies.

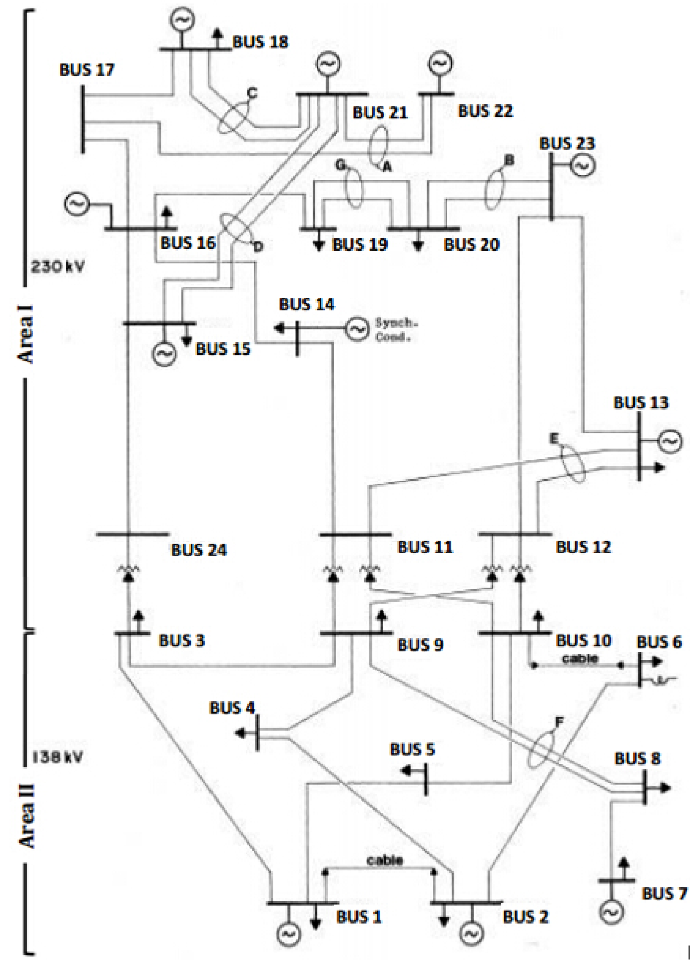


FIGURE 3.2: IEEE Reliability Test System diagram

As for the battery at bus 16, the proposed approaches were used to determine the battery capacity, where only those N-1 contingencies that require corrective actions of this battery are considered. The probability of contingency occurrence utilizes the corresponding permanent outage rate in [128]. The severity of overload is calculated as a percentage of the LTE rating using (3.1) through load flow analysis. Furthermore, the potential consequence of each contingency is measured, assuming that the contingency occurs together with the outage of the corresponding overloaded line, through the total number and average overload of further overloaded lines, where the overload is measured using (3.1) as well. The normalized weightings of each contingency with respect to the criteria or sub-criteria presented in Figure 3.1 are calculated, as shown in the Table 3.1, by computing the ratio of the actual value of a contingency evaluated for the corresponding (sub-)criterion (e.g. the annual outage rate or the overload severity of a contingency) and the sum of all contingencies' values for that (sub-)criterion. An example is given in (3.12). Furthermore, the composite and adjusted weightings of the N-1 contingencies that require battery actions are calculated as described in Section 3.2 and presented in Table 3.2. The corresponding charging or discharging capacities that are required at bus 16 to bring the load flow within LTE ratings are demonstrated in Table 3.2 as well, taking into account the battery's round-trip efficiency of 80%. For clarification, the equation to calculate the composite weighting w_{pa}^1 and adjusted weighting w_a^1 of contingency 1 are given as an example:

$$\begin{aligned} w_{pa}^1 = w_1^P \times 53.96\% + w_1^S \times 16.34\% + w_1^T \times 50\% \\ \times 29.7\% + w_1^A \times 50\% \times 29.7\%, \end{aligned} \tag{3.10}$$

$$w_a^1 = w_{pa}^1 / \sum_{i=1}^{j} w_{pa}^i, \tag{3.11}$$

where w_{pa}^i is the pre-adjustment composite weighting of ith contingency. Contingency 1 to j require charging actions as described previously. $w_1^P, w_1^S, w_1^T, w_1^A$ are the weightings of contingency 1 with respect to its criteria and sub-criteria, i.e. probability, severity, total number and average overload of overloaded lines, respectively, which is calculated via normalization. The equation to calculate w_1^P is given as an example:

$$w_1^P = \frac{Annual\ outage\ rate\ of\ contingency\ 1}{Sum\ of\ annual\ outage\ rates\ of\ all\ contingencies\ considered}, \tag{3.12}$$

where annual outage rate of each contingency can be found in [128].

Based on Table 3.2, the weighted averages of charging and discharging capacities are calculated to be 45.56 MWh and 81.47 MWh respectively, resulting in a total battery capacity of 127.03 MWh. However, when the relative importance between charging (60.1%) and discharging (39.92%) corrective actions is considered and the corresponding weightings of charging (1.2) and discharging (0.8) capacities are added, the resulting battery capacity is 119.84 MWh with 54.67 MWh for charge and 65.17 MWh for discharge. In this way, the battery capacity can save 7.19 MWh while achieving the same percentage of handleable N-1 contingencies.

TABLE 3.1: Weightings of the N-1 contingencies requiring battery actions at bus 16 with respect to each criterion or sub-criterion

Branch Outage		**Probability (53.96%)**	**Severity of Overload (16.34%)**	**Potential Consequence (29.70%)**	
From Bus	***To Bus***			***Total number (50%)***	***Average overload (50%)***
3	24	0.61%	25.56%	14.29%	7.35%
12	23	15.85%	10.41%	7.14%	8.27%
13	23	14.94%	15.66%	14.29%	4.26%
15	16	10.06%	2.31%	10.71%	12.95%
15	21	12.5%	2.16%	3.57%	20.20%
15	21	12.5%	2.16%	3.57%	20.20%
15	24	12.5%	25.56%	14.29%	7.35%
16	17	10.67%	0.42%	10.71%	12.95%
16	19	10.37%	15.77%	21.43%	6.48%

TABLE 3.2: Composite and adjusted weightings of N-1 contingencies with required battery capacities at bus 16

Branch Outage		**Composite Weight**	**Adjusted Weight**	**Charge (MWh)**	**Discharge (MWh)**
From Bus	***To Bus***				
3	24	7.72%	12.85%	69.76	
12	23	12.54%	20.87%	29.64	
13	23	13.38%	22.26%	38.10	
15	16	9.32%	23.35%		26.83
15	21	10.63%	26.63%		139.39
15	21	10.63%	26.63%		139.39
15	24	14.14%	23.53%	69.76	
16	17	9.34%	23.40%		4.11
16	19	12.32%	20.50%	26.91	

The percentage of handleable contingencies (P) in Figure 3.3 and Figure 3.6 is calculated as follows:

$$P = \frac{No.\ of\ handleable\ contingencies}{No.\ of\ all\ possible\ contingencies} \times 100\%, \tag{3.13}$$

where all possible contingencies include contingencies that cause thermal overload and those that cause no overload. The handleable contingencies contain the contingencies that can be handled by either the battery at bus 16 with the corresponding capacity or the battery at bus 6 with the capacity of 5.4 MWh as discussed earlier.

The minimum battery capacity proposed using either first or second approach for every possible percentage of handleable N-1 contingencies is calculated with the same ratio of charging to discharging capacity as that of the battery capacity proposed by the corresponding approach, as follows:

$$C^{min} = C_c^{min} + C_d^{min} \tag{3.14}$$

Using first approach based on (3.7):

$$C_c^{min} : C_d^{min} = C_c^1 : C_d^1 = 45.56 : 81.47 \tag{3.15}$$

Using second approach based on (3.8):

$$C_c^{min} : C_d^{min} = C_c^2 : C_d^2 = 54.67 : 65.17 \tag{3.16}$$

where C^{min} is the minimum battery capacity capable of handling a certain possible percentage of the N-1 contingencies, including a charging capacity of C_c^{min} and a discharging capacity of C_d^{min}. C_c^1 and C_d^1 are respectively the charging and discharging capacities proposed by the first approach, while C_c^2 and C_d^2 proposed by the second approach, which have been discussed earlier.

In the Figure 3.3, the red-filled circle represents the proposed battery capacity at Bus 16 calculated by the first approach based on (3.7), while the blue-filled square calculated by the second approach based on (3.8). Both of them depict the capability of handling 90.33% of total N-1 contingencies together with the battery at bus 6 with the previously discussed capacity. Furthermore, the red empty circles are the minimum battery capacities at bus 16 proposed for all possible percentages of handleable N-1 contingencies using the first approach. However, the set of blue empty squares are the minimum battery capacities calculated based on the second approach. It is obvious that the blue solid polyline connecting blue empty squares is mostly beyond the red dashed polyline connecting red empty circles, meaning that the second approach taking relative importance of charging and discharging capacities into account is generally better. In order for more detailed presentation, Figure 3.3 is separated into 2 figures, Figure3.4 and Figure 3.5, with each of them respectively demonstrating the corresponding charging and discharging part of the total battery capacities depicted in Figure 3.3.

The two different sets of minimum battery capacities proposed by the two different approaches are tested for all N-2 contingencies in Figure 3.6, with the red dashed polyline (first approach) below the blue solid line (second approach) for the most part. Through the first approach based on (3.7), the proposed battery capacity (red-filled circle) is able to handle 75.55% of the N-2 contingencies while the second approach (blue-filled square) based on (3.8) achieving 76.14% with 0.6% higher than the first approach.

From Figure 3.3 and Figure 3.6, it is clear that the second method has a more reasonable assignment of charging and discharging capacities within the battery resulting in a generally better result.

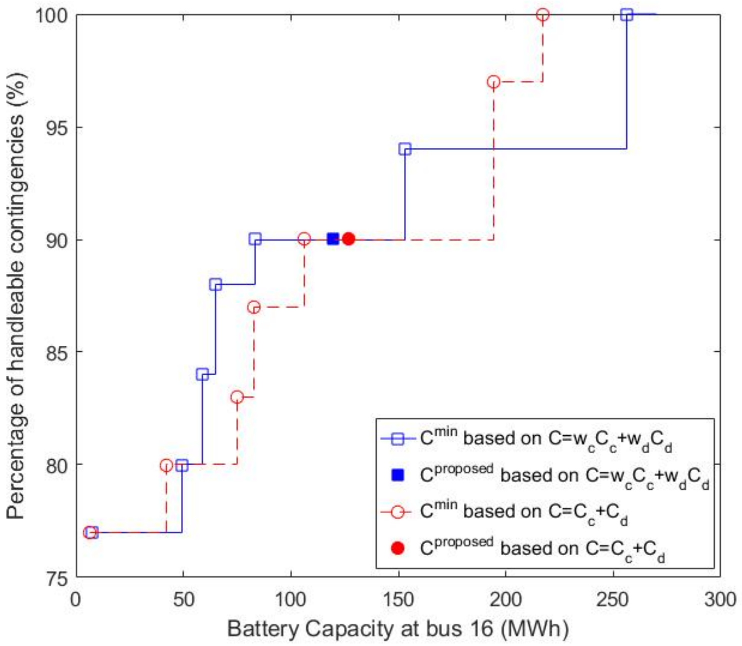


FIGURE 3.3: Percentage of handleable N-1 contingencies Versus Required Battery Capacity at Bus 16 Proposed Using Two Approaches

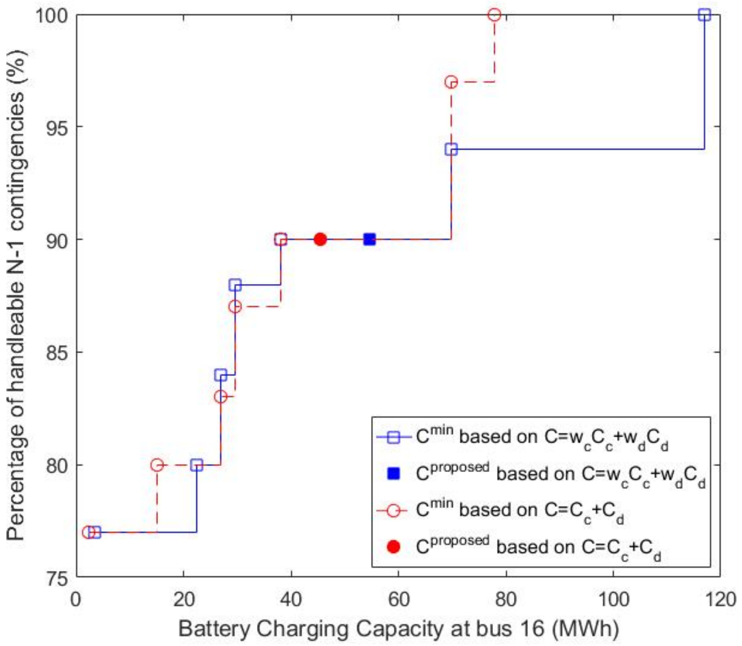


FIGURE 3.4: Percentage of handleable N-1 contingencies Versus Required Charging Capacity at Bus 16 Proposed Using Two Approaches

If the battery at bus 16 is sized based on the severest condition as in [22], then its capacity is estimated to be 70 MWh (with zero discharging capacity) taking into account the battery efficiency of 80%. Such a battery can handle 88% of the N-1 contingencies and 74% of the N-2 contingencies, which are less than those achieved by the proposed two

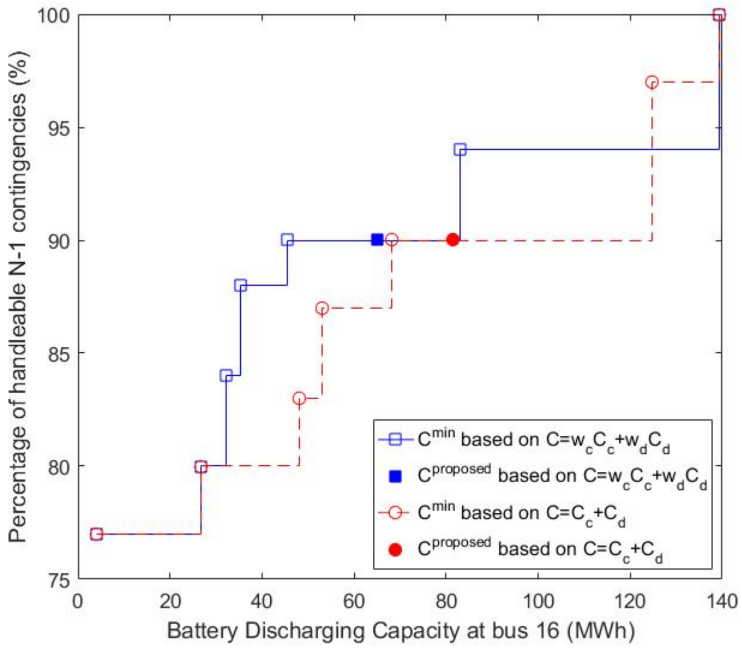


FIGURE 3.5: Percentage of handleable N-1 contingencies Versus Required Discharging Capacity at Bus 16 Proposed Using Two Approaches

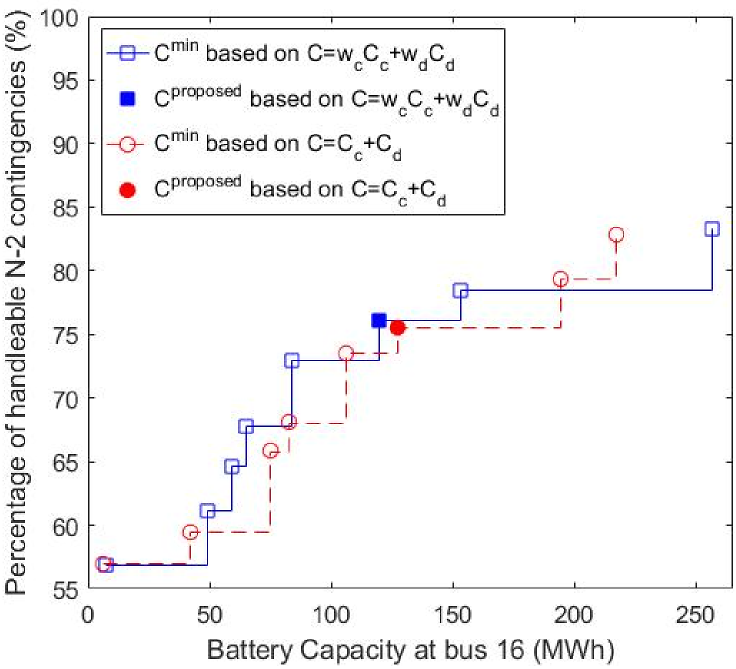


FIGURE 3.6: Proposed Battery Capacity at Bus 16 Versus Percentage of handleable N-2 contingencies using two approaches

approaches. The main problem of this method is that it cannot handle any contingency requiring discharging corrective action at all, meaning that when the contingency that requires discharging action occurs, no immediate corrective action can be taken within the first 15 minutes and the overloaded lines may be tripped out and more lines will be overloaded, which could result in additional aggravating consequences such as cascaded blackouts. However, the proposed battery capacities calculated by the two approaches can handle the contingencies that require the largest charging or discharging capacity to some extent. According to the calculation for the worst case, with proposed battery capacity calculated by either first or second approach, the affected line is loaded beyond LTE ratings for approximately 10 minutes at most. Because STE ratings can be sustained for 5–15 minutes for the security concern [22], the system is definitely more secure with the proposed battery capacity than with the battery capacity sized by the severest condition. Therefore, the proposed approaches are better in this respect.

3.4 Sensitivity Analysis

In order to understand how different assignment of criteria priorities affect the final result, the sensitivity analysis is conducted. In this work, the priorities of the 3 criteria probability, severity and potential consequences, are determined to be 53.96%, 16.34% and 29.7% respectively, as discussed earlier. Therefore, in the sensitivity analysis, these 3 priority scales are randomly assigned to the 3 criteria, resulting in another 5 possible relationship between them, in terms of relative importance, and 6 in total different results, in terms of proposed battery capacity and handleable percentage of contingencies, as shown in Figure 3.7 and Figure 3.8. The 6 different priority arrangements are listed in Table 3.3, while the corresponding results shown in the two figures (i.e. Figure 3.7 and 3.8) are quantified in Table 3.4.

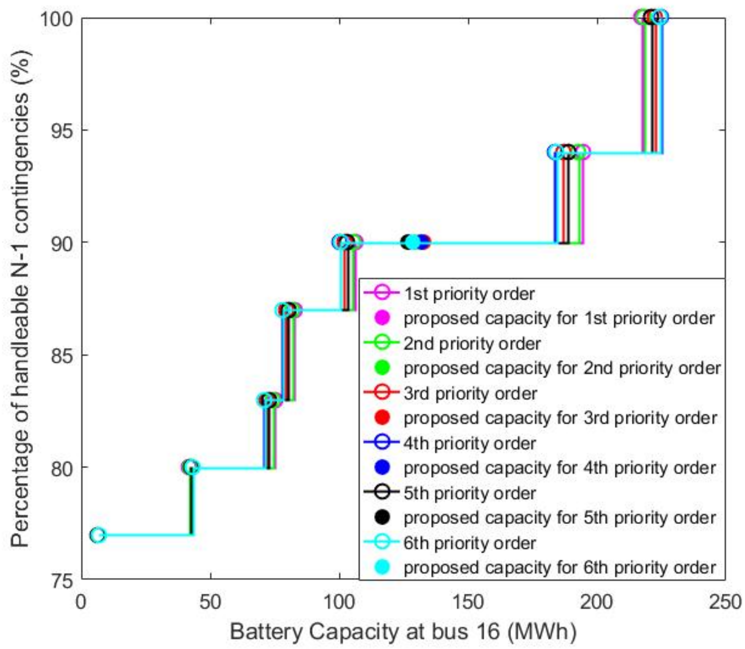


FIGURE 3.7: Percentage of handleable N-1 contingencies Versus Proposed Battery Capacity at Bus 16 using the first approach based on $C = C_c + C_d$

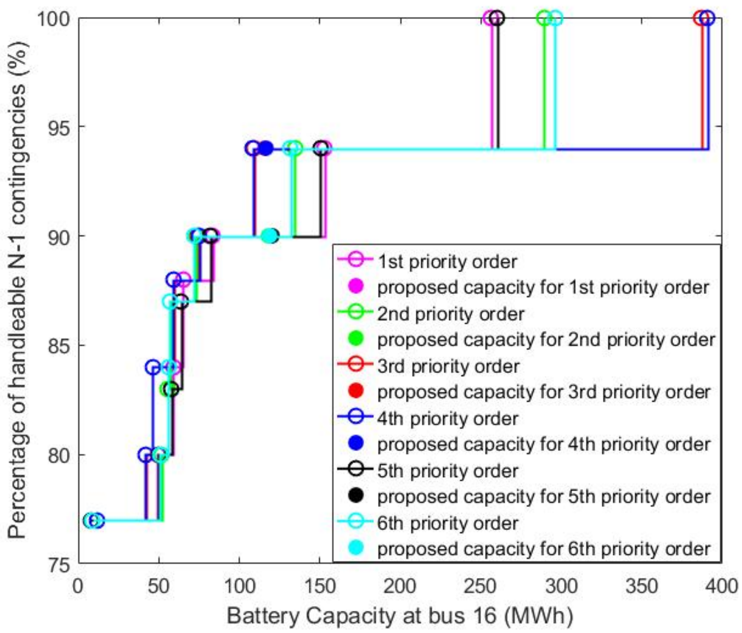


FIGURE 3.8: Percentage of handleable N-1 contingencies Versus Proposed Battery Capacity at Bus 16 using the second approach based on $C = w_c C_c + w_d C_d$

TABLE 3.3: Priority Arrangements for Sensitivity Analysis

No.	**Priority Arrangement**		
	Probability	***Severity***	***Potential Consequence***
1	53.96%	16.34%	29.7%
2	53.96%	29.7%	16.34%
3	29.7%	53.96%	16.34%
4	16.34%	53.96%	29.7%
5	29.7%	16.34%	53.96%
6	16.34%	29.7%	53.96%

TABLE 3.4: Corresponding Results for Different Priority Arrangements

No.	$\mathbf{C = C_c + C_d}$		$\mathbf{C = w_c C_c + w_d C_d}$	
	Proposed Battery Capacity (MWh)	***Percentage of Handleable N-1 Contingencies***	***Proposed Battery Capacity (MWh)***	***Percentage of Handleable N-1 Contingencies***
1	127.03	90%	119.84	90%
2	129.28	90%	118.20	90%
3	132.62	90%	115.92	94%
4	131.82	90%	116.23	94%
5	126.73	90%	120.32	90%
6	128.32	90%	119.07	90%

From Figure 3.7 and Table 3.4, it is clear that in this case, the different priority orders do not have much impact on the final results when the first approach based on $C = C_c + C_d$ is used, as all six polylines mostly overlap with each other and the proposed battery capacities are all around 130 MWh with no difference in the handleable percentage of N-1 contingencies. However, when the relative importance between charging and discharging corrective actions is considered, i.e. the second approach based on $C = w_c C_c + w_d C_d$ is used, the final results are most sensitive to the priority of severity, in this case, while the other two criteria's weightings do not affect the results much, which can be clearly seen in Figure 3.8. Those polylines that share the same priority scales for severity (i.e. No. 1&5, 2&6, 3&4) are almost overlap in Figure 3.8. In Table 3.4, when severity is considered as the most important (i.e. No. 3&4), the battery capacities proposed based on $C = w_c C_c + w_d C_d$ are able to cope with 94% of N-1 contingencies, which again demonstrates that the priority scale of severity can have apparent effects on the final results in this case. Compared to the first approach, the second approach is better at reflecting the real difference between the contingencies that require charging corrective actions and those that require discharging, which is the reason why those two approaches behave differently in this sensitivity analysis.

3.5 Conclusion

Power flow analysis and AHP were used to determine the capacity of the battery providing support during N-1 contingencies to alleviate thermal overload of transmission lines, taking into account the probability of occurrence, severity of thermal overload and potential consequences of the contingencies. The proposed approaches were tested on an IEEE Reliability Test System. The results have demonstrated the feasibility of the proposed approaches. When the relative importance of charging and discharging capacities is taken into account, the proposed battery capacity was found to be smaller than that obtained by simply summing up the charging and discharging capacities, while still capable of handling the same percentage of N-1 contingencies and higher percentage of N-2 contingencies. Furthermore, according to sensitivity analysis, the approach with weightings of charging and discharging corrective actions taken into account, is better, compared to the one without, at reflecting (i.e. more sensitive to) the real difference between the contingencies that require charging corrective actions and those that require discharging. The proposed approaches were found to be better, in terms of system security and the percentage of handleable contingencies, than the method proposed in [22] based on sizing the battery to cope with the severest condition.

In this work, the value of the AHP is in providing a rational informed basis for determining the weightings/priorities of different criteria, instead of assigning the weightings based on gut feeling without looking further into the relative importance between different criteria. There are many other approaches to determine the criteria weightings, as discussed in [131], and to size the battery as discussed in [6]. In comparison, the AHP is simpler to implement and more accommodating of uncertainty.

Flow batteries, which have been built in MW sizes, can be a good option for power system operational support such as overload alleviation. The capital cost of a typical flow battery (e.g. ZnBr and VRB) is $150-1000 per kWh. Thus, for a large-scale flow battery system of say about 100 MWh capacity as proposed in this work, the capital cost of this battery system is about $15M, at least, and it can last about 10-20 years with more than 10000 cycles. The cost of using battery storage to provide contingency reserve also includes charging cost and energy loss [132], which is estimated by the product of required regulation energy, hourly energy price, regulation energy use ratio (assumed to be 25%) and an efficiency loss rate of 20% (i.e., 5 MWh energy is consumed when providing 100 MW of regulation for each hour). The charging cost and energy loss cost are estimated to be around $7k per year. Hence, the total cost of using battery for contingency reserve is about $1M per year. If battery storage is not used, generators are utilized instead to provide spinning contingency reserves. The contingency reserve requirement is assumed to be about 4.5% of average load and constant for all hours of the year [132]. Hence, the resulting reserve cost including the opportunity cost and the additional operation cost of generators is estimated, using the data provided in [133], to be about $12M per year. Therefore, utilization of battery storage with the proposed capacity saves about $11M per year. Another alternative would be to upgrade the transmission system, which will cost about $50M for a regional scale network, as estimated in the WPS and Presidio projects [127]. Furthermore, the system upgrades can take several years to complete compared to the rapid installation of a battery.

Compared with a dump load or generation curtailment, it is expensive to use battery storage to absorb energy for overload relief only. Thus, in order to make the investment more worthwhile, the installed battery storage can also be utilized to provide other ancillary services for the power system, such as load levelling, operating reserve, interruption backup etc. as discussed earlier and in Chapter 1. But as discussed earlier, these ancillary services should not be at the expense of the overload relief service. A balance needs to be achieved between the different services so that the scheme remains effective and financially sound.

Furthermore, if ESSs are expected to support not only contingency thermal overload alleviation but also other power grid operations effectively and efficiently, more ESSs are needed to be distributed in the network with a proper dispatch strategy to achieve the desired system support. Next chapter will investigate the application and dispatch of EV batteries in the power system operational support.

Chapter 4

Dispatch of Vehicle-to-Grid Battery Storage Using an Analytic Hierarchy Process

The preceding chapter demonstrated the application of the stationary battery in thermal overload alleviation. However, many more batteries are required to be distributed in the system for the better power grid operational support including not only overload relief. The number of EVs is expected to increase significantly in the future to combat air pollution and reduce reliance on fossil fuels. This will impact the power system. However, with appropriate charging and discharging through vehicle-to-grid (V2G) operation, EV batteries could replace some stationary batteries to provide support for the power system and benefit the EV owners. This raises the questions of when and how EV battery storage should be dispatched, taking into account both vehicle users' and power system requirements and priorities, as well as the constraints of the battery system. This chapter proposes a novel decentralized dispatch strategy based on the Analytic Hierarchy Process (AHP) taking into account the relative importance of the different criteria such as cost, battery state of charge (SOC), power system contingency and load levelling. The proposed AHP-based dispatch strategy was tested on an IEEE Reliability Test System with different EV numbers and capacities to investigate the efficacy of such an approach. The simulation results demonstrate the feasibility and benefits of this dispatch strategy.

4.1 Introduction

Growing concerns over energy savings, emissions and the desire to reduce reliance on fossil fuels have resulted in ambitious plans for expanding the use of electric vehicles (EVs) [134, 135]. The large-scale penetration of EVs will challenge the power system as an additional load due to the substantial need for battery charging. However, due to the significant storage/generation capacity that EV batteries can offer to the grid if V2G is enabled, EVs, with an appropriate dispatch strategy, are capable of providing the grid operational support and the improvement of power system performance such as load levelling, spinning reserves and regulation [60, 61].

In this chapter, a novel decentralized dispatch strategy for EV batteries is developed based on AHP, taking into account the concerns and requirements of both EV users and the grid, including cost, sufficient SOC for the next journey, load levelling and alleviation of the thermal overload caused by N-1 contingency (i.e. only one component of transmission system is in outage). Using the proposed strategy, each EV determines the dispatch action by itself depending on its own situation and the information gathered from the power grid. The proposed dispatch strategy is then tested on an IEEE Reliability Test System for its effectiveness in satisfying the requirements of both EV users and grid. Comparisons with the rule-based dispatch strategy [20] are made to demonstrate the potential benefits of the proposed AHP-based dispatch strategy over previous approaches.

The rest of this chapter is organized as follows. A description of typical EV battery characteristics is presented in Section 4.2. In Section 4.3, the proposed dispatch strategy of EV battery storage is presented in detail. The results of simulations using MATLAB to verify its feasibility and efficacy, are presented and discussed in Section 4.4. This is followed by a presentation and discussion of the results of parameter sensitivity analysis in Section 4.5. Finally, the conclusions of the work are presented in Section 4.6.

This work has been accepted for publication in *IEEE Transactions on Vehicular Technology* in 2016.

4.2 Battery Characteristics

The capacity of a battery varies with the discharging current. This can be modelled based on the Peukert equation [136], [137], as follows:

$$C_p = I^k T \tag{4.1}$$

where C_p is the Peukert Capacity, k is the Peukert Coefficient (e.g. $k = 1.2$ for a lead acid battery), I is the constant discharging current in Amperes and T is the corresponding time the battery will last in hours. Furthermore, C_p can be calculated by substituting I and T with the nominal discharging current I_n and rated discharge time T_r, as follows:

$$C_p = I_n^k T_r \tag{4.2}$$

Therefore, given a constant discharging current i_d in Amperes, the corresponding available capacity C_a of a battery in Ampere-hours (Ah) can be obtained as:

$$C_a = \frac{i_d C_p}{i_d^k} = \frac{i_d I_n^k T_r}{i_d^k} \tag{4.3}$$

The state of charge (SOC) of a battery after being discharged for t hours at a constant discharging current i_d is calculated by (4.4).

$$SOC(t) = 1 - \frac{i_d t}{C_a} \tag{4.4}$$

As for the voltage at each time step when calculating charge/discharge power of a battery, the method proposed in [20] based on the simplified generic rechargeable battery model described in [138] is used due to the limited information on the detailed charge/discharge characteristics of actual EV batteries. A lookup table of the voltages of a typical battery (240V, 100Ah) corresponding to the different pairs of released capacities and charge/discharge currents is built and presented in Table 4.1. Therefore, given current SOC and charge/discharge current, the corresponding voltage can be obtained by linear interpolation of the data in Table 4.1. The main parameters of a typical battery that is used as an EV battery here, are calculated by (4.2) and (4.3) and demonstrated in Table 4.2.

TABLE 4.1: Battery voltage Versus released capacity at different charging/discharging current (240V, 100Ah)

Released Capacity (Ah)	Voltage(V)/Current(A)				
0	259.4/2	...	258.9/10	...	257.7/30
...	...	...	...	...	...
20	244.6/2	...	244.2/10	...	243/30
...	...	...	...	...	...
60	237.5/2	...	237/10	...	235.8/30
...	...	...	...	...	...
80	225.4/2	...	224.9/10	...	223.7/30
...	...	...	...	...	...
100	123.2/2	...	122.8/10	...	121.6/30

4.3 Dispatch Strategy of Electric Vehicle Battery Storage

As discussed above, by running the proposed V2G dispatch strategy for each EV, its dispatch action (i.e. whether charge or discharge and at what dispatch current) is determined by the power system operator (PSO) at the beginning of each time interval. In preparation for that, the PSO gathers real-time data, such as real-time pricing data, network fixed load demand (i.e. load without EVs) and overload information, a couple

TABLE 4.2: Characteristics of the typical EV battery (240V, 100Ah)

Rated Capacity (Ah)	Nominal Current (A)	Nominal Voltage (V)
100	20	240
Peukert Capacity (Ah)	Charge/ Discharge Current (A)	Effective Available Capacity (Ah)
182.06	2	158.5
	10	114.87
	30	92.21

of minutes prior to the beginning of each time interval. In this work, the AHP-based dispatch strategy is designed for V2G batteries to support power grid operations including load levelling and the alleviation of thermal overload caused by N-1 contingency, and to reduce cost to users while ensuring that the battery SOC is sufficient for the next journey. In order to jointly consider the different requirements of EV users and the grid, they should be weighed depending on their relative importance in determining how and when EV batteries should be dispatched and thus a hierarchy model is developed first.

4.3.1 AHP Hierarchy Model

A five-level hierarchy model for the dispatch of an EV battery is as shown in Figure 4.1.

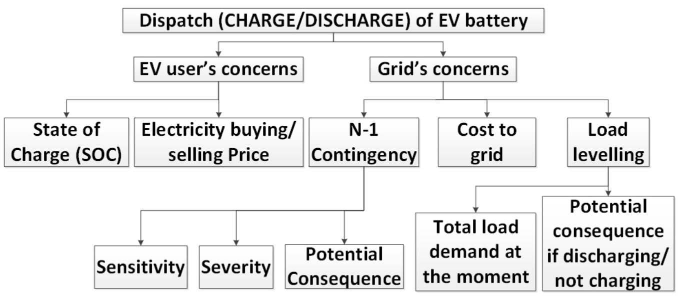


FIGURE 4.1: AHP Hierarchy Model for Dispatch of EV Battery

The top level is the objective: dispatch of the EV battery, which should take into account the requirements and concerns of both EV users and the grid (Second level criteria) as discussed earlier. EV users' main concerns are the SOC of the EV batteries and the cost to dispatch them (Third level sub-criteria). The grid might utilize the EVs to support system load levelling (i.e. valley filling and peak shaving) or to relieve thermal overload after an N-1 contingency while allowing for the costs of employing the storage/generation capacity of EV batteries (Third level sub-criteria). Several factors need to be considered regarding the support that an EV can provide in an N-1 contingency such as the sensitivity of the bus to which the EV is connected in alleviating thermal overload of the overloaded lines, the severity of the overload and the potential consequence if the contingency is not dealt with in 15 minutes (Fourth level sub-criteria). Furthermore, both current system demand and potential consequence of discharging/not charging the EV battery (Fourth level sub-criteria) should be taken into account when determining the preferred support that an EV can provide for load levelling. Finally, the battery can be either charged or discharged (alternatives of dispatch actions) as a result of this decision making process.

It is important to note, as mentioned earlier, that the dispatch system is owned by the PSO, not the EV owner. Aggregators may adjust the priorities to maximize their profit, which may change the weighting towards the grid or vice versa. Here, the concerns of EV users and the grid are considered as equally important and thus both weighed 50%. SOC and electricity buying/selling price are also weighed equally with 50% each. Among the three sub-criteria under grid's concerns, the cost to the grid is considered to be more important than the other two (i.e. twice as important as the other two), because the power grid can choose other kinds of energy storage for grid operational support if it is more expensive to use EV batteries. The overload alleviation under N-1 contingency and load levelling are two different kinds of grid support and thus are deemed to be equally important. A pairwise comparison matrix A_g is thus constructed as follows to derive the weightings of the three sub-criteria (N-1 contingency (NC), cost to grid (CG) and load levelling (LL)) under grid's concerns:

$$A_g = \begin{array}{c} \\ NC \\ CG \\ LL \end{array} \begin{array}{c} \begin{array}{ccc} NC & CG & LL \end{array} \\ \begin{pmatrix} 1 & \frac{1}{2} & 1 \\ 2 & 1 & 2 \\ 1 & \frac{1}{2} & 1 \end{pmatrix} \end{array} \tag{4.5}$$

By calculating the principle eigenvector of A_g, the weightings of NC, CG and LL, namely w_{NC}, w_{CG} and w_{LL}, are 25%, 50% and 25%, respectively. Among the three sub-criteria under N-1 contingency, sensitivity is considered to be the most important, because there is no need for the EV battery to deal with the contingency that occurs far from the bus to which it is connected, no matter how severe the overload and the potential consequence are. Potential consequence of the N-1 contingency is considered to be a bit more important than severity, because a contingency that does not cause severe overload but has a severe potential consequence is surely more important and needs more attention than one that causes severe overload but no severe potential consequence. Therefore, sensitivity is determined to be 4 times as important as severity and 3 times as important as potential consequence, with potential consequence twice as important as severity. The pairwise comparison matrix A_{NC} for weighing the three sub-criteria (Sensitivity (Sen), Severity (Sev) and Potential consequence (PoC)) under N-1 contingency is thus:

$$A_{NC} = \begin{array}{c} \\ Sen \\ Sev \\ PoC \end{array} \begin{array}{c} \begin{array}{ccc} Sen & Sev & PoC \end{array} \\ \begin{pmatrix} 1 & 4 & 3 \\ \frac{1}{4} & 1 & \frac{1}{2} \\ \frac{1}{3} & 2 & 1 \end{pmatrix} \end{array} \tag{4.6}$$

Calculating the principal eigenvector of A_{NC} gives the weightings of Sen, Sev and PoC, namely w_{Sen}, w_{Sev} and w_{PoC}, which are 62.5%, 13.7% and 23.9%, respectively. Moreover, the two sub-criteria under load levelling, viz. total load demand and potential consequence, are weighed 60% and 40%, respectively with total load demand considered more important than potential consequence.

The factors that can be included in the AHP hierarchy model are not limited to those in Figure 4.1. Other factors such as battery degradation, voltage constraints and integration of renewables can be readily included. For example, voltage limits, related to the distribution system for example, can be included by adding it as another grid concern in Figure 1. Indeed, additional sub-criteria specific to the distribution system can be added. But increasing the number of factors would increase complexity and implementation cost. Additionally, some factors may encompass others indirectly. For example, what an EV owner cares about most are the vehicle's basic function (i.e. driving) and the direct cost of utilizing this function (i.e. charging cost). But she or he implicitly understands (or should be told by the PSO as part of their duty of care) that participating in a V2G will accelerate battery degradation. If he or she decides to participate in the V2G scheme, then he or she would need to be convinced that the savings of charging cost, or even better getting some payment, is an adequate compensation for battery accelerated degradation. A PSO may also offer some incentives such as paying a battery degradation charge to convince the driver that participating in V2G is worth doing. In other words, the cost model includes some elements related to battery degradation. Similarly, the criteria which set a minimum and a maximum state of charge indirectly reduces the rate of battery degradation; battery life is known to improve if the depth of discharge is reduced [139], [140], for example it is well known that the cycle life of some lithium-ion batteries improves from 2000 deep cycles to more than 15,000 shallow cycles (from 30%–70% state of charge). The driver can have the option to limit the depth of discharge (by setting the minimum and maximum SOC), to reduce battery degradation.

4.3.2 Determination of the Dispatch Action

The priorities of alternatives are determined by evaluating how suitable it is to charge/discharge the EV battery with respect to every sub-criterion as follows:

State of Charge (SOC): The upper limit of SOC S_{max} is selected to avoid overcharge, while the lower limit S_{min} is set to ensure that sufficient electric energy remains within the battery for the next journey and for the protection of the health of the battery. In

V2G mode, S_{max} is set to be 0.8 and $S_{min} = 0.4 + S_n$ where S_n represents the per-unit capacity that is going to be consumed on the road to the next destination. It is clear that the lower the SOC the more favourable it is to charge the battery. The higher the SOC the more favourable it is to discharge it. Therefore, the priorities of charging and discharging with respect to SOC are defined as follows:

Charge:

$$P_{SOC}^{c} = \begin{cases} 1 & SOC \leqslant S_{min} \\ 1 - \frac{SOC - S_{min}}{S_{max} - S_{min}} & SOC > S_{min} \end{cases} \tag{4.7}$$

Discharge:

$$P_{SOC}^{d} = \begin{cases} 1 & SOC \geqslant S_{max} \\ 1 - \frac{S_{max} - SOC}{S_{max} - S_{min}} & SOC < S_{max} \end{cases} \tag{4.8}$$

Electricity Buying/Selling Price: In order to reduce cost, it is better to charge the battery when the buying price is low and discharge it when the selling price is high. Therefore, the priorities of charging and discharging in terms of electricity price are determined as follows:

Charge:

$$P_{EP}^{c} = \begin{cases} 1 & bp \leqslant hbp \\ 1 - \frac{bp - hbp}{0.1 \times mbp} & bp > hbp \end{cases} \tag{4.9}$$

Discharge:

$$P_{EP}^{d} = \begin{cases} 1 & sp \geqslant hsp \\ 1 - \frac{hsp - sp}{0.6 \times msp} & sp < hsp \end{cases} \tag{4.10}$$

where bp and sp are the electricity buying and selling prices determined by the real-time pricing information. hbp and hsp are the high buying and selling prices, respectively. hbp is defined as 90% of the maximum buying price mbp, while hsp is defined as 60% of the maximum selling price msp [20]. These four price values (hbp, hsp, mbp and msp) can be determined by using the day-ahead pricing information. In this work, real pricing data recorded in [141] are used to define these four values.

N-1 Contingency: The priority of charging is zero with respect to an N-1 contingency that requires discharging corrective action at the bus to which the EV is connected. Similarly, the priority of discharging is zero if the contingency requires a charging corrective action. The priorities of these actions are determined as follows:

Sensitivity: The sensitivity of load flow through a branch b overloaded by an N-1 contingency C to changes of power injection at a particular bus j can be evaluated using the sensitivity factor defined in [22]:

$$S_C^j = \frac{\Delta F_b}{\Delta P_j} \tag{4.11}$$

where ΔF_b is the change of power flow through the branch b resulting from the change in power injection at bus j (ΔP_j). As certain buses have relatively higher sensitivity factors with respect to the overloaded branch than others, the same power injections at these buses could make a more obvious difference to the alleviation of the overload. Thus, an EV connected to a bus with a high sensitivity factor will have a high priority of dispatch during an N-1 contingency, while the dispatch of an EV connected to a bus with a very low sensitivity factor is not recommended due to the little contribution it can make. By comparing the sensitivity of a bus to the bus with the highest sensitivity factor, the priority of charging/discharging an EV at a specific bus with respect to sensitivity P_{sen} can be defined as:

$$P_{sen} = \frac{\left|S_C^j\right|}{max\left\{\left|S_C^1\right|, \left|S_C^2\right| \ldots \left|S_C^{nb}\right|\right\}} \tag{4.12}$$

where nb is the total number of buses in the system.

Severity: The severity of overload can be quantified as a percentage of the Long Term Emergency (LTE) rating of the overloaded line, as follows:

$$Severity = \frac{Load\ Flow - LTE}{LTE} \times 100\%, \tag{4.13}$$

where LTE rating can be derived from [128], while $Load\ Flow$ is obtained by carrying out the power flow analysis for the network under an N-1 contingency. The priority of charging/discharging with respect to overloading severity P_{sev} is determined as:

$$P_{sev} = \frac{Sev_C}{Sev_{max}} \tag{4.14}$$

where Sev_C is the severity of overloading caused by contingency C and Sev_{max} is the severity of the severest overload caused by the severest contingency, which is derived by running the simulation of power flow analysis for all possible N-1 contingencies within the network, calculating the severity of overload they caused using (4.13) and then selecting the maximum.

Potential Consequence: Potential consequence of an N-1 contingency is determined by the number of overloaded lines, assuming that the branch overloaded by the contingency is broken due to the absence of an in-time measure. The priority of charging/discharging in terms of potential consequence P_{PoC} is calculated as follows:

$$P_{PoC} = \frac{Poc_C}{Poc_{max}} \tag{4.15}$$

where Poc_C is the potential consequence of contingency C and Poc_{max} is the severest potential consequence.

Cost to Grid: When charging the EV battery, the cost to the grid is C_c. However, when discharging it, the cost is C_d. From the grid's perspective, the lower the cost the better. Therefore, the priorities of charging and discharging in terms of the cost to grid are evaluated as follows:

Charge:

$$P_{CG}^{c} = \begin{cases} 1 & C_c \leqslant HCC \\ 0 & C_c > HCC \end{cases} \tag{4.16}$$

Discharge:

$$P_{CG}^{d} = \begin{cases} 1 & C_d \leqslant LDC \\ \frac{HDC - C_d}{HDC - LDC} & LDC < C_d < HDC \\ 0 & C_d \geqslant HDC \end{cases} \tag{4.17}$$

where HCC is the high cost to the grid for charging an EV and is set to be zero in this work because the cost to the grid to charge an EV is negative and the cost to discharge an EV is positive in this work. HDC and LDC are respectively the high and low costs to grid for discharging an EV and are set to be £0.02/KWh and £0.01/KWh respectively in this chapter [141].

Load Levelling:

Total Load Demand: In order for load levelling (i.e. peak shaving and valley filling) to be effective, it is better for EVs to be charged when the network's original load demand (i.e. the system load without EV integration) is low, so that EVs can store energy for the driving activities and grid operational support that might happen afterwards, while discharged to provide energy to the grid when the network's original load demand is high. Therefore, the priorities of charging and discharging in this case are defined as:

Charge:

$$P_{LD}^{c} = \begin{cases} 1 & d \leqslant LD \\ 1 - \frac{d - LD}{0.25(d_{max} - d_{min})} & LD < d < MD \\ 0 & d \geqslant MD \end{cases} \tag{4.18}$$

Discharge:

$$P_{LD}^{d} = \begin{cases} 1 & d \geqslant HD \\ 1 - \frac{HD - d}{0.25(d_{max} - d_{min})} & MD < d < HD \\ 0 & d \leqslant MD \end{cases} \tag{4.19}$$

where d_{max} and d_{min} are the maximum and minimum system demand during a day which can be determined using day-ahead load forecasting data or the historical data. In this work, real load demand data [141] are used to define these two values. MD is the mid-level system demand, calculated as

$$MD = 0.5 \times (d_{max} + d_{min}) \tag{4.20}$$

LD is the low-level system demand, determined by

$$LD = d_{min} + 0.25 \times (d_{max} - d_{min}) \tag{4.21}$$

HD is the high-level system demand, determined by

$$HD = d_{max} - 0.25 \times (d_{max} - d_{min}) \tag{4.22}$$

Potential Consequence: The potential consequence if discharging/not charging the battery is evaluated based on its SOC. If the EV battery is discharged/not charged at a given time, there might not be enough electric energy within the battery to be discharged during high load periods. Therefore, the priorities of charging and discharging are determined as follows:

Charge:

$$P^c_{LPoC} = \begin{cases} 1 & SOC \leqslant Sg_{min} \\ 1 - \frac{SOC - Sg_{min}}{Sg_{max} - Sg_{min}} & SOC > Sg_{min} \end{cases} \tag{4.23}$$

Discharge:

$$P^d_{LPoC} = \begin{cases} 1 & SOC \geqslant Sg_{max} \\ 1 - \frac{Sg_{max} - SOC}{Sg_{max} - Sg_{min}} & SOC < Sg_{max} \end{cases} \tag{4.24}$$

where Sg_{min} and Sg_{max} are low and high SOC from the perspective of grid and selected to be 0.4 and 0.8, respectively, for the normal operation of EV batteries.

The final priorities P_f of charging and discharging an EV are calculated in the same way, that is:

$$\begin{aligned} P_f = &(P_{SOC} \times w_{SOC} + P_{EP} \times w_{EP}) \times w_{EV} \\ &+ ((P_{sen} \times w_{sen} + P_{sev} \times w_{sev} + P_{PoC} \\ &\times w_{PoC}) \times w_{NC} + P_{CG} \times w_{CG} + (P_{LD} \\ &\times w_{LD} + P_{LPoC} \times w_{LPoC}) \times w_{LL}) \times w_G \end{aligned} \tag{4.25}$$

where P_{SOC}, P_{EP}, P_{sen}, P_{sev}, P_{PoC}, P_{CG}, P_{LD}, P_{LPoC} are the priorities of charge/discharge with respect to SOC, electricity price, sensitivity, severity, potential consequence of N-1 contingency, cost to grid, total load demand and potential consequence of load levelling, respectively. w_{SOC}, w_{EP}, w_{sen}, w_{sev}, w_{PoC}, w_{CG}, w_{LD}, w_{LPoC} are the corresponding weighting factors. w_{NC} and w_{LL} are the weightings of N-1 contingency and load levelling with respect to grid's concerns. w_{EV} and w_G are respectively the weightings of EV users' and grid's concerns in terms of the objective.

The above forms the main part of the decision making process for the dispatch of an EV's battery. It is important to note that the information about the next journey S_n is assumed to be available. The SOC of an EV battery is checked at the beginning of each time interval. If the SOC is less than S_{min}, the battery is charged during the current time interval. Otherwise, if the SOC is larger than S_{max}, it is discharged. The dispatch action is determined as the one that has the highest final priority if the SOC is between S_{min} and S_{max}. How fast the battery is charged/discharged depends on the final priority P_f of charge/discharge. If P_f is no less than 0.9, the EV is charged/discharged at the high current level set here to be 30A. If P_f is between 0.7 and 0.9, the EV is charged/discharged at the middle-level current of 10A. If P_f is between 0.4 and 0.7, the EV is charged/discharged at a low-level current of 2A. Otherwise, when P_f is lower than 0.4, the EV battery is idle during this time interval. The overall AHP-based dispatch strategy is shown in Figure 4.2, which is implemented at the beginning of each time interval for each EV by the PSO using the real-time data gathered a couple of minutes beforehand. The detailed data communication chart is illustrated in Figure 4.3, which describes the procedure of real-time data collection from EVs and system operators, the working process of the PSO and the transmission of dispatch action commands to EVs. The PSO collects data from the EVs, DSO and TSO and sends dispatch commands to the EVs. For comparison, the flow chart of the rule-based dispatch strategy described in [20] is given in Figure 2.4. The key parameters of the two dispatch strategies are set to be the same for the fair comparison, as shown in Table 4.3.

TABLE 4.3: Key Parameter Settings for AHP-based and Rule-based Dispatch strategies

Dispatch Strategy	SOC_{min}	SOC_{max}	High Buying Price *hbp* (£/KWh)	High Selling Price *hsp* (£/KWh)
AHP-based	0.4	0.8	0.15	0.11
Rule-based	0.4	0.8	0.15	0.11
Dispatch Strategy	**High Dispatch Current**	**Medium Dispatch Current**	**Low Dispatch Current**	**On-road Discharging Current**
AHP-based	30	10	2	20
Rule-based	30	10	2	20

4.4 Simulation Test

The proposed AHP-based dispatch strategy was tested on an IEEE Reliability Test System (RTS) [128], which is formed of 24 buses and 38 branches as shown in Figure

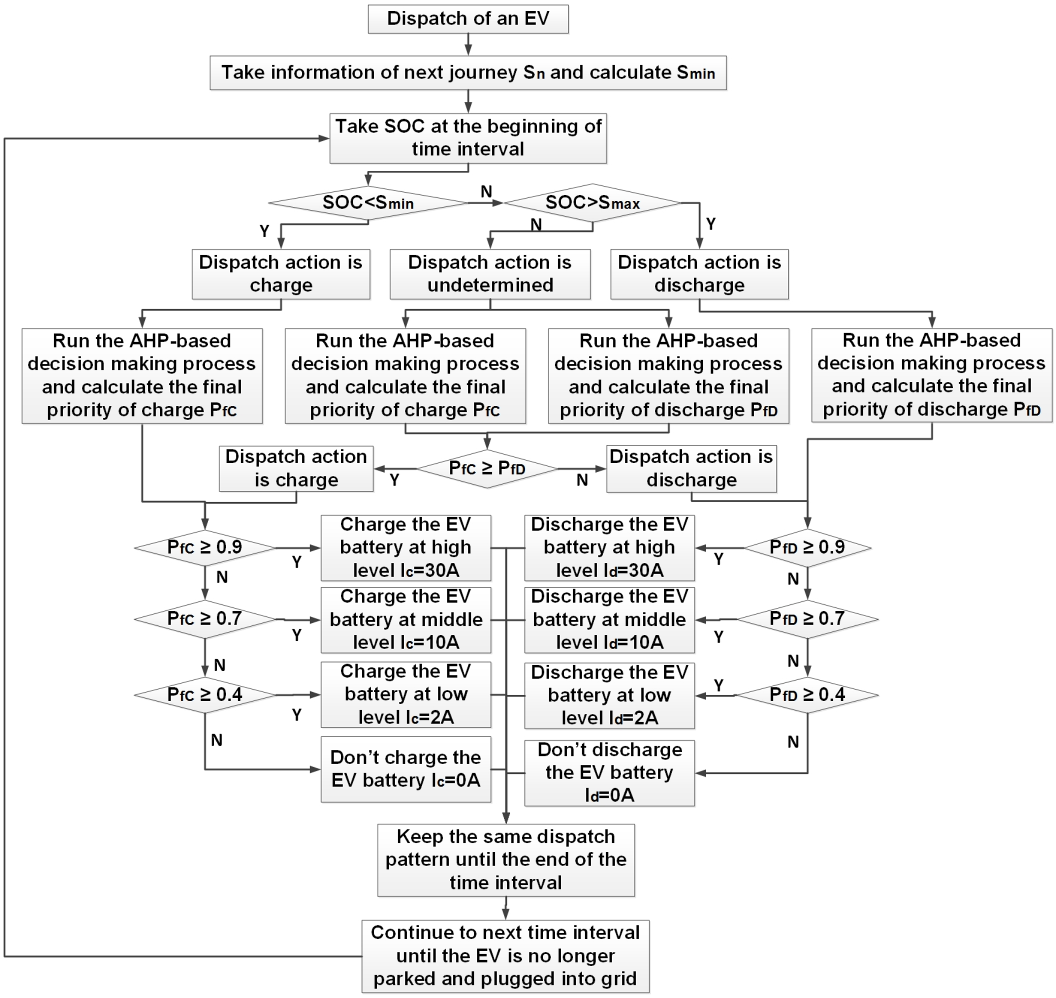


FIGURE 4.2: AHP-based Dispatch Strategy for an EV Battery

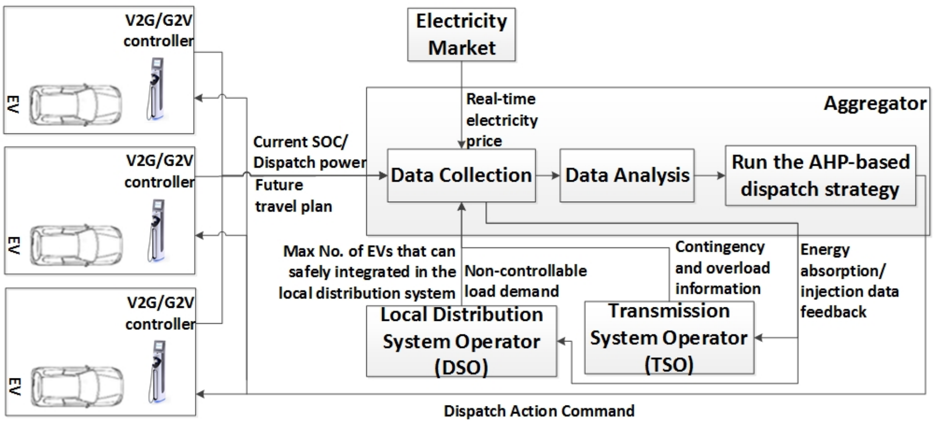


FIGURE 4.3: Data Communication Chart

4.4, and implemented using MATLAB and MATPOWER 4.1 [130]. The reason why RTS is used for simulation test is because this chapter mainly focuses on the application of V2G in the transmission system in terms of load levelling and overload alleviation under N-1 contingency. The EVs that are distributed within the local distribution network are aggregated and connected at a bus of transmission system that this local network is connected to, under the assumption that the integration of EVs will not overload the local distribution network.

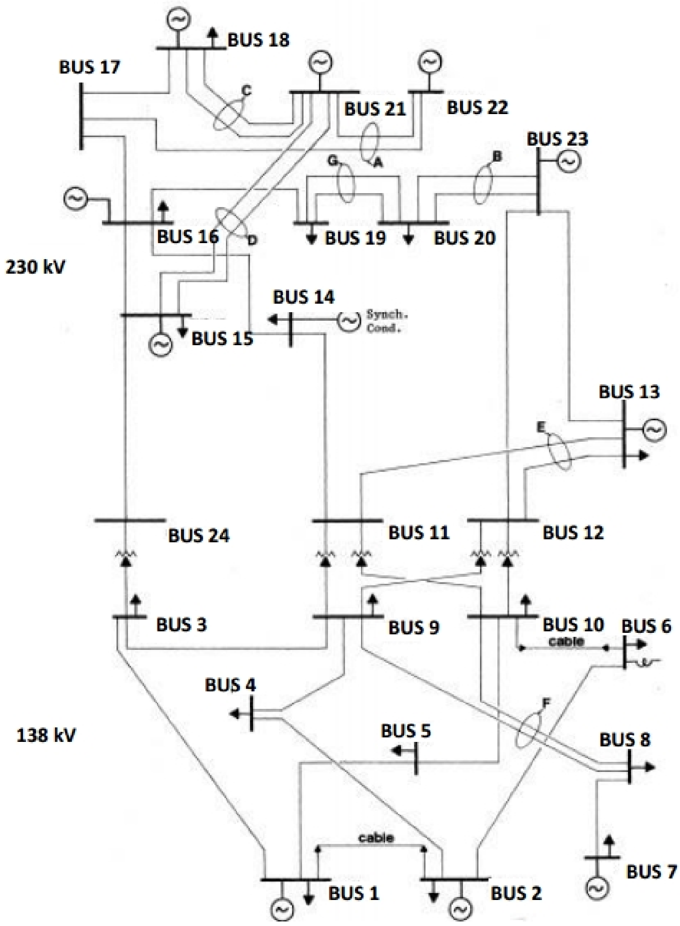


FIGURE 4.4: Diagram of IEEE Reliability Test System

The total load demand and system selling/buying price of electricity during a day was taken from [141]. The total load demand is scaled down so that the peak demand during a day is 3300 MW, which approximately represents the daily load demand in a large UK city, as shown in Figure 4.5. Due to a scarcity of information on the payments by/to

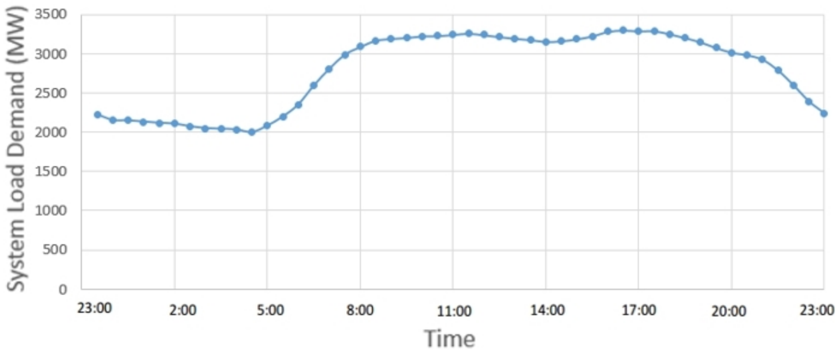


FIGURE 4.5: Total Load Demand During A Day without EV Load

EV users when the EVs are charged/discharged, the system selling/buying price [141], is adjusted based on domestic tariffs using the method proposed in [20] to represent EV users' selling/buying price of electricity. The adjusted system selling/buying price is derived by adding the difference between the average tariff value and the average system selling/buying price to the corresponding system selling/buying price, as shown in Figure 4.6:

$$ASP(t) = SP(t) + \left(\frac{\sum_{t=1}^{T} tariff(t)}{T} - \frac{\sum_{t=1}^{T} SP(t)}{T} \right), \tag{4.26}$$

where a day is divided into T time intervals. $ASP(t)$ and $SP(t)$ are the adjusted system selling/buying price and system selling/buying price at time interval t, respectively, while $tariff(t)$ represents the domestic tariff value at time interval t. The average values of ASSP and ASBP become the same and equal to the average of domestic tariff, therefore ASBP can be either higher or lower than ASSP. Therefore, the ASSP (i.e. adjusted system selling price) and ASBP (i.e. adjusted system buying price) in Figure 4.6 are used as the EV charging and discharging prices respectively, namely the *bp* in (4.9) and *sp* in (4.10) respectively. C_c in (4.16) is defined as the difference between system buying price and system selling price in Figure 4.6, namely $C_c = SSP - SBP$. C_d in (4.17) is set to be the difference between EV discharging price and charging price, that is, $C_d = ASBP - ASSP$. The time step is set to be 30 minutes. The dispatch actions are determined by the dispatch strategy at the beginning of every time interval and lasts for the entire time interval of 30 minutes.

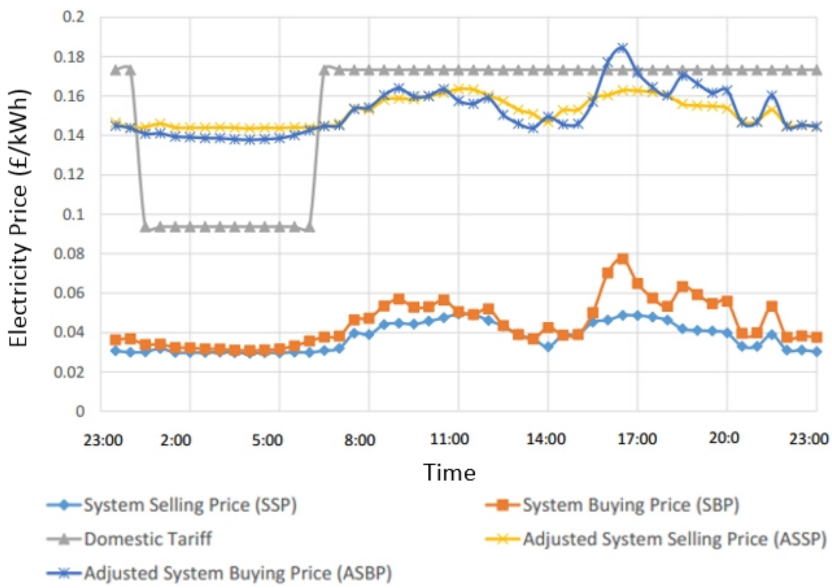


Figure 4.6: Daily Electricity Price

To test the AHP-based dispatch strategy, an EV is connected to bus 6 when it is parked and plugged into the grid. When the EV is on the road, the battery electric energy is assumed to be consumed at the nominal discharging current (20A) as assumed in [20]. The EV's travel pattern and electric capacity requirements for the next journey, S_n, during a day are described in Table 4.4. It is important to note that the travel pattern used here was randomly generated and utilized as an example. The EV can have any kind of travel pattern, which will not affect the feasibility of the proposed dispatch strategy. The simulation results of the EV's SOC and net cost to EV users (i.e. charging cost–discharging payment) during a day are shown in Table 4.5, compared with those generated by applying the rule-based dispatch strategy that has been developed in [20].

TABLE 4.4: EV's Travel Pattern and S_n during a day

Time	Travel Pattern	S_n
23:00–7:00	Parked	0.2
7:00–8:00	on road	n/a
8:00–12:00	Parked	0.1
12:00–12:30	on road	n/a
12:30–13:30	Parked	0.1
13:30–14:00	on road	n/a
14:00–18:00	Parked	0.2
18:00–19:00	on road	n/a
19:00–23:00	Parked	0.2

TABLE 4.5: SOC and daily cost of an EV – Comparisons between the proposed AHP-based dispatch strategy and the rule-based strategy

Dispatch Strategy	Lowest SOC	Highest SOC	Costs (£)
AHP-based	0.331	0.694	2.25
Rule-based	0.194	0.657	1.39

In Table 4.5, it is shown that under the same travel pattern the AHP-based dispatch strategy results in better SOC conditions of the EV though costing £0.86 more per day than the rule-based strategy. The improved SOCs produced by the AHP-based dispatch strategy remain in the main within the desired range [0.4, 0.8]. The lowest SOC is just below 0.4, which implies a more reliable driving experience, compared to the lowest SOC of 0.194 when using rule-based strategy. This could justify the higher daily cost of the AHP-based strategy. Furthermore, if the EV battery, without V2G service, is quickly fully charged after every use (i.e. using 30A charging current) to recover the electricity consumed on the road, the daily cost per vehicle will be £3.08 under the same travel pattern as shown in Table 4.4, which demonstrates the capability of AHP-based dispatch strategy to save costs while ensuring sufficient SOC in the battery for reliable driving.

As for the effectiveness of the AHP-based dispatch strategy in dealing with an N-1 contingency, the EV battery dispatch action can be affected by the occurrence of a severe N-1 contingency and the overloading it causes can be efficiently relieved by power injection/absorption at the bus to which the EV is connected. The simulation results using the AHP-based dispatch strategy show that a car's battery is charged at 479 W from 9:00 am to 9:30 am if no contingency occurred. However, if branch 10 (between bus 6 and 10) is broken (N-1 contingency) at 9 am, which results in the overloading of branch 5 (between bus 2 and 6), power injection will be required at bus 6 meaning that the connected EV battery is preferably discharged. By applying the AHP-based dispatch strategy, the EV is set to be discharged at 479 W. The rule-based dispatch strategy that was proposed in [20] does not take N-1 contingency into account and thus makes the same decision on the dispatch action of the EV battery regardless of the contingency — using this strategy an EV battery will in fact be charged at 468 W at 9:00 am, which will aggravate the overloading caused by the outage of branch 10.

The highest charging/discharging power of an EV is 7731 W, which is negligible compared to the total load demand of the system. Therefore, in order to test the efficacy of the proposed AHP-based dispatch strategy on load levelling, a total number of 625000 EVs are assumed to be in the system. This is a predicted number of EVs in a large UK city after 2035 (EVs are predicted to comprise 25% of the total car population after 2035, and there are about 2.5 million cars in a large UK city). These EVs are randomly allocated in the system while power flow analysis is used to make sure that no overload occurs. For the simplicity of simulation, EVs are grouped assuming that EVs within a group have similar SOCs, thus one SOC value is assigned to a group of EVs. Therefore, the initial SOCs of EVs are randomly assigned to the EV groups with a normal probability distribution ($\mu = 0.6, \sigma = 0.1$). A probability curve, illustrated in Figure 4.7 [20], is used to determine the number of parked EVs at each time interval by calculating the product of probability of parked cars at that moment and the total number of EVs available in the power network. When the probability of parked EVs decreases, a corresponding number of EVs are randomly selected to leave the parking lot. When the probability of parked EVs increases, an associated number of EVs arrive at the parking lot with SOCs randomly assigned at the arrival with a normal probability distribution of $\mu = 0.5, \sigma = 0.1$ (assuming that the average driving journey is about 30 minutes at a constant discharging current of 20A on road). The newly arrived EVs are then assigned to the EV groups having similar SOCs (the difference is less than 0.05) with them or constitute new EV groups if their SOCs are really different (the difference is larger than 0.05) from the existing EV groups.

In order to measure the total load demand and system transmission loss, the power injected at the slack bus of the test system (bus 13) is evaluated for all the cases including the cases where EVs are charged in an uncontrolled fashion (i.e. EVs are fast charged at a high current level (30A) after each journey) and where EVs are dispatched by the

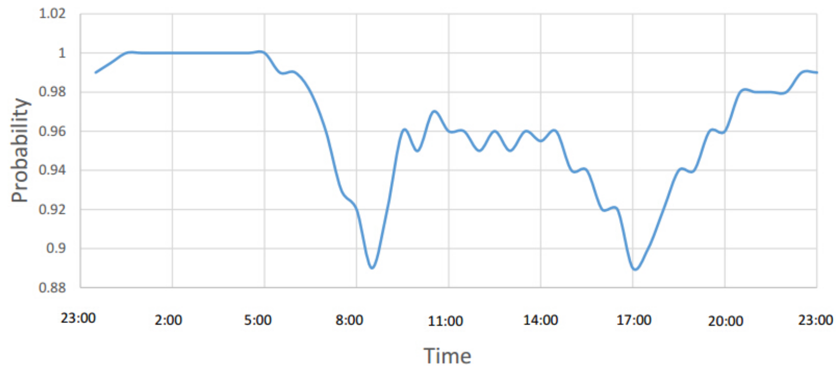


FIGURE 4.7: Probability of cars that are parked during a weekday

AHP-based strategy and the rule-based strategy. The power injection at the slack bus represents the power exchange between the power system under investigation and the rest of the grid. Simulation results for the power injections at the slack bus during 7 days are presented in Figure 4.8. The choppy transient response during the first day is due to the random assignment of the SOC of the EVs at the start of the simulation. The following days start with the SOC of EV batteries that have settled to a steady state value, hence the repeatable steady pattern of the power injected at the slack bus during the rest of the days, which confirms the stability of the dispatch strategies. Figure 4.9 shows power injections into the slack bus during the seventh day with the original curve of power injected at the slack bus when the grid is operated without EVs. Certain key data in Figure 4.9 is given in Table 4.6.

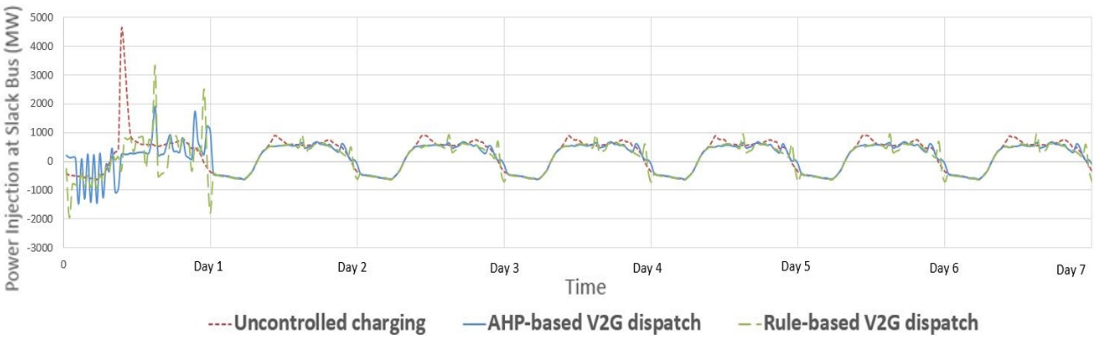


FIGURE 4.8: Power injections at the slack bus for all three cases during a week

TABLE 4.6: Numerical analysis of the efficacy of AHP-based dispatch strategy in load levelling compared with that of rule-based dispatch strategy and uncontrolled charging

Dispatch Strategy	Lowest Power Injection (MW)	Highest Power Injection (MW)
AHP-based	-626.3	673.4
Rule-based	-706.5	969.5
Uncontrolled	-630.5	877.5

As shown in Table 4.6, the rule-based dispatch strategy and uncontrolled charging result in very high peaks of power injections, which are respectively about 200 and 300 MW higher than the peak power injection when using AHP-based strategy. Therefore,

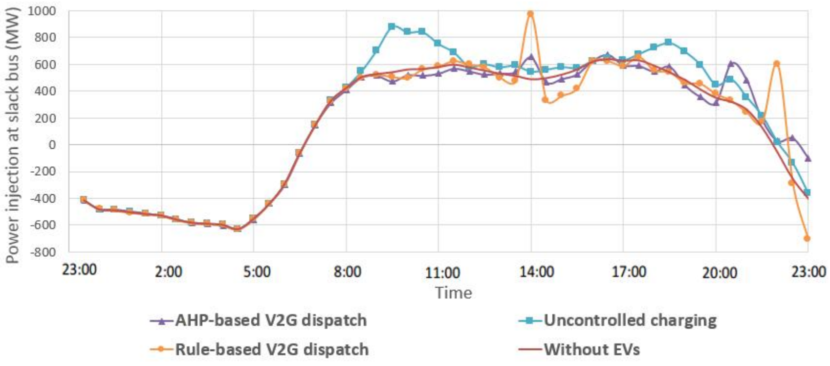


FIGURE 4.9: Power injections at the slack bus for all three cases during a day

AHP-based dispatch strategy has better performance in peak shaving compared to the two other strategies in the sense of creating much lower peak power injection from the main grid. During the off-peak period, the AHP-based dispatch strategy results in at most 626.3 MW power to be transferred to the main grid, while uncontrolled charging results in about 4 MW more power being absorbed by the main grid. The rule-based strategy results in 80 MW more power being transferred to the main grid. Thus, AHP-based strategy performs better in valley filling than rule-based strategy and uncontrolled charging. Compared with the original condition without EVs (631 MW at most being transferred to the main grid), AHP-based approach actually has effective valley filling, which is not visible in the figure though. Therefore, further development (inclusion of forecast model perhaps) should be made on the dispatch model in the future for better valley filling and peak shaving performance, as discussed in the Conclusions.

4.5 Sensitivity Analysis

In order to further investigate and understand the characteristics of the proposed AHP-based dispatch strategy, sensitivity analysis is carried out for many different scenarios. First of all, the effects of changing the weightings of EV users' and grid's concerns (i.e.w_{EV} and w_G respectively) on the EV daily SOC and cost are investigated, as well as the impacts of the relative importance between SOC w_{SOC} and electricity price w_{EP}.

As shown in Table 4.7, when the weighting of EV users' concerns w_{EV} increases, more attentions are paid to the EV user's benefits, hence the daily SOC condition of EV improves, as illustrated in Figure 4.10 (the corresponding current levels and final priorities of dispatch decisions are presented in Figure 4.11 and Figure 4.12, respectively), normally leading to an increasing charging cost. That is not hard to understand, because a better SOC condition implies more electric energy to be charged or less to be discharged, which increases the daily cost to EV users. However, when the EV's SOC condition is already good enough within the 40% and 80%, the increase of w_{EV} does not improve the daily SOC condition of the EV but reduces the daily cost for the EV (e.g. when $w_{SOC} = 80\%$,

$w_{EP} = 20\%$ with $w_{EV} = 50\%$ and 80% respectively, as can be seen in Figure 4.10). That is because when the EV's daily SOC condition is already good enough, due to having a high priority, the improvement space becomes very limited. Thus, increasing w_{EV} to continually increase its composite priority will not make significant improvement to the SOC condition. But, the increase of the composite priority of cost due to increasing w_{EV} will tend to charge the EV at the low buying price instead of scattering the charging process over a wide period to avoid increasing the peak load demand. Moreover, when keeping the weightings of EV user and grid concerns unchanged but adjusting the relative importance between the two criteria, SOC and electricity price, the results change as well, as demonstrated in Figure 4.13 (the corresponding current levels and final priorities of dispatch decisions are presented in Figure 4.14 and Figure 4.15, respectively). With a higher weighting given to electricity price (i.e. w_{EP} is higher but w_{SOC} is lower), it will be better to discharge an EV battery due to the high selling price, in order to save costs, than to charge it due to the low SOC. Thus, discharging might be chosen as the current dispatch action and the daily cost to the EV user, unsurprisingly, decreases as SOC condition worsens. However, with higher w_{SOC} but lower w_{EP}, the dispatch action recommended for ensuring that sufficient electricity remains in the battery will have a higher priority to be chosen as the current EV dispatch action than that recommended according to electricity price to save cost. The SOC condition is therefore, as expected, better with increasing daily cost.

Table 4.7: Sensitivity Analysis: the effects of changing the weightings of criteria on the EV daily SOC and cost

$\frac{w_{EV}}{w_G}$	$w_{SOC} = 20\%$ $w_{EP} = 80\%$		$w_{SOC} = 50\%$ $w_{EP} = 50\%$		$w_{SOC} = 80\%$ $w_{EP} = 20\%$	
	SOC	*Cost (£)*	*SOC*	*Cost (£)*	*SOC*	*Cost (£)*
$\frac{20\%}{80\%}$	[0.16, 0.69]	0.66	[0.18, 0.69]	0.82	[0.19, 0.69]	0.89
$\frac{50\%}{50\%}$	[0.19, 0.70]	1.59	[0.33, 0.69]	2.25	[0.47, 0.70]	2.62
$\frac{80\%}{20\%}$	[0.28, 0.75]	1.87	[0.36, 0.71]	2.35	[0.47, 0.70]	2.44

As for the impact of different settings of w_G and w_{EV} on the network, similar sensitivity analysis is carried out and the result is shown in Figure 4.16. The corresponding accumulated energy injections are presented in Table 4.8. It is clear in Figure 4.16 that the network load demand curve is improved, in terms of decreasing peak load demand, with higher weighting assigned to grid's concerns, which is just as expected and easy to understand. It is evident from Table 4.8 that when the weighting of grid concerns (w_G) becomes high at 80% the amount of energy injected at the slack bus reduces significantly with more energy absorbed by the network during off-peak period and less energy consumed during peak period. Compared with original situation without EVs, when w_G

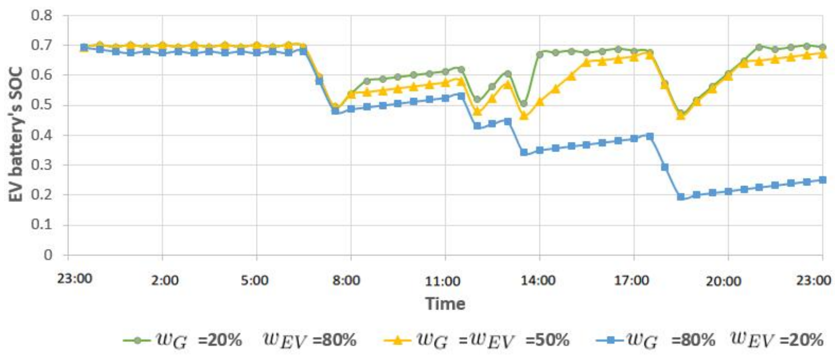


FIGURE 4.10: Daily SOC conditions of an EV battery under different settings of w_G and w_{EV} but constant w_{SOC} (80%) and w_{EP} (20%)

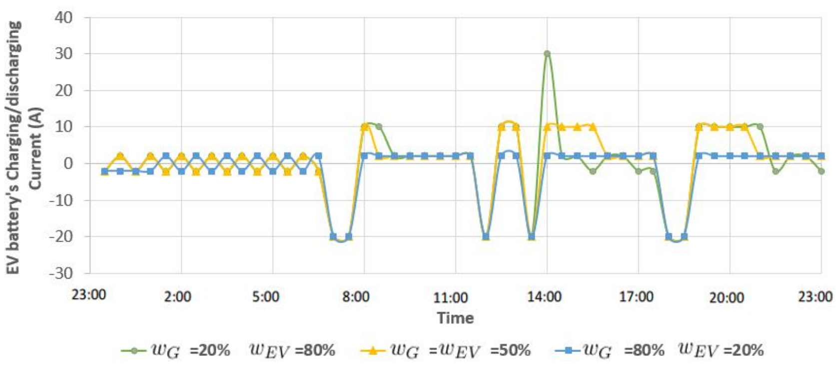


FIGURE 4.11: Charging/discharging currents of an EV battery under different settings of w_G and w_{EV} but constant w_{SOC} (80%) and w_{EP} (20%)

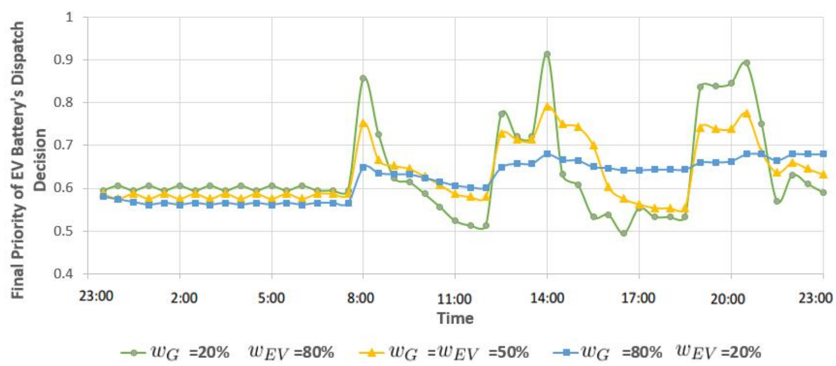


FIGURE 4.12: Final priorities of dispatch decisions of an EV battery under different settings of w_G and w_{EV} but constant w_{SOC} (80%) and w_{EP} (20%), which decide the dispatch current levels in Figure 4.11 and hence SOC conditions in Figure 4.10

becomes 80%, both valley filling and peak shaving are achieved. This is expected as load levelling takes priority as can be seen in Figure 4.16.

As mentioned earlier, the setting of weightings of the AHP model is based mainly on expert experience and common sense. In practice we envisage that the initial weighting determined based on common sense will be refined based on simulation results at the design stage or over time during the operation stage to produce sensible results. This may require some negotiations between the parties involved who may accept a compromise

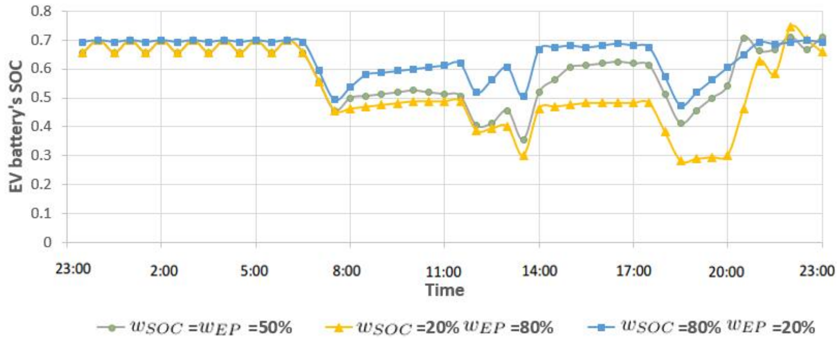


FIGURE 4.13: Daily SOC conditions of an EV battery under different settings of w_{SOC} and w_{EP} but constant w_G (20%) and w_{EV} (80%)

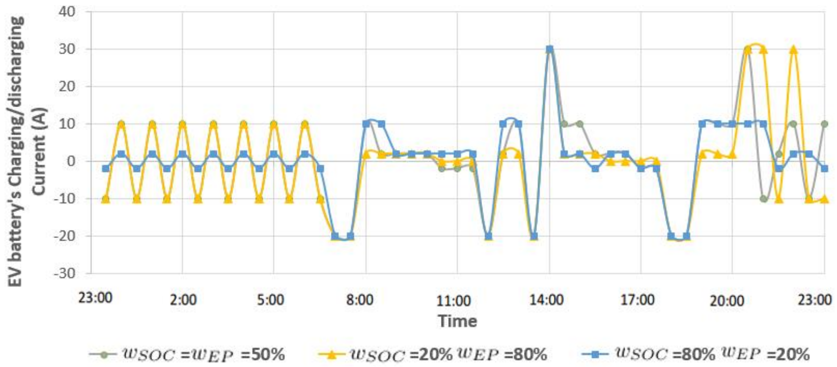


FIGURE 4.14: Charging/discharging currents of an EV battery under different settings of w_{SOC} and w_{EP} but constant w_G (20%) and w_{EV} (80%)

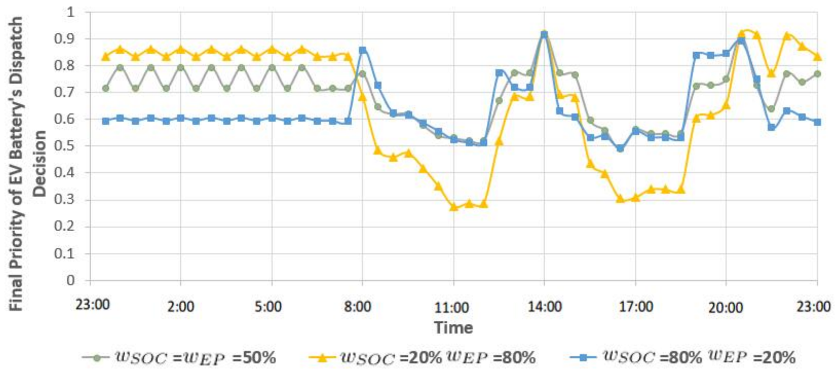


FIGURE 4.15: Final priorities of dispatch decisions of an EV battery under different settings of w_{SOC} and w_{EP} but constant w_G (20%) and w_{EV} (80%), which decide the dispatch current levels in Figure 4.14 and hence SOC conditions in Figure 4.13

once they are informed about the implications of their choices and perhaps after their fears are allayed. Indeed the weighting of the factors and even the factors themselves may need to be changed with time as the system evolves.

Furthermore, the impact of increasing the percentage of V2G-enabled EVs on load levelling is illustrated in Figure 4.17 and 4.18, where V2G-disabled EVs are charged in an uncontrolled way. Figure 4.17 shows the power injected at the slack bus during a 24-hour period. As the percentage of V2G-enabled EVs increases, the peak shaving increases. In

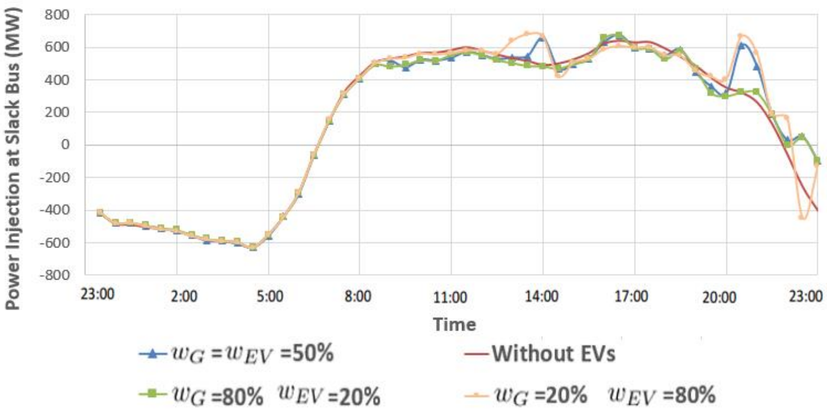


FIGURE 4.16: Daily power injections at the slack bus under different settings of w_G and w_{EV}

TABLE 4.8: Accumulated Energy injections at the slack bus under different settings of w_G and w_{EV} in 24 hour period

w_G	w_{EV}	Accumulated Energy during off-peak period (MWh)	Accumulated Energy during peak period (MWh)	Accumulated Energy in 24 hour period (MWh)
20%	80%	-3829	7708	3879
50%	50%	-3617	7413	3796
80%	20%	-3613	7039	3426
Without EVs		-3990	7321	3331

Figure 4.18, the total accumulated energy injected into the slack bus during a 24-hour period is plotted against the percentage of V2G-enabled EVs. This graph was obtained by repeating the simulation 40 times for each percentage of V2G-enabled EVs; each simulation starts with a randomly assigned travel pattern; the mean and standard deviation of injected energy at the slack bus during a 24-hour period are calculated for each percentage of V2G-enabled EVs from the results of the 40 simulations. The accumulated energy injection reduces when more EVs participate in the V2G operation. That is just as expected and can be easily understood; increasing the percentage of V2G-enabled EVs raises the available battery capacity that can be used to meet the power system demand.

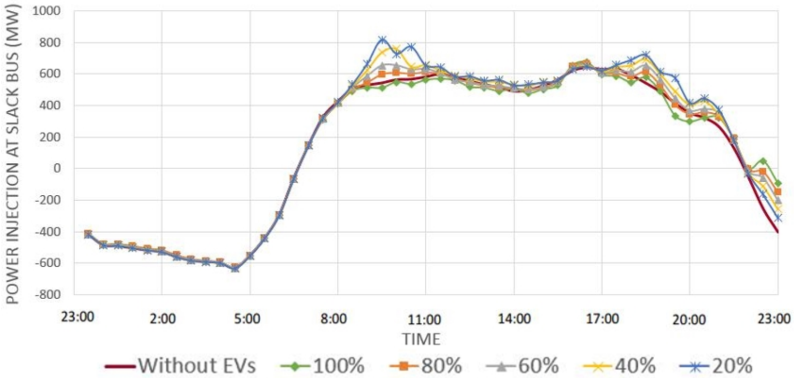


FIGURE 4.17: Power injections at the slack bus under different percentage of V2G-enabled EVs

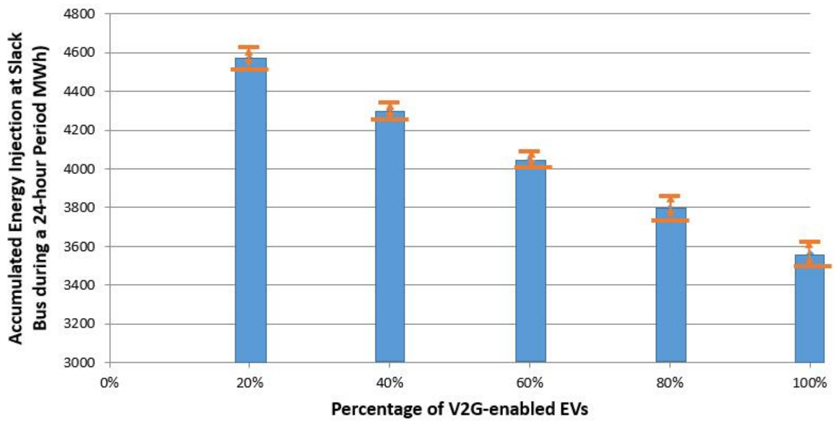


Figure 4.18: Average accumulated energy injections at the slack bus under different percentage of V2G-enabled EVs (Error bars represent ± standard deviation)

It is important to note that this work presents the proposed AHP-based dispatch strategy using an example setting of the criteria weightings in the AHP hierarchy model and the percentage of V2G batteries. However, the settings of weightings can be adjusted by the PSO to accommodate its requirements and the actual situation of the network and EV cluster. The proportion of V2G batteries in practice depends on the EV users' willingness and/or EV chargers' hardware.

Moreover, an estimate about the EV user's next journey is necessary for the proposed technique to carry out effective and customized dispatch. This information is expected to be supplied to the PSO either when the vehicle is plugged into the charging point or through the internet. This does not need to be a burden to the EV user. For example, users could have an App (a software agent) supplied by the PSO installed on their smart phones or computers that estimates the energy required for the next journey based on information in their calendar about their future activities (and the time and distance of the next journey), traffic conditions, weather, etc. They can send the energy requirement information (could be simply minimum SOC) to the PSO automatically through the internet or wireless communication channels, without revealing any of their activities and without impacting their privacy. Forecasting tools can also be employed, based on historical records of the EV's previous driving activities, to provide a forecast of EV user's travel pattern including an estimate of the next journey's energy usage. Further research is needed on how this may be implemented in practice.

Furthermore, the AHP-based dispatch strategy is proposed for practical application, therefore the computational efficiency is an important factor. It takes only 0.3s on average to make dispatch decisions for the total EV fleet of 625000 EVs at each time step if using a computer with a Quad Core 3.4GHz CPU and 16GB RAM as an example, which provides an insight into the computation time required by the proposed algorithm.

Some information is needed to implement the proposed dispatch strategy, apart from the EV user's driving activities, such as the real-time pricing, network's load demand and

N-1 contingency information. These are supposed to be notified a short while ahead of the beginning of every time interval, via wireless communication for example. However, if the communication is suspended due to a sudden accident, e.g. hacker attack, and some or all of the required information for the next time interval cannot be obtained in time, then an estimate or a forecast of the missing information will be used to dispatch the EVs. In more detail, if the exact pricing data is unknown, the PSO can either use the available day-ahead electricity price or use forecasting tools, which can be embedded into the proposed model, to predict the price for the next time interval. The uncertain load demand for the next time interval can be dealt with in a similar way, that is, the day-ahead network load demand forecast can be used to implement the proposed dispatch strategy. Besides, if N-1 contingency occurs without being able to inform the PSO, then EVs will be dispatched as if no contingency occurred. Due to the inherent redundancy of V2G and other energy storage that are in communication, this may not have a large impact as other EV clusters or stationary energy storage systems can participate in the alleviation of overload caused by the N-1 contingency. Note that it will be better to use exact real-time data in order for the proposed strategy to provide an accurate dispatch of EVs, however using well-estimated data for a certain time interval (when the communication breaks off) will not affect the dispatch decision of the proposed AHP-based strategy due to its robustness and inherent tolerance of uncertainty. One could envisage that the AHP priority factors may be adjusted depending on the level of confidence in the information, which in turn depends on whether they were based on actual information or estimated one, which could add further robustness. This matter deserves further investigation in the future.

4.6 Conclusions

This chapter has proposed a novel decentralized dispatch strategy for EV batteries using the AHP methodology taking into account the requirements of both EV users and the grid, such that the EV batteries can be dispatched to save cost while ensuring reliable driving experience with sufficient SOC left in the battery and help to support the grid for load levelling or under N-1 contingency. It is important to stress that this work is addressing this particular type of scenario but not limited to this. With slight adjustment, the proposed dispatch strategy can be flexibly applied to other kinds of scenarios that might come to fruition. For example, if home storage batteries are used instead of EV batteries, the dispatch strategy are almost the same except that the electric energy will not be stored for travelling but for home appliance usage during blackouts or peak periods. As the basic/core approach is highly independent of the different scenarios, it can be used as a tool to validate those scenarios and compare their different impacts.

The proposed dispatch strategy was tested on an IEEE Reliability Test System. The simulation results demonstrated the feasibility and efficacy of the proposed AHP-based dispatch strategy in satisfying the requirements of both EV users and the grid. Moreover, this strategy has proved to be generally better than the rule-based one proposed in [20]. Compared to the rule-based dispatch approach, it costs £0.86 more a day, per vehicle, but it improves the EV battery's SOC condition and performs better in terms of load levelling. Under the severe N-1 contingency that overloads the branch close to the bus to which the EV is connected, the AHP-based dispatch strategy correctly selects the dispatch action of the EV battery that can help to alleviate the overloading.

In the application of V2G batteries using the proposed AHP-based dispatch strategy, the grid operator gathers the real-time data a short while (e.g. 5 minutes) prior to the beginning of each time interval, as discussed earlier. Real-time pricing (RTP) data need to be downloaded from the electricity market on a half-hourly basis (i.e. before the start of every time interval of 30 minutes). RTP can be affected by the real-time demand and supply conditions, which include the dispatch of V2G batteries, i.e. the charging as a load or discharging as a source. Moreover, as the electricity price is considered in the V2G battery dispatch, RTP will have impact on the determination of battery dispatch actions. Hence, there is interactive effect between V2G battery dispatch and RTP, which will be investigated in the future. Furthermore, the proposed AHP-based dispatch strategy is actually utilized by the grid operator, not by the EV users, as discussed earlier. In order to encourage the V2G operation, incentive payments will be given to the EV users if they agree to let their vehicles participate in the grid operational support when needed. The settings of incentive payments and the relevant policy requires further investigations, as discussed in Chapter 7. Moreover, the stochastic modelling of EV travel patterns in this work is simple to use but not sophisticated enough to reflect the reality. In the future, EV travel patterns should be modelled in a more proper way using algorithms such as the Markov process, as discussed in [62], and Copula, as investigated in [124], and the historic data as utilized in [142]. Another improvements that could be made is to include the forecast of the grid's load level and electricity price in the next few hours into the dispatch model. In this way, the dispatch system is aware of the grid's environment in the near future, which might be more or less suitable for charging EVs than the current situation. Therefore, the dispatch system can make a better and more accurate decision in terms of whether to shift the EV charging load to future moments compared with the current situation, in order to achieve better valley-filling results.

In this work, renewable energy sources are not considered, and the EVs are dispatched in a power system without the integration of these sources.

Chapter 5

Optimal Coordination of Vehicle-to-Grid Batteries and Renewable Generators in A Distribution System

The previous Chapter proposed a dispatch strategy for EVs only. However, without RES, the CO_2 emissions cannot be effectively reduced. Therefore, RES should be integrated into the power grid and EVs should be at least partially charged from these sources. The increasing penetration of EVs and RESs will challenge the distribution network, since it has constrained capacity and most EVs and distributed renewable generators are directly connected to it. However, appropriate dispatch of electric vehicles via vehicle-to-grid operation in coordination with the distributed renewable generations can provide support for the grid, reduce the reliance on traditional fossil-fuel generators and also benefit EV users. This chapter develops a novel agent-based coordinated dispatch strategy for EVs and distributed renewable generators, taking into account both grid's and EV users' concerns and their priorities. This optimal dispatch problem is formulated as a distributed multi-objective constraint optimisation problem utilizing Analytic Hierarchy Process and is solved using a dynamic-programming-based algorithm. The proposed coordinated dispatch strategy is tested on a modified UK Generic Distribution System (UKGDS), while the electricity network model is simplified using virtual sub-node concept to alleviate the computation burden of a node (agent). The simulation results are presented and demonstrate the feasibility and stability of this dispatch strategy.

5.1 Introduction

To reap desired environmental benefits, RGs should provide at least a part of EVs' charging energy [8]. Therefore, a novel approach is required, so that EVs can be dispatched in coordination with renewable power.

In this work, a novel agent-based coordinated dispatch strategy for EVs and RGs is developed for real-time application, which aims at satisfying the concerns and requirements of both EV users and the grid, including 1. cost, as saving charging cost is a very common request of EV users, which are taken into account in many publications [66, 20, 15, 68]; 2. sufficient SOC for the next journey, which is important because mobility is the basic function of an EV, as considered in many researches [84, 62, 66, 68, 85]; 3. improved utilization of renewable energy, because it is crucial for carbon emission reduction, as discussed earlier and in [84, 43, 51, 53]; 4. load levelling, which is one of the main grid operational support that EV batteries can provide and is also discussed in many papers [58, 65, 68, 87].

In this strategy, each node in the network is represented by an software agent which is only aware of the elements that are locally connected to it and manages the dispatch of EVs and RGs connected to it, based on information received from the agents of other nodes that are directly linked to it, so that the stability of the network is ensured and all the objectives of dispatch are best achieved.

Being aware of a very large computational burden that could occur at a node's agent connecting with a great number of children nodes, a novel concept of a virtual sub-node is proposed to simplify the electricity network model in order to reduce this burden. Accordingly, the dispatch problem is formulated as a distributed multi-objective constraint optimization problem (DMOCOP) and then solved using a dynamic-programming-based algorithm to derive an optimal set of dispatch actions for EVs and RGs within a distribution network. The DMOCOP is developed from the distributed constraint optimization problem (DCOP) that was proposed in [98], using an Analytic Hierarchy Process (AHP) [116] to take several different objectives of dispatch, as discussed above, into account at the same time.

The proposed dispatch strategy is tested on a modified UKGDS, which is a radial distribution network, for its stability, feasibility and effectiveness at satisfying the requirements of both EV users and the grid. In practice, the aggregator or the distribution network operator (DNO) is supposed to be in charge of this optimal dispatch problem.

The rest of this chapter is organized as follows. In Section 5.2, the proposed coordinated dispatch strategy of EVs and RGs is presented in detail. The results of simulations using MATLAB to verify its feasibility and efficacy, are presented and discussed in Section 5.3. Finally, the conclusions of the work are presented in Section 5.4.

This work has been published in *Energy*, vol. 113, pp. 1250 – 1264, 2016 [94].

5.2 Coordinated Dispatch Strategy of Electric Vehicles and Renewable generators

The aim of the strategy is to realize optimal coordinated dispatch of EVs and RGs in the distribution network so that multiple objectives can be achieved such as saving charging cost to EV users while ensuring sufficient electricity remains in the batteries for the next journey, reducing waste of energy generated by RGs and supporting grid's operation like load levelling. It is assumed that the dispatch actions are conducted every 30 minutes.

The independent variables include the vehicles' driving patterns, which are initially randomly assigned as commonly done in many studies [67, 143]. The other independent variables including renewable power fluctuation, price of electricity and load pattern are determined based on historical data that can be found in [141, 144]. The dependent variables include cost, degree of utilization of RG, load levelling and SOC as discussed in Section 5.1.

5.2.1 General Description of the Electricity Network and Agents

Figure 5.1 shows a radial distribution network, which is derived from a UK Generic Distribution System (UKGDS)[145]. It includes renewable generators and EVs. The node v_0 is the slack bus, which is connected to the rest of power grid.

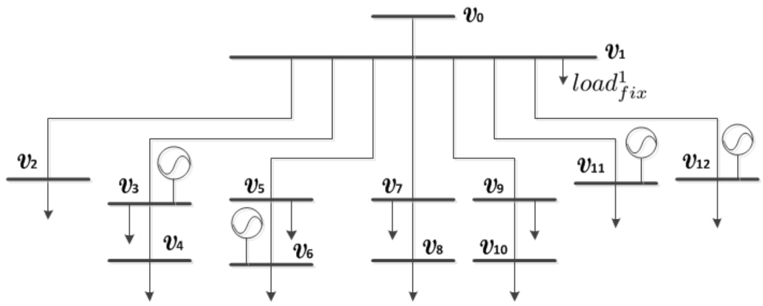


FIGURE 5.1: Diagram of a modified generic radial distribution network

In more detail, the network contains a set of renewable generators, which are denoted by $\mathbf{G} = \{g_1, \ldots, g_n\}$ with each generator g_i producing a certain amount of renewable power $p_i \in RG_i$ MW. $RG_i = \{0, \ldots, P_i^{max}\}$, where RG_i is discretized into 1 MW steps to simplify the computational burden and $P_i^{max} \in \mathbb{Z}^+$ is the maximum power output of renewable generator g_i rounded to the nearest whole MW at a particular moment in time; P_i^{max} will vary depending on the environmental conditions (e.g. cloud cover or wind speed variation). Let $\mathbf{p} = \{p_1, \ldots, p_n\}$ denote a set of power output variables for the renewable generators in $\mathbf{G}$.

Moreover, the network provides the capability of connecting a great number of EVs, each of which can be either charged or discharged. **EV** represents a set of EVs, where $\mathbf{EV} = \{ev_1, \ldots, ev_m\}$. Each EV has 7 choices of dispatch mode when it is parked and connected to the grid: charge (+) or discharge (–) at high (3), medium (2) or low-level (1) current, OR idle (0). Thus, each EV ev_i is dispatched in a certain mode $\delta_i \in DM_i = \{-3, -2, -1, 0, 1, 2, 3\}$, where the sign indicates the dispatch action (i.e. charge/discharge), and the absolute value represents the dispatch current level. Let $\boldsymbol{\delta} = \{\delta_1, \ldots, \delta_m\}$ denote a set of dispatch modes for EVs in **EV**.

$V = \{v_1, \ldots, v_k\}$ denotes the set of nodes within the network. A node v_i exchanges power with other nodes and contains a combination of fixed loads, EVs and RGs. $chi(v_i)$ denotes a set of children nodes of v_i, while $adj(v_i)$ represents a set of adjacent nodes of v_i, i.e., nodes that are directly connected to v_i via distribution cables, including its children nodes and its parent node. $load^i_{fix}$ represents the fixed load at v_i, while $\mathbf{EV}(v_i)$ and $\mathbf{G}(v_i)$ denote the sets of EVs and RGs connected at v_i, respectively. Note that $\mathbf{EV}(v_i) = \varnothing$ and $\mathbf{G}(v_i) = \varnothing$ respectively mean that v_i contains no EV and no RG.

The set of distribution cables in the network is denoted by $\mathbf{T}$ and t_{ij} refers to the distribution cable between nodes i and j. The power flow along the cable t_{ij} is denoted by f_{ij}, and cannot exceed the thermal capacity of the cable, C_{ij}. Distribution losses have not been considered in this work as is often done in many studies [146, 147]. The effect of distribution loss will be the subject of future investigations.

The network is controlled by agents in a decentralized way. Each node v_i is represented by an agent, which is aware of the output power domains of the local RGs, $\mathbf{G}(v_i)$, and all possible dispatch modes of the connected EVs, $\mathbf{EV}(v_i)$, and has control over the dispatch of renewable power output and EVs. Each agent has a utility function, carries out a part of the computation required to achieve optimal and stable operation of the network (i.e. utility computation) and communicates the computation results with its adjacent agents that map to the adjacent nodes $adj(v_i)$ of its designated node v_i. In this work, the utility function maps to the penalty cost at the designated node, which takes into account multiple objectives, using the Analytic Hierarchy Process (AHP), and measures how well these objectives are achieved. Based on the above definitions, this optimal coordinated dispatch problem can be formulated as a distributed multi-objective constraint optimisation problem (DMOCOP), which is developed from the distributed constraint optimisation problem (DCOP) proposed in [98], after simplifying the model using the virtual sub-node concept proposed in the following section to relieve the computational burden of a certain agent.

5.2.2 Simplification of The Model Using Virtual Sub-Node Concept

As will be discussed in the following section, an agent of a node performs computations based on the information received from its children and conducts the communication with them and the parent node. In a large network a node v_i may have a great number of children nodes, which increases the computational and communication burden. In order to solve this issue, v_i is considered to consist of several virtual sub-nodes, each of which is connecting some of v_i's children nodes and controlled by a sub-agent of the agent that maps to v_i. The sub-agents work simultaneously, with each sub-agent undertaking a part of the computation dealing with the information sent from the children of the corresponding virtual sub-node. A diagram of the distribution network with virtual sub-nodes is shown in Figure 5.3. In these figures, v_1^1, v_1^2 and v_1^3 are the three virtual sub-nodes of v_1. The load connected to v_1, $load_{fix}^1$, is connected directly to the parent node. Furthermore, the distribution cable between v_1 and v_0 is divided into three virtual sub-cables, whose thermal capacities are defined based on C_{01} and $load_{fix}^1$. As $load_{fix}^1$ is the fixed load at node v_1 consisting of active load $pload_{fix}^1$ and reactive load $qload_{fix}^1$. The total complex power that can be transmitted via v_1 to its children is limited in both active (real) and reactive (imaginary) parts, as follows:

$$Cp_{01}^n = \frac{\sum\limits_{s \in chi(v_1^n)} C_{s1}}{\sum\limits_{d \in chi(v_1)} C_{d1}} \times (pload_{fix}^1 \times \frac{(C_{01} - |load_{fix}^1|)}{|load_{fix}^1|}) \tag{5.1}$$

$$Cq_{01}^n = \frac{\sum\limits_{s \in chi(v_1^n)} C_{s1}}{\sum\limits_{d \in chi(v_1)} C_{d1}} \times (qload_{fix}^1 \times \frac{(C_{01} - |load_{fix}^1|)}{|load_{fix}^1|}) \tag{5.2}$$

where Cp_{01}^n and Cq_{01}^n are the capacities of the nth virtual sub-cable in terms of active and reactive power flows, respectively. $chi(v_1^n)$ is a set of child nodes of the virtual sub-node v_1^n, e.g. $chi(v_1^1) = [v_2, v_3]$ in Figure 5.3. Similarly, $chi(v_1)$ is a set of v_1's children, i.e. $[v_2, v_3, v_5, v_7, v_9, v_{11}, v_{12}]$. These 2 equations will be easier to interpret using Figure 5.2. In Figure 5.2, the total capacity of the virtual sub-cables should be $x = C_{01} - |load_{fix}^1|$. The absolute value of $load_{fix}^1$ is used to cater for both absorbing loads and generators. The apparent power capacity of a virtual sub-cable C_{01}^n is determined by multiplying x by the ratio of the total capacity of the cables connecting v_1^n to its children nodes, which equals $\sum\limits_{s \in chi(v_1^n)} C_{s1}$, to the total capacity of all the cables connecting the original v_1 node to its children nodes, i.e., $\sum\limits_{d \in chi(v_1)} C_{d1}$. Active and reactive power capacities are calculated by multiplying the apparent power capacity of each virtual sub-cable by $\frac{pload_{fix}^1}{|load_{fix}^1|}$ and $\frac{qload_{fix}^1}{|load_{fix}^1|}$, respectively.

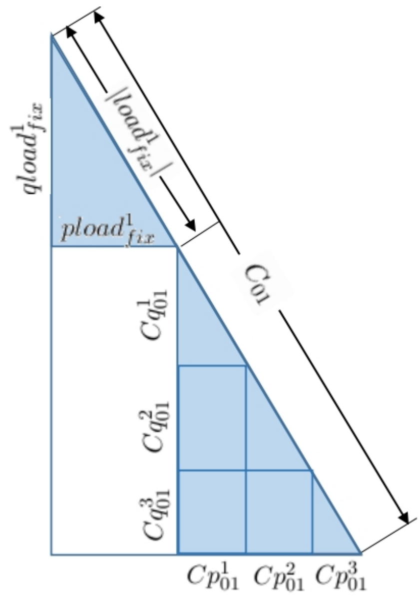


FIGURE 5.2: An illustration of how the virtual sub-cable capabilities are derived

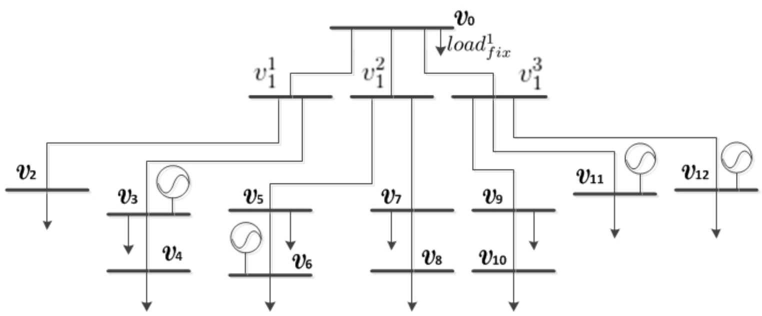


FIGURE 5.3: The diagram of the radial distribution network with virtual sub-nodes

The validity of the virtual sub-node approach has been verified via mathematical analysis and simulation (Details are given in the Appendix A), which demonstrate that it does not compromise the optimal dispatch results in this work.

5.2.3 Formulation of Distributed Multi-Objective Constraint Optimisation Problem (DMOCOP)

The DMOCOP extends the DCOP procedure described in [98] to a multi-objective optimization problem. It contains three main elements:

1. Variables: a set of h variables $X = \{x_1, \ldots, x_h\}$. In this work, x_i can be a renewable generator's power output or an EV's battery dispatch mode. Thus, in this work $X = \{\mathbf{p}, \boldsymbol{\delta}\}$.

2. Domains: a set of finite domains $D = \{d_1, \ldots, d_h\}$, which include all possible values of variables X. In this work, d_i can be represented as follows:

$$d_i = \begin{cases} RG_i & \text{when } x_i \text{ is a renewable generator} \\ DM_i & \text{when } x_i \text{ is an EV battery} \end{cases} \tag{5.3}$$

3. Utilities: a set of k utilities $U = \{U_1, \ldots, U_k\}$, each of which corresponding to an agent. In this work, U_i maps to the penalty cost at agent i, i.e., how unsatisfactory a combination of dispatch actions is in terms of the objectives. They are formed by using the AHP, as discussed in the following section.

The objectives include saving cost while leaving sufficient charge in an EV's battery, network load levelling and reduction of wasted renewable energy.

Objective 1 (RE): Reduce wasted renewable energy.

Assume that the weather forecast is accurate enough, and it is predicted that a renewable generator's power output $\widetilde{P_{RG}}$ is available during the following time interval of 30 minutes. The penalty cost of wasting renewable power C_{RG} is measured as follows:

$$C_{RG} = (\widetilde{P_{RG}} - P_{RG})/\widetilde{P_{RG}}, \tag{5.4}$$

where P_{RG} is the actual amount of power that is injected by a RG into the network.

Objective 2 (BS): Sufficient EV battery SOC.

Assuming that EVs' travel patterns are available (either directly entered by the user or estimated based on information in their diary or passed travel history), then the expected SOC at the end of the current time interval SOC_p is estimated to be:

$$SOC_p = \begin{cases} SOC & \text{if EV has no travel plan during the next 2 hours} \\ \frac{SOC_{pf} - SOC}{T_p} t + SOC & \text{if EV has travel plan within 2 hours and current SOC is not enough for its next journey (t=1 time interval)} \\ SOC_{pf} & \text{if EV has travel plan within 2 hours and has enough SOC for its next journey} \end{cases} \tag{5.5}$$

When the EV is expected to travel within the next 2 hours, a linear interpolation is used to determine the value of SOC_p as illustrated in Figure 5.4. T_p is the available preparation time before the next journey in multiples of dispatch action time interval, i.e. multiples of 30 minutes. SOC_{pf} is the desired SOC of the EV battery before the next journey, which may be lower than the current SOC value, in which case the EV will be able to participate in V2G operations.

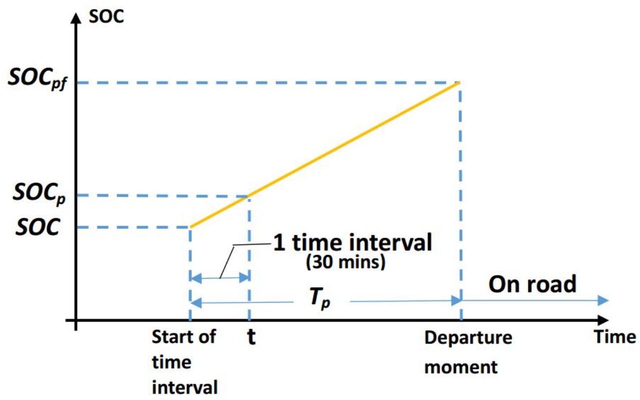


FIGURE 5.4: Illustration of the calculation of SOC_p when EV has travel plan within 2 hours and current SOC is not adequate

The aim is to keep the SOC at about 0.5. The penalty cost of insufficient SOC, C_{SOC}, is measured as follows:

1. If the EV has no travel plan during the next 2 hours and the battery is more than half full at the end of this time interval OR if the EV will be used in 2 hours and battery's SOC is not less than SOC_p at the end of the time interval, then:

$$C_{SOC} = 0. \tag{5.6}$$

2. If the EV has no travel plan during the next 2 hours and the battery is less than half full at the end of this time interval, then:

$$C_{SOC} = \frac{0.5 - SOC_n}{0.5 - S_{min}}, \tag{5.7}$$

   where SOC_n is the battery's SOC after a time interval of charging/discharging. In this work, the lower limit of SOC S_{min} is selected to be 0.4 to provide a good margin above the absolute minimum of 0.3.

3. If the EV has travel plans within 2 hours and the battery's SOC_n is lower than SOC_p:

$$C_{SOC} = \frac{SOC_p - SOC_n}{SOC_p - S_{min}}. \tag{5.8}$$

Objective 3 (CC): Save charging cost to EV users.

The penalty cost is sure to be low when an EV is charged at a low electricity buying price while discharged at a high selling price. And the penalty cost C_{ep} could vary depending on how fast the EV is charged/discharged, i.e., the charging/discharging current.

1. Charge at high current (i.e. 30 A):

$$C_{ep} = \begin{cases} \frac{bp - bp_{min}}{hbp - bp_{min}} & bp < hbp \\ 1 & bp \geqslant hbp \end{cases}. \tag{5.9}$$

2. Charge at mid-level current (i.e. 10 A):

$$C_{ep} = \begin{cases} \frac{|bp - (bp_{min} + hbp)/2|}{(hbp - bp_{min})/2} & bp < hbp \\ 1 & bp \geqslant hbp \end{cases}. \tag{5.10}$$

3. Charge at low current (i.e. 2 A):

$$C_{ep} = \begin{cases} \frac{hbp - bp}{hbp - bp_{min}} & bp < hbp \\ 0 & bp \geqslant hbp \end{cases}, \tag{5.11}$$

   where bp is the electricity buying price. bp_{max} and bp_{min} are respectively the top and bottom buying prices, while hbp is the high buying price threshold defined as $hbp = 0.9 * (bp_{max} - bp_{min}) + bp_{min}$. Equations (5.9)–(5.11) give progressively lower penalties for relatively high electricity buying prices as the charging current reduces.

   Similarly, the penalty cost for selling is defined as follows:

4. Discharge at high current (i.e. 30 A):

$$C_{ep} = \begin{cases} \frac{sp_{max} - sp}{sp_{max} - lsp} & sp > lsp \\ 1 & sp \leqslant lsp \end{cases}. \tag{5.12}$$

5. Discharge at mid-level current (i.e. 10 A):

$$C_{ep} = \begin{cases} \frac{|sp - (sp_{max} + lsp)/2|}{(sp_{max} - lsp)/2} & sp > lsp \\ 1 & sp \leqslant lsp \end{cases}. \tag{5.13}$$

6. Discharge at low current (i.e. 2 A):

$$C_{ep} = \begin{cases} \frac{sp - lsp}{sp_{max} - lsp} & sp > lsp \\ 0 & sp \leqslant lsp \end{cases}. \tag{5.14}$$

7. No dispatch (i.e. 0 A):

$$C_{ep} = \begin{cases} 0 & sp \leqslant lsp \& bp \geqslant hbp \\ 1 & otherwise \end{cases}, \tag{5.15}$$

where sp is the electricity selling price. sp_{max} and sp_{min} are respectively the top and bottom selling prices, while lsp is the low selling price threshold defined as $lsp = 0.1*(sp_{max} - sp_{min}) + sp_{min}$. These price data can be derived from historical data [141].

Objective 4 (LL): Load levelling in the distribution network.

The daily fixed (non-controllable) load demand is assumed to be available for both peak and off-peak periods. With the integration of EVs and RGs, the peak load could be pulled up further and might result in some spikes, or high power could be generated in the network and transferred to the rest of the grid through the slack bus during the off-peak period. Therefore, one of the objectives of coordinated dispatch of EVs and RGs is peak shaving and valley filling, i.e. load levelling. The penalty cost of failing to level the load, C_{ll}, is evaluated as follows:

$$C_{ll} = \begin{cases} 0 & load_l \leqslant load_{tot} \leqslant load_h \\ \frac{load_{tot} - load_h}{load_{max} - load_h} & load_h < load_{tot} < load_{max} \\ \frac{load_l - load_{tot}}{load_l - load_{min}} & load_{min} < load_{tot} < load_l \\ \frac{load_{min}}{load_{tot}} & 0 < load_{tot} \leqslant load_{min} \\ \frac{load_{tot}}{load_{max}} & load_{tot} \geqslant load_{max} \end{cases}, \tag{5.16}$$

where $load_{min}$ and $load_{max}$ are respectively the minimum and maximum fixed load demand during a day. $load_h$ and $load_l$ are the high and low fixed load thresholds, respectively, and they are determined by:

$$load_h = (load_{max} + load_{ave})/2, \tag{5.17}$$

$$load_l = (load_{min} + load_{ave})/2, \tag{5.18}$$

where $load_{ave}$ is the average of daily fixed load demand. $load_{tot}$ is the total load at a node, which can be calculated by

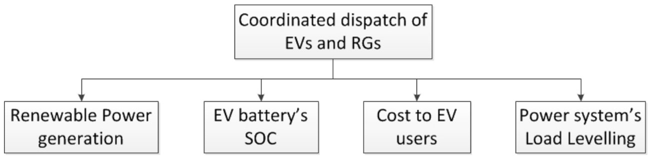
$$load_{tot} = load_{fix} + load_{ev}^c - load_{ev}^d - P_{RG}, \tag{5.19}$$

where $load_{fix}$ is the fixed load at a node, which is not controllable. $load_{ev}^c$ and $load_{ev}^d$ are the EVs' charging load and the EVs' discharging power, respectively. P_{RG} is the RG's output power.

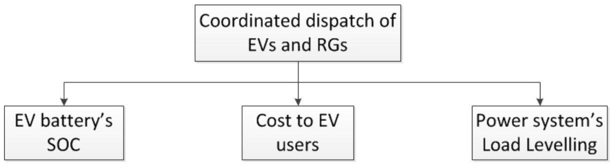
5.2.4 The Analytic Hierarchy Process

In order to jointly consider these objectives and thus the corresponding penalty costs in the optimization process, they should be weighted depending on their relative importance in determining how EV batteries and renewable generators should be dispatched in coordination.

Two different hierarchy models are respectively built up for the agents that only involve EV battery variables and those that have both EV battery and renewable generator variables, as shown in Figure 5.5.



(a) Agents that have both EV battery and renewable generator variables



(b) Agents that involve EV battery variables only

FIGURE 5.5: AHP Hierarchy Models of the Utilities of Two Types of Agents for the Coordinated Dispatch of EVs and RGs

The priorities of these agents' objectives are determined by forming a pairwise comparison matrix for each of the AHP hierarchy models shown in Figure 5.5; the scale numbers in the matrices are determined by the aggregator in practice based on experience and common sense. The matrices PC_a (Figure 5.5(a)) and PC_b (Figure 5.5(b)) are:

$$PC_a = \begin{array}{c c} & \begin{array}{cccc} RE & BS & CC & LL \end{array} \\ \begin{array}{c} RE \\ BS \\ CC \\ LL \end{array} & \begin{pmatrix} 1 & \frac{1}{2} & 2 & 2 \\ 2 & 1 & 2 & 2 \\ \frac{1}{2} & \frac{1}{2} & 1 & 1 \\ \frac{1}{2} & \frac{1}{2} & 1 & 1 \end{pmatrix} \end{array}, \tag{5.20}$$

$$PC_b = \begin{array}{c c} & \begin{array}{ccc} BS & CC & LL \end{array} \\ \begin{array}{c} BS \\ CC \\ LL \end{array} & \begin{pmatrix} 1 & 2 & 2 \\ \frac{1}{2} & 1 & 1 \\ \frac{1}{2} & 1 & 1 \end{pmatrix} \end{array}. \tag{5.21}$$

By calculating the principle eigenvectors of PC_a and PC_b, the priorities of RE, BS, CC and LL are found to be 27.81%, 39.52%, 16.34% and 16.34%, respectively for AHP model (a), and the priorities of BS, CC and LL are respectively 50%, 25% and 25% for AHP model (b).

The utilities U of agents that involve both EV battery and RG variables are then calculated as follows:

$$\begin{aligned} U = 27.81\% \times C_{RG} + 39.52\% \times C_{SOC} \\ +16.34\% \times C_{ep} + 16.34\% \times C_{LL}, \end{aligned} \tag{5.22}$$

while the utilities U of agents that only involve EV battery variables are calculated as follows:

$$U = 50\% \times C_{SOC} + 25\% \times C_{ep} + 25\% \times C_{LL}. \tag{5.23}$$

5.2.5 Constraints

For the stability of the electricity network and the better performance of EVs and RGs, several constraints are applied. The goal of agents is to find an assignment X^* for the variables in X (i.e. a combination of dispatch actions of EVs and RGs) that minimises the sum of penalty costs (i.e., the sum of utilities):

$$\arg\min_{X^*} \sum_{i=0}^{k} U_i, \tag{5.24}$$

subject to the following constraints:

Constraint 1: The sum of power flow into a node v_i should be equal to the sum of power flow out:

$$\sum_{j \in adj(v_i)} f_{ij} + load_{fix} + load_{ev}^{c} - load_{ev}^{d} - P_{RG} = 0, \tag{5.25}$$

where $adj(v_i)$ is the set of nodes that are connected to the node v_i. f_{ij} is the power flow from node i to j, and $f_{ij} = -f_{ji}$.

Constraint 2: The power flow along a distribution cable should not exceed its capacity:

$$|f_{ij}| \leqslant C_{ij}, \tag{5.26}$$

where C_{ij} is the thermal capacity of the distribution cable between nodes v_i and v_j.

Constraint 3: The SOC of EV batteries should be within the range from 0 to 1:

$$0 \leqslant SOC \leqslant 1. \tag{5.27}$$

Constraint 4: When the EV is not going to be used within the next 2 hours and the battery is currently less than half full of electricity, the EV has to be charged:

$$SOC_n \geqslant SOC, \quad \text{if } SOC < 0.5 \tag{5.28}$$

Constraint 5: When the EV has a travel plan within the next 2 hours and the battery's SOC is currently less than the desired at the end of the time interval, the EV has to be charged:

$$SOC_n \geqslant SOC, \quad \text{if } SOC < SOC_p \;. \tag{5.29}$$

5.2.6 Dynamic Programming Decentralized Optimal Dispatch (DPDOD)

In DPDOD, the agent representing a particular node computes the utility function corresponding to that node according to (5.22) or (5.23). A variable can only be assigned to an agent subject to the rule that an agent controls the EV battery and RG variables locally at its designated node.

Phase 1 — Value Calculation:

The calculation starts from leaf nodes (i.e., nodes that have no children) and ends at the root node (i.e. the node that has no parent node). Only after it receives all the computed results from its children does a node start its own computation. After that it sends its computing results to its parent node.

For each agent i (except agent 0 that controls v_0), the penalty cost is calculated for every possible combination of EV battery and RG dispatch actions, as well as the resulting power transfer along the distribution cable from the controlled node v_i to its parent node $\widehat{v_i}$. Hence, a Power Flow and the associated Penalty Cost ($PfPc$) message is formed as:

$$PfPc =< f_{i\hat{i}}, pec(f_{i\hat{i}}) >, \tag{5.30}$$

where $f_{i\hat{i}}$ is the power flow from v_i to $\widehat{v_i}$. $pec(f_{i\hat{i}})$ is the total penalty cost of the dispatch action combinations at v_i and all its children that result in the power flow value $f_{i\hat{i}}$. Every $PfPc$ is checked and deleted if an alternative $PfPc$ exists with the same $f_{i\hat{i}}$ but smaller $pec(f_{i\hat{i}})$. Thus, the remaining $PfPc$ messages are those that record the minimum penalty cost that can be achieved for the specific $f_{i\hat{i}}$'s, which are then formed into an array $Toparent_{i\to\hat{i}}$ defined as:

$$Toparent_{i\to\hat{i}} = [PfPc_1, \ldots, PfPc_m]. \tag{5.31}$$

Furthermore, each $PfPc$ in the $Toparent_{i\to\hat{i}}$ maps to a $LinkToPfDstate$, which describes the dispatch actions of EV batteries and RGs at v_i and the $PfPc$ messages of all its children that result in the total penalty cost described in the corresponding $PfPc$ message. Due to the different properties of nodes, there is a slight difference between the ways of construction of their $PfPc$ messages and $LinkToPfDstate$, which is explained below.

As leaf nodes have no child nodes, they only need to consider their own EV battery and RG dispatch actions when constructing their $PfPc$ messages. For each possible combination of dispatch actions at a leaf node v_i, a $PfPc$ message is constructed with power flow $f_{i\hat{i}}$ calculated as:

$$f_{i\hat{i}} = -load_{tot} = -load_{fix} - load_{ev}^{c} + load_{ev}^{d} + P_{RG}. \tag{5.32}$$

The corresponding penalty cost is then calculated by (5.22), if v_i has connections to both EVs and RGs, or (5.23), if v_i only has EVs connected to it. As discussed earlier, some $PfPc$ messages are filtered out due to alternative $PfPc$ messages available with the same $f_{i\hat{i}}$'s but lower penalty cost $pec(f_{i\hat{i}})$'s. Furthermore, each remaining $PfPc$ message maps to a $LinkToPfDstate$ which records the corresponding EV battery and RG dispatch actions.

For a node v_j that has at least one child node, all the $Toparent$ arrays that it receives from its children $chi(v_j)$ are considered along with its own EV and RG dispatch actions to compute its own $Toparent_{j\to\hat{j}}$ and construct the corresponding $LinkToPfDstate$. For each possible combination of the dispatch actions of the EV batteries and RGs at v_i along with every possible combination of the $PfPc$ messages received from its children (with one from each child's $Toparent$ array), the power flow $f_{j\hat{j}}$ is calculated as:

$$\begin{aligned} f_{j\hat{j}} &= -load_{tot} + \sum_{c\in chi(v_j)} f_{cj} \\ &= -load_{fix} - load_{ev}^{c} + load_{ev}^{d} + P_{RG} + \sum_{c\in chi(v_j)} f_{cj}, \end{aligned} \tag{5.33}$$

where $\sum_{c\in chi(v_j)} f_{cj}$ is the sum of power flows recorded in the chosen $PfPc$ messages from each of v_j's children. For each resultant power flow $f_{j\hat{j}}$, the minimum penalty cost that can be realized is thus:

$$\min_{f_{j\hat{j}}} pec(f_{j\hat{j}}), \tag{5.34}$$

where $pec(f_{j\hat{j}})$ is defined by:

$$pec(f_{j\hat{j}}) = U_j + \sum_{c\in chi(v_j)} pec(f_{cj}), \tag{5.35}$$

where $\sum_{c\in chi(v_j)} pec(f_{cj})$ is the sum of penalty costs calculated in the chosen $PfPc$ messages from each of v_j's children. U_j is the penalty cost calculated at v_j for a chosen combination of EV battery and RG dispatch actions, by using either (5.22) or (5.23) depending on whether EVs or RGs are connected to v_j. Each $PfPc$ message maps to a $LinkToPfDstate$, which also records the corresponding EV battery and RG dispatch actions at v_j as well as the associated combination of $PfPc$ messages, one from each child.

Phase 2 — Value Propagation:

Once Phase 1 has been completed, and the root node has received the $Toparent$ arrays from all of its children, its agent starts to examine every possible combination of the $PfPc$ messages, one from each $Toparent$ array, in terms of whether the demand can be balanced with the supply and which combination minimizes the total penalty cost. A combination is considered to be feasible if the required power flow from the root node to satisfy all the corresponding loads within the network is within its given feasible domain. Therefore, the feasible combination that minimizes the total penalty cost is selected as the optimum state of the network. Then, the power flow values from each of its $PfPc$ messages are sent to all of the root node's children, telling them which of their $PfPc$ messages minimize the total penalty cost. The child retrieves the correct $PfPc$ message that has the same power flow value as that received from the root node. Thus, its corresponding $LinkToPfDstate$ specifies the optimal way to dispatch the EV batteries and RGs at this node, as well as the $PfPc$ messages whose power flow values need to be sent to the corresponding children of this node. By iterating this procedure, the power flow values are propagated to the leaf nodes at the end. Thus, all nodes in the network know in which way their EV batteries and RGs should be dispatched to minimize the total penalty cost and best satisfy the objectives.

The operating procedure of DPDOD is concretely illustrated, taking a certain node's agent as an example, in Figure 5.6.

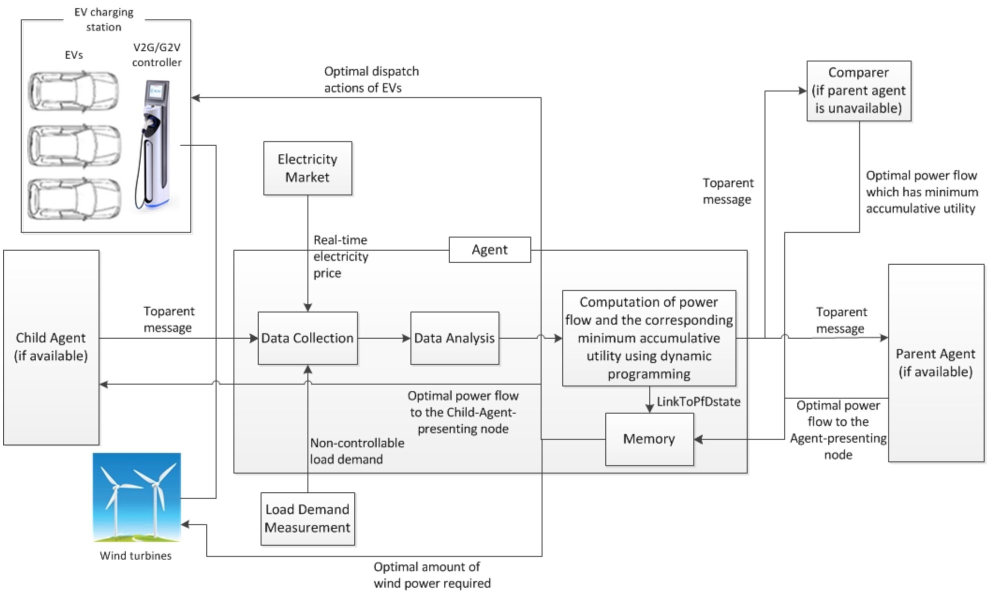


FIGURE 5.6: The Operation of DPDOD at A Certain Node's Agent

5.3 Complexity Discussion

The pseudo codes for the construction of the $Toparent$ array at a leaf node and a node that has children nodes are presented below. The $\prod$ in the codes represents the Cartesian product.

Algorithm 1: Construct $Toparent$ array at a leaf node v_i

1. $DM_i := \prod_{ev \in EV_i} DM_i^{ev};$
2. $for\ each\ p_i\ in\ RG_i\ \{$
3. $\quad for\ each\ \boldsymbol{\delta_i}\ in\ DM_i\ \{$
4. $\quad\quad load_{ev} := EVpowercal(\boldsymbol{\delta_i}, \boldsymbol{SOC_i});$
5. $\quad\quad f_{i\hat{i}} := -load_{fix} - load_{ev} + p_i;$
6. $\quad\quad pec(f_{i\hat{i}}) := U(p_i, \boldsymbol{\delta_i});$
7. $\quad\quad if\ (\min_{f_{i\hat{i}}} pec(f_{i\hat{i}}))\{$
8. $\quad\quad\quad PfPc(f_{i\hat{i}}, pec(f_{i\hat{i}}));$
9. $\quad\quad\quad LinkToPfDstate(PfPc, p_i, \boldsymbol{\delta_i});$
10. $\quad\quad \}$
11. $\quad \}$
12. $\}$
13. $Toparent();$
14. $Send\ Toparent\ array\ to\ parent\ node\ \widehat{v_i};$

Algorithm 2: Construct $Toparent$ array at a node v_i that has children nodes

1. $DM_i := \prod_{ev \in EV_i} DM_i^{ev};$
2. $ChildComToparent := \prod_{c \in chi(v_i)} Toparent_{c \to i};$
3. $for\ each\ p_i\ in\ RG_i\ \{$
4. $\quad for\ each\ \boldsymbol{\delta_i}\ in\ DM_i\ \{$
5. $\quad\quad for\ each\ \boldsymbol{ChiPfPc}\ in\ ChildComToparent\{$
6. $\quad\quad\quad load_{ev} := EVpowercal(\boldsymbol{\delta_i}, \boldsymbol{SOC_i});$
7. $\quad\quad\quad f_{i\hat{i}} := -load_{fix} - load_{ev} + p_i + \sum_{c \in chi(v_i)} f_{ci};$
8. $\quad\quad\quad pec(f_{i\hat{i}}) := U(p_i, \boldsymbol{\delta_i}) + \sum_{c \in chi(v_i)} pec(f_{ci});$
9. $\quad\quad\quad if\ (\min_{f_{i\hat{i}}} pec(f_{i\hat{i}}))\{$
10. $\quad\quad\quad\quad PfPc(f_{i\hat{i}}, pec(f_{i\hat{i}}));$
11. $\quad\quad\quad\quad LinkToPfDstate(PfPc, p_i, \boldsymbol{\delta_i}, \boldsymbol{ChiPfPc});$
12. $\quad\quad\quad \}$
13. $\quad\quad \}$
14. $\quad \}$
15. $\}$
16. $Toparent();$
17. $Send\ Toparent\ array\ to\ parent\ node\ \widehat{v_i};$

According to algorithm 1 (lines 1–3), the computational complexity grows linearly at a leaf node with the increase of its RG maximum power output and exponentially with the number of EVs it connects in $O(N_p 7^{N_{ev}})$, because it needs to iterate through all states in the Cartesian products of its own RG power output values and EVs' dispatch actions (each EV has 7 possible dispatch actions, i.e. charge/discharge at high, middle and low currents and idle, thus N_{ev} EVs have $7^{N_{ev}}$ possible dispatch actions in total). However, at a node with children nodes, it needs to iterate through all states in the Cartesian products of its own RG power output values, its EVs' dispatch actions and all of its children's states, that is, the computational complexity at its agent also grows exponentially with the number of its children nodes, hence in $O(N_p 7^{N_{ev}} M^{N_{chi}})$, as presented in algorithm 2, line 5. N_p, N_{ev} and N_{chi} are the number of discrete RG power output values, the number of EVs connected at a node and the number of children nodes a node has, respectively. M is the number of states a child has. The total size of messages that are sent by the DPDOD increases linearly with the size of the network in $O(N_v)$, since it equals to the sum of the size of messages that each node creates and sends. N_v is the number of nodes in a network. Moreover, as stated in lines 7–9 in algorithm 1 and lines 9–11 in algorithm 2, the communication complexity is also in inverse proportion to how many states converge to the same state, which is due to the fundamental of this proposed algorithm of dynamic programming.

In contrast, as discussed in [98], the computational complexity of a centralized algorithm, like the simplex method, grows exponentially with the size of the network, because, unlike DPDOD, it doesn't take the network's topology into account. Therefore, with the increasing penetration of EVs and RGs and network expansion, it will quickly become infeasible for a centralized algorithm to solve an optimal dispatch problem.

5.4 Simulations

The proposed coordinated dispatch strategy of EV batteries and RGs was tested on the modified generic radial distribution network shown in Figure 6.18. In this distribution network, four renewable generators are located at nodes v_3, v_6, v_{11} and v_{12}, respectively. Moreover, 33000 EVs are assumed to exist in the network with each node, except v_0 and v_1, capable of connecting 3000 EVs to the grid at the same time. All the network data, including thermal capacity of distribution cables, are shown in Table 5.1. The fixed load at each node, in Table 5.2, are derived from UKGDS model [145]. The parameters of EV batteries are obtained from [20]. The simulation is implemented on a laptop with a Dual Core 2GHz CPU and 8GB RAM using MATLAB.

The total load demand and system selling/buying prices of electricity during a day were taken from [141].The total load demand is scaled down so that the peak demand during a day is 350 MW, which approximately represents the daily load demand in a typical UK regional distribution network, as shown in Figure 5.7.

Due to scarcity of information on the payments by/to EV users when the EVs are charged/discharged, the system selling/buying prices are adjusted based on domestic

Table 5.1: Thermal Capacity of Distribution Cables

Distribution Cable		Thermal Capacity (MVA)
From Node	**To Node**	
v_0	v_1	500
v_1	v_2	52
v_1	v_3	154
v_1	v_5	102
v_1	v_7	76
v_1	v_9	76
v_1	v_{11}	54
v_1	v_{12}	154
v_3	v_4	81
v_5	v_6	110
v_7	v_8	46
v_9	v_{10}	51

Table 5.2: Fixed Load at Each Node of the Distribution Network

Node	Fixed Active Load (MW)	Fixed Reactive Load (MVAR)
v_1	38.48	11.28
v_2	22.98	9.11
v_3	56.05	3.63
v_4	36.63	9.77
v_5	25.32	5.55
v_6	55.54	6.19
v_7	17.70	0
v_8	15.73	2.28
v_9	13.14	0
v_{10}	20.93	3.70
v_{11}	99.05	21.09
v_{12}	87.07	16.58

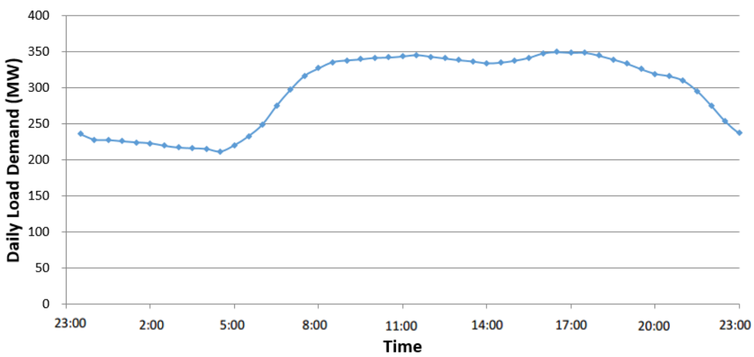


Figure 5.7: Total Fixed Load Demand During A Day

tariffs to represent EV users' selling/buying prices of electricity [20]. The adjusted system selling/buying price is derived as described in Chapter 4 and shown in Figure 4.6. The ASSP (i.e. adjusted system selling price) and ASBP (i.e. adjusted system buying price) in Figure 4.6 are respectively used as the EV charging and discharging prices , namely the *bp* and *sp* in *Objective 3*, respectively.

Furthermore, the renewable generators erected in this network are assumed to be wind turbines, and the daily weather forecast is assumed to be accurate enough. The historical wind power data recorded in [144] are used in the simulations. These daily wind power data are repeatedly utilized for the simulation of several successive days, as shown in Figure 5.8, to test the proposed strategy's performance and its stability in the same daily environment. In the future, the impacts of continuously varying environment on the dispatch system's performance should also be tested, as discussed in Chapter 6.

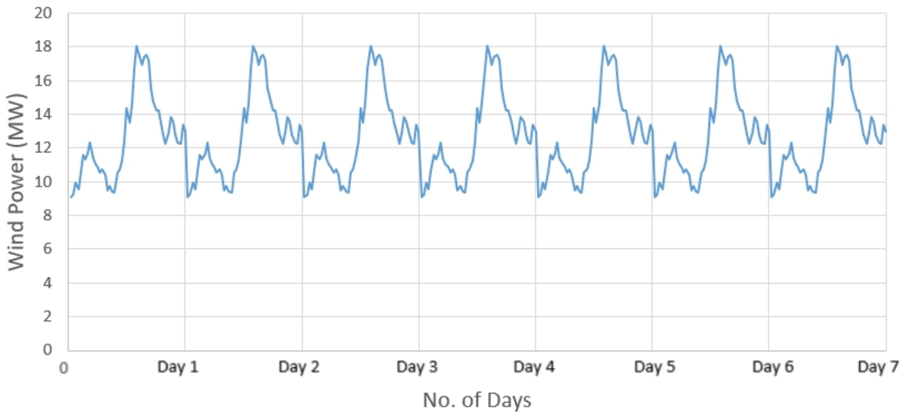


FIGURE 5.8: Repeated daily wind power generated in the network

As for the EVs' travel patterns, they are randomly generated based on the probability of parked cars during a weekday as shown in Figure 5.9. When the EV is on the road, the battery electric energy is assumed to be consumed at the nominal discharging current (20A) as assumed in [20]. Furthermore, the initial SOC of EVs are randomly assigned with a normal probability distribution ($\mu = 0.6, \sigma = 0.1$). The time step is set to be 30 minutes. The dispatch actions are determined by the dispatch strategy at the beginning of every time interval and lasts for the entire time interval of 30 minutes, as mentioned earlier.

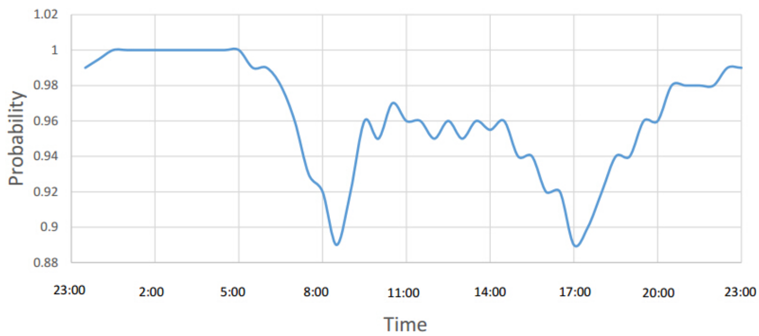


FIGURE 5.9: Probability of cars that are parked during a weekday

The simulation was run many times, each time starting with a different randomly assigned SOC, to determine the daily costs to EV users (i.e. charging costs–discharging payments) on average, as shown in Table 5.3. In the table, the average daily charging cost of 33000 EVs within the network is shown to be £12303 in total (by simply summing up the average daily cost at every node), thus the average daily cost of each EV is

TABLE 5.3: Daily costs of EVs calculated from simulations starting with different initial SOC

Daily Costs to EV Users (£)	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}
Average	3020	877	-309	1326	-83	1701	1944	1189	1408	873	357
Standard Deviation	77	38	136	54	35	40	101	67	30	87	65

calculated to be only £0.37, compared to a cost of £1.53 per day per vehicle when EV is charged in an uncontrolled way (based on simulations assuming that EVs are charged when their SOCs are less than 0.8 and renewable energy is utilized as much as possible without overloading the cables). However, the EVs parked at certain nodes tend to cost more than the EVs at the other nodes and the amount of costs/payments within the same node also varies (shown in Table 5.3 as non-zero standard deviations), depending on the driving patterns of EVs, their initial SOCs and local network constraints. Furthermore, the simulations confirmed that the dispatch strategy ensures that EVs can complete their daily journeys without running out of electricity on the road and always ensures that enough energy (over 31% of battery's available capacity) remains in their batteries, as demonstrated in Table 5.4. Figure 5.10 presents an EV's SOC variation during a day under the proposed dispatch strategy, which includes 2 driving activities from 8:30 to 9:30 and from 14:00 to 14:30.

TABLE 5.4: minimum and maximum SOCs of EVs during a day

Node	**Minimum SOC**	**Maximum SOC**	**Node**	**Minimum SOC**	**Maximum SOC**
v_2	0.37	1.00	v_3	0.36	1.00
v_4	0.40	1.00	v_5	0.41	1.00
v_6	0.43	1.00	v_7	0.33	0.99
v_8	0.33	1.00	v_9	0.31	1.00
v_{10}	0.33	0.99	v_{11}	0.44	1.00
v_{12}	0.42	0.99			

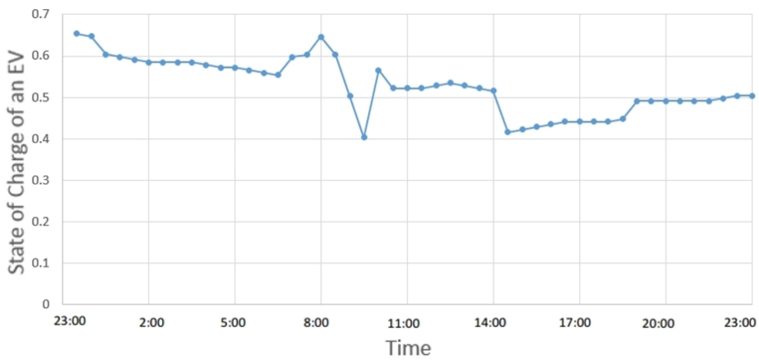


FIGURE 5.10: The variation of SOC of an EV at node 2

As for the renewable generators, one of the aims of this dispatch strategy is to utilize as much of their energy as possible, i.e., wind energy in this work. The simulation result shown in Figure 5.11 (Case 1, where the objectives' priorities are given in (5.22)) demonstrates consistency with this aim, i.e., the daily usage rate of wind power within the network is calculated to be 100%, with no wind power wasted. Compared to Case 1, if the relative priority of objective RE is lowered (i.e., Case 2 in Figure 5.11: priorities of RE, BS, CC, LL are 14.04%, 33%, 19.96% and 33% for example), 1.4% of wind power cannot be directly absorbed and utilized by the network if no extra energy storage device is located at the RG. This occurs because when the RE's priority is relatively low, the strategy will choose to save charging cost to EV users and achieve valley filling at the expense of abandoning some of the wind power and using less energy to charge EVs, as expected.

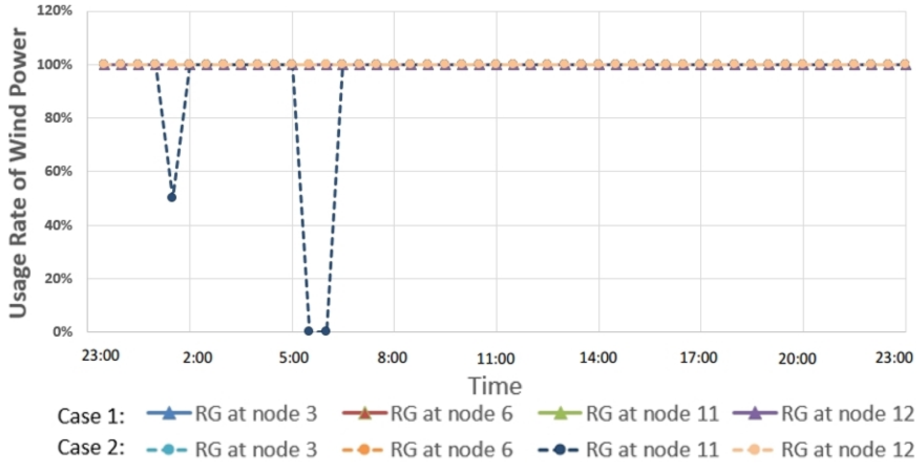


FIGURE 5.11: The usage rate of wind power at each renewable generator (Case 1: Relatively high priority is given to the objective RE, i.e. the case where priority settings are as demonstrated in (5.20) and (5.21). Case 2: Relatively low priority is given to the objective RE.)

Moreover, the changes of network load demand due to the integration of EVs and RGs are demonstrated in Figure 5.12, where positive values refer to load decrease (peak shaving) while negative values imply load increase (valley filling). It is shown in Figure 5.12, that a significant load levelling is fulfilled by the coordinated dispatch strategy, in comparison with uncontrolled dispatch of EVs and RGs. From this figure, it is clear to see that peak shaving (EV load=-14.2MW, RG output=13.5MW on average) and valley filling (EV load=28.5MW, RG output=11.2MW on average) for the distribution network's daily load demand has been realized by the coordinated integration of EVs and RGs. Coordinated dispatch results in a better performance than an uncontrolled one.

To verify the stability of the proposed coordinated dispatch strategy, the simulation described above continues running to check EVs and RGs' performance for 7 days. The 7-day load curves of the network with the integration of EVs and RGs are shown in Figure 5.13, from which it is obvious that load levelling is well achieved and the load curves for the 7 days approximately overlay each other, meaning that the dispatch system

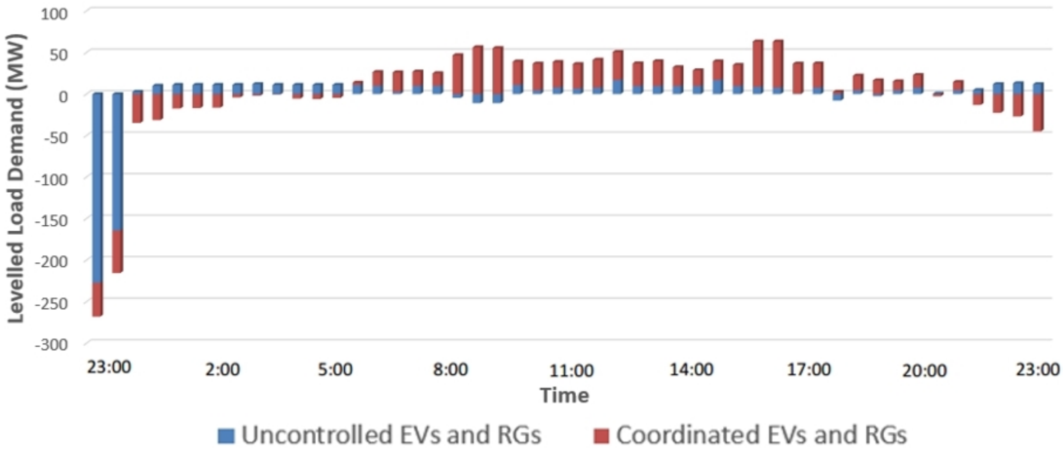


FIGURE 5.12: The change of total network's daily load demand that is caused by the integration of EVs and RGs in 2 different ways (i.e. uncontrolled and coordinated ways, respectively)

is very stable. The EV SOCs at the end of a day are checked as well to see whether they remain in a similar probability distribution curve from day to day. As shown in Figure 5.14, from Day 1 to 7, the EVs' SOC distributions are similar with most of the EVs' SOCs in the range from 0.5 to 0.6 and the rest distributed within the range from 0.6 to 1.0. Moreover, when EVs start with different assignments of initial SOCs, they will stabilise at a very similar SOC distribution at the end of every day with a mean $\mu = 0.56$ on average and a standard deviation $\sigma = 0.15$ on average, as shown in Table 5.5.

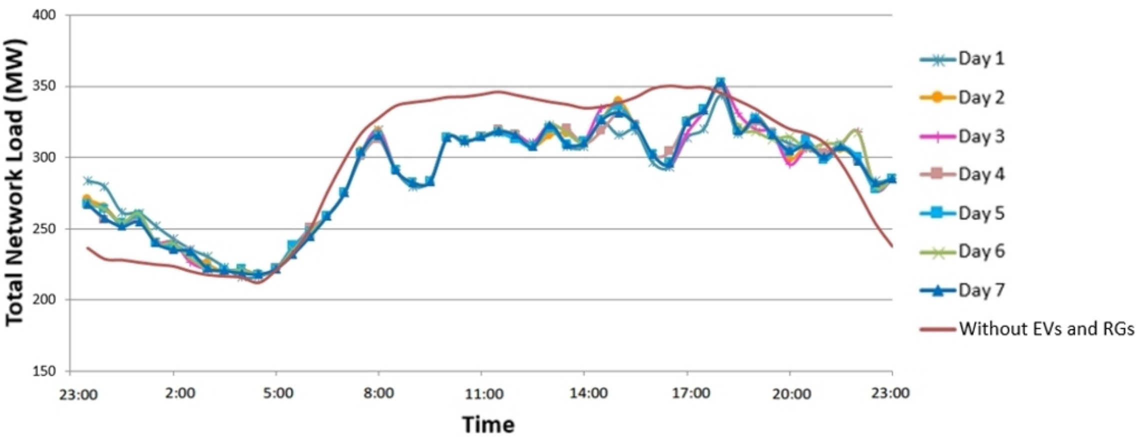


FIGURE 5.13: The load demand of the network within a week

5.5 Conclusion

This chapter has proposed a novel coordinated dispatch strategy for EVs and RGs within a distribution network, taking into account the requirements of both EV users and the grid, such that the EV batteries can be dispatched in coordination with each

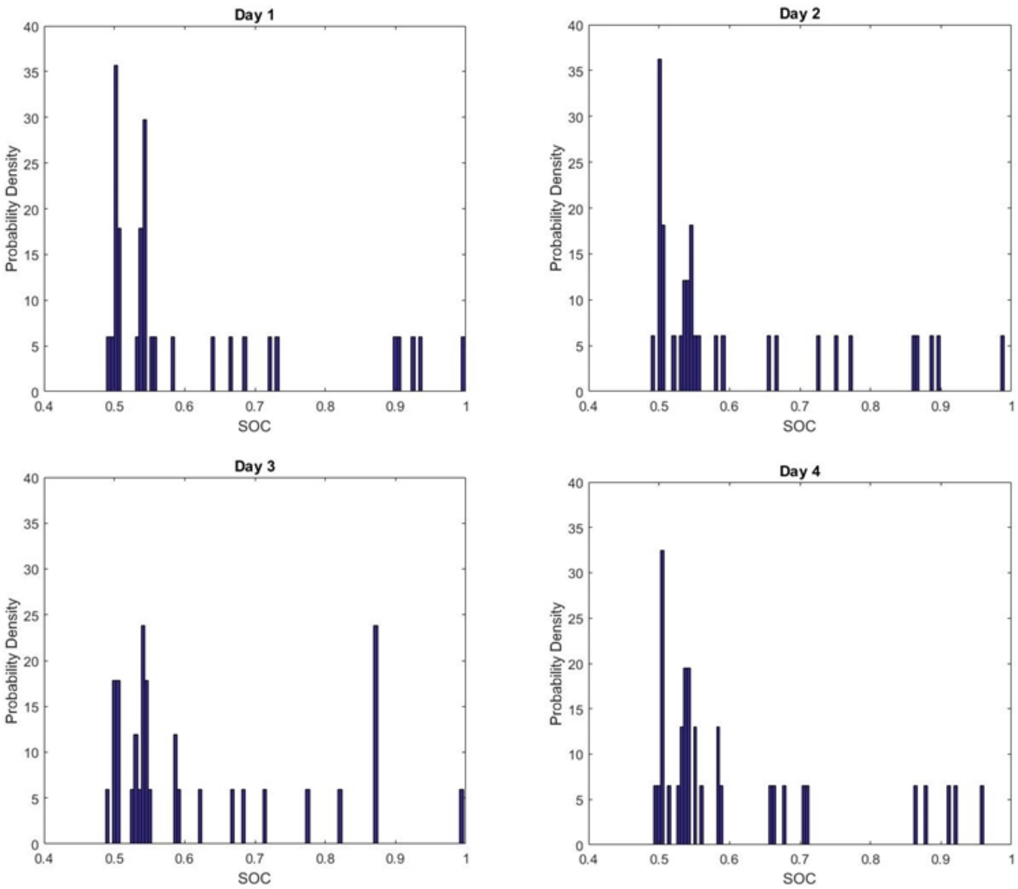
TABLE 5.5: Key factors of EVs' SOC distribution at the end of a day tested with different mean values of the random assignments of the initial SOC

Initial SOC assignment	**Mean of SOCs**		**Standard deviation of SOCs**	
	Mean	Standard deviation	Mean	Standard deviation
$\boldsymbol{\mu = 0.5, \sigma = 0.1}$	0.57	0.004	0.15	0.008
$\boldsymbol{\mu = 0.6, \sigma = 0.1}$	0.56	0.004	0.15	0.006
$\boldsymbol{\mu = 0.7, \sigma = 0.1}$	0.56	0.006	0.15	0.006

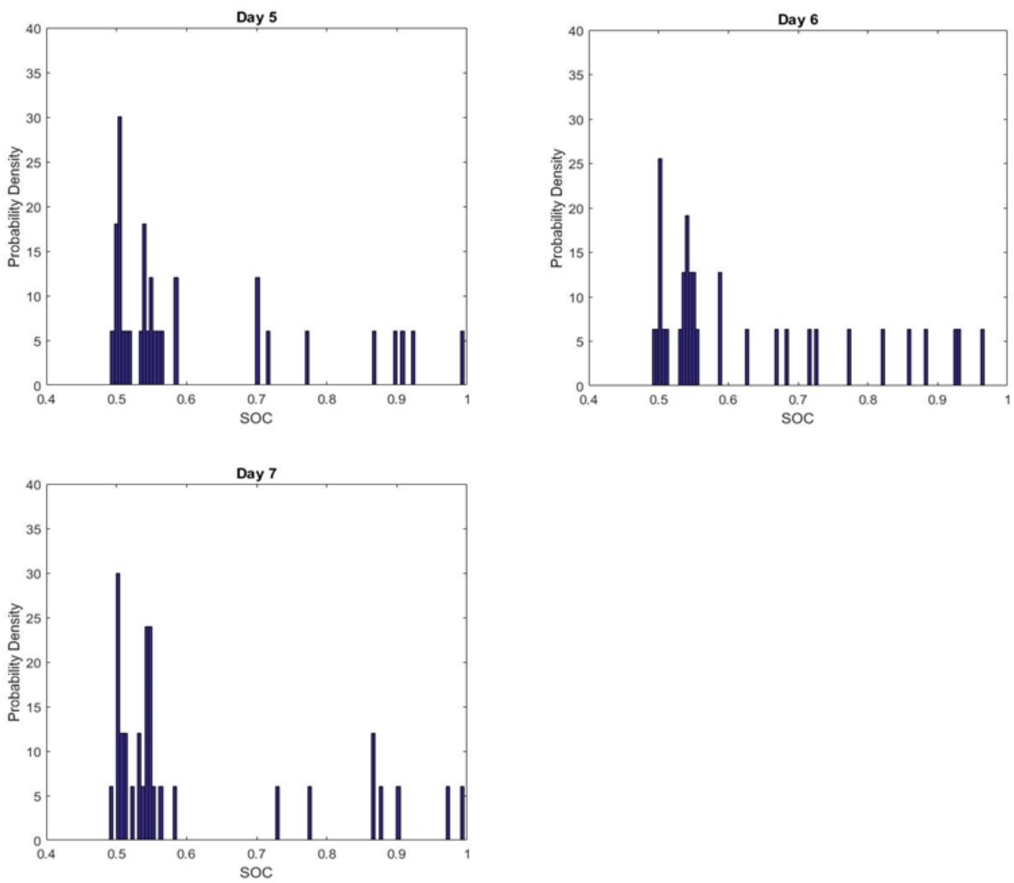
other and with RGs to provide support to the electricity network for load levelling while saving cost to EV users, ensuring reliable driving experience with sufficient SOC left in the EV battery, and reducing the waste of renewable energy. This multi-objective optimal dispatch problem has been formulated as DMOCOP, which is developed from DCOP using AHP. The electricity network model is simplified using virtual sub-node concept when it has a great number of EVs and RGs and a node that are connected by many children nodes. The problem is then solved using a dynamic-programming-based algorithm DPDOD to derive an optimal combination of the dispatch actions of EVs and RGs. The simulation results have verified the feasibility and stability of the proposed coordinated dispatch strategy. It only costs £0.37 per vehicle per day averagely while all journeys are completed with at least 31% electricity remaining in the EV batteries. Because the EVs connected at different nodes have different travel patterns and initial SOCs, those parked at certain nodes may tend to cost more than the others. Moreover, the cost/payment to the same set of EVs may also varies from day to day, because of the different start-up SOCs of the day. 100% of the wind power generated is directly injected into the network. EVs and RGs cooperate to well achieve the load levelling of the network's load demand. The performance of EVs and RGs under the proposed coordinated dispatch strategy has been proved via simulation to be continually stable. Furthermore, the distribution of EVs' daily end-up SOCs is very similar from day to day, even with different initial SOCs assigned, which also verifies the stability of the proposed dispatch strategy.

In this work, the proposed dynamic-programming-based algorithm discretize the dispatch variables, i.e., EV dispatch currents and RG power outputs, to simplify the solution process and save computational cost. However, the optimal solution could be compromised compared with the centralized continuous optimization result, as the discretization of dispatch variables is likely to miss the global optimum.

In this work, the optimal dispatch problem is solved using dynamic programming algorithm, which under certain circumstances tends to do the computations that are actually superfluous thus increases the cost of solution. In the next chapter, a new algorithm is introduced which conceptually will solve this problem. The comparisons between these two algorithms are also made.



(a)



(b)

FIGURE 5.14: EVs' SOC distribution curves at the end of every day

Chapter 6

Decentralized Dispatch of A Distribution Network Using A*

The previous chapter proposed an agent-based optimal dispatch strategy of EVs and RGs that used a dynamic-programming-based algorithm to find the optimal solution. However, as mentioned at the end of last chapter, this algorithm might make certain redundant computations that could be omitted to save time and cost. In this chapter, a new algorithm based on A search is introduced for the energy management of a distribution network. The applications of two algorithms in a relatively simple case — the optimal dispatch of distributed generators is discussed. In this chapter, a distributed constrained optimization problem is formulated with the aim of minimizing carbon emission and then solved using an A*-based algorithm and a dynamic programming algorithm, respectively. The two approaches are tested on an example distribution network and compared to discuss the differences between them and the potential advantages of the proposed A*-based optimal dispatch algorithm. The application of A*-based approach to the optimal coordination of EVs and RGs is also made and discussed in terms of its feasibility and efficacy.*

6.1 Introduction

In this chapter, a novel decentralized optimal dispatch approach based on the A* search procedure [148] is proposed for the energy management of a radial distribution network. The optimal dispatch problem is first formulated as an agent-based distributed constrained optimisation problem (DCOP) based on [98] and then solved using the A* search procedure. Each node is managed by an agent, which is aware of the elements that are locally connected to its designated node and their properties. The agent is also in charge of a part of the computation and communication that are required to solve the optimization problem using the A* search procedure and dispatches its local elements according to the final optimal solution, such that the network's load demand is just met by the power supply without overloading any cable and the system's optimization objective is achieved.

The A* search procedure is enhanced to increase its speed in order to better serve this sort of optimal dispatch problem. The efficacy of the approach is tested using a 10-node radial distribution network. The DCOP is also solved with the dynamic programming decentralized optimal dispatch (DYDOP) algorithm published in [98] for comparisons and to verify the efficacy and potential advantages of the proposed A*-based algorithm.

The application of the A*-based algorithm in a more complicated problem — the decentralized coordination of RGs and EVs within a distribution network is also investigated. This optimal dispatch problem is formulated in Chapter 5 as a distributed multi-objective constraint optimization problem (DMOCOP) [94], which is developed from DCOP and based on agents. The objectives of the optimal coordination of EVs and RGs satisfy both EV users and grids' concerns and requirements, including EV charging cost saving, sufficient energy throughout a day to support any necessary journey, stability of the network without overloading any cable and network's load levelling. Moreover, the inherent uncertainty of EV travel patterns and renewable power generation is modelled and simulated using Gaussian copulas, so that the correlation between each pair of random variables are taken into account to better reflect the characteristics of the real case. The proposed A*-based optimal dispatch algorithm is tested on a radial distribution network, a modified UKGDS, for its stability, feasibility and efficacy at satisfying the requirements of both EV users and the grid. In practice, the aggregator or the distribution network operator (DNO) is supposed to be in charge of this optimal dispatch problem.

The rest of the chapter is organized as follows: A general model of the electricity network that is used for optimal dispatch is described in Section 6.2. The optimal dispatch problem is formulated as DCOP in Section 6.3. Section 6.4 illustrates the procedure of using the A* to solve DCOP. The case study of applying the A*-based dispatch approach to DG dispatch is presented in Section 6.5. In Section 6.6, the decentralized coordination of RGs and EVs using A* algorithm is discussed. The results of simulations using MATLAB to verify its feasibility and efficacy, are presented and discussed in both Section 6.5 and 6.6. Finally, the conclusions of the work are presented in Section 6.7.

This work has been written as two conference papers, one of which has been presented at IEEE POWERCON 2016 [149] and another one will be presented at the 26th IEEE ISIE 2017, and two journal papers, which are under review.

6.2 A General Model of Electricity Network and Agents

A network contains a set of n controllable items, either load or supply, which can be RGs or EVs. Their control variables, such as the power outputs of RGs and the dispatch modes of EVs, are represented by $\mathbf{X} = \{x_1, \ldots, x_n\}$.

$V = \{v_1, \ldots, v_k\}$ denotes the set of nodes within the network. A node v_i exchanges power with other nodes and contains a combination of uncontrollable and controllable loads and/or power supplies. $adj(v_i)$ represents a set of adjacent nodes of v_i, i.e., nodes that are directly connected to v_i via distribution cables, including its children nodes $chi(v_i)$ and its parent node $par(v_i)$. $load^i_{fix}$ represents the fixed load at v_i.

The set of distribution cables in the network is denoted by $\mathbf{T}$ and t_{ij} refers to the distribution cable between nodes i and j. The power flow along the cable t_{ij} is denoted by f_{ij}, which cannot exceed the thermal capacity of the cable, C_{ij}, and $f_{ij} = -f_{ji}$. In this work, it is reasonable to assume that the capacity of the parent cable is no less than the sum of the children's cable capacities, i.e. $C_{i\hat{i}} \geqslant \sum\limits_{c \in chi(v_i)} C_{ci}$, where $\widehat{v_i}$ is the parent node of v_i.

Furthermore, each node, v_i, is managed by an agent, which only has knowledge of the locally connected elements, and sends dispatch commands to those under its control. They work in a decentralized way to undertake the computation and communication that are required to solve the optimization problem. Each agent has a utility function. In the case study of DG dispatch, the utility function maps to the carbon emission at the designated node, as an example; the utility function can also be formulated to describe cost, load variance etc. to fulfil another objective or multiple objectives. Based on the above definitions, this optimal coordinated dispatch problem can be formulated as a distributed constraint optimisation problem (DCOP) based on [98]. The 10-node electricity network presented in Figure 6.1 will be used as a case study example.

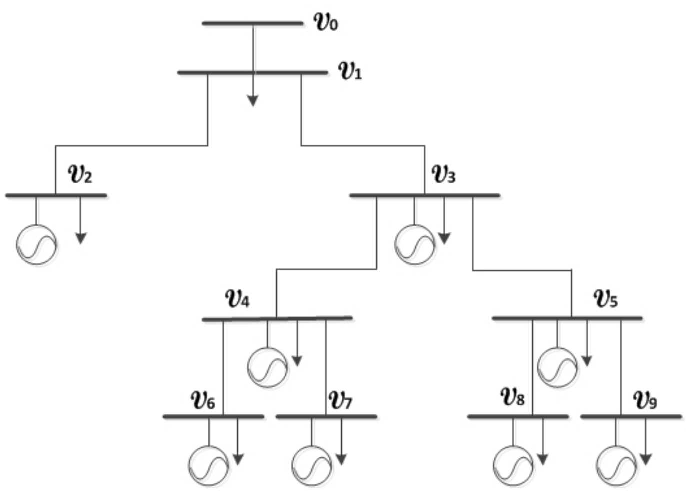


FIGURE 6.1: Diagram of a 10-node radial distribution network

6.3 Distributed Constraint Optimization Problem (DCOP)

A set of h control variables of DCOP is $X = \{x_1, \ldots, x_h\}$. x_i refers to the control variable of ith controllable player in the distribution network. Each variable, x_i, has a feasible domain, d_i, that includes all the possible values of x_i. Hence, a set of domains are denoted by $D = \{d_1, \ldots, d_h\}$. As discussed above, each node, v_i, maps to an agent, which has a utility function to compute the corresponding utility value U_i. Therefore, the objective of DCOP is to find an assignment of control variables X^* that optimizes the sum of utilities of all agents within the network:

$$\arg\min_{X^*} \sum_{i=0}^{k} U_i \tag{6.1}$$

subjected to the following constraints:

1. The sum of power flow into a node v_i should be equal to the sum of power flow out:

$$\sum_{j \in adj(v_i)} f_{ij} + load^i_{fix} + load^i_{con} - p^i_{con} = 0, \tag{6.2}$$

where $load^i_{con}$ and p^i_{con} are the controllable load and power supply at node v_i which are determined by v_i's control variables X_{v_i}.

2. The power flow along a distribution cable should not exceed its capacity:

$$|f_{ij}| \leqslant C_{ij}, \tag{6.3}$$

where C_{ij} is the thermal capacity of the distribution cable between nodes v_i and v_j.

3. The control variables should be assigned with values within their feasible domains:

$$x_i \in d_i \tag{6.4}$$

6.4 A* Optimal Dispatch Procedure

In order to solve the DCOP defined earlier, A* is used to find the optimal assignment of the control variables X^* in the network.

A* is a well known optimal path search procedure, discovering the shortest route between a starting point and a destination. A simple example is as follows:

Figure 6.2 describes a route map from city A to city D (numbers present the distance between two cities), and the objective is to find the shortest path from A to D. Starting from city A, two paths can be expanded: $A - B$ (A to B) and $A - C$ (A to C) with path lengths of 10 and 5, respectively. Thus, two states can be formed: $\{A - B, 10\}$ and $\{A - C, 5\}$, which comprise a state queue, Q, in which the states are lined up in an ascending order of path length:

$$Q = [\{A - C, 5\}, \{A - B, 10\}]. \tag{6.5}$$

Only the first state of the queue is extended: $\{A - C - B, 20\}$ and $\{A - C - D, 17\}$. These two new states replace the old state $\{A - C, 5\}$ in the queue and the queue is sorted again with the shortest patch in the front. The state queue Q now becomes:

$$Q = [\{A - B, 10\}, \{A - C - D, 17\}, \{A - C - B, 20\}] \tag{6.6}$$

Again, the first state of the queue is extended: $\{A - B - C, 25\}$ and $\{A - B - D, 16\}$, which replace the old state in the queue and the states in the queue are reordered. The new state queue Q becomes:

$$Q = [\{A - B - D, 16\}, \{A - C - D, 17\}, \{A - C - B, 20\}, \{A - B - C, 25\}] \tag{6.7}$$

Now, the first state, which has the shortest length, terminates at the destination D. The A* search procedure ends, and the optimal path is $A - B - D$ with a length of 16.

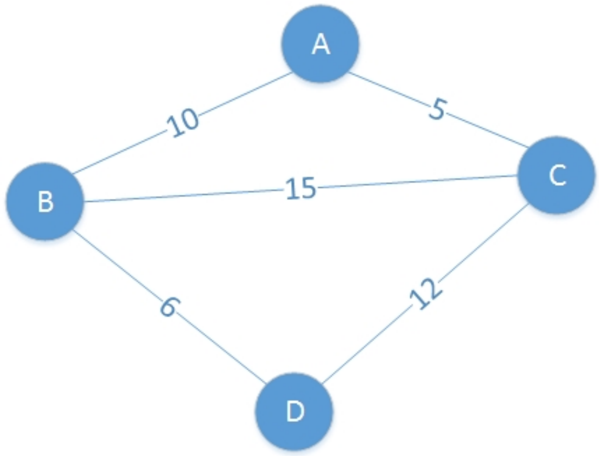


FIGURE 6.2: A route map

The procedure is also illustrated graphically in Figure 6.3. We refer the interested reader to [148] for a detailed description of A* search procedure.

6.4.1 Conventional A* Procedure

In this section we describe how A* is applied to our specific case of optimal dispatch of DGs in a distribution network.

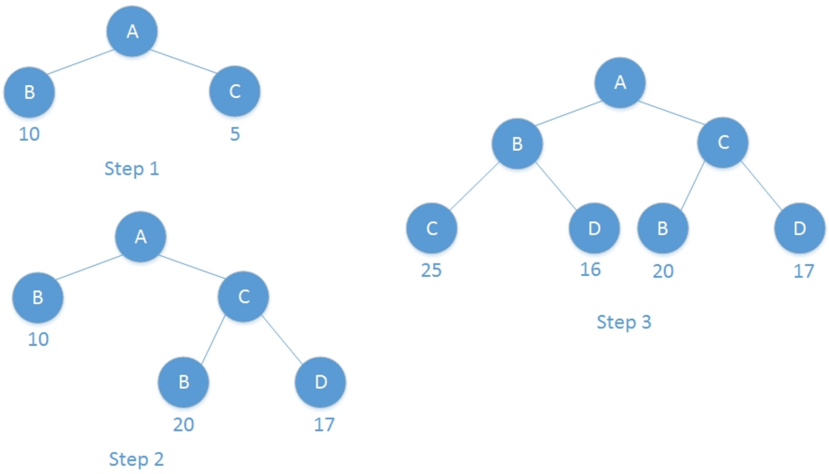


FIGURE 6.3: Optimal path searching using A*

This problem can be interpreted by comparison with the optimal path search problem described earlier. As an electricity network has more than one leaf node, this problem is like an optimal path search problem with multiple starting points. The path length in the path search problem maps to the accumulated utilities in this optimal dispatch problem, as they are respectively the objective function of each problem that needs to be minimized. The path maps to the power flow from a node to its parent node. Both path and power flow are extendible and their extensions are dependent on the current states: given power flow from a node v_i to its parent node $\widehat{v_i}$, the power flow from node $\widehat{v_i}$ to $\widehat{v_i}$'s parent node can be calculated.

The A*-based optimal dispatch of a distribution network can be applied as follows:

Each node is managed by an agent, as discussed earlier. The computational results of the agents are stored and managed in a central memory for the future usage in the A* search procedure. Figure 6.4 illustrates a thorough procedure of A*-based dispatch. The communication of central memory and agents is illustrated in Figure 6.5.

The A* terminology used in these figures is as follows:

- State and State Queue: A state S_j in a state queue Q_i consists of two elements the power flow value $f_{i\hat{i}}$ from the node v_i to its parent node $\widehat{v_i}$ and the corresponding accumulated utility $U_{\sum}(f_{i\hat{i}})$ that are resulted from a certain given assignment of control variables $X_{v_i}(S_j)$ at this node v_i and $X_{v_c}(S_j)$ at its children nodes $chi(v_i)$. Therefore,

$$S_j = \{f_{i\hat{i}}, U_{\sum}(f_{i\hat{i}})\}, \tag{6.8}$$

where $|f_{i\hat{i}}| \leqslant C_{i\hat{i}}$ for the stability of power system. Each state S_j maps to a $Pstate$ which records the values of control variables at v_i and its children nodes that

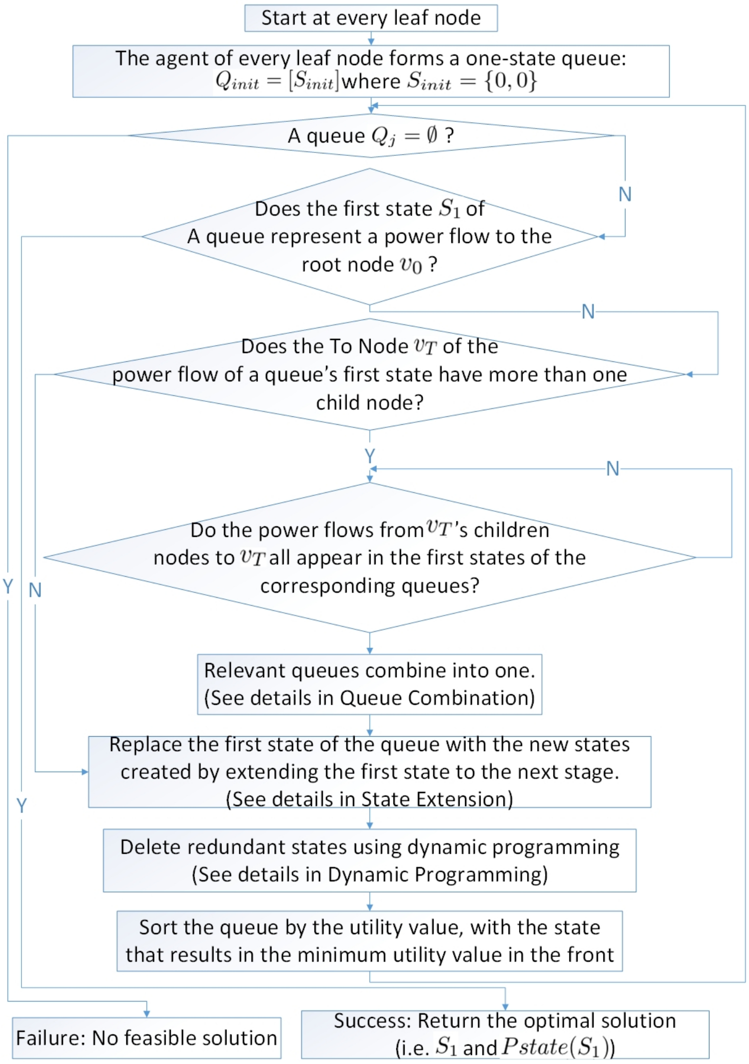


FIGURE 6.4: The A*-based dispatch algorithm

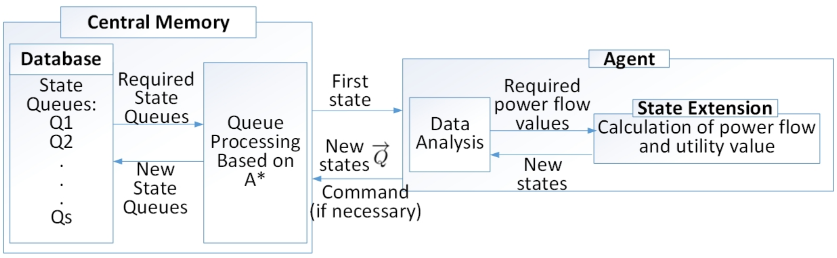


FIGURE 6.5: A general graph of data communication within A*-based optimal dispatch

results in the accumulated utility described by the function $U_{\sum}$:

$$Pstate(S_j) = \{S_j, X(S_j)\}, \tag{6.9}$$

where $X(S_j) = \{X_{v_i}(S_j), X_{v_c}(S_j) | v_c \in chi(v_i)\}$.

A state queue Q_i consists of M states that are generated from the corresponding agents and lined up in an ascending order of the utility value $U_{\sum}$:

$$Q_i = [S_1, S_2 \ldots, S_M] \tag{6.10}$$

At some points during the A* search procedure (e.g. after queue combination), a state may contain more than one power flow value, associated with different distribution cables, and its $Pstate$ develops along with it, as will be explained in the following context.

- State Extension: Every time, the central memory will only send the first state S_1 of a queue to the To node v_i of its power flow(s) f_{ci} for state extension. To extend a state, which contains the power flow(s) f_{ci} from v_i's child(ren) node(s) $v_c \in chi(v_i)$ to itself, all the possible power flow values $f_{i\hat{i}}$ from v_i to its parent node $\widehat{v_i}$ that satisfy the capacity constraint (6.3) and the associated accumulated utility $U_{\sum}(f_{i\hat{i}})$ are evaluated, based on the state S_1 and all the possible assignments of the control variables X_{v_i} at node v_i:

$$f_{i\hat{i}} = \sum_{v_c \in chi(v_i)} f_{ci} + p^i_{con} - load^i_{con} - load^i_{fix}, \tag{6.11}$$

$$U_{\sum}(f_{i\hat{i}}) = U_{\sum current} + U_i(X_{v_i}), \tag{6.12}$$

where $\sum_{v_c \in chi(v_i)} f_{ci}$ and $U_{\sum current}$ are given in the current first state of the queue S_1. $U_i(X_{v_i}$ is calculated using the utility function at node v_i given a certain assignment of the control variables X_{v_i} at node v_i.

The agent at v_i will then send the new states to the central memory in the form of an array $\overrightarrow{Q}$:

$$\overrightarrow{Q} = [Sp_1, Sp_2 \ldots, Sp_m], \tag{6.13}$$

where $\overrightarrow{Q}$ is a new state message array, which is an array of m Sp messages. A Sp_j message is formed of two elements: a new state $Snew$ and the assignment of the control variables X_{v_i} at node v_i that results in the power flow of $f_{i\hat{i}}$ and the associated utility $U_{\sum}(f_{i\hat{i}})$ of $Snew$, as follows:

$$Sp_j = \{f_{i\hat{i}}, U_{\sum}(f_{i\hat{i}}), X_{v_i}\} = \{Snew_j, X_{v_i}(Snew_j)\}. \tag{6.14}$$

Once the $\overrightarrow{Q}$ is received at the central memory, the original first state S_1 of the queue is replaced by the new states that are created from it, i.e., $Snew_1, Snew_2, \ldots, Snew_m$

in $\overrightarrow{Q}$, and the corresponding $Pstate(Snew_j)$ is developed from the $Pstate(S_1)$ of S_1:

$$Pstate(Snew_j) = \{Snew_j, X(Snew_j)\}, \tag{6.15}$$

where $X(Snew_j) = \{X_{v_i}(Snew_j), X(S_1)\}$.

- Dynamic Programming: In A* procedure, according to dynamic programming, if there are more than one state representing the same power flow, only the state with minimum accumulated utility value remains while the others are redundant and deleted.

- Queue Combination: When the procedure proceeds to a point where the first state of a queue represents a power flow to a node v_i that has more than one child node, the extension of this state requires the state(s) from other queue(s) that contain(s) the power flow(s) from v_i's other child(ren) node(s) to v_i. Therefore, those involved queues need to be combined, but only after their first states all represents a power flow to the same node v_i. Combining state queues $Q_1, Q_2, \ldots, Q_s$ is to perform Cartesian product over those queues to form a new state queue Q_{comb}:

$$Q_{comb} = \{Q_1 \times Q_2 \times \cdots \times Q_s\}, \tag{6.16}$$

with the state within becoming

$$S_{comb} = \{(f_{vq_1\widehat{vq_1}}, f_{vq_2\widehat{vq_2}} \ldots f_{vq_s\widehat{vq_s}}), \sum_{x=1}^{s} U_{\sum}(f_{vq_x\widehat{vq_x}})\}, \tag{6.17}$$

where $f_{vq_x\widehat{vq_x}}$ is a power flow value presented by a state S_j^x in the queue $Q_x \in \{Q_1, Q_2, \ldots, Q_s\}$. And the $Pstate(S_{comb})$ of S_{comb}becomes:

$$Pstate(S_{comb}) = \{S_{comb}, X(S_{comb})\}, \tag{6.18}$$

where $X(S_{comb}) = \{X(S_\alpha^1), X(S_\beta^2), \ldots, X(S_\gamma^s)\}$.

As shown in Figure 6.4, the A* procedure starts from every leaf node with a initial state $S_{init} = \{0, 0\}$. At a leaf node v_i, the agent extends S_{init} using (6.11) and (6.12) to form new states. Figure 6.6 illustrates the state extension at a leaf node v_6 in Figure 6.1 for an example. These new states, together with the corresponding control variables' values, are sent to the central memory in the form of $\overrightarrow{Q}$ and stored in the database for the future usage in the A* procedure, as demonstrated in Figure 6.5. After $\overrightarrow{Q}$ is received at the central memory, the corresponding first state of the queue is replaced with the new states in the received $\overrightarrow{Q}$. The redundant states are deleted according to dynamic programming. The queue is then sorted in an ascending order of the accumulated utility values, as presented in Figure 6.4. Thus the new first state of the queue will have the minimum accumulated utility value so far and will be sent to the To Node of the state's power flow (e.g. node v_j of f_{ij}) for its agent to extend, as presented in Figure 6.5. Therefore, every time, in every queue the state with the least accumulated utility value will be extended to a set of new states, which will be sent back to the central memory and with the original state queue they will form into a new state queue according to A* procedure.

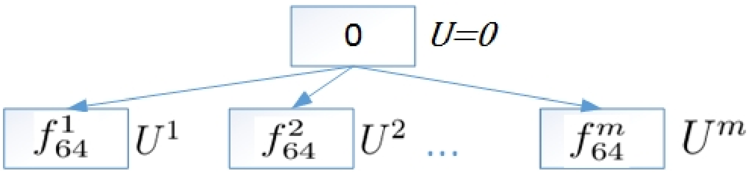


FIGURE 6.6: An illustration of the A* procedure at a leaf node v_6 in the distribution network shown in Figure 6.1

Queue combination will be conducted using (6.16) when the procedure reaches a stage where a queue's first state represents a power flow $f_{i\hat{i}}$ to a node $\widehat{v_i}$ that has more than one child node. The agent of node $\widehat{v_i}$ will wait till it receives from central memory the associated combined queue's first state $S_{1,comb}$ which contains all the power flow values from its children nodes $\{f_{c\hat{i}} | c \in chi(\widehat{v_i})\}$. It will then extend the received $S_{1,comb}$ to a set of new states using (6.11) and (6.12). The combined queue will then be updated with the new states, as stated in Figure 6.4 and Figure 6.5. Taking node v_4 in Figure 6.1 as an example, Figure 6.7 illustrates how the state queues for nodes v_6 and v_7 combine for the agent at node v_4 to extend their first states and create new states.

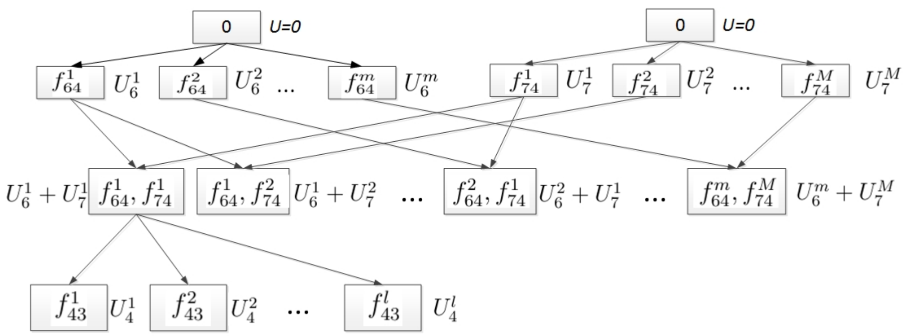


FIGURE 6.7: An illustration of state queues' combination and extension of the A* procedure at a node v_4 in the distribution network shown in Figure 6.1

The aforementioned A* search procedure will iterate until terminating at the root node of power network (i.e. node v_0 in Figure 6.1), at which stage the first state S_1 of a queue is the optimal solution that optimize the total utility of the network and thus best achieves the objective of dispatch. The optimal assignment of all control variables X^* within the power network are recorded in the corresponding $Pstate(S_1)$. The optimal dispatch solution is then returned and success is announced, as stated in Figure 6.4. However, if the procedure ends up with an empty queue, no feasible solution can be found for this optimal dispatch problem and thus failure is announced.

6.5 Case Study 1: The Application of A*-Based Approach in DG Dispatch in A Distribution Network

In this case, the control variable $X = \mathbf{p}$, and the utility function is defined to be the carbon emission of DGs:

$$U_i = CI_i p_i, \tag{6.19}$$

where CI_i is the carbon intensity of the DG at node v_i in units of kg CO_2e/kWh, which measures the carbon emission of a generator or power plant in both direct (i.e., fuel combustion) and indirect ways; a renewable generator has indirect carbon emissions, which occur during the manufacturing process. Therefore, the objective of this optimal dispatch problem is to find an assignment of X^*, which minimizes the total carbon emission within the network:

$$\arg\min_{X^*} \sum_{i=0}^{k} CI_i p_i \tag{6.20}$$

subjected to three constraints as defined earlier:

Constraint 1 (6.2) becomes:

$$\sum_{j \in adj(v_i)} f_{ij} + load_i - p_i = 0. \tag{6.21}$$

Constraint 2 (6.3) unchanged.

Constraint 3 (6.4) becomes:

$$0 \leqslant p_i \leqslant P_i^{max}. \tag{6.22}$$

6.5.1 Improved A* Procedure

With the definition above, the A* algorithm is customized to solve the DCOP of DG dispatch. In order to accelerate the process to better serve this sort of dispatch problem, some improvements can be made in the queue combination step while the rest of procedure remains the same as conventional A* procedure. Instead of the entire state queues, the Cartesian product will be performed on the first N_c states of every involved queue Q_c. The reason for doing so is that every time only the state with the least accumulated carbon emission in the queue is extended and the states in the front of a queue are most likely to be extended in the future and lead to the final optimal solution. However, those at the rear of a state queue cause the highest accumulated carbon emissions, which might even cover less nodes' generators than those in the front, and thus are very unlikely to be extended in the future and become the final optimal solution. Therefore, those states in the back of a queue can be abandoned during the procedure to simplify and accelerate the process.

There are two ways to determine N_c for every $Q_c \in \{Q_1, \ldots, Q_s\}$, a set of state queues that share the same To Node v_i of their first states' power flows:

1. Maximum Power Flow Method: The first state of a queue has the least carbon emission, which implies that the power output of the generators is assigned to its minimum possible and hence the power flow required through the cable from node v_i to its child node will be at its maximum. Therefore, the maximum power flow to v_i from its parent node $\widehat{v_i}$ would be

$$|f_{i\hat{i}}|_{max} = \sum_{c \in chi(v_i)} |f_{ci}| + load_i - (p_i)_{min}, \tag{6.23}$$

   where $\sum_{c \in chi(v_i)} |f_{ci}|$ is the summation of the absolute values of the power flows recorded by the first states of involved queues $Q_1, \ldots, Q_s$. $(p_i)_{min}$ denotes the minimum power output of the generator at node v_i. If $|f_{i\hat{i}}|_{max}$ is larger than the cable's capacity $C_{i\hat{i}}$ by a value of μ, where

$$\mu = |f_{i\hat{i}}|_{max} - C_{i\hat{i}}, \tag{6.24}$$

   then N_c is determined as the number of states that are listed before the state with a power flow of $(|f_{ci}| - \mu)$ to make sure that the states that have relatively high potential to become the final optimal solution proceed to further investigation. However, if $|f_{i\hat{i}}|_{max}$ is smaller than the cable's capacity $C_{i\hat{i}}$, then N_c is set to be 1, because in this case the network is not heavily constrained and the first states of involved queues will lead to the final optimal solution in the future.

2. Cable Capacity Method: When the power network is heavily constrained or loaded, the first states' power flows $|f_{ci}|$ of involved queues are very likely to be close to the cable's thermal capacity C_{ci}. Therefore, instead of using $(|f_{ci}| - \mu)$ to be the threshold as in the maximum power flow method, the difference between a child cable capacity and a new $\mu = \sum_{c \in chi(v_i)} C_{ci} + load_i - C_{i\hat{i}}$, $(C_{ci} - \mu)$ can be used as the threshold. In particular, when $C_{i\hat{i}}$ is close to the sum of children cables' capacities, $\sum_{c \in chi(v_i)} C_{ci}$, the threshold formula can be simplified to be $(C_{ci} - load_i)$, the difference between a child cable capacity and the local load. N_c is then determined by the number of states that are listed before the first state that has a power flow that is not larger than the threshold value.However, if $(C_{ci} - load_i) > |f_{ci}|$, which implies that the network is not heavily constrained, N_c is set to be 1.

6.5.2 Simulation Results

The proposed A*-based optimal dispatch procedure is tested on the 10-node radial distribution network shown in Figure 6.1 with the objective of minimizing the total carbon emissions. This work will mainly focus on the improved A* procedure, the cable capacity method in particular due to the nature of the generic distribution network, which will be tested in several different scenarios and compared with DYnamic programming Decentralised OPtimal dispatch (DYDOP) that is published in [98] to investigate the potential advantage of the A*-based dispatch approach.

In this work, the power network is assumed to have a distributed generator (DG) at every node. The detailed data of the distribution network used in this work are presented in Table 6.1 and 6.2, where $load_i = \frac{1}{2}C_{i\hat{i}}$ and $P_i^{max} = \frac{1}{2}load_i$. In this case, the network is heavily loaded and the DGs' power outputs are not sufficient.

Table 6.3 shows the optimal assignments of the DGs' power outputs under A* optimal dispatch and DYDOP, which are slightly different for DGs at v_7 and v_8 but the total carbon emissions are the same (1.35 kg). This verifies the feasibility of the proposed A*-based optimal dispatch: it is capable of finding an optimal solution.

TABLE 6.1: Parameters of Load and Distributed Generators of the Radial Distribution Network

Node	Load (kW)	DG Ratings (kW)	Carbon Intensity (kg CO_2e/kWh)
v_1	0	0	0
v_2	10	5	0.05
v_3	40	20	0.03
v_4	20	10	0.05
v_5	20	10	0.03
v_6	10	5	0.05
v_7	10	5	0.05
v_8	10	5	0.05
v_9	10	5	0.03

If the number of discrete values of the output power of a DG in $DG_i = \{0, \ldots, P_i^{max}\}$ increases, in other words, if DG_i is discretized into $\frac{1}{\delta}$ kW steps with $\delta > 1$, then the computation and communication burden will increase even though the optimization problem remains the same. The simulation results in Figure 6.8 and Table 6.4 illustrate the computation and communication burdens as a function of δ, respectively. It is clear in Figure 6.8 that DYDOP requires many more utility computations (i.e., the computation of carbon emissions at agents) than the A*-based approach. Therefore, the A*-based approach outperforms DYDOP in terms of agents' workload in computation. This result is easy to understand as DYDOP is basically a breadth-first search approach, which

Table 6.2: Capacity of Cables of the Radial Distribution Network

From Node	To Node	Capacity (kW)
v_1	v_0	100
v_2	v_1	20
v_3	v_1	80
v_4	v_3	40
v_5	v_3	40
v_6	v_4	20
v_7	v_4	20
v_8	v_5	20
v_9	v_5	20

Table 6.3: Optimal Assignment of Distributed Generators' Power Outputs of the Radial Distribution Network

Methodology	v_2 (kW)	v_3 (kW)	v_4 (kW)	v_5 (kW)	v_6 (kW)	v_7 (kW)	v_8 (kW)	v_9 (kW)	Total Carbon Emission (kg)
A* Optimal Dispatch	1	20	1	10	1	2	1	5	1.35
DYDOP	1	20	1	10	1	1	2	5	1.35

checks every possible set of DGs' power assignments, while the A*-based approach is more of a best-first search procedure, which only considers the current best set of DG dispatch actions. However, Table 6.4 shows that agents need to send out many more messages using A*-based approach than using DYDOP.

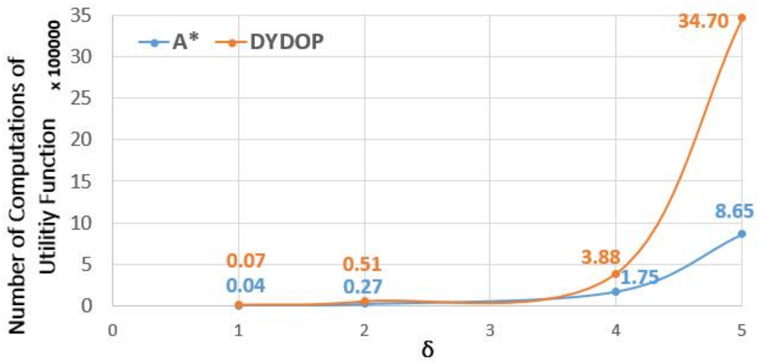


Figure 6.8: The Number of Utility Computations Required to Solve the Optimal Dispatch Problem Using A*-based approach and DYDOP

When the value of load demand is a real number instead of an integer, which is usually the case in reality and the network is less loaded than the previous case, as shown in Table 6.5, the agents' computation and communication burden of solving this optimal dispatch problem is evaluated for A*-based approach and DYDOP [98], respectively, as depicted

TABLE 6.4: Number of Messages Sent from the Agents within the Distribution Network

Methodology	**Increase rate of DGs (δ)**			
	1	2	4	5
A*	4455	26603	175020	865105
DYDOP	115	213	409	1163

in Figure 6.9 and 6.10. Comparison with the first case where the network's load demands are as shown in Table 6.1 are made with δ set to be 5. It is shown in Figure 6.9 that with the loads presented in Table 6.5 (Case 2), using the A*-based approach, the network's agents required fewer utility computations than that in the previous case, while the computation burden increases considerably when using DYDOP. Similar observation can be made about the agents' communication burden, as shown in Figure 6.10; the number of messages sent from the network's agents are considerably fewer than that in the first case when using the A*-based approach while the messages sent from agents under DYDOP are more than twice that in the previous case.

TABLE 6.5: Nodal Loads of the Radial Distribution Network

Node	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
Load (kW)	0	9.11	22.975	13.142	11.283	9.767	6.187	8.182	9.726

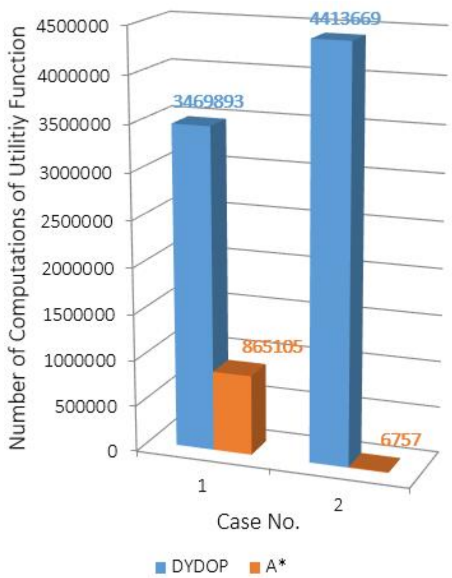


FIGURE 6.9: The Number of Utility Computations Required to Solve the Optimal Dispatch Problem Using A*-based approach and DYDOP in Case 1 and 2 Respectively (Case 1: Network's load demands are as shown in Table 1; Case 2: Network's load demands are as shown in Table 5)

These results reveal a very straightforward fact that in the case where dynamic programming is less effective, DYDOP suffers from much higher computation and communication burden at the agents. This is not hard to understand. It is a breadth-first search method, as discussed earlier; states that remain after using dynamic programming to delete redundant states will be sent out in a given form of message and will be investigated

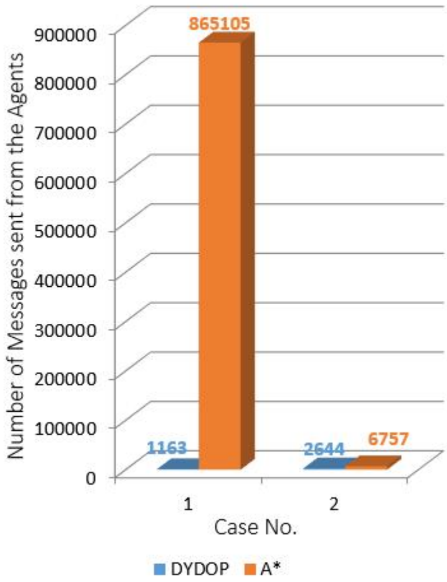


FIGURE 6.10: Number of Messages Sent from the Agents within the Distribution Network Using A*-based approach and DYDOP in Case 1 and 2 Respectively (Case 1: Network's load demands are as shown in Table 1; Case 2: Network's load demands are as shown in Table 5)

further at the next stage. Therefore, in Case 2, where the load demands change to those values that are shown in Table 6.5, the number of redundant states decreases, which implies that the number of states that remain after conducting dynamic programming increases, hence more messages will be sent out and an increasing number of states will be investigated further, which requires many more utility computations. This result is expected for any algorithm based on dynamic programming.

However, using the A*-based approach under the situation where the effect of dynamic programming diminishes, the agents' workload should have also increased according to the analysis above, since this algorithm also leverages dynamic programming, but it actually didn't and even greatly reduced. Why? The only reason is that in this case the network's load demand is reduced. As the A* extends only the best state at each time, as discussed earlier, those relatively better states that result in less carbon emissions but higher power flows on a cable are checked first, while those relatively worse states will not be considered unless the better ones turn out to be infeasible, i.e., overloading a cable. When the network is less loaded, the resulting power flow of an assignment of DG power outputs reduces, thus the number of states that cause a power flow exceeding the cable's capacity will be relatively small. Therefore, the A* procedure needs to check a relatively small number of states and filter out fewer states that are relatively better but infeasible before finding the optimal solution, meaning that in this case it is faster and easier to reach the final optimal solution.

Therefore, DYDOP is more sensitive to how effective the dynamic programming functions in a case, while the A*-based approach is more sensitive to the network's load demand.

6.6 Case Study 2: Optimal Decentralized Coordination of Electric Vehicles and Renewable Generators in A Distribution Network Using A* Search

In this case, the optimal dispatch problem of RGs and EVs is formulated as a distributed multi-objective constraint optimization problems (DMOCOP), which has been presented in Chapter 5 [94] and is developed from DCOP. The objectives include cost saving for EV users while ensuring enough energy within the batteries to complete their driving activities, load levelling for the network and increase the usage of renewable power. For the details of formulation of each objective function and constraints, please refer to Section 5.2.3. The DMOCOP contains three main elements:

1. Control Variables: $X = \{x_1, \ldots, x_h\}$. In this work, x_i can be a renewable generator's power output, p_i, or an EV's battery dispatch mode, δ_i. Thus, $X = \{\mathbf{p}, \boldsymbol{\delta}\}$.
2. Feasible Domains: Each control variable x_i has its own feasible domain d_i, which include all possible values of x_i. In this work, d_i can be represented as follows:

$$d_i = \begin{cases} RG_i = \{0, \ldots, P_i^{max}\} & \text{when } x_i \text{ is a renewable generator} \\ DM_i = \{-3, -2, -1, 0, 1, 2, 3\} & \text{when } x_i \text{ is an EV battery} \end{cases} \tag{6.25}$$

3. Utilities: Every agent has its own utility function $U_i \in U = \{U_1, \ldots, U_k\}$. In this case, U_i maps to the penalty cost at agent i, which evaluates how unsatisfactory a combination of dispatch actions is in terms of the objectives.

The utility functions of agents, with the objectives of saving cost while leaving sufficient charge in an EV's battery, network load levelling and reduction of wasted renewable energy, incorporate all the associated objective functions, which have been defined in Chapter 5 and have their priorities determined using the Analytic Hierarchy Process (AHP). For those involve both EV battery and RG variables, the utility functions are:

$$\begin{aligned} U_i = {} & 27.81\% \times C_{RG} + 39.52\% \times C_{SOC} \\ & +16.34\% \times C_{ep} + 16.34\% \times C_{LL}, \end{aligned} \tag{6.26}$$

while for the agents that only involve EV battery variables, their utility functions are as follows:

$$U_i = 50\% \times C_{SOC} + 25\% \times C_{ep} + 25\% \times C_{LL}. \tag{6.27}$$

With the definition of control variables X and the optimal dispatch problem DMOCOP, the A* algorithm is customized for this application of coordinated dispatch of RGs and EVs.

6.6.1 Stochastic Modelling of Uncertainties

In order to test the A* optimal dispatch algorithm on the EVs and RGs within a distribution network, the information of EV travel patterns and renewable energy sources (only wind power are incorporated in this case) is required. However, due to the intrinsic uncertainties, stochastic modelling of EV travel patterns and wind power were made using Copula [124, 150] — see Section 2.3.2.

6.6.1.1 Modelling of EV Travel Patterns and On-Road Energy Consumption

In this case, we focus on domestic dispatch of EV batteries, i.e., charging or discharging of EVs when parked at home. Therefore, three key parameters of EV driving activities are needed by the proposed optimal dispatch algorithm: time of departure from home, travel distance, time of arrival at home. In order to model the stochastic driving behaviours of EV users, the methodology presented in [124] is utilized, which leveraged Copula to simulate the values of these parameters considering their dependence on each other.

As introduced in [124], 75% of the users had a single home-to-home (h2h) trip in a day while 21% take double h2h trips and the rest 4% take more than two h2h trips. Therefore, the modelling focuses mainly on the single and double h2h trips as they accounts for most of the daily driving activities.

Modelling EVs' travel patterns requires capturing the stochastic behaviour of the three parameters mentioned earlier. That, in terms of single h2h trip, incorporates three random variables representing it [124]: departure time T_d^s, arrival time T_a^s and travel distance D^s of the h2h trip, while six random variables are needed to model double h2h trips: departure time, arrival time and travel distance of the first h2h trip (T_d^{d1}, T_a^{d1}, D^{d1}) and the second h2h trip (T_d^{d2}, T_a^{d2}, D^{d2}). Dependence between each pair of random variables are an essential element in modelling together with the marginal distribution of each random variable. Gaussian copula, which is described earlier, is used in this stochastic modelling.

The historical data used and presented in [124] are directly leveraged to simulate the stochastic EV driving activities in this work. The marginal distributions of random variables of single and double h2h trips are shown in Figure 6.11, 6.12 and 6.13. The rank correlation matrix (Spearman's ρ) of the 3 random variables representing single h2h trips, ρ_s^s, is obtained from [124] as follows:

$$\rho_s^s = \begin{array}{c c} & \begin{array}{ccc} T_d^s & T_a^s & D^s \end{array} \\ \begin{array}{c} T_d^s \\ T_a^s \\ D^s \end{array} & \begin{pmatrix} 1 & 0.13 & -0.35 \\ 0.13 & 1 & 0.29 \\ -0.35 & 0.29 & 1 \end{pmatrix} \end{array}, \tag{6.28}$$

while the rank correlation matrix of 6 random variables of double h2h trips, ρ_s^d, is:

$$\rho_s^s = \begin{array}{c c} & \begin{array}{cccccc} T_d^{d1} & T_a^{d1} & D^{d1} & T_d^{d2} & T_a^{d2} & D^{d2} \end{array} \\ \begin{array}{c} T_d^{d1} \\ T_a^{d1} \\ D^{d1} \\ T_d^{d2} \\ T_a^{d2} \\ D^{d2} \end{array} & \begin{pmatrix} 1 & -0.06 & -0.37 & -0.02 & -0.03 & -0.02 \\ -0.06 & 1 & 0.42 & 0.70 & 0.57 & -0.06 \\ -0.37 & 0.42 & 1 & 0.3 & 0.24 & 0.17 \\ -0.02 & 0.70 & 0.30 & 1 & 0.80 & -0.04 \\ -0.03 & 0.57 & 0.24 & 0.80 & 1 & 0.20 \\ -0.02 & -0.06 & 0.17 & -0.04 & 0.20 & 1 \end{pmatrix} \end{array}, \tag{6.29}$$

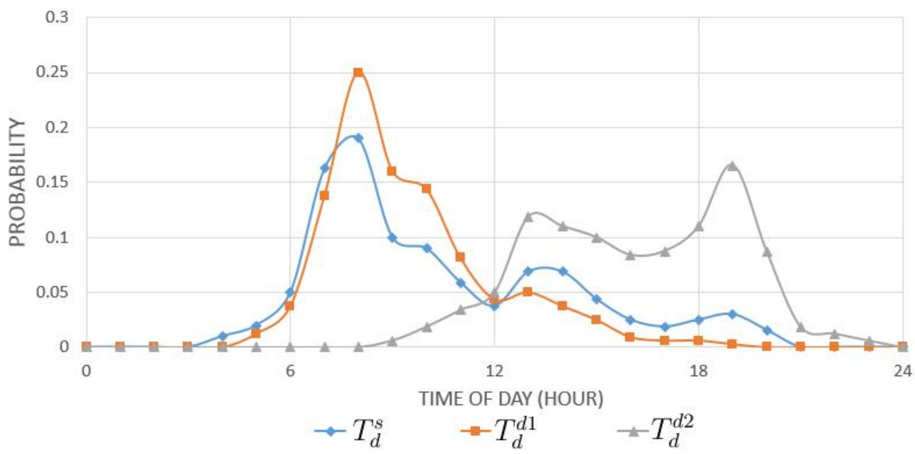


FIGURE 6.11: Marginal Distributions of departure time of single and double h2h trips

The procedure of stochastic modelling using Gaussian Copulas is then carried out to simulate random variables of single and double h2h trips. The CDF of each random variable is derived by calculating the integral function of the associated marginal distribution. The linear correlation ρ of underlying multivariate standard normal distribution of each type of h2h trips is computed on the corresponding rank correlation (ρ_s^s, ρ_s^d) using the 1-1 mapping formula between ρ and ρ_s described in (2.6).

As the simulated EV driving activities need to be viable, the departure time must be earlier than arrival time of a h2h trip: $T_d^s < T_a^s$, $T_d^{d1} < T_a^{d1}$ and $T_d^{d2} < T_a^{d2}$. If the simulated values of random variables don't meet this constraint, the stochastic simulation

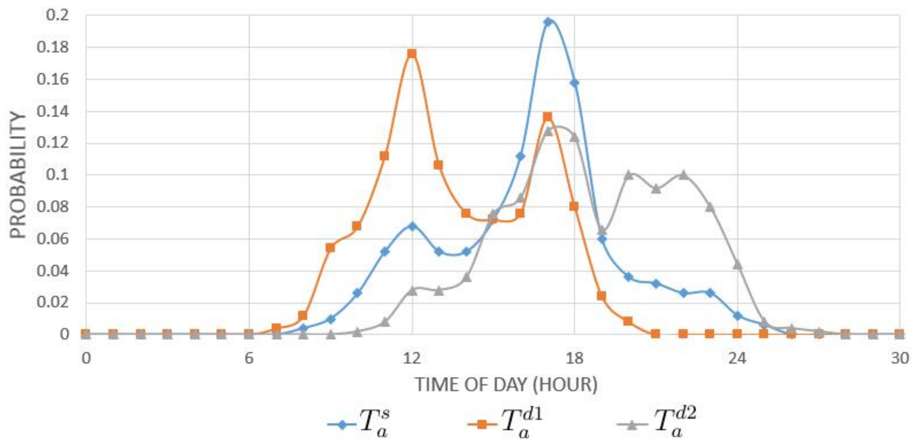


FIGURE 6.12: Marginal Distributions of arrival time of single and double h2h trips

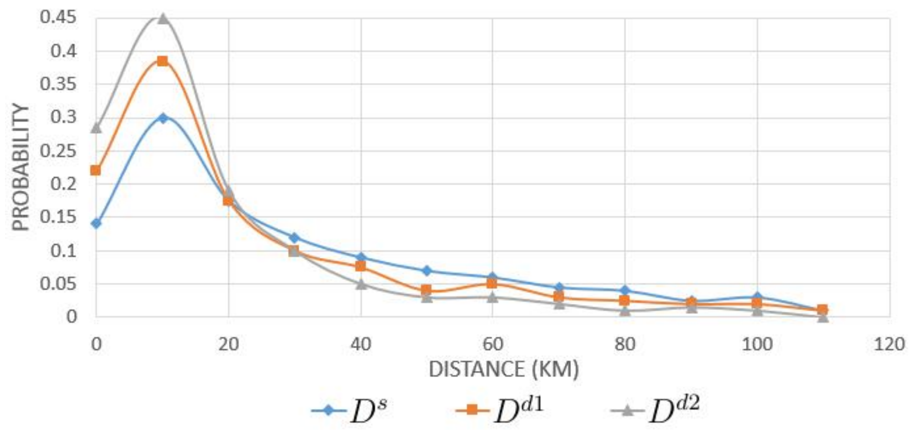


FIGURE 6.13: Marginal Distributions of travelled distance of single and double h2h trips

using Gaussian Copulas described above will be re-run until this viability constraint is satisfied and the simulated values are ensured to be viable.

In order to compute the energy consumed on road, the value of EV driving velocity is essential and needs to be simulated. Due to the lack of historical velocity data, and thus its actual marginal distribution and correlation with other random variables, Monte Carlo simulation is adopted to derive the stochastic values of EV driving velocity, where a normal distribution is assumed with a mean value μ of 40 mph and a standard deviation σ formulated as

$$\sigma = 0.5 \times \min(v_{max} - \mu,\ \mu - v_{min}), \tag{6.30}$$

where v_{max} is set to be 70 mph, which, generally speaking, is the maximum velocity a vehicle can drive in the urban area. $v_{min} = \frac{D}{T_a - T_d}$ is the minimum average velocity an EV must reach in order to travel a distance of D (the travelled distance of a h2h trip, which can be D^s, D^{d1} or D^{d2}) within the time interval of $T_a - T_d$ (the duration of a h2h trip, i.e. the difference between the departure and arrival time of a h2h trip, which can be $T_a^s - T_d^s$, $T_a^{d1} - T_d^{d1}$ or $T_a^{d2} - T_d^{d2}$). If the simulated velocity is outside the range $[v_{min}, v_{max}]$, the above Monte Carlo simulation is re-run until the simulated velocity is within that reasonable range.

Therefore, the energy consumed on a h2h trip is computed as follows.

It is assumed that EV driving velocity is constant on road, therefore the propulsion force is equal to the sum of rolling friction and air resistance:

$$F = C_r mg + \frac{1}{2}\rho_a A_f C_D v^2 \tag{6.31}$$

where rolling resistance coefficient $C_r = 0.015$, mass of the vehicle $m = 1150kg$, air density $\rho_a = 1.29kg/m^3$, area of vehicle's face $A_f = 1.5^2 m^2$, and air resistance coefficient $C_D = 0.3$. Hence, the power of propulsion is:

$$P = Fv = C_r mgv + \frac{1}{2}\rho_a A_f C_D v^3. \tag{6.32}$$

The energy of propulsion E_w is thus the integral of P:

$$E_w = \int Pdt = C_r mg \int vdt + \frac{1}{2}\rho_a A_f C_D \int v^3 dt = D(A + Bv^2), \tag{6.33}$$

where $A = C_r mg$, $B = \frac{1}{2}\rho_a A_f C_D$ and D is the travelled distance of a h2h trip. Therefore, the electric energy consumed E_e is:

$$E_e = E_w/\eta, \tag{6.34}$$

where the efficiency of the electrical engine $\eta = 70\%$ [151]. With the on-road electric energy consumption calculated, the state of charge (SOC) of an EV battery after a h2h trip can be estimated.

6.6.1.2 Wind Power Modelling

In this work, distributed wind power generators are considered only and are spread in the distribution network under investigation. As these wind turbines are located in proximity, the wind speeds at these turbine sites have strong correlation with each other. Due to the lack of historical data, the correlation matrix calculated in [150] is used in this work:

$$\rho_{wind} = \begin{array}{c} \\ S_w^1 \\ S_w^2 \\ S_w^3 \\ S_w^4 \\ S_w^5 \end{array} \begin{array}{c} \begin{array}{ccccc} S_w^1 & S_w^2 & S_w^3 & S_w^4 & S_w^5 \end{array} \\ \begin{pmatrix} 1 & 0.82 & 0.85 & 0.74 & 0.78 \\ 0.82 & 1 & 0.83 & 0.74 & 0.82 \\ 0.85 & 0.83 & 1 & 0.81 & 0.74 \\ 0.74 & 0.74 & 0.81 & 1 & 0.65 \\ 0.78 & 0.82 & 0.74 & 0.65 & 1 \end{pmatrix} \end{array}. \tag{6.35}$$

The marginal distributions of wind speeds at different sites are derived from [152], as shown in Figure 6.14 and 6.15.

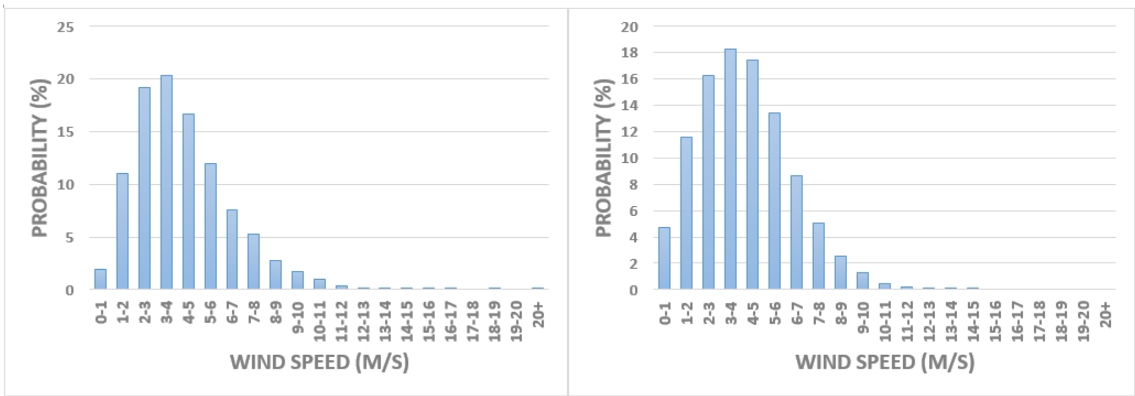


FIGURE 6.14: Marginal Distributions of wind speed at site 1 (left) & 2 (right)

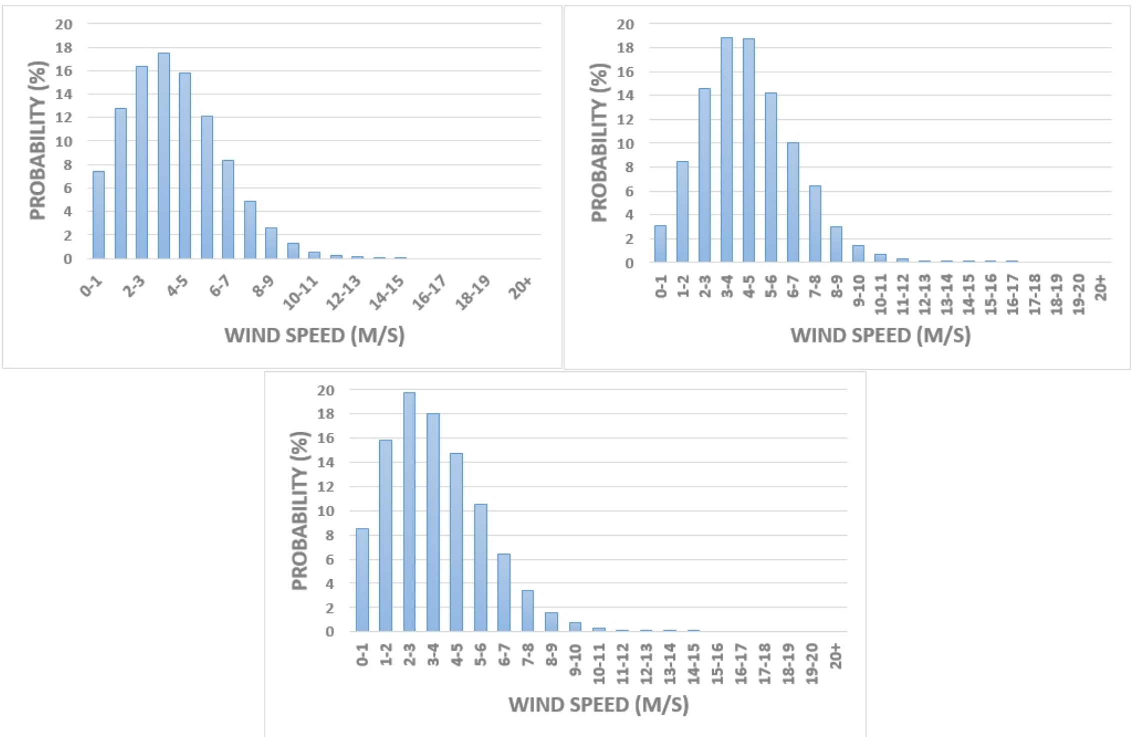


FIGURE 6.15: Marginal Distributions of wind speed at site 3 (up left), 4 (up right) & 5 (middle)

The copula-based stochastic modelling procedure discussed earlier is then applied to simulate random wind speeds at these different sites.

Provided the simulated wind speed, the wind power generated at a wind turbine generator (WTG) confronting the wind with given speed is calculated according to Figure

6.16[1], where the cut-in, nominal and cut-out wind speeds are 3.5, 11.2 and 25 m/s and its nominal power is 2 MW.

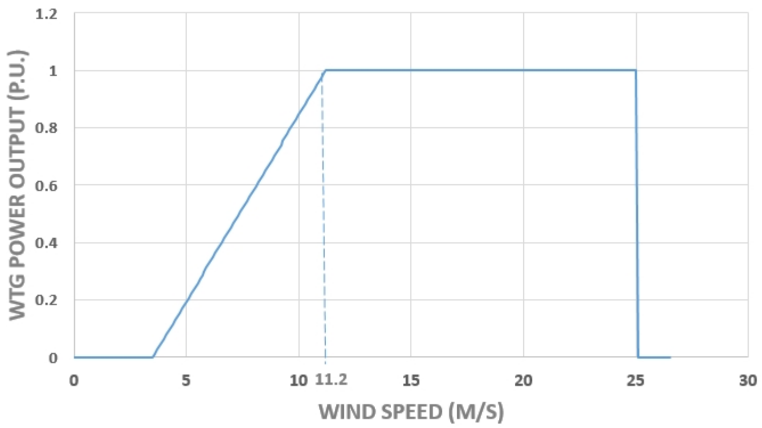


FIGURE 6.16: WTG Wind Speed/Power Characteristics

Due to lack of information in autocorrelation of wind speed in time series, the simulated wind speed data are arranged manually in temporal dimension only, so that the variation of wind speed is reasonable and more realistic and the correlations between different WTG sites remain untouched. However, in the future, autocorrelation should also be considered in the stochastic modelling of wind speed in order to better reflect the reality. Figure 6.17 shows the sum of simulated wind power generated at five WTG sites in the network.

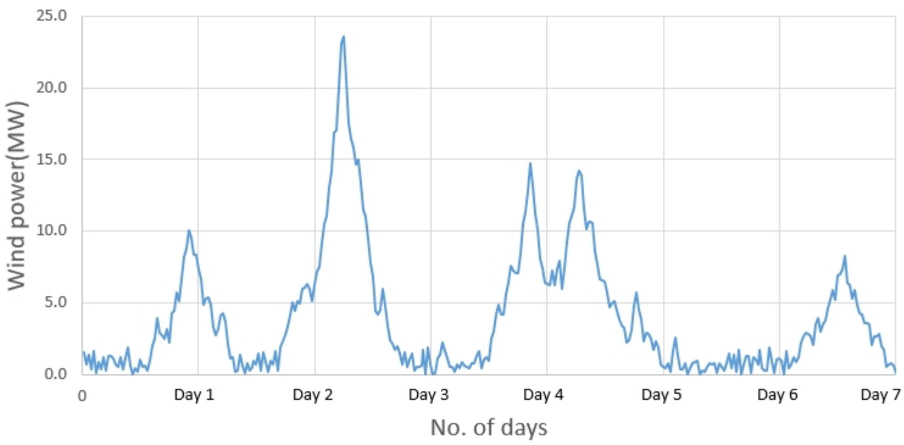


FIGURE 6.17: Simulated Wind Power Generated in the Power Network during a Week

[1]The figure is from [150]. In the real case, when wind speed $\leqslant$ nominal speed, WTG power output $\propto$ wind speed^3, but for the simplification of modelling, we use a first approximation here, i.e., WTG power output $\propto$ wind speed.

6.6.2 Simulation Results

The proposed A*-based optimal dispatch algorithm is tested in a radial distribution network, a modified UKGDS shown in Figure 6.18. In this distribution network, five renewable generator sites are located at nodes v_2, v_3, v_6, v_{11} and v_{12}, respectively, each of which site has 3 wind turbines, thus a total nominal wind power of 6 MW can be generated at each site. A total number of 33000 EVs spread in the network and are capable of connecting to the grid via nodes $v_2, v_3, \ldots, v_{12}$, each of which can provide synchronous V2G/G2V services for at most 3000 EVs. The random EV travel patterns and wind power are simulated using the stochastic models discussed earlier. The results of the simulated EV travel patterns and wind power are used in the simulation test of proposed A*-based dispatch algorithm.

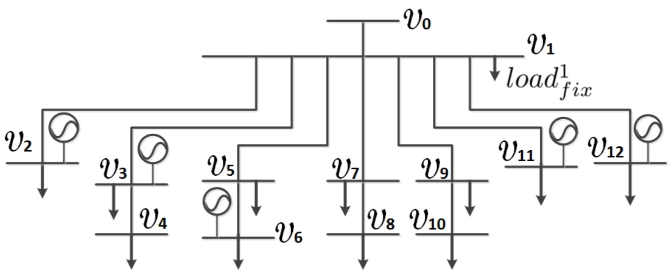


FIGURE 6.18: Diagram of a modified generic radial distribution network

The thermal capacity of distribution cables are derived from UKGDS model [145], as shown in Table 5.1 and Table 5.2. The parameters of EV batteries are obtained from [20]. The simulation is implemented on a laptop with a Dual Core 2GHz CPU and 8GB RAM using MATLAB.

The total load demand and system selling/buying prices of electricity during a day were taken from [141].The total load demand is scaled down so that the peak demand during a day is 350 MW, which approximately represents the daily load demand in a typical UK regional distribution network, as shown in Figure 5.7.

Due to scarcity of information on the payments by/to EV users when the EVs are charged/discharged, the ASSP (i.e. adjusted system selling price) and ASBP (i.e. adjusted system buying price) in Figure 4.6 are respectively used as the EV charging and discharging prices.

The time step is set to be 30 minutes. The dispatch actions are determined by the proposed A*-based optimal dispatch strategy at the beginning of every time interval and lasts for the entire time interval of 30 minutes, as mentioned earlier.

In order to test the potential economic benefits that participating in the proposed optimal coordinated scheme can bring to EV users, the net costs of EVs are calculated and presented in Table 6.6. An average cost of £0.45 per vehicle per day is expected for those adopt the proposed A*-based dispatch strategy, in comparison with an average cost of £0.73 for an uncontrolled way of treating EV batteries, which indicates an approximate amount of £103 annually for each EV user. Furthermore, the simulation results in Table 6.7 shows that the proposed dispatch strategy is safe in the sense that it prevents EVs from overcharging their batteries (the stored energy is never exceed 100% of battery's capacity) and also from running out of energy during the day with at least 34% of battery's available capacity remains.

Table 6.6: Daily costs of EVs calculated from simulations starting with different initial SOC

	Total Daily Costs to EV Users in a network (£)
Average	14773.5
Standard Deviation	1184.6

Table 6.7: minimum and maximum SOCs of EVs during a day

Node	**Minimum SOC**	**Maximum SOC**	**Node**	**Minimum SOC**	**Maximum SOC**
v_2	0.41	1.00	v_3	0.65	1.00
v_4	0.50	1.00	v_5	0.49	1.00
v_6	0.49	1.00	v_7	0.64	1.00
v_8	0.49	1.00	v_9	0.51	1.00
v_{10}	0.57	1.00	v_{11}	0.34	0.99
v_{12}	0.50	1.00			

As for the renewable generators, the simulation results show that overall 99.5% of wind power are utilized to support network load demand and/or EV charging demand, as shown in Table 6.8. The difference between wind power utilization rates at different WTG sites is because the wind speed, the battery conditions of locally parked EVs and local fixed load demand varies from site to site.

Table 6.8: Utilization of Wind Power Generated at WTG sites of the Distribution Network during A Week

Site No.	1	2	3	4	5
Utilization of Generated Wind Power	99.6%	100%	98.1%	100%	100%

In terms of the network load variance due to the integration of RGs and EVs, the simulation results are demonstrated in Figure 6.19, where the comparison between the proposed optimal dispatch and uncontrolled charging is made and the load variance value $\Delta load$ is calculated by:

$$\Delta load = load(with\ EVs\ \&\ RGs) - load(without\ EVs\ \&\ RGs). \tag{6.36}$$

Therefore, the positive value implies valley filling while negative value peak shaving. From Figure 4.9, it is obvious to find that A*-based optimal dispatch strategy works well in levelling the network's load demand (peak shaving with EV load=-5.7MW and RG output=3.7MW on average, and valley filling with EV load=22.5MW and RG output=4.1MW on average), while uncontrolled charging of EVs pushes the peak of network's load even higher.

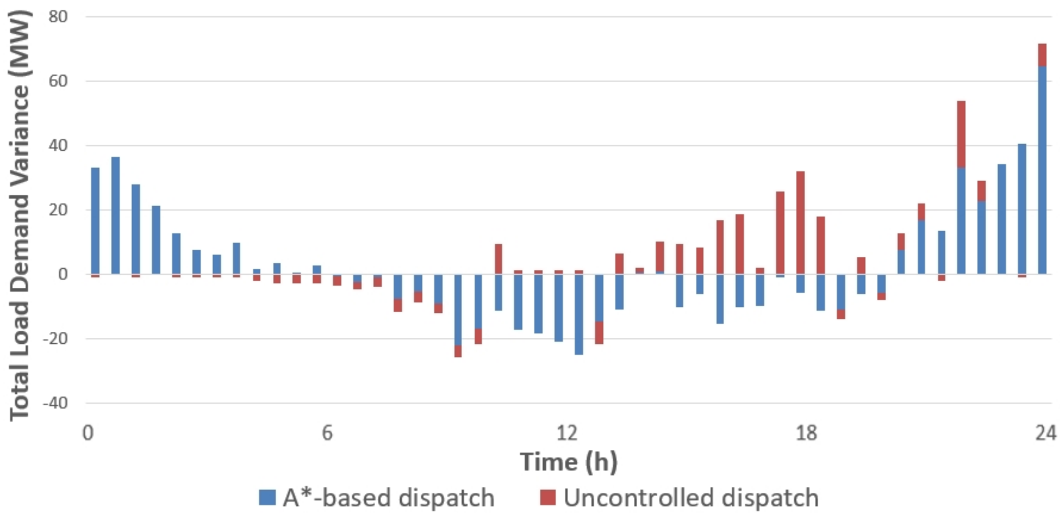


Figure 6.19: Network Load Variance Caused by the Integration of RGs and EVs

Furthermore, in order to test the stability of the proposed A*-based optimal dispatch strategy, the simulation for 7 incessant days was run and the results are demonstrated in Figure 6.20. It is clear that these 7 daily load curves are mostly overlapped with each other, indicating the stability of proposed dispatch strategy. This figure also shows that load levelling is generally achieved on the daily basis without increasing the peak load demand of the network.

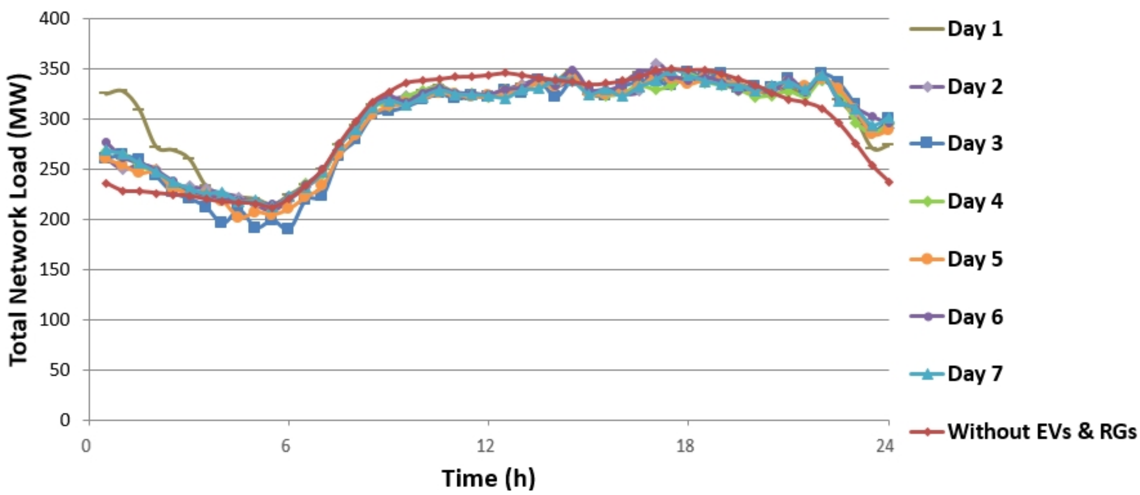


Figure 6.20: Network Load Demand during a Week

Further testing the stability of the proposed dispatch algorithm is done by examining the distribution the EV batteries' SOCs at the start of each day. In Figure 6.21, it is easy to find out that the EVs' SOCs at the end of every day have a similar probability distribution, with most of EVs' SOCs in the range [0.9,1] and the smallest SOC no less than 0.5. If the initial SOCs of EVs at the beginning of the simulation is assigned based on a different normal distribution with different means, the distribution of EV SOCs at the start of following days are evaluated and shown in Table 6.9. It is clear in this table that starting form different assignments of initial SOCs, the EVs' SOCs at the outset of a day always stabilize to a distribution that has a mean of 0.88 and a standard deviation of 0.15 on average, with very slight variation. Again, the stability of the proposed algorithm is verified.

TABLE 6.9: Key factors of EVs' SOC distribution at the end of a day tested with different mean values of the random assignments of the initial SOCs

Initial SOC assignment	**Mean of SOCs**		**Standard deviation of SOCs**	
	Mean	Standard deviation	Mean	Standard deviation
$\boldsymbol{\mu = 0.5, \sigma = 0.1}$	0.88	0.01	0.15	0.01
$\boldsymbol{\mu = 0.6, \sigma = 0.1}$	0.88	0.01	0.15	0.01
$\boldsymbol{\mu = 0.7, \sigma = 0.1}$	0.88	0.01	0.15	0.01

6.7 Complexity Discussion

In Section 6.5, the comparisons between the proposed A* algorithm and the dynamic-programming-based algorithm are made and discussed in terms of the computational and communication complexities. Here, from the A* algorithm itself, the computational and communication complexities are discussed further, with the pseudo code presented below.

The number of states it iterates grows linearly with the number of possible assignments of X, as can be seen from line 16 of the pseudo code. If the control variables X contain several different variables such as DG power output and EV dispatch action in Case study 2, a node needs to iterate through all states in the Cartesian products of its own RG power output values and EVs' dispatch actions (each EV has 7 possible dispatch actions, i.e. charge/discharge at three different current levels and idle, thus N_{ev} EVs have $7^{N_{ev}}$ possible dispatch actions in total), hence the computational complexity at a node grows linearly with the number of discretized DG power output values N_p and grows exponentially with the number of EVs connected to this node N_{ev} in $O(N_p 7^{N_{ev}})$. If considering the worst case, where the state that has the maximum utility value is the only feasible state and the optimal solution, all possible states have to be iterated. In this case, it needs to iterate through all states in the Cartesian product of all of its children's states, hence the computational complexity at a node with N_{chi} children nodes grows

A* algorithm

1. Initialize $closedlist := the\ empty\ set$;
2. Initialize an $openlist$ for each $leafnode$;
3. $openlist_i := \{(leafnode_i, 0)\}$;
4. $U(leafnode_i, 0) := 0$;
5. while any $openlist$ is not empty
6. $for\ each$ non-empty $openlist$
7. $currentnode :=$ the node in $openlist$ having the lowest utility value U;
8. $if\ currentnode = root\ node$
9. {We have found the solution and thus stop the search;}
10. $if\ N_{chi}(currentnode) > 1$
11. $\{N_{chi}\ openlists\ that\ have\ the\ same\ currentnode\ are\ combined;$
12. $Remove\ (currentnode, \boldsymbol{f_{chi \to current}})\ from\ the\ openlist_{comb};\}$
13. $else$
14. $\{Remove\ (currentnode, f_{chi \to current})\ from\ the\ openlist;\}$
15. $Generate\ Nextnode := par(currentnode)$;
16. $for\ each\ possible\ assignment\ of\ X\ at\ currentnode$
17. $P_X := powercal(X)$;
18. $U_X := utilitycal(X)$;
19. $f_{current \to next} := sum(\boldsymbol{f_{chi \to current}}) + P_X$;
20. $if\ f_{current \to next} > C_{current \to next}$
21. $\{continues;\}$
22. $UtilityValue := U(currentnode, \boldsymbol{f_{chi \to current}}) + U_X$;
23. $if\ (Nextnode, f_{current \to next})\ is\ in\ the\ openlist$
24. $\{if\ U(Nextnode, f_{current \to next}) > UtilityValue$
25. $\{U(Nextnode, f_{current \to next}) := UtilityValue;$
26. $Pstate(Nextnode, f_{current \to next}) := \{X, Pstate(currentnode, \boldsymbol{f_{chi \to current}})\};\}$
27. $else$
28. $\{continue;\}\}$
29. $if\ (Nextnode, f_{current \to next})\ is\ in\ the\ closedlist$
30. $\{if\ U(Nextnode, f_{current \to next}) > UtilityValue$
31. $\{U(Nextnode, f_{current \to next}) := UtilityValue;$
32. $Pstate(Nextnode, f_{current \to next}) := \{X, Pstate(currentnode, \boldsymbol{f_{chi \to current}})\};\}$
33. $else$
34. $\{continue;\}\}$
35. $Add\ (Nextnode, f_{current \to next})\ to\ the\ corresponding\ openlist;$
36. end
37. $Add\ (currentnode, \boldsymbol{f_{chi \to current}})\ to\ the\ closedlist;$
38. end
39. end

exponentially with N_{chi} in $O(M^{N_{chi}})$. M is the number of different states a child node generates. However, this case is rare and only possible in a network that is very strictly constrained and severely loaded. Generally, as stated in line 7 of the pseudo code, the algorithm picks the best state each time. Therefore, the computational complexity is inversely proportional to how fast bad states turn distinctively bad but proportional to the level of capacity constraint of the network.

As the states generated by the A* algorithm will be sent to the central memory for comparisons and for future usage (lines 22–35), the communication complexity grows linearly with the computational complexity. Moreover, the total size of messages sent by the A* algorithm equals to the sum of the size of messages that each node creates and sends, hence increases linearly with the size of the network in $O(N_v)$. N_v is the number of nodes in a network.

In contrast, the computational complexity of a centralized algorithm, such as simplex algorithm, which doesn't explicitly consider the topology of the network, grows exponentially with the size of the network, as discussed in [98]. Therefore, it will quickly become infeasible for a centralized algorithm to solve an optimal dispatch problem when the network expands or integrate with more EVs and RGs.

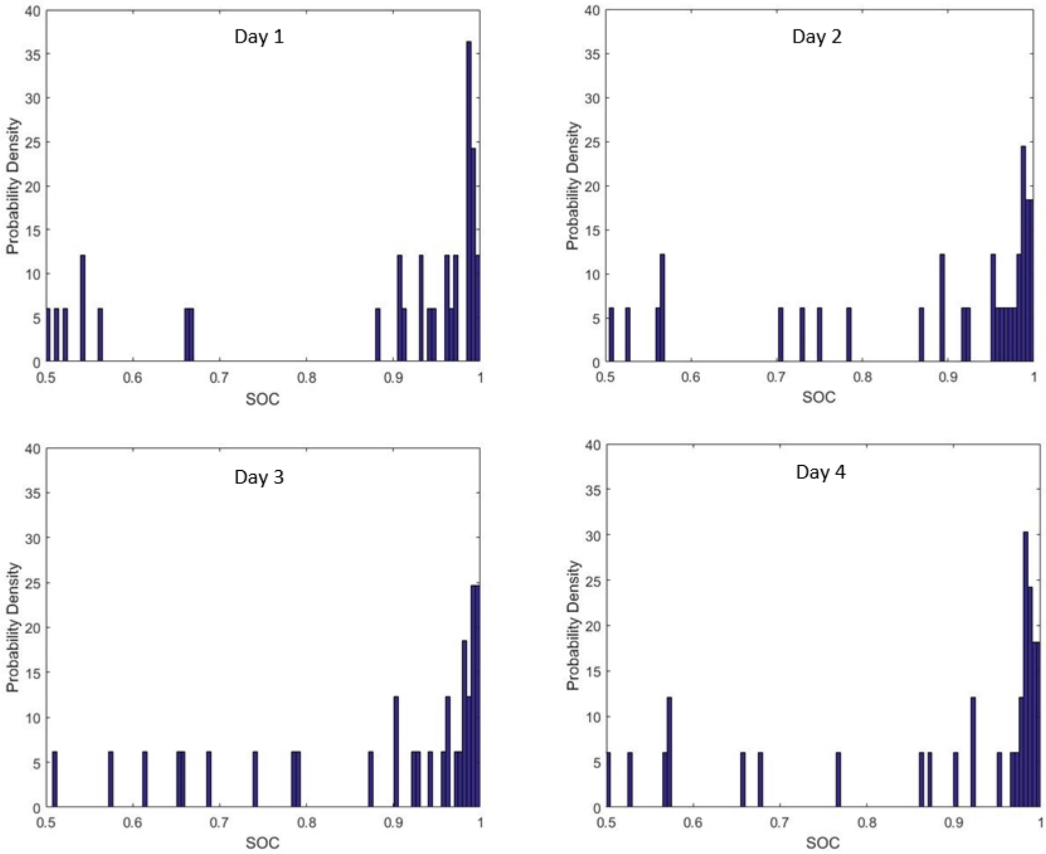
6.8 Conclusions

In this chapter, an A*-based optimal dispatch algorithm is proposed and discussed thoroughly. The feasibility and efficacy of the A* algorithm are verified by the simulation results of the two different applications of it.

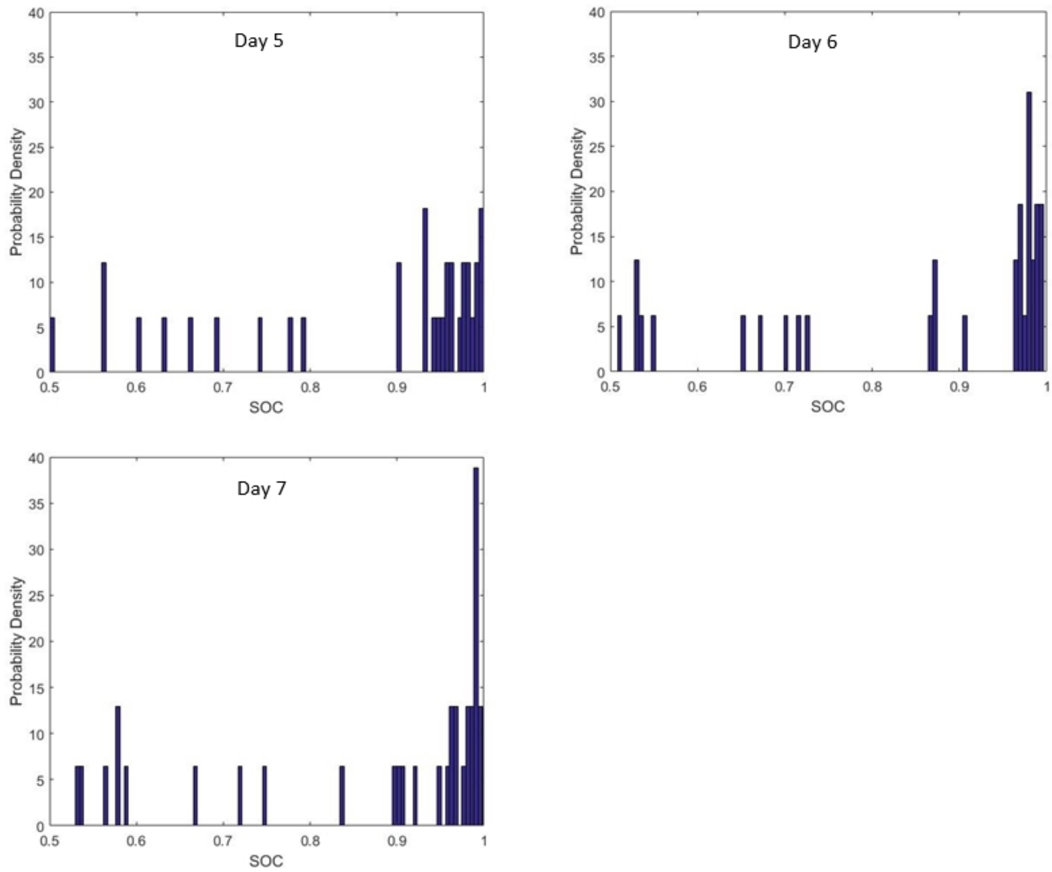
In the case study of DG dispatch, an improved A* search procedure is utilized in this approach. Its efficacy in finding the optimal solution of DG dispatch problem has been verified. The A*-based approach outperforms another decentralized dispatch algorithm, DYDOP, that was published in [98] in terms of requiring fewer computations by agents, at the expense of the increase of the communication burden. A* is therefore more suited to systems with fast communication while DYDOP is better suited to system with slow communication channels, but fast computers. Furthermore, even though dynamic programming is utilized in both the A*-based approach and DYDOP to simplify the process, it is only in DYDOP that it has a dominant effect. Moreover, when the network's load demand reduces, the computation and communication amount decreases significantly when using the A*-based approach. The A*-based approach shows a higher sensitivity to the network's load level, while DYDOP is much more sensitive to whether dynamic programming is heavily used in the solution process. In this case study, the A* search algorithm is applied to a relatively simple case, for the simplicity of explanation and comparison.

In the case study 2, the application of A* search algorithm to the main problem — the coordinated dispatch of EVs and RGs is investigated, where the uncertainties of RG power outputs and EVs' driving activities are also modelled and simulated. Its stability and feasibility in dispatching the RGs and EVs to best achieve the objectives demonstrate its robustness to overcome the uncertainties of RGs ad EVs and ensure a stable and good performance of the power network. It costs EV users £0.45 per vehicle per day averagely, which saves EV users around £103 per year per vehicle compared with uncontrolled charging of EVs, while ensuring all driving activities complete with at least 34% of battery's capacity remains. Moreover, overall more than 99% of generated wind power are absorbed and utilized by the network to support various load demand. Different local network conditions and wind speeds result in the discrepant wind power utilization rates among different WTG sites. The proposed optimal coordination of EVs and RGs proves to work well and effectively in load levelling without intensifying the peak loading of a constrained distribution network. Tests in terms of total load demand curves of 7 successive days and distributions of EV SOCs at the start of each day all verifies the stability of proposed A*-based optimal dispatch strategy, as they all follow a stable and repeatable pattern, respectively, since day 2 (The results for day 1 is mostly dominated by transient responses of network).

In the future, other published algorithms will be tested in the same scenario for further comparison and investigation, such as the centralized optimal dispatch model proposed in [54] that minimize load variance, the hierarchical control approach proposed in [86] that minimizes operating cost, the fuzzy logic controller proposed in [153] that improves the voltage profile of a network, the two-stage stochastic programming approach proposed in [154] that optimizes pricing and maximizes the energy aggregator's profits, and so forth. By examining these existing approaches in the optimal coordination of EVs and RGs, different characteristics of these approaches can be better understood such as computational and communication complexity and impacts on the power network and EVs. Therefore, better decision can be made in terms of which approach will be applied in reality.



(a)



(b)

FIGURE 6.21: SOC Distributions at the End of Every Day

Chapter 7

Conclusions & Future Studies

This chapter gives conclusions of the research presented in this thesis, and suggests future studies.

7.1 Conclusions

Four related pieces of work with the overall aim of improving the smart grid have been presented in this thesis: battery sizing for the relief of thermal overload caused by N-1 contingency; design of a dispatch strategy for EV batteries based on V2G concept; and coordinated dispatch of RGs and EVs using two different algorithms.

Battery sizing for the thermal overload alleviation

Two AHP-based approaches for determining the battery capacity were proposed in Chapter 3 and tested on an IEEE Reliability Test System using numerical simulation. The difference between these two approaches is related to whether or not the relative importance of charging and discharging corrective actions is taken into account when calculating the total capacity of the battery given the proposed charging and discharging capacities. The results lead to the following conclusions:

1. A smaller battery capacity was obtained using the approach taking into account the relative importance of charging and discharging capacities than that of simply summing up the desired charging and discharging capacities.

2. These two approaches have similar performance when dealing with N-1 contingencies, while the battery capacity proposed by the approach that takes into account the relative importance of charging and discharging corrective actions has a better capability of handling N-2 contingencies.

3. Both of these two approaches were proved to be better, in terms of system security and the percentage of handleable contingencies, than the battery sizing method in [22] focusing on the severest condition.

However, if not only contingency thermal overload alleviation but also other power grid operations are expected to be supported by energy storage, the system will need more storage capacity to be distributed in the network with proper dispatch strategy in order for effective system support and the cost savings to be achieved.

Dispatch of EV batteries based on V2G concept using the Analytic Hierarchy Process

As EVs become increasingly popular and their large-scale penetration is inevitable, EV batteries with appropriate control can be used as energy storage providing support for the grid. Here, an AHP-based dispatch strategy for V2G batteries is demonstrated in Chapter 4. The test of feasibility and efficacy of the strategy was conducted on an IEEE Reliability Test System. The simulation results demonstrate the following conclusions:

1. With the AHP-based dispatch strategy the EV battery is capable of providing sufficient electricity for the EV's on-road journeys at a reasonable cost while helping to support the power system's load levelling and alleviate thermal overload caused by a severe N-1 contingency that overloads the branch close to the bus the EV is connected to.

2. The comparisons with a rule-based dispatch approach [20] have been made and show that the AHP-based dispatch strategy is generally better. With the AHP-based strategy, the EV battery has a better SOC situation and better performance in load levelling and thermal overload alleviation during a severe N-1 contingency at a £0.8 higher cost per day, per vehicle.

However, the integration of intermittent renewable generation was not considered in the dispatch strategy of V2G batteries, and the coordination among EVs would need to be improved.

Optimal coordinated dispatch of RGs and EVs in a distribution network

In Chapters 5 & 6, a novel agent-based coordinated dispatch strategy was investigated and developed for both EVs and RGs in a radial distribution network. Two different algorithms were utilized to solve this optimal dispatch problem of EVs and RGs which was formulated as a distributed multi-objective constraint optimization problem: dynamic programming and the A* search procedure. The proposed dispatch strategy using either algorithm has been tested on a modified UK generic radial distribution system in terms of its feasibility, efficacy and stability. Based on the simulation results it can be concluded that:

1. With this agent-based coordinated dispatch strategy, EVs are able to coordinate with each other and with RGs (only wind power is considered in this work) to provide support to the electricity network for load levelling while saving cost to EV users (tens to hundreds of pounds saving per year is expected), ensuring reliable driving experience with sufficient SOC left in the EV battery (at least one third of the battery's available capacity remains at any time during a day), and reducing the waste of renewable power (i.e., nearly 100% of wind power is utilized).

2. Compared to the previous work demonstrated in Chapter 4, the performance of EVs is much better in coordination with each other and with RGs, including load levelling, where the demand peak of network is shaved while the valley is filled by this coordinated dispatch strategy.

3. The stability of the proposed strategy using either dynamic-programming-based algorithm or the A*-based algorithm is tested as well by continuing to run the simulation to determine the behaviour of the system over a period of one week. The similarity of the pattern of performance over a 7-day period verifies the stability of the proposed strategies.

4. The travel patterns of EVs and wind speed were considered as random variables in Chapter 6; they were modelled and simulated stochastically using Gaussian Copulas. The simulation results in Chapter 6 indicate the efficacy of proposed A*-based optimal dispatch strategy and its capability and robustness when dealing with these uncertainties. The dynamic-programming-based optimal dispatch strategy proposed in Chapter 5 was tested using the historic data of wind power and the driving activities of EVs that were randomly generated (using Monte Carlo simulation) from the probability distribution of parked cars over 24 hours. Although this way of simulating random variables is not sophisticated enough to reflect the characters of real data, the simulation results still show the robustness of the proposed dynamic-programming-based optimal dispatch strategy to a certain degree. However, the proposed dynamic-programming-based optimal dispatch strategy is expected to have the same robustness as the A*-based strategy, because their cardinal optimal dispatch problems are exactly the same. Only the algorithms used to solve this problem are different, which will only affect the cost of solution in terms of computation and communication instead of the final solution. This was demonstrated in Chapter 6 in details.

7.2 Future Studies

This research has identified several other areas of research that need to be investigated further. One is the stability of electricity network when changing the dispatch actions of EVs and RGs. The main factor that will be focused on will be the frequency of the network. Different dispatch of EVs and RGs mainly change the active power injection/absorption to/from the system, which will result in the unbalance between active power supply and demand, hence frequency variation [155, 156]. Frequency regulation using either stationary energy storage or EVs has been investigated in many articles. In [157], Lucas et al. discuss the efficacy and potential of a vanadium redox flow battery storage system to provide multi-ancillary services such as frequency regulation and peak shaving for the grid. Xinran Li et al. [158] utilize batteries for primary frequency regulation via variable droop control. V2G batteries are used by Janfeshan et al. [159] to drive the frequency back to its nominal value using fuzzy logic control. In [160], Baboli et al. utilize V2G batteries in the primary frequency control (PFC) of micro-grids showing that the increasing penetration of V2G batteries improves both the transient and steady state frequency of the power system. Izadkhast et al. [161] propose an aggregate model of EVs for provision of PFC while also considering network power loss and cables' current capacity. Therefore, frequency regulation of a distribution network containing EVs and RGs needs to be considered in the future design of dispatch strategy for the transient stability of electricity network.

Moreover, EVs and RGs are very different from their conventional counterparts that are currently prevalent. As they are the new players in the power system and their numbers are increasing significantly, new rules in the electricity market and power system operation need to be set, which requires both the economic and technical analysis of the integration of EVs and RGs: in [70, 71], the environmental and economical assessment of EV and RG systems are presented; Ruan et al. [162] evaluate the economic cost and benefit of EV market; in [163] the revenue of EV market integration concerning energy policy and different market conditions is evaluated; Agarwal et al. [164] present an economic analysis of V2G operations considering market price and battery degradation. New rules should encompass incentive payment for V2G participation, electricity pricing for trades between different suppliers and demanders in a power system with large-scale integration of EVs and RGs. Other standardization policies should also incorporated such as which cars are eligible to provide V2G services, how the power system should be operated and managed in terms of using new techniques or algorithms to meet new goals in addition to the current goals and how the power system structure should evolve. Furthermore, since the future smart grid requires a lot of measurements and communications in order for the accurate and optimal control, cyber security and ICT are also the essential aspects that need to be considered in the context of smart grid, as discussed in the Introduction (Chapter 1). As the field of smart grid related research is very wide, here we focus on the frequency regulation and electricity pricing approach. The future tests of practical operation are also discussed.

7.2.1 Transient Stability — Frequency Regulation

The stability in the steady state is not hard to ensure, but due to the limited response speed of generators and other reserve devices, transient stability of the network may be compromised. In order to make sure the stability in transient state, fast-response devices should be used with appropriate control to keep the variation of frequency within permissible limits.

Many studies discussed the potential of and techniques for using EV batteries for frequency regulation. Falahati et al. [155] propose a fuzzy controller for the smart charging of EVs with respect to frequency regulation of the power grid and vehicles' SOC. Janfeshan et al. [159] also utilize fuzzy logic in V2G batteries control to stabilize frequency. Fuzzy logic is also used in [165] to develop a bidirectional power flow controller for EVs to realize frequency regulation in a system with PV panels. In [166], a droop control strategy is used for EVs to provide primary frequency regulation (PFR). This study also demonstrates great cost saving of PFR and carbon emission reduction in the case of PFR provision from EVs. A decentralized V2G control is proposed in [167] using adaptive droop control, where two different modes of EV management are allowed: any-time mode — EVs can be used at any time by the drivers and participate only in primary frequency regulation; fixed-time mode — only fixed time usage of EVs is allowed and EVs are used for both primary and secondary frequency regulation.

Therefore, the two control approaches that are frequently used in frequency regulation of EV batteries are fuzzy logic and droop control. However, most of the published studies focus on the centralized control of V2G batteries for the frequency regulation. Thus, in the future, by taking into account the dynamics of generators, the decentralized dispatch system for EVs and RGs can be improved and become more consummate. In this future dispatch system, one of the two aforementioned control approaches can be utilized to coordinate EVs and RGs so that the network's frequency varies within a safe range while they gradually change to the optimal dispatch actions determined by the proposed dispatch strategy in this thesis.

7.2.2 Electricity Pricing

With the increasing integration of EVs, RGs and energy storage systems, the constitution of power load and supplies becomes more diverse. The power system's load demand and generation capacity can vary continuously in time and/or space. Furthermore, in the future smart grid, demand response is considered as one of its key characteristics, which indicates that the consumers can adjust their electric load demand according to the electricity price. Therefore, dynamic time-varying pricing strategy needs to be developed in order to better reflect the capacity of generation and load demand and their variations and guide the residential energy consumption in order to realize better operation of the grid and benefit consumers.

Several publications have investigated this area. Toru Namerikawa et al. [19] devise a real-time pricing mechanism using game-theoretic approach, which guarantees greater benefit for consumers and suppliers to participate in a real-time pricing market than in a fixed-price market. Parvania et al. [168] propose a continuous-time marginal electricity pricing as a function of incremental generation and its incremental ramping cost rates, so that it can manifest the behaviour of continuously varying load and generation schedule in a power network. In [169], a minimum electricity price model is proposed for micro-grid network customers by minimizing the cost of electricity generation and transmission. In [170], the real-time price is set by a stochastic optimization problem to schedule EV charging demand with the aim of maximizing the utility's net profit.

As the electricity pricing for optimal EV dispatch in coordination with RGs hasn't been investigated widely in depth. In future studies, real-time pricing strategy can be developed for the dispatch of EVs in coordination with RGs, by taking into account EV users' responses to the different prices related to their G2V/V2G decisions and using stochastic optimization techniques, so that the objectives of dispatch including those discussed in this thesis can be achieved.

7.2.3 Practical Operation Examination

In order to test and refine the methodologies in a real environment and to better understand the evolution and changing nature of supply and demand, a power system lab can be built in the future, which could include a micro-grid testbed, or we can cooperate with other universities that already have this kind of laboratories and testing facilities. The algorithms proposed can be developed into some sorts of software and then installed in the testbed for further experiments. We can also run the tests on a small area of real power network such as our faculty buildings or our campus. The practical experiments will provide the real reflections of a power system on the control methodologies applied. Therefore, appropriate adjustments and further developments can be made in order for the control schemes to better serve the power grid and make the grid truly smarter.

7.2.4 Further Research to Address the Limitations

As discussed in the thesis, there are several limitations in the proposed algorithms and approaches, which should be resolved in the future studies.

As for the battery sizing, the approach proposed in Chapter 3 is not an optimal sizing and is evaluated for the corrective actions following a contingency. Therefore, in the future the sizing model of battery storage should be developed using optimization algorithms and taking into account various power system operational supports that can be provided by battery storage. This can be done by formulating an optimal sizing problem with the objectives of minimizing the penalty costs for being unable to provide certain required system operational supports and solved with existing optimization algorithms such as those embedded within CPLEX optimizer or stochastic optimization algorithm such as particle swarm optimization. The optimal battery sizing model can then be tested by various real case scenarios to verify its feasibility and refine it if needed.

Regarding to the coordinated dispatch of EVs and RGs, the optimal dispatch approaches proposed in this thesis use discrete control variables for the simplification of computation. This however might compromise the optimization result as it might miss the global optimum that can be found by continuous optimization. Therefore, in the future, continuous dispatch of EV charging/discharging currents and RG power outputs will be investigated using the same scenario of this work to derive the global optimal solution for the coordinated dispatch of EVs and RGs in a network. This optimal dispatch problem will be formulated as a centralized continuous optimization problem and solved using CPLEX optimizer. Then comparisons can be made in terms of the optimal solutions derived for discrete and continuous dispatch, respectively, and their communication and computation complexity. Therefore, the pros and cons of these approaches can be further investigated and discussed, which will provide some advices for decision makers to determine which approach is more suitable for practical application. Furthermore, more renewable energy resources can be incorporated into the dispatch model, such as solar energy which is one of the main renewable energy sources but has different behaviours as wind energy, as it is only available during the daytime. Historical data of solar energy in a certain region can be used to form a stochastic model. The simulated stochastic solar power data can then be input to the dispatch model proposed to test its efficacy of coordinating both EVs and different types of RGs.

Reference

[1] IET, "What is smart grid?" [Online], Available: http://www.theiet.org/factfiles/energy/smart-grids-page.cfm.

[2] S. Abu-Sharkh, R. Arnold, J. Kohler, R. Li, T. Markvart, J. Ross, K. Steemers, P. Wilson, and R. Yao, "Can microgrids make a major contribution to UK energy supply?" *Renewable and Sustainable Energy Reviews*, vol. 10, no. 2, pp. 78 – 127, 2006.

[3] A. Soni and C. Ozveren, "Renewable energy market potential in u.k." in *Universities Power Engineering Conference, 2007. UPEC 2007. 42nd International*, Sept 2007, pp. 717–720.

[4] S. Teleke, M. E. Baran, A. Q. Huang, S. Bhattacharya, and L. Anderson, "Control strategies for battery energy storage for wind farm dispatching," *IEEE Transactions on Energy Conversion*, vol. 24, no. 3, pp. 725–732, Sept 2009.

[5] G. Carpinelli, G. Celli, S. Mocci, F. Mottola, F. Pilo, and D. Proto, "Optimal integration of distributed energy storage devices in smart grids," *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 985–995, June 2013.

[6] H. Zhao, Q. Wu, S. Hu, H. Xu, and C. N. Rasmussen, "Review of energy storage system for wind power integration support," *Applied Energy*, vol. 137, pp. 545 – 553, 2015.

[7] S. Alahakoon, "Significance of energy storages in future power networks," *Energy Procedia*, vol. 110, pp. 14 – 19, 2017.

[8] N. Masuch, J. Keiser, M. Lutzenberger, and S. Albayrak, "Wind power-aware vehicle-to-grid algorithms for sustainable ev energy management systems," in *2012 IEEE International Electric Vehicle Conference (IEVC)*, March 2012, pp. 1–7.

[9] K. Zafred, J. Nieto-Martin, and E. Butans, "Electric vehicles - effects on domestic low voltage networks," in *2016 IEEE International Energy Conference (ENERGYCON)*, April 2016, pp. 1–6.

[10] S. Rezaee, E. Farjah, and B. Khorramdel, "Probabilistic analysis of plug-in electric vehicles impact on electrical grid through homes and parking lots," *IEEE Transactions on Sustainable Energy*, vol. 4, no. 4, pp. 1024–1033, Oct 2013.

[11] Y. Yao and D. W. Gao, "Charging load from large-scale plug-in hybrid electric vehicles: Impact and optimization," in *Innovative Smart Grid Technologies (ISGT), 2013 IEEE PES*, Feb 2013, pp. 1–6.

[12] M. Z. Fortes, D. F. Silva, T. P. Abud, P. P. M. Jr, R. S. Maciel, and D. H. N. Dias, "Impact analysis of plug-in electric vehicle connected in real distribution network," *IEEE Latin America Transactions*, vol. 14, no. 5, pp. 2239–2245, May 2016.

[13] F. Mwasilu, J. J. Justo, E.-K. Kim, T. D. Do, and J.-W. Jung, "Electric vehicles and smart grid interaction: A review on vehicle to grid and renewable energy sources integration," *Renewable and Sustainable Energy Reviews*, vol. 34, no. 0, pp. 501 – 516, 2014.

[14] S. S. S. R. Depuru, L. Wang, and V. Devabhaktuni, "Smart meters for power grid: Challenges, issues, advantages and status," *Renewable and Sustainable Energy Reviews*, vol. 15, no. 6, pp. 2736 – 2742, 2011.

[15] O. Erdinc, N. Paterakis, T. Mendes, A. Bakirtzis, and J. Catalao, "Smart household operation considering bi-directional ev and ess utilization by real-time pricing-based dr," *IEEE Transactions on Smart Grid*, vol. 6, no. 3, pp. 1281–1291, May 2015.

[16] V. Gungor, B. Lu, and G. Hancke, "Opportunities and challenges of wireless sensor networks in smart grid," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 10, pp. 3557–3564, Oct 2010.

[17] R. Liu, C. Vellaithurai, S. Biswas, T. Gamage, and A. Srivastava, "Analyzing the cyber-physical impact of cyber events on the power grid," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–1, 2015.

[18] P. Samadi, H. Mohsenian-Rad, V. Wong, and R. Schober, "Real-time pricing for demand response based on stochastic approximation," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 789–798, March 2014.

[19] T. Namerikawa, N. Okubo, R. Sato, Y. Okawa, and M. Ono, "Real-time pricing mechanism for electricity market with built-in incentive for participation," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–1, 2015.

[20] Y. Ma, T. Houghton, A. Cruden, and D. Infield, "Modeling the benefits of vehicle-to-grid technology to a power system," *IEEE Transactions on Power Systems*, vol. 27, no. 2, pp. 1012–1020, May 2012.

[21] EPRI, "Epri-doe handbook of energy storage for transmission & distribution applications," 2003.

[22] A. Del Rosso and S. Eckroad, "Energy storage for relief of transmission congestion," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 1138–1146, March 2014.

[23] M. Arita, A. Yokoyama, and Y. Tada, "Evaluation of battery system for frequency control in interconnected power system with a large penetration of wind power generation," in *2006 International Conference on Power System Technology*, Oct 2006, pp. 1–7.

[24] N. Wade, P. Taylor, P. Lang, and J. Svensson, "Energy storage for power flow management and voltage control on an 11kv uk distribution network," in *20th International Conference on Electricity Distribution*, June 2009, pp. 1–4.

[25] S. Teleke, M. Baran, A. Huang, S. Bhattacharya, and L. Anderson, "Control strategies for battery energy storage for wind farm dispatching," *IEEE Transactions on Energy Conversion*, vol. 24, no. 3, pp. 725–732, Sept 2009.

[26] W.-Y. Chiu, H. Sun, and H. Poor, "Demand-side energy storage system management in smart grid," in *Proc. 2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, 2012, pp. 73–78.

[27] C. U. EPRI, Palo Alto, "Application of storage technoloy for transmission system support: interim report," 2012,1025418.

[28] U. EPRI, Palo Alto, "Use of energy storage to increase transmission capacity," 2011,1024586.

[29] Y. Wen, C. Guo, and S. Dong, "Coordinated control of distributed and bulk energy storage for alleviation of post-contingency overloads," *Energies*, vol. 7, no. 3, pp. 1599–1620, 2014.

[30] P. Ribeiro, B. Johnson, M. Crow, A. Arsoy, and Y. Liu, "Energy storage systems for advanced power applications," *Proceedings of the IEEE*, vol. 89, no. 12, pp. 1744–1756, Dec 2001.

[31] C. Apapipat and K. Audomvongseree, "Determination of the optimal battery capacity of a wind generation system with power fluctuation consideration," in *Proc. 2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2013, pp. 1–4.

[32] K. Ogimi, A. Yoza, A. Yona, T. Senjyu, and T. Funabashi, "A study on optimum capacity of battery energy storage system for wind farm operation with wind power

forecast data," in *Proc. 2012 IEEE 15th International Conference on Harmonics and Quality of Power (ICHQP)*, June 2012, pp. 118–123.

[33] W. Pan, B. Feng, and J. Zhu, "An analysis of capacity requirements of battery energy storage based on forecasted wind power correction," in *Proc. International Conference on Sustainable Power Generation and Supply (SUPERGEN 2012)*, Sept 2012, pp. 1–5.

[34] F. Millo, L. Rolando, R. Fuso, and F. Mallamo, "Real CO2 emissions benefits and end user's operating costs of a plug-in hybrid electric vehicle," *Applied Energy*, vol. 114, no. 0, pp. 563 – 571, 2014.

[35] M. Sorrentino, G. Rizzo, and L. Sorrentino, "A study aimed at assessing the potential impact of vehicle electrification on grid infrastructure and road-traffic green house emissions," *Applied Energy*, vol. 120, no. 0, pp. 31 – 40, 2014.

[36] "Global ev outlook – understanding the electric vehicle landscape to 2020," [Online], Available: https://www.iea.org/publications/globalevoutlook_2013.pdf.

[37] International Energy Agency, "Global ev outlook 2016," [Online], May 2016, Available: http://www.iea.org/publications/freepublications/publication/global-ev-outlook-2016.html.

[38] S. Jeong, Y. J. Jang, and D. Kum, "Economic analysis of the dynamic charging electric vehicle," *IEEE Transactions on Power Electronics*, vol. 30, no. 11, pp. 6368–6377, Nov 2015.

[39] S. Jung, H. Lee, C. S. Song, J. H. Han, W. K. Han, and G. Jang, "Optimal operation plan of the online electric vehicle system through establishment of a dc distribution system," *IEEE Transactions on Power Electronics*, vol. 28, no. 12, pp. 5878–5889, Dec 2013.

[40] J. Shin, S. Shin, Y. Kim, S. Ahn, S. Lee, G. Jung, S. J. Jeon, and D. H. Cho, "Design and implementation of shaped magnetic-resonance-based wireless power transfer system for roadway-powered moving electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1179–1192, March 2014.

[41] "Global EV market forecast," [Online], Dec 2012, Available: http://www.greencarcongress.com/2012/12/anderman-20121218.html.

[42] J. Ayre, "Europe electric car sales reached 1.24% of car sales in 2015," [Online], March 2016, Available: http://cleantechnica.com/2016/03/08/europe-electric-car-sales-reached-1-24-of-car-sales-in-2015/.

[43] A. Verma, R. Raj, M. Kumar, S. Ghandehariun, and A. Kumar, "Assessment of renewable energy technologies for charging electric vehicles in Canada," *Energy*, vol. 86, pp. 548 – 559, 2015.

[44] T. S. Ustun, A. Zayegh, and C. Ozansoy, “Electric vehicle potential in australia: Its impact on smartgrids,” *IEEE Industrial Electronics Magazine*, vol. 7, no. 4, pp. 15–25, Dec 2013.

[45] T. W. Ching and T. H. Cheong, “Cost analysis of electric vehicles in macau,” in *TENCON Spring Conference, 2013 IEEE*, April 2013, pp. 342–346.

[46] B. C. Adolfo Perujo, “The introduction of electric vehicles in the private fleet: Potential impact on the electric supply system and on the environment. a case study for the province of milan, italy,” *Energy Policy*, vol. 38, no. 8, pp. 4549 – 4561, 2010.

[47] B. K. Huib van Essen, “Impact of electric vehicles – summary report,” [Online], April 2011, Available: http://ecologic.eu/8271.

[48] C. M. F. H. B. C. v. D. Florian Salah, Jens P. Ilg, “Impact of electric vehicles on high-voltage grids: A swiss case study,” [Online], May 2013, Available: http://ssrn.com/abstract=2209841.

[49] B. . DfT, “Investigation into the scope for the transport sector to switch to electric vehicles and plugin hybrid vehicles,” [Online], October 2008, Available: http://www.emic-bg.org/files/file48653.pdf.

[50] P. Papadopoulos, S. Skarvelis-Kazakos, I. Grau, L. M. Cipcigan, and N. Jenkins, “Electric vehicles’ impact on british distribution networks,” *IET Electrical Systems in Transportation*, vol. 2, no. 3, pp. 91–102, September 2012.

[51] C. Li, C. Ahn, H. Peng, and J. Sun, “Synergistic control of plug-in vehicle charging and wind power scheduling,” *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1113–1121, May 2013.

[52] J. Peas Lopes, P. Almeida, and F. Soares, “Using vehicle-to-grid to maximize the integration of intermittent renewable energy resources in islanded electric grids,” in *2009 International Conference on Clean Electrical Power*, June 2009, pp. 290–295.

[53] D. Dallinger, S. Gerda, and M. Wietschel, “Integration of intermittent renewable power supply using grid-connected vehicles – a 2030 case study for california and germany,” *Applied Energy*, vol. 104, no. 0, pp. 666 – 682, 2013.

[54] X. Jiang, J. Wang, Y. Han, and Q. Zhao, “Coordination dispatch of electric vehicles charging/discharging and renewable energy resources power in microgrid,” *Procedia Computer Science*, vol. 107, pp. 157 – 163, 2017.

[55] R. Bessa and M. Matos, “Optimization models for an EV aggregator selling secondary reserve in the electricity market,” *Electric Power Systems Research*, vol. 106, pp. 36 – 50, 2014.

[56] L. Chang, X. Jiu, M. Mao, and Y. Wang, "Voltage regulation of microgrids containing electric vehicles," in *2016 IEEE 8th International Power Electronics and Motion Control Conference (IPEMC-ECCE Asia)*, May 2016, pp. 3756–3763.

[57] K. M. Tan, V. K. Ramachandaramurthy, and J. Y. Yong, "Three-phase bidirectional electric vehicle charger for vehicle to grid operation and grid voltage regulation," in *2016 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific)*, June 2016, pp. 007–012.

[58] H. Fathabadi, "Utilization of electric vehicles and renewable energy sources used as distributed generators for improving characteristics of electric power distribution systems," *Energy*, vol. 90, Part 1, pp. 1100 – 1110, 2015.

[59] M. Carrion and R. Zarate-Minano, "Operation of renewable-dominated power systems with a significant penetration of plug-in electric vehicles," *Energy*, vol. 90, Part 1, pp. 827 – 835, 2015.

[60] W. Kempton and J. Tomi, "Vehicle-to-grid power fundamentals: Calculating capacity and net revenue," *Journal of Power Sources*, vol. 144, no. 1, pp. 268 – 279, 2005.

[61] J. Lopes, F. Soares, and P. Almeida, "Integration of electric vehicles in the electric power system," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 168–183, Jan 2011.

[62] E. B. Iversen, J. M. Morales, and H. Madsen, "Optimal charging of an electric vehicle using a markov decision process," *Applied Energy*, vol. 123, no. 0, pp. 1 – 12, 2014.

[63] J. Yang, L. He, and S. Fu, "An improved pso-based charging strategy of electric vehicles in electrical distribution grid," *Applied Energy*, vol. 128, no. 0, pp. 82 – 92, 2014.

[64] P. Richardson, D. Flynn, and A. Keane, "Optimal charging of electric vehicles in low-voltage distribution systems," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 268–279, Feb 2012.

[65] M. Singh, P. Kumar, and I. Kar, "Implementation of vehicle to grid infrastructure using fuzzy logic controller," *IEEE Transactions on Smart Grid*, vol. 3, no. 1, pp. 565–577, March 2012.

[66] L. Yang, J. Zhang, and H. Poor, "Risk-aware day-ahead scheduling and real-time dispatch for electric vehicle charging," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 693–702, March 2014.

[67] M. Yazdani-Damavandi, M. Moghaddam, M.-R. Haghifam, M. Shafie-khah, and J. Catalao, "Modeling operational behavior of plug-in electric vehicles' parking lot

in multienergy systems," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–1, 2015.

[68] L. Hua, J. Wang, and C. Zhou, "Adaptive electric vehicle charging coordination on distribution network," *IEEE Transactions on Smart Grid*, vol. 5, no. 6, pp. 2666–2675, Nov 2014.

[69] B. Dollinger and K. Dietrich, "Storage systems for integrating wind and solar energy in spain," in *Renewable Energy Research and Applications (ICRERA), 2013 International Conference on*, Oct 2013, pp. 361–365.

[70] J. J. Garca, R. Enrich, and M. Torrent-Moreno, "A greening energy positive tool for energy management in infrastructures and buildings of public use: Technical and economic assessment framework for the integrated management of pv and ev systems," in *Sustainable Technologies (WCST), 2013 World Congress on*, Dec 2013, pp. 42–46.

[71] G. R. C. Mouli, M. Leendertse, V. Prasanth, P. Bauer, S. Silvester, S. van de Geer, and M. Zeman, "Economic and co2 emission benefits of a solar powered electric vehicle charging station for workplaces in the netherlands," in *2016 IEEE Transportation Electrification Conference and Expo (ITEC)*, June 2016, pp. 1–7.

[72] J. Garca-Villalobos, I. Zamora, J. S. Martn, F. Asensio, and V. Aperribay, "Plug-in electric vehicles in electric distribution networks: A review of smart charging approaches," *Renewable and Sustainable Energy Reviews*, vol. 38, pp. 717 – 731, 2014.

[73] J. P. Lopes, N. Hatziargyriou, J. Mutale, P. Djapic, and N. Jenkins, "Integrating distributed generation into electric power systems: A review of drivers, challenges and opportunities," *Electric Power Systems Research*, vol. 77, no. 9, pp. 1189 – 1203, 2007.

[74] W. Steel, "Spain closes in on 50 percent renewable power generation," [Online], September 2016, Available: http://www.renewableenergyworld.com/articles/2016/09/spain-closes-in-on-50-percent-renewable-power-generation.html.

[75] H. Jacobsen, "Denmark breaks its own world record in wind energy," [Online], 2016, Available:http://www.euractiv.com/section/climate-environment/news/denmark-breaks-its-own-world-record-in-wind-energy/.

[76] E. Commission, "2020 energy strategy," [Online], 2010, Available: https://ec.europa.eu/energy/en/topics/energy-strategy/2020-energy-strategy.

[77] P. Asmus, "Microgrids, virtual power plants and our distributed energy future," *The Electricity Journal*, vol. 23, no. 10, pp. 72 – 82, 2010.

[78] P. Karimyan, M. Abedi, S. H. Hosseinian, and R. Khatami, "Stochastic approach to represent distributed energy resources in the form of a virtual power plant in energy and reserve markets," *IET Generation, Transmission Distribution*, vol. 10, no. 8, pp. 1792–1804, May 2016.

[79] S. R. Dabbagh and M. K. Sheikh-El-Eslami, "Risk assessment of virtual power plants offering in energy and reserve markets," *IEEE Transactions on Power Systems*, vol. 31, no. 5, pp. 3572–3582, Sept 2016.

[80] A. F. Raab, M. Ferdowsi, E. Karfopoulos, I. G. Unda, S. Skarvelis-Kazakos, P. Papadopoulos, E. Abbasi, L. M. Cipcigan, N. Jenkins, N. Hatziargyriou, and K. Strunz, "Virtual power plant control concepts with electric vehicles," in *Intelligent System Application to Power Systems (ISAP), 2011 16th International Conference on*, Sept 2011, pp. 1–6.

[81] B. Jansen, C. Binding, O. Sundstrom, and D. Gantenbein, "Architecture and communication of an electric vehicle virtual power plant," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, Oct 2010, pp. 149–154.

[82] M. Vasirani, R. Kota, R. L. G. Cavalcante, S. Ossowski, and N. R. Jennings, "An agent-based approach to virtual power plants of wind power generators and electric vehicles," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1314–1322, Sept 2013.

[83] J. Soares, N. Borges, C. Lobo, and Z. Vale, "Vpp energy resources management considering emissions: The case of northern portugal 2020 to 2050," in *Computational Intelligence, 2015 IEEE Symposium Series on*, Dec 2015, pp. 1259–1266.

[84] Z. Li, Q. Guo, H. Sun, Y. Wang, and S. Xin, "Emission-concerned wind-ev coordination on the transmission grid side with network constraints: Concept and case study," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1692–1704, Sept 2013.

[85] T. Zhang, W. Chen, Z. Han, and Z. Cao, "Charging scheduling of electric vehicles with local renewable energy under uncertain electric vehicle arrival and grid power price," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 6, pp. 2600–2612, July 2014.

[86] S. Gao, K. Chau, C. Liu, D. Wu, and C. Chan, "Integrated energy management of plug-in electric vehicles in power grid with renewables," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 7, pp. 3019–3027, Sept 2014.

[87] E. L. Karfopoulos and N. D. Hatziargyriou, "Distributed coordination of electric vehicles providing v2g services," *IEEE Transactions on Power Systems*, vol. 31, no. 1, pp. 329–338, Jan 2016.

[88] C. Wang, Y. Liu, X. Li, L. Guo, L. Qiao, and H. Lu, "Energy management system for stand-alone diesel-wind-biomass microgrid with energy storage system," *Energy*, vol. 97, pp. 90 – 104, 2016.

[89] V. Mohan, J. G. Singh, and W. Ongsakul, "An efficient two stage stochastic optimal energy and reserve management in a microgrid," *Applied Energy*, vol. 160, pp. 28 – 38, 2015.

[90] C. Chen, F. Wang, B. Zhou, K. W. Chan, Y. Cao, and Y. Tan, "An interval optimization based day-ahead scheduling scheme for renewable energy management in smart distribution systems," *Energy Conversion and Management*, vol. 106, pp. 584 – 596, 2015.

[91] M. Peik-herfeh, H. Seifi, and M. K. Sheikh-El-Eslami, "Two-stage approach for optimal dispatch of distributed energy resources in distribution networks considering virtual power plant concept," *International Transactions on Electrical Energy Systems*, vol. 24, no. 1, pp. 43–63, 2014.

[92] L. Yu, T. Zhao, Q. Chen, and J. Zhang, "Centralized bi-level spatial-temporal coordination charging strategy for area electric vehicles," *CSEE Journal of Power and Energy Systems*, vol. 1, no. 4, pp. 74–83, Dec 2015.

[93] M. Granada, M. Rider, J. R. Mantovani, and M. Shahidehpour, "Multi-areas optimal reactive power flow," in *IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, Colombia, Aug 2008, pp. 1–6.

[94] L. Wang, S. Sharkh, and A. Chipperfield, "Optimal coordination of vehicle-to-grid batteries and renewable generators in a distribution system," *Energy*, vol. 113, pp. 1250 – 1264, 2016.

[95] Y. Zhang, N. Gatsis, and G. Giannakis, "Robust energy management for microgrids with high-penetration renewables," *IEEE Transactions on Sustainable Energy*, vol. 4, no. 4, pp. 944–953, Oct 2013.

[96] C. Dou, W. Wang, D. Hao, and X. Li, "Mas-based solution to energy management strategy of distributed generation system," *International Journal of Electrical Power & Energy Systems*, vol. 69, pp. 354 – 366, 2015.

[97] F. Ren, M. Zhang, and D. Sutanto, "A multi-agent solution to distribution system management by considering distributed generators," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1442–1451, May 2013.

[98] S. Miller, S. D. Ramchurn, and A. Rogers, "Optimal decentralised dispatch of embedded generation in the smart grid," in *Proceedings of 11th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Spain, June 2012, pp. 281–288.

[99] J. Rezaei, "Best-worst multi-criteria decision-making method: Some properties and a linear model," *Omega*, vol. 64, pp. 126 – 130, 2016.

[100] H. Zhang, "Group decision making based on incomplete multiplicative and fuzzy preference relations," *Applied Soft Computing*, vol. 48, pp. 735 – 744, 2016.

[101] F. Jin, Z. Ni, H. Chen, and Y. Li, "Approaches to decision making with linguistic preference relations based on additive consistency," *Applied Soft Computing*, vol. 49, pp. 71 – 80, 2016.

[102] B. Wu, X. Yan, Y. Wang, and C. G. Soares, "Selection of maritime safety control options for NUC ships using a hybrid group decision-making approach," *Safety Science*, vol. 88, pp. 108 – 122, 2016.

[103] Y. Yan, B. Zhang, and J. Guo, "An adaptive decision making approach based on reinforcement learning for self-managed cloud applications," in *2016 IEEE International Conference on Web Services (ICWS)*, June 2016, pp. 720–723.

[104] P. V. Krishna, S. Misra, D. Nagaraju, V. Saritha, and M. S. Obaidat, "Learning automata based decision making algorithm for task offloading in mobile cloud," in *2016 International Conference on Computer, Information and Telecommunication Systems (CITS)*, July 2016, pp. 1–6.

[105] Z. Hao, Z. Xu, H. Zhao, and R. Zhang, "Novel intuitionistic fuzzy decision making models in the framework of decision field theory," *Information Fusion*, vol. 33, pp. 57 – 70, 2017.

[106] S. Wibowo and H. Deng, "Evaluating the performance of cloud services: A fuzzy multicriteria group decision making approach," in *2016 International Symposium on Computer, Consumer and Control (IS3C)*, July 2016, pp. 327–332.

[107] P. Ekel, I. Kokshenev, R. Parreiras, W. Pedrycz, and J. P. Jr., "Multiobjective and multiattribute decision making in a fuzzy environment and their power engineering applications," *Information Sciences*, vol. 361 - 362, pp. 100 – 119, 2016.

[108] T. J. Ross, *Fuzzy Logic with Engineering Applications.* John Wiley & Sons, 2010.

[109] MathWorks, "What is fuzzy logic?" [Online], 2016, Available: http://uk.mathworks.com/help/fuzzy/what-is-fuzzy-logic.html.

[110] ——, "Foundations of fuzzy logic," [Online], 2016, Available: http://uk.mathworks.com/help/fuzzy/foundations-of-fuzzy-logic.html.

[111] K. K. Pedapenki, S. P. Gupta, and M. K. Pathak, "Comparison of fuzzy logic and neuro fuzzy controller for shunt active power filter," in *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, Dec 2015, pp. 1247–1250.

[112] M. N. Uddin and S. R. Ronald, "Neuro-fuzzy and fuzzy logic controllers based speed control of ipmsm drive — a torque ripple optimization approach," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, Nov 2010, pp. 2242–2247.

[113] S. Vedula, N. K. Pandey, and R. Nese, "An adaptive neuro-fuzzy logic based systematic approach of modelling virtual driver for real world driving," in *2015 IEEE International Transportation Electrification Conference (ITEC)*, Aug 2015, pp. 1–12.

[114] X. Bajrami, A. Drmaku, N. Demaku, S. Maloku, A. Kikaj, and A. Kokaj, "Genetic and fuzzy logic algorithms for robot path finding," in *2016 5th Mediterranean Conference on Embedded Computing (MECO)*, June 2016, pp. 195–199.

[115] S. M. Odeh, "Hybrid algorithm: Fuzzy logic-genetic algorithm on traffic light intelligent system," in *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*, Aug 2015, pp. 1–7.

[116] T. L. Saaty, *The Analytic Hierarchy Process*. New York: McGraw Hill, 1980.

[117] J. Zhu and M. Irving, "Combined active and reactive dispatch with multiple objectives using an analytic hierarchical process," *IEE Proceedings–Generation, Transmission and Distribution*, vol. 143, no. 4, pp. 344–352, Jul 1996.

[118] J. Zhu and J. A. Momoh, "Optimal VAr pricing and VAr placement using analytic hierarchical process," *Electric Power Systems Research*, vol. 48, no. 1, pp. 11 – 17, 1998.

[119] R. L. G. Latimier, B. Multon, H. B. Ahmed, F. Baraer, and M. Acquitter, "Stochastic optimization of an electric vehicle fleet charging with uncertain photovoltaic production," in *2015 International Conference on Renewable Energy Research and Applications (ICRERA)*, Nov 2015, pp. 721–726.

[120] V. Kekatos, G. Wang, and G. B. Giannakis, "Stochastic loss minimization for power distribution networks," in *North American Power Symposium (NAPS), 2014*, Sept 2014, pp. 1–6.

[121] W. Wei, F. Liu, and S. Mei, "Charging strategies of ev aggregator under renewable generation and congestion: A normalized nash equilibrium approach," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1630–1641, May 2016.

[122] Q. Huang, Q. S. Jia, Z. Qiu, X. Guan, and G. Deconinck, "Matching ev charging load with uncertain wind: A simulation-based policy improvement approach," *IEEE Transactions on Smart Grid*, vol. 6, no. 3, pp. 1425–1433, May 2015.

[123] G. Wang, J. Zhao, F. Wen, Y. Xue, and G. Ledwich, "Dispatch strategy of phevs to mitigate selected patterns of seasonally varying outputs from renewable generation," *IEEE Transactions on Smart Grid*, vol. 6, no. 2, pp. 627–639, March 2015.

[124] A. Lojowska, D. Kurowicka, G. Papaefthymiou, and L. van der Sluis, "Stochastic modeling of power demand due to evs using copula," *IEEE Transactions on Power Systems*, vol. 27, no. 4, pp. 1960–1968, Nov 2012.

[125] R.B.Nelsen, *An Introduction to Copulas*. New York: Springer, 1999.

[126] A. Sklar, "Fonctions de rpartition n dimensions et leurs marges," *Publications de lInstitut de Statistique de LUniversit de Paris*, vol. 8, pp. 229–231, 1959.

[127] B. K. M. J. Muhsin Abdurrahman, Scott Baker, "Energy storage as a transmission asset," [Online], 2012, Available: https://www.pjm.com/~/media/markets-ops/advanced-tech-pilots/xtreme-power-storage-as-transmission.ashx.

[128] P. M. Subcommittee, "IEEE reliability test system," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-98, no. 6, pp. 2047–2054, Nov 1979.

[129] L. Wang, S. Sharkh, and A. Chipperfield, "Determination of battery capacity for alleviation of thermal overload," in *IEEE PES Innovative Smart Grid Technologies, Europe*, Oct 2014, pp. 1–6.

[130] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "Matpower: Steady-state operations, planning and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12–19, Feb 2011.

[131] V. Cancer, "Criteria weighting by using the 5ws & h technique," *Business Systems Research*, vol. 3, no. 2, pp. 41–48, September 2012.

[132] M. O. Brendan Kirby, Ookie Ma, "The value of energy storage for grid applications," [Online], May 2013, Available: http://www.nrel.gov/docs/fy13osti/58465.pdf.

[133] J. J. D. P. B. K. O. M. Marissa Hummon, Paul Denholm, "Fundamental drivers of the cost and price of operating reserves," [Online], July 2013, Available: http://www.nrel.gov/docs/fy13osti/58491.pdf.

[134] A. Foley, B. Tyther, P. Calnan, and B. O. Gallachoir, "Impacts of electric vehicle charging under electricity market operations," *Applied Energy*, vol. 101, no. 0, pp. 93 – 102, 2013.

[135] H. Khayyam, J. Abawajy, B. Javadi, A. Goscinski, A. Stojcevski, and A. Bab-Hadiashar, "Intelligent battery energy management and control for vehicle-to-grid

via cloud computing network," *Applied Energy*, vol. 111, no. 0, pp. 971 – 981, 2013.

[136] J. Larminie and J. Lowry, *Electric Vehicle Technology Explained.* England: John Wiley & Sons, 2003.

[137] D. Doerffel and S. M. Sharkh, "A critical review of using the peukert equation for determining the remaining capacity of lead-acid and lithium-ion batteries," *Journal of Power Sources*, vol. 155, no. 2, pp. 395 – 400, 2006.

[138] O. Tremblay, L. Dessaint, and A. Dekkiche, "A generic battery model for the dynamic simulation of hybrid electric vehicles," in *2007 IEEE Vehicle Power and Propulsion Conference*, Sept 2007, pp. 284–289.

[139] S. Han, S. Han, and H. Aki, "A practical battery wear model for electric vehicle charging applications," *Applied Energy*, vol. 113, pp. 1100 – 1108, 2014.

[140] SAFT, "Lithium-ion battery life," [Online], Available: http://www.saftbatteries.com/force_download/li_ion_battery_life__TechnicalSheet_en_0514_Protected.pdf.

[141] "UK national grid," [Online], Jan 2014, Available: http://www.bmreports.com/bwx_reporting.htm.

[142] Y. Wang, D. Infield, and S. Gill, "Smart charging for electric vehicles to minimize charging cost," *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, pp. 1–15, 12 2016.

[143] M. Shafie-khah and J. P. S. Catalao, "A stochastic multi-layer agent-based model to study electricity market participants behavior," *IEEE Transactions on Power Systems*, vol. 30, no. 2, pp. 867–881, March 2015.

[144] "UK national grid status," [Online], Jan 2014, Available: http://www.gridwatch.templar.co.uk/index.php.

[145] "UK generic distribution network," [Online], Oct 2013, Available: http://www.sedg.ac.uk/ukgds.htm.

[146] E. Sortomme, M. M. Hindi, S. D. J. MacPherson, and S. S. Venkata, "Coordinated charging of plug-in hybrid electric vehicles to minimize distribution system losses," *IEEE Transactions on Smart Grid*, vol. 2, no. 1, pp. 198–205, March 2011.

[147] K. Clement-Nyns, E. Haesen, and J. Driesen, "The impact of charging plug-in hybrid electric vehicles on a residential distribution grid," *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 371–380, Feb 2010.

[148] P. H. Winston, *Artificial Intelligence.* USA: Addison Wesley, 1992.

[149] L. Wang, S. Sharkh, and A. Chipperfield, "Decentralized coordination of distributed generators in a distribution network using A star," in *2016 IEEE International Conference on Power System Technology (POWERCON)*, Sept 2016, pp. 1–6.

[150] G. Papaefthymiou and D. Kurowicka, "Using copulas for modeling stochastic dependence in power system uncertainty analysis," *IEEE Transactions on Power Systems*, vol. 24, no. 1, pp. 40–49, Feb 2009.

[151] M. Markowitz, "Wells to wheels: electric car efficiency," [Online], Feb 2013, Available: https://matter2energy.wordpress.com/2013/02/22/wells-to-wheels-electric-car-efficiency/.

[152] "Wind speed distribution for different sites," [Online], 2016, Available: http://www.rensmart.com/Weather/Metar.

[153] S. Faddel, A. A. Mohamed, and O. A. Mohammed, "Fuzzy logic-based autonomous controller for electric vehicles charging under different conditions in residential distribution systems," *Electric Power Systems Research*, vol. 148, pp. 48 – 58, 2017.

[154] J. Soares, M. A. F. Ghazvini, N. Borges, and Z. Vale, "Dynamic electricity pricing for electric vehicles using stochastic programming," *Energy*, vol. 122, pp. 111 – 127, 2017.

[155] S. Falahati, S. A. Taher, and M. Shahidehpour, "A new smart charging method for EVs for frequency control of smart grid," *International Journal of Electrical Power & Energy Systems*, vol. 83, pp. 458 – 469, 2016.

[156] M. Marinelli, S. Martinenas, K. Knezovic, and P. B. Andersen, "Validating a centralized approach to primary frequency control with series-produced electric vehicles," *Journal of Energy Storage*, vol. 7, pp. 63 – 73, 2016.

[157] A. Lucas and S. Chondrogiannis, "Smart grid energy storage controller for frequency regulation and peak shaving, using a vanadium redox flow battery," *International Journal of Electrical Power & Energy Systems*, vol. 80, pp. 26 – 36, 2016.

[158] X. Li, Y. Huang, J. Huang, S. Tan, M. Wang, T. Xu, and X. Cheng, "Modeling and control strategy of battery energy storage system for primary frequency regulation," in *Power System Technology (POWERCON), 2014 International Conference on*, Oct 2014, pp. 543–549.

[159] K. Janfeshan, M. Masoum, and S. Deilami, "V2g application to frequency regulation in a microgrid using decentralized fuzzy controller," in *Modelling, Identification Control (ICMIC), 2014 Proceedings of the 6th International Conference on*, Dec 2014, pp. 361–364.

[160] P. T. Baboli, F. Fallahi, M. P. Moghaddam, and E. Alishahi, "Micro-grid's primary frequency control by supporting vehicle-to-grid concept," in *Control, Instrumentation and Automation (ICCIA), 2011 2nd International Conference on*, Dec 2011, pp. 74–78.

[161] S. Izadkhast, P. Garcia-Gonzalez, P. Fras, L. Ramrez-Elizondo, and P. Bauer, "An aggregate model of plug-in electric vehicles including distribution network characteristics for primary frequency control," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 2987–2998, July 2016.

[162] R. Qiantu, X. Wei, and W. Libo, "Research of market potential and policy mechanism on how power grid enterprises make roles in ev charging infrastructure," in *Electricity Distribution (CICED), 2012 China International Conference on*, Sept 2012, pp. 1–11.

[163] B. Illing and O. Warweg, "Achievable revenues for electric vehicles according to current and future energy market conditions," in *2016 13th International Conference on the European Energy Market (EEM)*, June 2016, pp. 1–5.

[164] L. Agarwal, W. Peng, and L. Goel, "Using ev battery packs for vehicle-to-grid applications: An economic analysis," in *2014 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, May 2014, pp. 663–668.

[165] D. Karun and S. T. K, "Fuzzy logic based load frequency control of grid connected distributed generators," in *2015 International Conference on Advancements in Power and Energy (TAP Energy)*, June 2015, pp. 432–437.

[166] F. Teng, Y. Mu, H. Jia, J. Wu, P. Zeng, and G. Strbac, "Challenges on primary frequency control and potential solution from {EVs} in the future {GB} electricity system," *Applied Energy*, 2016.

[167] L. Chen, Y. Jiang, X. Li, L. Yao, X. Xu, and T. Geng, "Frequency regulation strategy for decentralized v2g control," in *2015 5th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, Nov 2015, pp. 2626–2629.

[168] M. Parvania and R. Khatami, "Continuous-time marginal pricing of electricity," *IEEE Transactions on Power Systems*, 2016.

[169] M. H. K. Tushar and C. Assi, "Optimal electricity pricing in a microgrid network," in *2016 IEEE/PES Transmission and Distribution Conference and Exposition (T D)*, May 2016, pp. 1–5.

[170] N. Y. Soltani, S. J. Kim, and G. B. Giannakis, "Real-time load elasticity tracking and pricing for electric vehicle charging," *IEEE Transactions on Smart Grid*, vol. 6, no. 3, pp. 1303–1313, May 2015.

Appendix A

Verification of Virtual Sub-Node Concept

In order to verify that using the virtual sub-node concept to simplify the model will not compromise the optimal dispatch results in this work, it needs to be proved that the following two optimization problems are equivalent (i.e., result in the same optimal solutions):

Problem 1:

$$\begin{aligned} &\min \sum_{c\in chi(v_1)} pec(f_{c1}) \\ s.t. \quad &\sum_{c\in chi(v_1)} f_{c1} + load^1_{fix} \leqslant C_{01} \end{aligned} \tag{A.1}$$

Problem 2:

$$\begin{aligned} &\min \sum_{c\in chi(v_1^n)} pec(f_{c1}) \\ s.t. \quad &\sum_{c\in chi(v_1^n)} f_{c1} \leqslant C^n_{01}, \quad n=1,2,3. \\ &\sum_{n=1}^{3} C^n_{01} + load^1_{fix} = C_{01} \\ C^1_{01} : C^2_{01} : C^3_{01} = &\sum_{d\in chi(v_1^1)} C_{d1} : \sum_{d\in chi(v_1^2)} C_{d1} : \sum_{d\in chi(v_1^3)} C_{d1}. \end{aligned} \tag{A.2}$$

First of all, because the three virtual sub-nodes are independent, it is clear that:

$$\begin{aligned} \min \sum_{c \in chi(v_1)} pec(f_{c1}) &= \sum_{n=1}^{3} \left(\min \sum_{c \in chi(v_1^n)} pec(f_{c1}) \right) \\ s.t. \qquad \sum_{c \in chi(v_1)} f_{c1} &+ load_{fix}^1 \leqslant C_{01}, \end{aligned} \tag{A.3}$$

which can be easily proved by contradiction. Then, the next step is to prove that problem 2 shares the same optimal solution with the following problem:

$$\begin{aligned} &\min \sum_{c \in chi(v_1^n)} pec(f_{c1}) \\ s.t. \qquad &\sum_{c \in chi(v_1)} f_{c1} + load_{fix}^1 \leqslant C_{01}, \end{aligned} \tag{A.4}$$

which is an equivalent substitute of problem 1 thus denoted as problem 1'.

As the proposed simplification process (i.e., problem 2) is adding more constraints on the optimization problem compared to problems 1 and 1', the optimum solved from problems 1 or 1' might be cut out. Therefore, a direct way is to prove that the global optimum stays within the constraints of problem 2. By running the simulation to confirm that the optimal solution of problem 2 is the same as that of the unconstrained optimization problem:

$$\min \sum_{c \in chi(v_1^n)} pec(f_{c1}), \tag{A.5}$$

the global optimum is verified to meet the constraints of problem 2, that is, the optimal solutions of problem 1, 1' and 2 are all the same and equal to the global optimum. In other words, as long as the capacities of the virtual sub-cables are set such that they are in proportion to the total capacities of the downstream cables, as defined in (5.1) and (5.2), this simplification process realizes the equivalence in terms of deriving the same optimal dispatch solution while reducing the computation burden of the agent.

Appendix B

MATALAB Code Produced for the Research

Codes produced for studies in Chapter 4

Main Function:

```
define_constants;
mpc=loadcase('case24_ieee_rts1');
EVnumber=xlsread('bus_number.xlsx','Sheet1','N26:N49');
EVpark=xlsread('parking.xlsx','Sheet1','A1:A48');
con=xlsread('day_contingency.xlsx','Sheet1','B1:B48'); %no contingency

[loadp,loadq,tot_demand,mindemand,maxdemand]=demand;
[sbp, ssp, asbp, assp, hbp, hsp]=price;
percentEV=1;
iter_energy=zeros(1,1);
for iteration=1:1

day=7;
groupev=zeros(24,48*day);
cur_parkEV=zeros(24,48*day);
groupnum=zeros(24,48*day);
normgroupev=zeros(11,24);
addEV=zeros(24,48*day);
addgroup=zeros(24,48*day);
SOC=zeros(11,24);
grouppower=zeros(1000,24);
groupSOC=zeros(1000,24);
evnum=zeros(1000,24);
gnum=zeros(24,1);
gnum_time=zeros(7,24);
buspower=zeros(24,48*day);
buspower_test=zeros(24,48*day);
load=zeros(48*day,1);
day_energy=zeros(48,1);
energy=0;
d=0;
for time=1:48*day
    if mod(time,48)==0
        tt=48;
    else
        tt=mod(time,48);
    end

    mpc.bus(1:24,PD)=loadp(:,tt);
    %mpc.bus(1:24,QD)=loadq(:,time);
    display(time);
    t_mpc=mpc;
    for bus=1:24
        display(bus);
        cur_parkEV(bus,time)=percentEV*EVnumber(bus)*EVpark(tt);%total number of EVs at this bus this moment, unit is 1000xA%.
        if tt==1
            if time==1
                if bus~=7 && bus~=13
                    groupev(bus,time)=1000*percentEV*EVpark(tt); %how many EVs per group
                    groupnum(bus,time)=EVnumber(bus);%how many groups
                else
                    if bus==7
                        groupev(bus,time)=5000*percentEV*EVpark(tt); %how many EVs per group
                        groupnum(bus,time)=EVnumber(bus)/5;%how many groups
                    else
                        if bus==13
                            groupev(bus,time)=80000*percentEV*EVpark(tt); %how many EVs per group
                            groupnum(bus,time)=EVnumber(bus)/80;%how many groups
                        end
                    end
                end
                int_SOC=normrnd(0.6,0.1,groupnum(bus,time),1);
                [groupmember,newSOC]=group(int_SOC,groupnum(bus,time));
                normgroupev(:,bus)=groupmember; %how many groups in every norm distribution interval
                for i=1:11
                    SOC(i,bus)=(i-1)/10;
                end

                for i=1:11
                    if normgroupev(i,bus)~=0
```

```
                    gnum(bus)=gnum(bus)+1; %how many groups in total at each bus
                    groupSOC(gnum(bus),bus)=SOC(i,bus);
                    evnum(gnum(bus),bus)=groupev(bus,time)*normgroupev(i,bus); %how many cars per group
                end
            end
            for g=1:gnum(bus)

                [powerout, SOCnexttime]=totalEVV2G(groupSOC(g,bus),bus, sbp(tt), ssp(tt), asbp(tt), assp(tt), hbp, hsp, con(tt),loadp(:,tt), tot_demand(tt),mindemand, maxdemand);
                %[powerout, SOCnexttime]=totalEVold(groupSOC(g,bus),tt,asbp,assp,hbp,hsp);
                %[powerout, SOCnexttime]=totalEVnodispatch(groupSOC(g,bus),tt);

                groupSOC(g,bus)=SOCnexttime;%update the SOC

                grouppower(g,bus)=powerout*evnum(g,bus); % unit is Watt

            end
        else

            [groupmember, newevnum]=regroup(groupSOC(:,bus),gnum(bus),evnum(:,bus));
            gnum=zeros(24,1);
            normgroupev(:,bus)=groupmember;
            groupSOC(:,bus)=zeros;
            for i=1:11
                if normgroupev(i,bus)~=0
                    gnum(bus)=gnum(bus)+1; %how many groups in total at each bus
                    groupSOC(gnum(bus),bus)=SOC(i,bus);
                    evnum(gnum(bus),bus)=newevnum(i); %how many cars per group
                end
            end
            for g=1:gnum(bus)

                [powerout, SOCnexttime]=totalEVV2G(groupSOC(g,bus),bus, sbp(tt), ssp(tt), asbp(tt), assp(tt), hbp, hsp, con(tt),loadp(:,tt), tot_demand(tt),mindemand, maxdemand);
                %[powerout, SOCnexttime]=totalEVold(groupSOC(g,bus),tt,asbp,assp,hbp,hsp);
                %[powerout, SOCnexttime]=totalEVnodispatch(groupSOC(g,bus),tt);

                groupSOC(g,bus)=SOCnexttime;%update the SOC

                grouppower(g,bus)=powerout*evnum(g,bus); % unit is Watt

            end

        end

    else
        if cur_parkEV(bus,time)<=cur_parkEV(bus,time-1)

            for g=1:gnum(bus)
                if tt==1
                    evnum(g,bus)=evnum(g,bus)/EVpark(48)*EVpark(tt);
                else
                    evnum(g,bus)=evnum(g,bus)/EVpark(tt-1)*EVpark(tt);
                end
                [powerout, SOCnexttime]=totalEVV2G(groupSOC(g,bus),bus, sbp(tt), ssp(tt), asbp(tt), assp(tt), hbp, hsp, con(tt),loadp(:,tt), tot_demand(tt),mindemand, maxdemand);
                %[powerout, SOCnexttime]=totalEVold(groupSOC(g,bus),tt,asbp,assp,hbp,hsp);
                %[powerout, SOCnexttime]=totalEVnodispatch(groupSOC(g,bus),tt);
                groupSOC(g,bus)=SOCnexttime;
                grouppower(g,bus)=powerout*evnum(g,bus);% unit is W


            end
        else
            addEV(bus,time)=cur_parkEV(bus,time)-cur_parkEV(bus,time-1); % parking cars increase
            if addEV(bus,time)>10 %actual number >10000
                addgroup(bus,time)=floor(addEV(bus,time)/10);
                addSOC=normrnd(0.5,0.1,addgroup(bus,time),1);
                [groupmember, maddnewSOC]=group(addSOC,addgroup(bus,time));
                mnormgroupev=groupmember;
                raddnum=(addEV(bus,time)/10-addgroup(bus,time))*10000;
                raddSOC=normrnd(0.5,0.1,1,1);
                [groupmember, raddnewSOC]=group(raddSOC,1);
                rnormgroupev=groupmember;
                addevnum=10000*mnormgroupev+raddnum*rnormgroupev;
                normgroupev(:,bus)=mnormgroupev+rnormgroupev;
                for i=1:11
                    if normgroupev(i,bus)~=0
                        gnum(bus)=gnum(bus)+1;
                        groupSOC(gnum(bus),bus)=SOC(i,bus);
                        evnum(gnum(bus),bus)=addevnum(i);
                    end
                end
            else
                if addEV(bus,time)>1 %actual number >1000
                    addgroup(bus,time)=floor(addEV(bus,time));
                    addSOC=normrnd(0.5,0.1,addgroup(bus,time),1);
```

```
                [groupmember,~]=group(addSOC,addgroup(bus,time));
                mnormgroupev=groupmember;
                raddnum=(addEV(bus,time)-addgroup(bus,time))*1000;
                raddSOC=normrnd(0.5,0.1,1,1);
                [groupmember,newSOC]=group(raddSOC,1);
                rnormgroupev=groupmember;
                addevnum=1000*mnormgroupev+raddnum*rnormgroupev;
                normgroupev(:,bus)=mnormgroupev+rnormgroupev;
                for i=1:11
                    if normgroupev(i,bus)~=0
                        gnum(bus)=gnum(bus)+1;
                        groupSOC(gnum(bus),bus)=SOC(i,bus);
                        evnum(gnum(bus),bus)=addevnum(i);
                    end
                end


            else
                if addEV(time)>0.1
                    addgroup(bus,time)=floor(addEV(bus,time)*10);
                    addSOC=normrnd(0.5,0.1,addgroup(bus,time),1);
                    [groupmember,~]=group(addSOC,addgroup(bus,time));
                    mnormgroupev=groupmember;
                    raddnum=(addEV(bus,time)*10-addgroup(bus,time))*100;
                    raddSOC=normrnd(0.5,0.1,1,1);
                    [groupmember,newSOC]=group(raddSOC,1);
                    rnormgroupev=groupmember;
                    addevnum=100*mnormgroupev+raddnum*rnormgroupev;
                    normgroupev(:,bus)=mnormgroupev+rnormgroupev;
                    for i=1:11
                        if normgroupev(i,bus)~=0
                            gnum(bus)=gnum(bus)+1;
                            groupSOC(gnum(bus),bus)=SOC(i,bus);
                            evnum(gnum(bus),bus)=addevnum(i);
                        end
                    end
                else
                    if addEV(time)>0.01
                        addgroup(bus,time)=floor(addEV(bus,time)*100);
                        addSOC=normrnd(0.5,0.1,addgroup(bus,time),1);
                        [groupmember,~]=group(addSOC,addgroup(bus,time));
                        mnormgroupev=groupmember;
                        raddnum=(addEV(bus,time)*100-addgroup(bus,time))*10;
                        raddSOC=normrnd(0.5,0.1,1,1);
                        [groupmember,newSOC]=group(raddSOC,1);
                        rnormgroupev=groupmember;
                        addevnum=10*mnormgroupev+raddnum*rnormgroupev;
                        normgroupev(:,bus)=mnormgroupev+rnormgroupev;
                        for i=1:11
                            if normgroupev(i,bus)~=0
                                gnum(bus)=gnum(bus)+1;
                                groupSOC(gnum(bus),bus)=SOC(i,bus);
                                evnum(gnum(bus),bus)=addevnum(i);
                            end
                        end
                    else
                        addgroup(bus,time)=floor(addEV(bus,time)*1000);
                        addSOC=normrnd(0.5,0.1,addgroup(bus,time),1);
                        [groupmember,newSOC]=group(addSOC,addgroup(bus,time));
                        normgroupev(:,bus)=groupmember;
                        for i=1:11
                            if normgroupev(i,bus)~=0
                                gnum(bus)=gnum(bus)+1;
                                groupSOC(gnum(bus),bus)=SOC(i,bus);
                                evnum(gnum(bus),bus)=1*normgroupev(i,bus);
                            end
                        end
                    end
                end
            end
        end
        for g=1:gnum(bus)


            [powerout,SOCnexttime]=totalEVV2G(groupSOC(g,bus),bus, sbp(tt), ssp(tt), asbp(tt), assp(tt), hbp, hsp, con(tt),loadp(:,tt), tot_demand(tt),mindemand,maxdemand);
            %[powerout,SOCnexttime]=totalEVold(groupSOC(g,bus),tt,asbp,assp,hbp,hsp);
            %[powerout,SOCnexttime]=totalEVnodispatch(groupSOC(g,bus),tt);

            groupSOC(g,bus)=SOCnexttime;%update the SOC

```

```
                    grouppower(g,bus)=powerout*evnum(g,bus); % unit is Watt

                end

            end


        end

        buspower(bus,time)=sum(grouppower(1:gnum(bus),bus));

        t_mpc.bus(bus,PD)=mpc.bus(bus,PD)+buspower(bus,time)/1e6; %unit is MW
        display(buspower(bus,time));
    end

    pf=runpf(t_mpc);
    load(time)=pf.gen(12,PG)+pf.gen(13,PG)+pf.gen(14,PG);
    if time>48*6
        energy=energy+load(time)*0.5;
        day_energy(tt)=energy;
    end
end
iter_energy(iteration)=day_energy(48);
end
mu=mean(iter_energy);
std_dev=std(iter_energy);
```

Functions that refer to different dispatch algorithms:

1. Proposed V2G dispatch algorithm:

```
function[powerout,SOCnexttime]=totalEVV2G(SOC,location, sbp, ssp, asbp, assp, hbp, hsp, contingency_status,loadp, tot_demand,mindemand,maxdemand)

Smin=0.4;
Smax=0.8;

if SOC<Smin
    action='charge';
else
    if SOC>Smax
        action='discharge';
    else
        action='undetermined';
    end
end

if strcmp(action,'charge') || strcmp(action,'undetermined')

    evbp=assp;
    massp=max(assp);
    if evbp<=hbp
        ev_ep=1;
    else
        %ev_ep=1-(evbp-hbp)/hbp;
        ev_ep=1-(evbp-hbp)/(0.1*massp);
    end

    g_margin=sbp-ssp;
    if g_margin>=0
        ev_gm=1;
    else
        ev_gm=0;
    end

    if contingency_status~=0
        [sen, sev, potcon]=contingency(contingency_status, location, 'charge', loadp);
        ev_con=0.625*sen+0.1365*sev+0.2385*potcon;
    else
        ev_con=0;
    end

    ev_LL=loadleveling(SOC,'charge',tot_demand,mindemand,maxdemand);

    if strcmp(action,'charge')
        ev=0.2*(0.25*ev_con+0.5*ev_gm+0.25*ev_LL)+0.8*0.5*(ev_ep+1);
    end
    if strcmp(action,'undetermined')
        ev_soc=1-(SOC-Smin)/(Smax-Smin);
        ev_ch=0.2*(0.25*ev_con+0.5*ev_gm+0.25*ev_LL)+0.8*0.5*(ev_ep+ev_soc);%test
    end

end
if strcmp(action,'discharge') || strcmp(action,'undetermined')

    evsp=asbp;
    masbp=max(asbp);
    if evsp>=hsp
        ev_ep=1;
    else
        ev_ep=1-(hsp-evsp)/(0.6*masbp);
    end

    g_margin=assp-asbp;
    if g_margin>=-0.01
        ev_gm=1;
    else
        if g_margin<=-0.02
            ev_gm=0;
        else
            ev_gm=(g_margin+0.02)/0.01;
        end
    end

    if contingency_status~=0
        [sen, sev, potcon]=contingency(contingency_status, location, 'discharge', loadp);
        ev_con=0.625*sen+0.1365*sev+0.2385*potcon;
    else
        ev_con=0;
    end

    ev_LL=loadleveling(SOC,'discharge',tot_demand,mindemand,maxdemand);

    if strcmp(action,'discharge')
        ev=0.2*(0.25*ev_con+0.5*ev_gm+0.25*ev_LL)+0.8*0.5*(ev_ep+1);
    end
    if strcmp(action,'undetermined')
        ev_soc=1-(Smax-SOC)/(Smax-Smin);
        ev_disch=0.2*(0.25*ev_con+0.5*ev_gm+0.25*ev_LL)+0.8*0.5*(ev_ep+ev_soc);
    end
end
if strcmp(action,'undetermined')

    ev=max(ev_ch,ev_disch);
    if ev==ev_ch
        action='charge';
    else
        action='discharge';
    end
end

if strcmp(action,'charge')
    if ev>=0.9
        volt=voltage(SOC,30);
```

```
            powerout=30*volt;
            SOCnexttime=SOC+15/92.21;

        else
            if ev>=0.7
                volt=voltage(SOC,10);
                powerout=10*volt;
                SOCnexttime=SOC+5/114.87;

            else
                if ev>=0.4
                    volt=voltage(SOC,2);
                    powerout=2*volt;
                    SOCnexttime=SOC+1/158.5;
                else
                    powerout=0;
                    SOCnexttime=SOC;
                end
            end
        end
    end
    if strcmp(action,'discharge')
        if ev>=0.9
            volt=voltage(SOC,30);
            powerout=-30*volt;
            SOCnexttime=SOC-15/92.21;

        else
            if ev>=0.7
                volt=voltage(SOC,10);
                powerout=-10*volt;
                SOCnexttime=SOC-5/114.87;

            else
                if ev>=0.4
                    volt=voltage(SOC,2);
                    powerout=-2*volt;
                    SOCnexttime=SOC-1/158.5;

                else
                    powerout=0;
                    SOCnexttime=SOC;
                end
            end
        end
    end
```

2. Published rule-based dispatch algorithm:

```
function [powerout,SOCnexttime]=totalEVold(SOC,tt,asbp,assp,hbp,hsp)
hcurrent=30;
mcurrent=10;
lcurrent=2;
if tt<=48
    V2GR=0;
    if SOC>=0.75
        EVR=-1;
    else
        if SOC<=0.5
            EVR=1;
        else
            EVR=2;
        end
    end
    if EVR==-1
        V2GR=-1;
    else
        if EVR==1;
            V2GR=1;
        else
            if EVR==2;
                V2GR=2;
            end
        end
    end
    display(V2GR);
    if tt<3 || tt>17
        if V2GR==1
            if assp(tt)>=hbp
                current=lcurrent;
            else
                current=hcurrent;
            end
        else
            if V2GR==-1
                if asbp(tt)>=hsp
                    current=-hcurrent;
                else
                    current=-lcurrent;
                end
            else
                if V2GR==2;
                    if asbp(tt)>=hsp
                        current=-mcurrent;
                    else
                        if assp(tt)<=hbp
                            current=hcurrent;
                        else
                            current=-lcurrent;
                        end
                    end
                end
            end
        end
    else
        if V2GR==1
            current=lcurrent;
        else
            if V2GR==-1
                current=-lcurrent;
            else
                if V2GR==2
                    current=-lcurrent;
                end
            end
        end
    end
    display(current);
    display(SOC);
    volt=voltage(SOC,abs(current));
    powerout=current*volt;


    if abs(current)==hcurrent;
        SOCnexttime=SOC+0.5*current/92.21;

    else
        if abs(current)==mcurrent;
            SOCnexttime=SOC+0.5*current/114.87;

        else
            if abs(current)==lcurrent;
                SOCnexttime=SOC+0.5*current/158.5;
            end

        end
    end



end

end
```

3. Uncontrolled dispatch

```
function[powerout,SOCnexttime]=totalEVnodispatch(SOC,time)

if time>=19 && time<=48
    if SOC<=0.9
        volt=voltage(SOC,30);
        powerout=30*volt;
        SOCnexttime=SOC+15/92.21;
```

```
    else
        powerout=0;
        SOCnexttime=SOC;
    end
else
    powerout=0;
    SOCnexttime=SOC;
end
```

Codes produced for studies in Chapter 5

Main Function:

```
nodestate=networkdata;
cap=brcap;
branchcap=zeros(12,1);
for ag=1:12
    branchcap(ag)=cap(ag+1,nodestate(ag).par+1);
end
Pnc_min=zeros(48,3); %unit is MW
Pnc_max=zeros(48,3);
totPnc_min=xlsread('networkdata.xlsx','contingency','B2:B49');
totPnc_max=xlsread('networkdata.xlsx','contingency','C2:C49');
[asbp, assp, hbp, lsp, maxsp, minbp]=price;
[Pload,Qload,aveload,load_max,load_min,load_h,load_l]=loadcalculate; %node 0 has no load
virtualnode=subagent;
virtualbrcap_p=zeros(3,1);

tplan=zeros(12,48,3);
for node=2:12
    tplan(node,1:48,1:3)=travelpattern(node);
end
opobj=zeros(48,3);
oppflow=zeros(48,3);
opSflow=zeros(48,3);
opchildcom=zeros(48,3,3);
opdispatchstate=zeros(48,12,17);
opdispatch=zeros(48,12,4);
soc=zeros(48,12,3);
nodeload=zeros(48,12);
chcost=zeros(48,12);
nodeRG=zeros(48,12);

for time=1:48
    display(time);
    node1=0;
    node2=0;
    node3=0;
    num_dsre=zeros(12,1);
    pwind=zeros(12,1);
    dsrecord=zeros;
    toparentnode=zeros;
    cnode=zeros(3,3);

    while node1==0
        while node2==0
            while node3==0 %leaf node
                for agent=1:12 % search for leaf node first
                    if nodestate(agent).calorder==0 %leaf node
                        display(agent);
                        num_flow=0; %total number of different power flow from this agent%
                        if nodestate(agent).EV==1 %this node can connect 3*1000 EVs.%
                            soc_p=zeros(3:1);
                            soc_pf=zeros(3:1);
                            parkflag=zeros(12,3);
                            for evgroup=1:3
                                if time==1;
                                    soc_init=normrnd(0.6,0.1);
                                    while soc_init>1 || soc_init<0
                                        soc_init=normrnd(0.6,0.1);
                                    end
                                    soc(time,agent,evgroup)=soc_init;
                                end
                                if tplan(agent,time,evgroup)~=0
                                    parkflag(agent,evgroup)=0; % the ev is currently on road
                                else
                                    parkflag(agent,evgroup)=1; % the ev is currently parked
                                    onroadtime=0;
                                    setouttime=0;
                                    preparetime=0;
                                    for chtime=1:4 %if driver have a travel plan in 2 hours, battery needs to prepare for that%
                                        if time+chtime<=48
                                            if tplan(agent,time+chtime,evgroup)~=0
                                                onroadtime=tplan(agent,time+chtime,evgroup);
                                                setouttime=time+chtime;
                                                preparetime=chtime;
                                                break;
                                            end
                                        end
                                    end
                                    if onroadtime==0
                                        soc_p(evgroup)=0; %if ev has no travel plan in 2 hours, soc needs to be controlled between 0.4 and 1.0%
                                    else
                                        soc_pf(evgroup)=0.4+onroadtime*0.2; %this is the SOC expected at last.
                                        if soc(time,agent,evgroup)<soc_pf(evgroup)
                                            soc_p(evgroup)=abs(soc(time,agent,evgroup)-soc_pf(evgroup))/preparetime+soc(time,agent,evgroup);
                                        else
                                            soc_p(evgroup)=soc_pf(evgroup);
                                        end
                                    end
                                end
```

```
                        end
                    end
                    if nodestate(agent).DG==1
                        pwind(agent)=wind(agent,time)*5;
                    else
                        pwind(agent)=0;
                    end

                    %%%%%%% coordination%%%%%%%%%
                    [combination,tot_num,nodetype]=coordination(nodestate(agent).EV,parkflag(agent,:),pwind(agent));
                    brflow=zeros;
                    lineSflow=zeros;
                    U=zeros;
                    opcomb=zeros;
                    nextsoc=zeros;
                    objdetail=zeros;
                    obj=zeros(tot_num,1);
                    pflow=zeros(tot_num,1);
                    Sflow=zeros(tot_num,1);
                    validflag=zeros(tot_num,1);
                    soc_n=zeros(tot_num,3);
                    obj_info=zeros(tot_num,1);

                    for cn=1:tot_num
                        [obj(cn),pflow(cn),Sflow(cn),validflag(cn),soc_n(cn,1:3),obj_info(cn)]=Utility_leafnode(nodetype,1000,3,0,combination(cn,:),parkflag(agent,:),pwind(agent),soc_p,soc(time,agent,:),assp(time),hbp,minbp,asbp(time),maxsp,lsp,Pload(time,agent),Qload(time,agent),load_max(agent),load_min(agent),load_h(agent),load_l(agent),branchcap(agent));
                        if validflag(cn)==1
                            if num_flow>0
                                sameflag=0;
                                for nf=1:num_flow
                                    if round(pflow(cn))==round(brflow(nf,1))
                                        sameflag=1;
                                        if obj(cn)<U(nf,1)
                                            brflow(nf,1)=pflow(cn);
                                            U(nf,1)=obj(cn);
                                            l=1:length(combination(cn,:));
                                            opcomb(nf,l)=combination(cn,l);
                                            nextsoc(nf,1:3)=soc_n(cn,:);
                                            lineSflow(nf,1)=Sflow(cn);
                                            objdetail(nf,1:4)=obj_info(cn);

                                        end
                                    end
                                end
                                if sameflag==0
                                    num_flow=num_flow+1;
                                    brflow(num_flow,1)=pflow(cn);
                                    lineSflow(num_flow,1)=Sflow(cn);
                                    U(num_flow,1)=obj(cn);
                                    l=1:length(combination(cn,:));
                                    opcomb(num_flow,l)=combination(cn,l);
                                    nextsoc(num_flow,1:3)=soc_n(cn,:);
                                    objdetail(num_flow,1:4)=obj_info(cn);
                                end
                            else
                                num_flow=num_flow+1;
                                brflow(num_flow,1)=pflow(cn);
                                lineSflow(num_flow,1)=Sflow(cn);
                                U(num_flow,1)=obj(cn);
                                l=1:length(combination(cn,:));
                                opcomb(num_flow,l)=combination(cn,l);
                                nextsoc(num_flow,1:3)=soc_n(cn,:);
                                objdetail(num_flow,1:4)=obj_info(cn);
                            end
                        end
                    end
                    [staterecord,toparentrecord,totnum_record]=flowcostcomb(0,brflow,lineSflow,U,opcomb,[],0,nextsoc,objdetail,num_flow); %objdetail refers to local obj details of the node%
                    [row,column]=size(staterecord);
                    r=1:row;
                    c=1:column;
                    dsrecord(agent,r,c)=staterecord(r,c);
                    toparentnode(agent,r,1:3)=toparentrecord(r,:);
                    num_dsre(agent)=totnum_record;
dsrecord_test(agent,r,c)=dsrecord(agent,r,c);
                end
                if agent==12 % search for leaf node is complete%
                    node3=1;
```

```
                end
            end
        end
        %%%%%%%%%% 3rd-level node%%%%%%%%%%%%%%%%%%
        for agent=1:12
            if nodestate(agent).calorder==1
                display(agent);
                num_flow=0; % total number of different power flow from this agent%
                if nodestate(agent).EV==1 %this node can connect 3*1000 EVs.%
                    soc_p=zeros(3:1);
                    soc_pf=zeros(3:1);
                    parkflag=zeros(12,3);
                    for evgroup=1:3
                        if time==1;
                            soc_init=normrnd(0.6,0.1);
                            while soc_init>1 || soc_init<0
                                soc_init=normrnd(0.6,0.1);
                            end
                            soc(time,agent,evgroup)=soc_init;
                        end
                        if tplan(agent,time,evgroup)~=0
                            parkflag(agent,evgroup)=0; % the ev is currently on road
                        else
                            parkflag(agent,evgroup)=1; % the ev is currently parked
                            onroadtime=0;
                            setouttime=0;
                            preparetime=0;
                            for chtime=1:4 %if driver have a travel plan in 2 hours, battery needs to prepare for that%
                                if time+chtime<=48
                                    if tplan(agent,time+chtime,evgroup)~=0
                                        onroadtime=tplan(agent,time+chtime,evgroup);
                                        setouttime=time+chtime;
                                        preparetime=chtime;
                                        break;
                                    end
                                end
                            end
                            if onroadtime==0
                                soc_p(evgroup)=0; %if ev has no travel plan in 2 hours, soc needs to be controlled between 0.4 and 1.0%
                            else
                                soc_pf(evgroup)=0.4+onroadtime*0.2; %this is the SOC expected at last.
                                if soc(time,agent,evgroup)<soc_pf(evgroup)
                                    soc_p(evgroup)=abs(soc(time,agent,evgroup)-soc_pf(evgroup))/preparetime+soc(time,agent,evgroup);
                                else
                                    soc_p(evgroup)=soc_pf(evgroup);
                                end
                            end
                        end
                    end
                end
                if nodestate(agent).DG==1
                    pwind(agent)=wind(agent,time)*5;
                else
                    pwind(agent)=0;
                end

                %%%%%%% coordination%%%%%%%%%
                [combination,tot_num,nodetype]=coordination(nodestate(agent).EV,parkflag(agent,:),pwind(agent));
                brflow=zeros;
                lineSflow=zeros;
                U=zeros;
                opcomb=zeros;
                opchild=zeros;
                opchildno=zeros;
                nextsoc=zeros;
                objdetail=zeros;
                obj=zeros(tot_num,num_dsre(nodestate(agent).chi));
                pflow=zeros(tot_num,num_dsre(nodestate(agent).chi));
                Sflow=zeros(tot_num,num_dsre(nodestate(agent).chi));
                validflag=zeros(tot_num,num_dsre(nodestate(agent).chi));
                soc_n=zeros(tot_num,num_dsre(nodestate(agent).chi),3);
                obj_info=zeros(tot_num,num_dsre(nodestate(agent).chi));

                for cn=1:tot_num
                    for fn=1:num_dsre(nodestate(agent).chi)
                        [obj(cn,fn),pflow(cn,fn),Sflow(cn,fn),validflag(cn,fn),soc_n(cn,fn,1:3),obj_info(cn)]=Utility_node2(nodetype,1000,3,0,combination(cn,:),parkflag(agent,:),toparentnode(nodestate(agent).chi,fn,:),pwind(agent),soc_p,soc(time,agent,:),assp(time),hbp,minbp,asbp(time),maxsp,lsp,Pload(time,agent),Qload(time,agent),load_max(agent),load_min(agent),load_h(agent),load_l(agent),branchcap(agent));
```

```
                                if validflag(cn,fn)==1
                                    if num_flow>0
                                        sameflag=0;
                                        for nf=1:num_flow
                                            if round(pflow(cn))==round(brflow(nf,1))
                                                sameflag=1;
                                                if obj(cn,fn)<U(nf,1)
                                                    brflow(nf,1)=pflow(cn,fn); %although they are closed,but still a little bit different. better update.%
                                                    U(nf,1)=obj(cn,fn);
                                                    l=1:length(combination(cn,:));
                                                    opcomb(nf,l)=combination(cn,l);
                                                    opchild(nf,1:3)=toparentnode(nodestate(agent).chi,fn,:);
                                                    opchildno(nf,1)=fn;
                                                    nextsoc(nf,1:3)=soc_n(cn,fn,:);
                                                    lineSflow(nf,1)=Sflow(cn,fn);
                                                    objdetail(nf,1:4)=obj_info(cn);
                                                end
                                            end
                                        end
                                        if sameflag==0
                                            num_flow=num_flow+1;
                                            brflow(num_flow,1)=pflow(cn,fn);
                                            lineSflow(num_flow,1)=Sflow(cn,fn);
                                            U(num_flow,1)=obj(cn,fn);
                                            l=1:length(combination(cn,:));
                                            opcomb(num_flow,l)=combination(cn,l);
                                            opchild(num_flow,1:3)=toparentnode(nodestate(agent).chi,fn,:);
                                            opchildno(num_flow,1)=fn;
                                            nextsoc(num_flow,1:3)=soc_n(cn,fn,:);
                                            objdetail(num_flow,1:4)=obj_info(cn);
                                        end
                                    else
                                        num_flow=num_flow+1;
                                        brflow(num_flow,1)=pflow(cn,fn);
                                        lineSflow(num_flow,1)=Sflow(cn,fn);
                                        U(num_flow,1)=obj(cn,fn);
                                        l=1:length(combination(cn,:));
                                        opcomb(num_flow,l)=combination(cn,l);
                                        opchild(num_flow,1:3)=toparentnode(nodestate(agent).chi,fn,:);
                                        opchildno(num_flow,1)=fn;
                                        nextsoc(num_flow,1:3)=soc_n(cn,fn,:);
                                        objdetail(num_flow,1:4)=obj_info(cn);
                                    end
                                end
                            end
                        end

                        [staterecord,toparentrecord,totnum_record]=flowcostcomb(1,brflow,lineSflow,U,opcomb,opchild,opchildno,nextsoc,objdetail,num_flow);
                        [row,column]=size(staterecord);
                        r=1:row;
                        c=1:column;
                        dsrecord(agent,r,c)=staterecord(r,c);
                        toparentnode(agent,r,1:3)=toparentrecord(r,:);
                        num_dsre(agent)=totnum_record;
    dsrecord_test(agent,r,c)=dsrecord(agent,r,c);
                    end

                    if agent==12 % search for 3rd level node is complete%
                        node2=1;
                    end
                end
            end

            %%%%%%%%%%%% V1 %%%%%%%%%%%%%%%
            for agent=1:12
                if nodestate(agent).calorder==2
                    display(agent);
                    %%%%%%%%%% sub-agents of V1 %%%%%%%%%%%%
                    if totPnc_min(time)>0 % need a certain amount of power to be transferred from V0 to the rest of grid.%
                        Pnc_min(time,1)=(7731*3000*3/1e6+pwind(3)-sum(Pload(time,2:4)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                        Pnc_max(time,1)=(7731*3000*3/1e6+pwind(3)-sum(Pload(time,2:4)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));
```

```
                Pnc_min(time,2)=(7731*3000*4/1e6+pwind(6)-sum(Pload(time,5:8)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                Pnc_max(time,2)=(7731*3000*4/1e6+pwind(6)-sum(Pload(time,5:8)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));

                Pnc_min(time,3)=(7731*3000*4/1e6+sum(pwind(11:12))-sum(Pload(time,9:12)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                Pnc_max(time,3)=(7731*3000*4/1e6+sum(pwind(11:12))-sum(Pload(time,9:12)))/(7731*33000/1e6+sum(pwind)-sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));
            else
                if totPnc_min(time)<0 % need a certain amount of power to be transferred the rest of grid to V0.%
                    Pnc_min(time,1)=(sum(Pload(time,2:4)))/(sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                    Pnc_max(time,1)=(sum(Pload(time,2:4)))/(sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));

                    Pnc_min(time,2)=(sum(Pload(time,5:8)))/(sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                    Pnc_max(time,2)=(sum(Pload(time,5:8)))/(sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));

                    Pnc_min(time,3)=(sum(Pload(time,9:12)))/(sum(Pload(time,2:12)))*(totPnc_min(time)+Pload(time,1));
                    Pnc_max(time,3)=(sum(Pload(time,9:12)))/(sum(Pload(time,2:12)))*(totPnc_max(time)+Pload(time,1));
                end
            end
            for vn=1:3
                virtualbrcap_p(vn)=sum(branchcap(virtualnode(vn).chi))/sum(branchcap(nodestate(1).chi))*(Pload(time,1)*(branchcap(1)/abs(Pload(time,1)+1i*Qload(time,1))-1));
            end
            for subag=1:3
                vc=1;
                for chirn1=1:num_dsre(virtualnode(subag).chi(vc))
                    if vc==length(virtualnode(subag).chi)
                        [obj,pflow,Sflow,validflag]=Utility_V1(toparentnode(virtualnode(subag).chi(vc),chirn1,:),virtualbrcap_p(subag),Pnc_min(time,subag),Pnc_max(time,subag));
                        %%%%%% optimal solution selection %%%%%%%%
                        if validflag==1
                            if chirn1==1
                                opobj(time,subag)=obj;
                                oppflow(time,subag)=pflow;
                                opSflow(time,subag)=Sflow;
                                opchildcom(time,subag,1:3)=[chirn1,0,0];
                            else
                                if opobj(time,subag)>obj
                                    opobj(time,subag)=obj;
                                    oppflow(time,subag)=pflow;
                                    opSflow(time,subag)=Sflow;
                                    opchildcom(time,subag,1:3)=[chirn1,0,0];
                                end
                            end
                        end

                    else
                        vc=vc+1;
                        for chirn2=1:num_dsre(virtualnode(subag).chi(vc))
                            if vc==length(virtualnode(subag).chi)
                                [obj,pflow,Sflow,validflag]=Utility_V1([toparentnode(virtualnode(subag).chi(vc-1),chirn1,:);toparentnode(virtualnode(subag).chi(vc),chirn2,:)],virtualbrcap_p(subag),Pnc_min(time,subag),Pnc_max(time,subag));
                                %%%%%% optimal solution selection %%%%%%%%
                                if validflag==1
                                    if chirn1==1 && chirn2==1
                                        opobj(time,subag)=obj;
                                        oppflow(time,subag)=pflow;
                                        opSflow(time,subag)=Sflow;
                                        opchildcom(time,subag,1:3)=[chirn1,chirn2,0];
                                    else
                                        if opobj(time,subag)>obj
                                            opobj(time,subag)=obj;
```

```
                        oppflow(time,subag)=pflow;
                        opSflow(time,subag)=Sflow;
                        opchildcom(time,subag,1:3)=[chirn1,chirn2,0];
                    end
                end
            end
        else
            vc=vc+1;
            for chirn3=1:num_dsre(virtualnode(subag).chi(vc))
                [obj,pflow,Sflow,validflag]=Utility_V1([toparentnode(virtualnode(subag).chi(vc-2),chirn1,:);toparentnode(virtualnode(subag).chi(vc-1),chirn2,:);toparentnode(virtualnode(subag).chi(vc),chirn3,:)],virtualbrcap_p(subag),Pnc_min(time,subag),Pnc_max(time,subag));
                %%%%%% optimal solution selection %%%%%%%%
                if validflag==1
                    if chirn1==1 && chirn2==1 && chirn3==1
                        opobj(time,subag)=obj;
                        oppflow(time,subag)=pflow;
                        opSflow(time,subag)=Sflow;
                        opchildcom(time,subag,1:3)=[chirn1,chirn2,chirn3];
                    else
                        if opobj(time,subag)>obj
                            opobj(time,subag)=obj;
                            oppflow(time,subag)=pflow;
                            opSflow(time,subag)=Sflow;
                            opchildcom(time,subag,1:3)=[chirn1,chirn2,chirn3];
                        end
                    end
                end
            end
            vc=vc-1;
        end
                    end
                    vc=vc-1;
                end
            end

            end


        end
        if agent==12 %search for 3rd level node is complete %
            node1=1;
        end
    end
end

for subag=1:3
    for ch=1:3
        display(subag); display(ch);
        if opchildcom(time,subag,ch)~=0

            if ~isempty(nodestate(virtualnode(subag).chi(ch)).chi) %this node has child node
                opdispatchstate(time,virtualnode(subag).chi(ch),1:17)=dsrecord(virtualnode(subag).chi(ch),opchildcom(time,subag,ch),:);
                opdispatch(time,virtualnode(subag).chi(ch),1:4)=opdispatchstate(time,virtualnode(subag).chi(ch),4:7);
                soc(time+1,virtualnode(subag).chi(ch),1:3)=opdispatchstate(time,virtualnode(subag).chi(ch),12:14);
                nodeload(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),16);
                chcost(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),17);
                if nodestate(virtualnode(subag).chi(ch)).DG==1
                    nodeRG(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),15);
                end


                cnode(subag,ch)=nodestate(virtualnode(subag).chi(ch)).chi; %child node

                opdispatchstate(time,cnode(subag,ch),1:13)=dsrecord(cnode(subag,ch),opdispatchstate(time,virtualnode(subag).chi(ch),11),1:13);
                opdispatch(time,cnode(subag,ch),1:4)=opdispatchstate(time,cnode(subag,ch),4:7);
                soc(time+1,cnode(subag,ch),1:3)=opdispatchstate(time,cnode(subag,ch),8:10);
                nodeload(time,cnode(subag,ch))=opdispatchstate(time,cnode(subag,ch),12);
                chcost(time,cnode(subag,ch))=opdispatchstate(time,cnode(subag,ch),13);
                if nodestate(cnode(subag,ch)).DG==1
```

```
                        nodeRG(time,cnode(subag,ch))=opdispatchstate(time,cnode(subag,ch),11);
                    end

                else %this node is a leaf node
                    opdispatchstate(time,virtualnode(subag).chi(ch),1:13)=dsrecord(virtualnode(subag).chi(ch),opchildcom(time,subag,ch),1:13);
                    opdispatch(time,virtualnode(subag).chi(ch),1:4)=opdispatchstate(time,virtualnode(subag).chi(ch),4:7);
                    soc(time+1,virtualnode(subag).chi(ch),1:3)=opdispatchstate(time,virtualnode(subag).chi(ch),8:10);
                    nodeload(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),12);
                    chcost(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),13);
                    if nodestate(virtualnode(subag).chi(ch)).DG==1
                        nodeRG(time,virtualnode(subag).chi(ch))=opdispatchstate(time,virtualnode(subag).chi(ch),11);
                    end
                end
            end
        end
    end
end
node=1:12;
dailycost(node)=sum(chcost(:,node));
```

Utility Calculation at leaf nodes:

```
function [obj,pflow,Sflow,validflag,SOC_n,obj_info]=Utility_leafnode(nodetype,n,group,~,combination,parkflag,Prg_p,SOC_p,SOC_c,bp,hbp,bpmin,sp,spmax,lsp,fixloadp,fixloadq,load_max,load_min,load_h,load_l,branchcap)
validflag=1;
Prg=combination(4);
level=combination(1:3);
SOC_n=zeros(1,group);
diffsoc=zeros(1,group);
Po=zeros(1,group);
cost=zeros(1,group);
total_diffsoc=0;
total_evload=0;
total_cost=0;
evcost=zeros(1,group);
total_evcost=0;
obj_info=zeros(1,3);

if nodetype==1 || nodetype==2 %node=1 or 2 means DG-connected nodes
    RG=(Prg_p-Prg)/Prg_p;
else
    RG=inf;
end

if nodetype==2 || nodetype==3 %node=2 or 3 means EV-connected nodes
    for ev=1:group %group is the total number of local EV groups%
        if parkflag(ev)==1 % ev is currently parked
            if abs(level(ev))==3
                delta_soc=15/92.21;
            else
                if abs(level(ev))==2
                    delta_soc=5/114.87;
                else
                    if abs(level(ev))==1;
                        delta_soc=1/158.5;
                    else
                        delta_soc=0;
                    end
                end
            end

            if level(ev)<0
                delta_soc=-1*delta_soc;
            end

            SOC_n(ev)=SOC_c(ev)+delta_soc; %level is an array that presents the charging(+)/discharging(-) speeds of all local EV groups%

           if SOC_n(ev)>1
               validflag=0;
           end

           if SOC_p(ev)~=0
               if SOC_n(ev)<SOC_p(ev)
                   if SOC_n(ev)>=SOC_c(ev)
                       diffsoc(ev)=(SOC_p(ev)-SOC_n(ev))/(SOC_p(ev)-0.4);
                   else
                       validflag=0;
                   end
               else
                   diffsoc(ev)=0;
               end


           else
               if SOC_n(ev)<0.5
                   if SOC_c(ev)<0.5 && SOC_n(ev)>=SOC_c(ev)
                       %diffsoc(ev)=1;
                       %diffsoc(ev)=0;
                       diffsoc(ev)=(0.5-SOC_n(ev))/(0.5-0.4);
                   else
                       validflag=0;
                   end
               else
                   diffsoc(ev)=0; %when ev has no plan in 2 hours and soc is between 0.5 and 1, this dispatch action is acceptable.
               end
           end

            Po(ev)=evpower(SOC_c(ev),level(ev));
            if level(ev)<0
                Po(ev)=-Po(ev);
```

```
                cost(ev)=sp*Po(ev)*0.5/1000*n; %price unit is pounds/kwh, one time interval is 0.5 hours.%
            else
                if level(ev)>0
                    cost(ev)=bp*Po(ev)*0.5/1000*n; % n is the number of EVs per group%
                else
                    cost(ev)=0;
                end
            end
            if level(ev)==3
                if bp<hbp
                    evcost(ev)=(bp-bpmin)/(hbp-bpmin);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==2
                if bp<hbp
                    evcost(ev)=abs(bp-(bpmin+hbp)/2)/((hbp-bpmin)/2);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==1
                if bp<hbp
                    evcost(ev)=(hbp-bp)/(hbp-bpmin);
                else
                    evcost(ev)=0;
                end
            end
            if level(ev)==0
                if sp<=lsp || bp>=hbp
                    evcost(ev)=0;
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==-1
                if sp>lsp
                    evcost(ev)=(sp-lsp)/(spmax-lsp);
                else
                    evcost(ev)=0;
                end
            end
            if level(ev)==-2
                if sp>lsp
                    evcost(ev)=abs(sp-(lsp+spmax)/2)/((spmax-lsp)/2);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==-3
                if sp>lsp
                    evcost(ev)=(spmax-sp)/(spmax-lsp);
                else
                    evcost(ev)=1;
                end
            end
        else
            delta_soc=-0.1; %ev is currently on road
            SOC_n(ev)=SOC_c(ev)+delta_soc;
            cost(ev)=0;
            evcost(ev)=0;
            Po(ev)=0;
            diffsoc(ev)=0;
        end
        total_evload=total_evload+Po(ev)*n/1e6;%load unit is MVA
        total_cost=total_cost+cost(ev);
        total_evcost=total_evcost+evcost(ev);
        total_diffsoc=total_diffsoc+diffsoc(ev);
    end
end

totload=total_evload+fixloadp-Prg;
if totload>=load_l && totload<=load_h
    LL=0;
end
if totload<load_max && totload>load_h
    LL=(totload-load_h)/(load_max-load_h);
end
if totload>load_min && totload<load_l
    LL=(load_l-totload)/(load_l-load_min);
end
if totload<=load_min && totload>0
    LL=load_min/totload;
else
    if totload<=0
        LL=(load_min-totload)/load_min;
    end
end
if totload>=load_max
    LL=totload/load_max;
end

pflow=-totload; %power flow from this node to its parent node%
qflow=-fixloadq;
Sflow=pflow+1i*qflow;
if abs(Sflow)>branchcap
    validflag=0;
end

if validflag==1

    if nodetype==1
        display(RG); display(LL);% test
        obj=0.5*RG+0.5*LL;
    else
        if nodetype==2
            obj=0.2781*RG+0.3952*total_diffsoc/3+0.1634*total_evcost/3+0.1634*LL;
            obj_info=[RG,total_diffsoc/3,total_evcost/3,LL];
        else
            if nodetype==3
                obj=0.5*total_diffsoc/3+0.25*total_evcost/3+0.25*LL;


            end
        end
    end

else
    obj=inf;
end

display(obj);
```

Utility Calculation at mid-level nodes:

```
function [obj,pflow,Sflow,validflag,SOC_n,obj_info]=Utility_node2(nodetype,n,group,~,combination,parkflag,childpfobj,Prg_p,SOC_p,SOC_c,bp,hbp,bpmin,sp,spmax,lsp,fixloadp,fixloadq,load_max,load_min,load_h,load_l,branchcap)

validflag=1;
Prg=combination(4);
level=combination(1:3);
SOC_n=zeros(1,group);
```

```
diffsoc=zeros(1,group);
Po=zeros(1,group);
cost=zeros(1,group);
childpf=childpfobj(1);
childsf=childpfobj(2);
childqf=imag(childsf);
childobj=childpfobj(3);
total_diffsoc=0;
total_evload=0;
total_cost=0;
evcost=zeros(1,group);
total_evcost=0;
obj_info=zeros(1,3);

if nodetype==1 || nodetype==2 % node=1 or 2 means DG-connected nodes
    RG=(Prg_p-Prg)/Prg_p;
else
    RG=inf;
end

if nodetype==2 || nodetype==3 % node=2 or 3 means EV-connected nodes
    for ev=1:group %group is the total number of local EV groups%
        if parkflag(ev)==1 % ev is currently parked
            if abs(level(ev))==3
                delta_soc=15/92.21;
            else
                if abs(level(ev))==2
                    delta_soc=5/114.87;
                else
                    if abs(level(ev))==1
                        delta_soc=1/158.5;
                    else
                        delta_soc=0;
                    end
                end
            end

            if level(ev)<0
                delta_soc=-1*delta_soc;
            end

            SOC_n(ev)=SOC_c(ev)+delta_soc; %level is an array that presents the charging(+)/discharging(-) speeds of all local EV groups%

           if SOC_n(ev)>1
               validflag=0;
               %continue;
           end

          if SOC_p(ev)~=0
              if SOC_n(ev)<SOC_p(ev)
                  if SOC_n(ev)>=SOC_c(ev)
                      diffsoc(ev)=(SOC_p(ev)-SOC_n(ev))/(SOC_p(ev)-0.4);
                  else
                      validflag=0;
                  end
              else
                  diffsoc(ev)=0;
              end

           else
              if SOC_n(ev)<0.5
                  if SOC_c(ev)<0.5 && SOC_n(ev)>=SOC_c(ev)
                      diffsoc(ev)=(0.5-SOC_n(ev))/(0.5-0.4);
                  else
                      validflag=0;
                  end
              else
                  diffsoc(ev)=0; %when ev has no plan in 2 hours and soc is between 0.5 and 1, this dispatch action is acceptable.
              end
          end

            Po(ev)=evpower(SOC_c(ev),level(ev));
            if level(ev)<0
                Po(ev)=-Po(ev);
                cost(ev)=sp*Po(ev)*0.5/1000*n; %price unit is pounds/kwh, one time interval is 0.5 hours.%
            else
                if level(ev)>0
                    cost(ev)=bp*Po(ev)*0.5/1000*n; % n is the number of EVs per group%
                else
                    cost(ev)=0;
                end
            end

            if level(ev)==3
                if bp<hbp
                    evcost(ev)=(bp-bpmin)/(hbp-bpmin);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==2
                if bp<hbp
                    evcost(ev)=abs(bp-(bpmin+hbp)/2)/((hbp-bpmin)/2);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==1
                if bp<hbp
                    evcost(ev)=(hbp-bp)/(hbp-bpmin);
                else
                    evcost(ev)=0;
                end
            end
            if level(ev)==0
                if sp<=lsp || bp>=hbp
                    evcost(ev)=0;
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==-1
                if sp>lsp
                    evcost(ev)=(sp-lsp)/(spmax-lsp);
                else

                    evcost(ev)=0;
                end
            end
            if level(ev)==-2
                if sp>lsp
                    evcost(ev)=abs(sp-(lsp+spmax)/2)/((spmax-lsp)/2);
                else
                    evcost(ev)=1;
                end
            end
            if level(ev)==-3
                if sp>lsp
                    evcost(ev)=(spmax-sp)/(spmax-lsp);
                else
                    evcost(ev)=1;
```

```
                end
            end



        else
            delta_soc=-0.1; %ev is currently on road
            SOC_n(ev)=SOC_c(ev)+delta_soc;
            cost(ev)=0;
            evcost(ev)=0;
            Po(ev)=0;
            diffsoc(ev)=0;
        end
        total_evload=total_evload+Po(ev)*n/1e6;%load unit is MVA
        total_cost=total_cost+cost(ev);
        total_evcost=total_evcost+evcost(ev);
        total_diffsoc=total_diffsoc+diffsoc(ev);
    end
end

totload=total_evload+fixloadp-Prg;
if totload>=load_l && totload<=load_h
    LL=0;
end
if totload<load_max && totload>load_h
    LL=(totload-load_h)/(load_max-load_h);
end
if totload>load_min && totload<load_l
    LL=(load_l-totload)/(load_l-load_min);
end
if totload<=load_min && totload>0
    LL=load_min/totload;
else
    if totload<=0
        LL=(load_min-totload)/load_min;
    end
end
if totload>=load_max
    LL=totload/load_max;
end

pflow=-totload+childpf; %power flow from this node to its parent node%
qflow=-fixloadq+childqf;
Sflow=pflow+1i*qflow;
if abs(Sflow)>branchcap
    validflag=0;
end

if validflag==1
    if nodetype==1
        nodeobj=0.5*RG+0.5*LL;
    else
        if nodetype==2
            nodeobj=0.2781*RG+0.3952*total_diffsoc/3+0.1634*total_evcost/3+0.1634*LL;
            obj_info=[RG,total_diffsoc/3,total_evcost/3,LL];
        else
            if nodetype==3
                nodeobj=0.5*total_diffsoc/3+0.25*total_evcost/3+0.25*LL;
            end
        end
    end
    obj=nodeobj+childobj;

else
    obj=inf;
end
```

Utility Calculation at top-level nodes:

```
function [obj,pflow,Sflow,validflag]=Utility_V1(childpfobj,branchcap_p,Pnc_min,Pnc_max)

validflag=1;
childpf=childpfobj(:,1);
childsf=childpfobj(:,2);
childqf=imag(childsf);
childobj=childpfobj(:,3);


pflow=sum(childpf); %power flow from this node to its parent node%
qflow=sum(childqf);
Sflow=pflow+1i*qflow;
if abs(pflow)>branchcap_p
    validflag=0;
end
if Pnc_min~=0
   if pflow<Pnc_min || pflow>Pnc_max
        validflag=0;
   end
end

if validflag==1

    obj=sum(childobj);

else
    obj=inf;
end
```

Possible combinations of the dispatch actions of EVs and RGs:

```
function [combination,tot_num,nodetype]=coordination(EV,parkflag,pwind) combination=zeros;
wlevel=pwind;
n=0;
if EV==1
    for ev1=-3*parkflag(1):3*parkflag(1)
        for ev2=-3*parkflag(2):3*parkflag(2)
            for ev3=-3*parkflag(3):3*parkflag(3)
                for wind=0:wlevel
                    n=n+1;
                    combination(n,1:4)=[ev1,ev2,ev3,wind];
                end

            end
        end
    end
    tot_num=n;
else
    for wind=0:wlevel
        n=n+1;
        combination(n,1:4)=[0,0,0,wind];
    end
    tot_num=n;
end

%%%%nodetype definition%%%%%%%
if EV==1
    if pwind==0
        nodetype=3; %only EV connected
    else
        nodetype=2; %has both EV and DG
    end
else
    if pwind==0
        nodetype=0; %slack bus
    else
        nodetype=1; %only DG connected
```

```
        end
    end
```

Creation of the new state message array:

```
function [staterecord,toparentrecord,totnum_record]=flowcostcomb(nodetype,brflow,lineSflow,U,opcomb,opchild,opchildno,nextsoc,objdetail,num)

if nodetype==0 %leafnode
    staterecord=[brflow,lineSflow,U,opcomb,nextsoc,objdetail];
    toparentrecord=[brflow,lineSflow,U];
else
    if nodetype==1 %not leafnode staterecord=[brflow,lineSflow,U,opcomb,opchild,opchildno,nextsoc,objdetail];
        toparentrecord=[brflow,lineSflow,U];
    end
end

totnum_record=num;
```

Codes produced for studies in Chapter 6

Main Function of DG dispatch:

```
nodestate=distrinetworktest_loadchange;
memory=struct;
mem_assign=struct;
n1=0; n2=0; n3=0; n4=0;
level1=zeros; level2=zeros; level3=zeros; level4=zeros;
global Num_Ucompute Num_message gen_inc %no. of data messages sent to central memory
Num_Ucompute=0; Num_message=0; gen_inc=1;
for vn=1:9
    if nodestate(vn).calorder==1
        n1=n1+1;
        level1(n1)=vn;
    else
        if nodestate(vn).calorder==2
            n2=n2+1;
            level2(n2)=vn;
        else
            if nodestate(vn).calorder==3;
                n3=n3+1;
                level3(n3)=vn;
            else
                if nodestate(vn).calorder==4;
                    n4=n4+1;
                    level4(n4)=vn;
                end
            end
        end
    end
end

for r=1:n1 %leafnode
    node=level1(r);
    chinode=nodestate(node).chi;
    gen=nodestate(node).DG;
    CI=nodestate(node).CI;
    load=nodestate(node).load;
    cap=nodestate(node).cap;
    nodememory=mem_create(node,gen,CI,load,cap);
    memory(node).co2=nodememory.co2;
    memory(node).pf=nodememory.pf;
    memory(node).gen=nodememory.gen;
end

feasible=0;
repeat=0;
chifeasible=ones(1,9);
for r=1:n2 %nodes that have calorder=2
    node=level2(r);
    chinode=nodestate(node).chi;

    chinum=length(chinode);
    if ~isempty(chinode)
        nodememory=chi_combinetest(memory(chinode),chinode,node);

        memory(node).co2=nodememory.co2;
        memory(node).pf=nodememory.pf;
        memory(node).gen=nodememory.gen;
        while memory(node).pf(1,node)==0 && ~isempty(memory(node).co2) % minimum of utility sequence is at level 1
            gen=nodestate(node).DG;
            CI=nodestate(node).CI;
            load=nodestate(node).load;
            cap=nodestate(node).cap;
            memory_row=mem_continue(gen,CI,load,cap,memory(node),node,node,nodestate(node).chi,nodestate,[]);

            memory(node).co2=memory_row.co2; %general memory has all the details.
            memory(node).pf=memory_row.pf;
            memory(node).gen=memory_row.gen;

        end
        if isempty(memory(node).co2)
            chifeasible(node)=0;

        end
    end
end
display('level1');

for r=1:n3
    node=level3(r);
    chinode=nodestate(node).chi;
    chinum=length(chinode);
    if ~isempty(chinode) && prod(chifeasible(chinode))==1
        memory(node)=chi_combinetest(memory(chinode),chinode,node);
        testmemory=memory(node);
        tst=0;
        tstmemory=struct;
        tstsequence=zeros;
        while memory(node).pf(1,node)==0 && ~isempty(memory(node).co2)

            while prod(memory(node).pf(1,chinode))==0
                memory_chi=struct;
                for c=1:chinum
                    if memory(node).pf(1,chinode(c))==0

                        memory_chi(c).co2(1)=memory(node).co2(1);
                        memory_chi(c).pf(1,:)=memory(node).pf(1,:);
```

```
                    memory_chi(c).gen(1,:)=memory(node).gen(1,:);
                    while memory_chi(c).pf(1,chinode(c))==0
                        gen=nodestate(chinode(c)).DG;
                        CI=nodestate(chinode(c)).CI;
                        load=nodestate(chinode(c)).load;
                        cap=nodestate(chinode(c)).cap;
                        memory_row=mem_continue(gen,CI,load,cap,memory_chi(c),level2,chinode(c),nodestate(chinode(c)).chi,nodestate,level3);

                        memory_chi(c)=memory_row;

                    end

                end
            end


            memory_row=mem_rearrange(memory_chi,memory(node),chinode,level3);

            memory(node)=memory_row;
        end
        if memory(node).pf(1,node)==0 %after extending memory_chi and adding it to memory(r) and reordering the sequence, the best state needs to be cheched again
            gen=nodestate(node).DG;
            CI=nodestate(node).CI;
            load=nodestate(node).load;
            cap=nodestate(node).cap;
            memory_row=mem_continue(gen,CI,load,cap,memory(node),node,node,chinode,nodestate,[]);

            memory(node)=memory_row;

        end
    end
    if isempty(memory(node).co2)
        chifeasible(node)=0;

    end
  else
    gen=nodestate(node).DG;
    CI=nodestate(node).CI;
    load=nodestate(node).load;
    cap=nodestate(node).cap;
    nodememory=mem_create(node,gen,CI,load,cap);
    memory(node).co2=nodememory.co2;
    memory(node).pf=nodememory.pf;
    memory(node).gen=nodememory.gen;
  end
end
display('level2');

for r=1:n4
    mem_chi=struct;
    node=level4(r);
    chinode=nodestate(node).chi;
    chinum=length(chinode);
    if prod(chifeasible(chinode))==1
        for chi_col=1:chinum
            mem_chi(chi_col).co2=memory(chinode(chi_col)).co2;
            mem_chi(chi_col).pf=memory(chinode(chi_col)).pf;
            mem_chi(chi_col).gen=memory(chinode(chi_col)).gen;

        end
        memory(node)=chi_combinetest(mem_chi,chinode,node);
        validflag=0;

        while memory(node).pf(1,node)==0 && ~isempty(memory(node).co2)

            while prod(memory(node).pf(1,chinode))==0
                memory_chi=struct;
                for c=1:chinum
                    grandchi=nodestate(chinode(c)).chi;
                    granchinum=length(grandchi);

                    if memory(node).pf(1,chinode(c))==0
                        memory_chi(c).co2(1)=memory(r).co2(1);
                        memory_chi(c).pf(1,:)=memory(r).pf(1,:);
                        memory_chi(c).gen(1,:)=memory(r).gen(1,:);
                        if prod(memory_chi(c).pf(1,grandchi))==0
                            while prod(memory_chi(c).pf(1,grandchi))==0
                                memory_grandchi=struct;

                                for cc=1:granchinum
                                    if memory_chi(c).pf(1,grandchi(cc))==0

                                        memory_grandchi(cc).co2(1)=memory_chi(c).co2(1);
                                        memory_grandchi(cc).pf(1,:)=memory_chi(c).pf(1,:);
                                        memory_grandchi(cc).gen(1,:)=memory_chi(c).gen(1,:);
                                        while memory_grandchi(cc).pf(1,grandchi(cc))==0
                                            gen=nodestate(grandchi(cc)).DG;
                                            CI=nodestate(grandchi(cc)).CI;
                                            load=nodestate(grandchi(cc)).load;
                                            cap=nodestate(grandchi(cc)).cap;
                                            memory_row=mem_continue(gen,CI,load,cap,memory_grandchi(cc),level2,grandchi(cc),nodestate(grandchi(cc)).chi,nodestate,level3);

                                            memory_grandchi(cc)=memory_row;

                                        end
```

```
            end
        end
        memory_chi(c)=mem_rearrange(memory_grandchi,memory_chi(c),grandchi,level3);
                        end
                    else
                        while memory_chi(c).pf(1,chinode(c))==0

        gen=nodestate(chinode(c)).DG;
        CI=nodestate(chinode(c)).CI;
        load=nodestate(chinode(c)).load;
        cap=nodestate(chinode(c)).cap;
        memory_row=mem_continue(gen,CI,load,cap,memory_chi(c),level3,chinode(c),nodestate(chinode(c)).chi,nodestate,level3);
        memory_chi(c)=memory_row;
                        end
                    end

                end
            end
            memory(node)=mem_rearrange(memory_chi,memory(node),chinode,level3);


        end
        if memory(node).pf(1,node)==0 %after extending memory_chi and adding it to memory(r) and reordering the sequence, the best state needs to be cheched again
            gen=nodestate(node).DG;
            CI=nodestate(node).CI;
            load=nodestate(node).load;
            cap=nodestate(node).cap;
            memory_row=mem_continue(gen,CI,load,cap,memory(node),node,node,chinode,nodestate,[]);
            memory(node)=memory_row;
            display(length(memory(node).co2));
        end
    end
    if ~isempty(memory(node).co2)
        feasible=1;

    end
end

end
display('level3');
```

State creation:

```
function memory=mem_create(node,gen,CI,load,cap)
num=0;
global Num_Ucompute Num_message gen_inc

for g=1:gen_inc:gen
    accuco2=CI*g;
    Num_Ucompute=Num_Ucompute+1;
    pf=g-load;
    if abs(pf)<=cap
        if num==0
            num=num+1;
            Num_message=Num_message+1;
            nodememory.co2(num)=accuco2;

            nodememory.pf(num,1:9)=zeros; %initialize
            nodememory.gen(num,1:9)=zeros;

            nodememory.pf(num,node)=pf;
            nodememory.gen(num,node)=g;
        else
            for n=1:num
                if pf==nodememory.pf(n,node)
                    if accuco2-nodememory.co2(n)<-1e-5
                        nodememory.co2(num)=accuco2;
                        nodememory.gen(num,node)=g;
                    end
                    break;
                end
                if n==num
                    num=num+1;
                    Num_message=Num_message+1;
                    nodememory.co2(num)=accuco2;
                    nodememory.pf(num,node)=pf;
                    nodememory.gen(num,node)=g;
                end
            end
        end
    end
end

[~,index]=sort(nodememory.co2);
memory.co2=nodememory.co2(index);
memory.pf=nodememory.pf(index,:);
memory.gen=nodememory.gen(index,:);
```

State extension:

```
function memory_new=mem_continue(gen,CI,load,cap,memory,parall,par,chi,nodestate,ancesters)
num=0;
global Num_Ucompute Num_message gen_inc

memory_new=memory;
accu_co2=memory.co2(1);
chi_pf=sum(memory.pf(1,chi));
chi_gen=memory.gen(1,:);
if gen~=0
    for g=1:gen_inc:gen
        co2=accu_co2+CI*g;
        Num_Ucompute=Num_Ucompute+1;
        Num_message=Num_message+1;
        pf=chi_pf+g-load;
        tot_pf=memory.pf(1,:); % initialize with every g
        tot_pf(par)=pf;
        parpf=tot_pf(parall);
        if abs(pf)<=cap
            for n=1:length(memory_new.co2)
                cur_size=length(memory_new.co2);
                if ~isempty(ancesters)

                    if memory_new.pf(n,ancesters)==tot_pf(ancesters)

```

```
                        if prod(parpf)~=0
                            if prod(memory_new.pf(n,parall))~=0 && sum(memory_new.pf(n,parall))==sum(parpf)

                                if co2-memory_new.co2(n)<-1e-5
                                    memory_new.co2(n)=co2;
                                    memory_new.pf(n,:)=tot_pf;
                                    memory_new.gen(n,:)=chi_gen;
                                    memory_new.gen(n,par)=g;

                                end
                                break;

                            end
                        else
                            zeropar=zeros;
                            par0=0;
                            for pl=1:length(parall)
                                if parpf(pl)==0
                                    par0=par0+1;
                                    zeropar(par0)=parall(pl);
                                end
                            end
                            if memory_new.pf(n,parall)==parpf
                                flag=0;
                                for k=1:par0
                                    if tot_pf(nodestate(zeropar(k)).chi)~=memory_new.pf(n,nodestate(zeropar(k)).chi)
                                        flag=1;
                                        break;
                                    end
                                end
                                if flag==0
                                    if co2-memory_new.co2(n)<-1e-5
                                        memory_new.co2(n)=co2;
                                        memory_new.pf(n,:)=tot_pf;
                                        memory_new.gen(n,:)=chi_gen;
                                        memory_new.gen(n,par)=g;

                                    end
                                    break;
                                end
                            end

                        end
                    end
                else
                    if prod(parpf)~=0
                        if prod(memory_new.pf(n,parall))~=0 && sum(memory_new.pf(n,parall))==sum(parpf)

                            if co2-memory_new.co2(n)<-1e-5
                                memory_new.co2(n)=co2;
                                memory_new.pf(n,:)=tot_pf;
                                memory_new.gen(n,:)=chi_gen;
                                memory_new.gen(n,par)=g;

                            end
                            break;

                        end
                    else
                        zeropar=zeros;
                        par0=0;
                        for pl=1:length(parall)
                            if parpf(pl)==0
                                par0=par0+1;
                                zeropar(par0)=parall(pl);
                            end
                        end
                        if memory_new.pf(n,parall)==parpf
                            flag=0;
                            for k=1:par0
                                if tot_pf(nodestate(zeropar(k)).chi)~=memory_new.pf(n,nodestate(zeropar(k)).chi)
                                    flag=1;
                                    break;
                                end
                            end
                            if flag==0
                                if co2-memory_new.co2(n)<-1e-5
                                    memory_new.co2(n)=co2;
                                    memory_new.pf(n,:)=tot_pf;
                                    memory_new.gen(n,:)=chi_gen;
                                    memory_new.gen(n,par)=g;

                                end
                                break;
                            end
                        end

                    end
                end
                if n==cur_size
                    if num==0
                        num=num+1; %num=1 replace the best state
                        memory_new.co2(num)=co2;
                        memory_new.pf(num,:)=tot_pf;
                        memory_new.gen(num,par)=g;
                    else
                        num=num+1; %add at the end of the sequence, which doesn't matter as the sequence will be sorted later.
                        memory_new.co2(cur_size+1)=co2;
                        memory_new.pf(cur_size+1,:)=tot_pf;
                        memory_new.gen(cur_size+1,:)=chi_gen;
                        memory_new.gen(cur_size+1,par)=g;
```

```
                    end
                end
            end
        end
    end

    if num==0
        for n=1:length(memory_new.co2)-1
            memory_new.co2(n)=memory_new.co2(n+1);
            memory_new.pf(n,:)=memory_new.pf(n+1,:);
            memory_new.gen(n,:)=memory_new.gen(n+1,:);
        end
    end

else %gen=0 slack bus
    co2=accu_co2;
    pf=chi_pf;
    if abs(pf)<=cap
        tot_pf=memory.pf(1,:); % initialize with every g
        tot_pf(par)=pf;
        parpf=tot_pf(parall);

        for n=1:length(memory_new.co2)
            cur_size=length(memory_new.co2);
            if ~isempty(ancesters)
                if memory_new.pf(n,ancesters)==tot_pf(ancesters)
                    if prod(parpf)~=0
                        if prod(memory_new.pf(n,parall))~=0 && sum(memory_new.pf(n,parall))==sum(parpf)

                            if co2-memory_new.co2(n)<-1e-5
                                memory_new.co2(n)=co2;
                                memory_new.pf(n,:)=tot_pf;
                                memory_new.gen(n,:)=chi_gen;
                                memory_new.gen(n,par)=0;

                            end
                            break;

                        end
                    else
                        zeropar=zeros;
                        par0=0;
                        for pl=1:length(parall)
                            if parpf(pl)==0
                                par0=par0+1;
                                zeropar(par0)=parall(pl);
                            end
                        end
                        if memory_new.pf(n,parall)==parpf
                            flag=0;
                            for k=1:par0
                                if tot_pf(nodestate(zeropar(k)).chi)~=memory_new.pf(n,nodestate(zeropar(k)).chi)
                                    flag=1;
                                    break;
                                end
                            end
                            if flag==0
                                if co2-memory_new.co2(n)<-1e-5
                                    memory_new.co2(n)=co2;
                                    memory_new.pf(n,:)=tot_pf;
                                    memory_new.gen(n,:)=chi_gen;
                                    memory_new.gen(n,par)=0;

                                end
                                break;
                            end
                        end

                    end
                end
            else
                if prod(parpf)~=0
                    if prod(memory_new.pf(n,parall))~=0 && sum(memory_new.pf(n,parall))==sum(parpf)

                        if co2-memory_new.co2(n)<-1e-5
                            memory_new.co2(n)=co2;
                            memory_new.pf(n,:)=tot_pf;
                            memory_new.gen(n,:)=chi_gen;
                            memory_new.gen(n,par)=0;

                        end
                        break;

                    end
                else
                    zeropar=zeros;
                    par0=0;
                    for pl=1:length(parall)
                        if parpf(pl)==0
                            par0=par0+1;
                            zeropar(par0)=parall(pl);
                        end
                    end
                    if memory_new.pf(n,parall)==parpf
                        flag=0;
                        for k=1:par0
                            if tot_pf(nodestate(zeropar(k)).chi)~=memory_new.pf(n,nodestate(zeropar(k)).chi)
                                flag=1;
                                break;
                            end
                        end
                        if flag==0
                            if co2-memory_new.co2(n)<-1e-5
                                memory_new.co2(n)=co2;
                                memory_new.pf(n,:)=tot_pf;
                                memory_new.gen(n,:)=chi_gen;
                                memory_new.gen(n,par)=0;

                            end
                            break;
                        end
                    end

                end
            end
            if n==cur_size
                if num==0
```

```
                        num=num+1; % num=1 replace the best state
                        memory_new.co2(num)=co2;
                        memory_new.pf(num,:)=tot_pf;
                        memory_new.gen(num,par)=0;
                    else
                        num=num+1; % add at the end of the sequence, which doesn't matter as the sequence will be sorted later.
                        memory_new.co2(cur_size+1)=co2;
                        memory_new.pf(cur_size+1,:)=tot_pf;
                        memory_new.gen(cur_size+1,:)=chi_gen;
                        memory_new.gen(cur_size+1,par)=0;
                    end
                end

            end
        end
        if num==0
            for n=1:length(memory_new.co2)-1
                memory_new.co2(n)=memory_new.co2(n+1);
                memory_new.pf(n,:)=memory_new.pf(n+1,:);
                memory_new.gen(n,:)=memory_new.gen(n+1,:);
            end
            memory_new.co2(length(memory_new.co2))=[];
            memory_new.pf(length(memory_new.co2),:)=[];
            memory_new.gen(length(memory_new.co2),:)=[];
        end
    end


    [~,index]=sort(memory_new.co2);
    memory_new.co2=memory_new.co2(index);
    memory_new.pf=memory_new.pf(index,:);
    memory_new.gen=memory_new.gen(index,:);
```

Queue combination:

```
function memory=chi_combinetest(memory_chi,chi,node)
len=zeros(1,9);
len1=length(memory_chi(1).co2);
len2=length(memory_chi(2).co2);

if node==1
    len(chi(1))=1;
    len(chi(2))=1;
end
if node==4
    len(chi(1))=len1;
    len(chi(2))=len2;
end
if node==5
    len(chi(1))=len1;
    len(chi(2))=len2;
end

if node==3

    for l=1:len1
        if memory_chi(1).pf(l,chi(1))==0 || abs(memory_chi(1).pf(l,chi(1)))>=17.025
            len(chi(1))=len(chi(1))+1;
        end
        if abs(memory_chi(1).pf(l,chi(1)))~=0 && abs(memory_chi(1).pf(l,chi(1)))<17.025
            break;
        end

    end
    for l=1:len2
        if memory_chi(2).pf(l,chi(2))==0 || abs(memory_chi(2).pf(l,chi(2)))>=17.025
            len(chi(2))=len(chi(2))+1;
        end
        if abs(memory_chi(2).pf(l,chi(2)))~=0 && abs(memory_chi(2).pf(l,chi(2)))<17.025
            break;
        end

    end

end

num=0;

for l1=1:len(chi(1))
    for l2=1:len(chi(2))
        if num==0
            num=num+1;
            memory.co2(num)=memory_chi(1).co2(l1)+memory_chi(2).co2(l2);
memory_chi(1).pf(l1,length(memory_chi(1).pf(l1,:))+1:9)=zeros;
memory_chi(2).pf(l2,length(memory_chi(2).pf(l2,:))+1:9)=zeros;
memory.pf(num,1:9)=memory_chi(1).pf(l1,:)+memory_chi(2).pf(l2,:); memory.gen(num,1:9)=memory_chi(1).gen(l1,:)+memory_chi(2).gen(l2,:);
        else
            co2=memory_chi(1).co2(l1)+memory_chi(2).co2(l2);
pf(1:9)=memory_chi(1).pf(l1,:)+memory_chi(2).pf(l2,:);
gen(1:9)=memory_chi(1).gen(l1,:)+memory_chi(2).gen(l2,:);
            for n=1:num
                if prod(memory.pf(n,chi))~=0 && prod(pf(chi))~=0
                    if sum(memory.pf(n,chi))== sum(pf(chi))
                        if co2-memory.co2(n)<-1e-5
                            memory.co2(n)=co2;
                            memory.pf(n,:)=pf;
                            memory.gen(n,:)=gen;

                        end
                        break;
                    end

                end
                if n==num
                    num=num+1;
                    memory.co2(num)=co2;
                    memory.pf(num,:)=pf;
                    memory.gen(num,:)=gen;
                end
            end


        end
    end
end
[~,index]=sort(memory.co2);
memory.co2=memory.co2(index);
memory.pf=memory.pf(index,:);
memory.gen=memory.gen(index,:);
```

Queue processing:

```
function memory_new=mem_rearrange(memory_chi,memory,allpar,ancesters) %allpar has 2 elements here
len=length(memory_chi); %in this case, len<=2
num=0;
nonzero=zeros;

for l=1:len
    if ~isempty(memory_chi(l).co2)
        num=num+1;
        nonzero(num)=l;
    end
end
if num==1
    memory_new=memory;
    differ=0;
    for ch=1:length(memory_chi(nonzero).co2)
        for mem_s=1:length(memory_new.co2)
            cur_size=length(memory_new.co2);
            if ~isempty(ancesters)
                if memory_new.pf(mem_s,ancesters)==memory_chi(nonzero).pf(ch,ancesters)
                    if prod(memory_new.pf(mem_s,allpar))~=0 && prod(memory_chi(nonzero).pf(ch,allpar))~=0
                        if sum(memory_new.pf(mem_s,allpar))==sum(memory_chi(nonzero).pf(ch,allpar))
                            % display(mem_s);
                            if memory_chi(nonzero).co2(ch)-memory_new.co2(mem_s)<-1e-5
                                memory_new.co2(mem_s)=memory_chi(nonzero).co2(ch);
                                memory_new.pf(mem_s,:)=memory_chi(nonzero).pf(ch,:);
                                memory_new.gen(mem_s,:)=memory_chi(nonzero).gen(ch,:);

                            end
                            break;
                        end
                    end
                end
            else
                if prod(memory_new.pf(mem_s,allpar))~=0 && prod(memory_chi(nonzero).pf(ch,allpar))~=0
                    if sum(memory_new.pf(mem_s,allpar))==sum(memory_chi(nonzero).pf(ch,allpar))
                        %display(mem_s);
                        if memory_chi(nonzero).co2(ch)-memory_new.co2(mem_s)<-1e-5

                            memory_new.co2(mem_s)=memory_chi(nonzero).co2(ch);
                            memory_new.pf(mem_s,:)=memory_chi(nonzero).pf(ch,:);
                            memory_new.gen(mem_s,:)=memory_chi(nonzero).gen(ch,:);

                        end
                        break;
                    end
                end
            end
            if mem_s==cur_size
                if differ==0 % replace the best state with first different pf
                    differ=differ+1;
                    memory_new.co2(1)=memory_chi(nonzero).co2(ch);
                    memory_new.pf(1,:)=memory_chi(nonzero).pf(ch,:);
                    memory_new.gen(1,:)=memory_chi(nonzero).gen(ch,:);
                else
                    %add at the end of the sequence, which doesn't matter as the sequence will be sorted later.
                    differ=differ+1;
                    memory_new.co2(cur_size+1)=memory_chi(nonzero).co2(ch);
                    memory_new.pf(cur_size+1,:)=memory_chi(nonzero).pf(ch,:);
                    memory_new.gen(cur_size+1,:)=memory_chi(nonzero).gen(ch,:);

                end
            end
        end

    end
else
    n=0;
    oldco2=memory.co2(1);
    oldpf=memory.pf(1,:);
    oldgen=memory.gen(1,:);
    for ch1=1:length(memory_chi(1).co2)
        for ch2=1:length(memory_chi(2).co2)

            newco2=memory_chi(1).co2(ch1)+memory_chi(2).co2(ch2)-oldco2;
            newpf=memory_chi(1).pf(ch1,:)+memory_chi(2).pf(ch2,:)-oldpf;
            newgen=memory_chi(1).gen(ch1,:)+memory_chi(2).gen(ch2,:)-oldgen;

            if n==0
                n=n+1;
                memory_chi_combine.co2(n)=newco2;
                memory_chi_combine.pf(n,:)=newpf;
                memory_chi_combine.gen(n,:)=newgen;
            else
                for nc=1:n
                    if sum(newpf(allpar))==sum(memory_chi_combine.pf(nc,allpar)) %other elements sure to be same.
                        if newco2-memory_chi_combine.co2(nc)<-1e-5
                            memory_chi_combine.co2(nc)=newco2;
                            memory_chi_combine.pf(nc,:)=newpf;
                            memory_chi_combine.gen(nc,:)=newgen;
                        end
                        break;
                    else
                        if nc==n
                            n=n+1;
```

```
                            memory_chi_combine.co2(n)=newco2;
                            memory_chi_combine.pf(n,:)=newpf;
                            memory_chi_combine.gen(n,:)=newgen;
                        end
                    end
                end
            end
        end
    end

    memory_new=memory;
    differ=0;
    for ch=1:length(memory_chi_combine.co2)
        for mem_s=1:length(memory_new.co2)
            cur_size=length(memory_new.co2);
            if ~isempty(ancesters)
                if memory_new.pf(mem_s,ancesters)==memory_chi_combine.pf(ch,ancesters)
                    if prod(memory_new.pf(mem_s,allpar))~=0 && prod(memory_chi_combine.pf(ch,allpar))~=0
                        if sum(memory_new.pf(mem_s,allpar))==sum(memory_chi_combine.pf(ch,allpar))
                            if memory_chi_combine.co2(ch)-memory_new.co2(mem_s)<-1e-5
                                memory_new.co2(mem_s)=memory_chi_combine.co2(ch);
                                memory_new.pf(mem_s,:)=memory_chi_combine.pf(ch,:);
                                memory_new.gen(mem_s,:)=memory_chi_combine.gen(ch,:);

                            end
                            break;
                        end
                    end
                end
            else
                if prod(memory_new.pf(mem_s,allpar))~=0 && prod(memory_chi_combine.pf(ch,allpar))~=0
                    if sum(memory_new.pf(mem_s,allpar))==sum(memory_chi_combine.pf(ch,allpar))
                        if memory_chi_combine.co2(ch)-memory_new.co2(mem_s)<-1e-5
                            memory_new.co2(mem_s)=memory_chi_combine.co2(ch);
                            memory_new.pf(mem_s,:)=memory_chi_combine.pf(ch,:);
                            memory_new.gen(mem_s,:)=memory_chi_combine.gen(ch,:);

                        end
                        break;
                    end
                end
            end

            if mem_s==cur_size
                if differ==0 % replace the best state with first different pf
                    differ=differ+1;
                    memory_new.co2(1)=memory_chi_combine.co2(ch);
                    memory_new.pf(1,:)=memory_chi_combine.pf(ch,:);
                    memory_new.gen(1,:)=memory_chi_combine.gen(ch,:);
                else
                    %add at the end of the sequence, which doesn't matter as the sequence will be sorted later.
                    differ=differ+1;
                    memory_new.co2(cur_size+1)=memory_chi_combine.co2(ch);
                    memory_new.pf(cur_size+1,:)=memory_chi_combine.pf(ch,:);
                    memory_new.gen(cur_size+1,:)=memory_chi_combine.gen(ch,:);

                end
            end
        end

    end
end
if differ==0
    for n=1:length(memory_new.co2)-1
        memory_new.co2(n)=memory_new.co2(n+1);
        memory_new.pf(n,:)=memory_new.pf(n+1,:);
        memory_new.gen(n,:)=memory_new.gen(n+1,:);
    end
    memory_new.co2(length(memory_new.co2))=[];
    memory_new.pf(length(memory_new.co2),:)=[];
    memory_new.gen(length(memory_new.co2),:)=[];
end

[~,index]=sort(memory_new.co2);
memory_new.co2=memory_new.co2(index);
memory_new.pf=memory_new.pf(index,:);
memory_new.gen=memory_new.gen(index,:);
```

Main Function of RG&EV dispatch:

```
global Num_Ucompute Num_message % no. of data messages sent to central memory
Num_Ucompute=0; Num_message=0;
global gen_inc asbp assp hbp lsp maxsp minbp Pload Qload aveload load_max load_min load_h load_l
gen_inc=1;
[asbp, assp, hbp, lsp, maxsp, minbp]=price;
[Pload,Qload,aveload,load_max,load_min,load_h,load_l]=loadcalculate;

nodestate=distrinetwork1;
day=7;
EV_dispatch=zeros(48*day,12,3);
RG_disptach=zeros(48*day,12);
networkload=zeros(48*day,1);
networkcost=zeros(48*day,1);
EV_travel=struct;
n1=0; n2=0; n3=0; n4=0;
level1=zeros; level2=zeros; level3=zeros;


```

```
%%%%%%% Wind power %%%%%%%%%%%%%%%%%%%%
if day==1
    tot_RG=xlsread('networkdata.xlsx','wind','BT2:BX49');
else
    tot_RG=xlsread('networkdata.xlsx','wind','AW2:BA337');
end
n_RG=0;
node_RG=zeros(length(tot_RG(:,1)),12);
for vn=1:12
    if nodestate(vn).DG==1
        n_RG=n_RG+1;
        node_RG(:,vn)=round(tot_RG(:,n_RG));
    end
end
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%% EV travel %%%%%%%%%%%%%%%%%%%%
commute_state=randsample([0,1,2],33,true,[0.17 0.65 0.18]); %0:idle all day 1:single h2h trip 2: double h2h trip
EVcommuting=zeros(1,12);
EV_start=zeros(2,12);
EV_end=zeros(2,12);
EV_dist=zeros(2,12);
singletrip=xlsread('networkdata.xlsx','EV','V3:Y102');
doubletrip=xlsread('networkdata.xlsx','EV','AA3:AH102');
n_EV=0;
n_EV_single=0;
n_EV_double=0;
m=1150; A=0.015*9.8*1150; B=0.5*1.29*1.5^2*0.3; efficiency=0.7; %parameters of energy formula
voltage=240; %battery rating:240V 100Ah
k=1.2; %peukert coefficient
Cp=20^k*5;

for vn=1:12

    if nodestate(vn).EV==1
        for nodeEVgroup=1:3
            n_EV=n_EV+1;
            EV_travel(vn).SOC(1,nodeEVgroup)=normrnd(0.5,0.1); %initial SOC
            while EV_travel(vn).SOC(1,nodeEVgroup)>1 || EV_travel(vn).SOC(1,nodeEVgroup)<0
                EV_travel(vn).SOC(1,nodeEVgroup)=normrnd(0.5,0.1); %initial SOC
            end
            EV_travel(vn).commute(nodeEVgroup)=commute_state(n_EV);
            if EV_travel(vn).commute(nodeEVgroup)==0
                EV_travel(vn).SOCp(1:48,nodeEVgroup)=zeros;
            end
            if EV_travel(vn).commute(nodeEVgroup)==1 %% single h2h trip
                n_EV_single=n_EV_single+1;
                EV_travel(vn).starttime(1,nodeEVgroup)=EVtraveltime_round(singletrip(n_EV_single,1))*2; %transfer time to time interval
                EV_travel(vn).endtime(1,nodeEVgroup)=EVtraveltime_round(singletrip(n_EV_single,2))*2;

                if EV_travel(vn).endtime(1,nodeEVgroup)>48
                    EV_travel(vn).endtime(1,nodeEVgroup)=EV_travel(vn).endtime(1,nodeEVgroup)-48;
                end

                EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-singletrip(n_EV_single,4)))); %km/h
                while EV_travel(vn).speed(1,nodeEVgroup)<=singletrip(n_EV_single,4)
                    EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-singletrip(n_EV_single,4)))); %km/h
                end
                EV_travel(vn).dist(1,nodeEVgroup)=singletrip(n_EV_single,3); %km
                EV_travel(vn).duration(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)/EV_travel(vn).speed(1,nodeEVgroup); %hours
                EV_travel(vn).energy(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(1,nodeEVgroup)*1000/3600)^2); %J
                EV_travel(vn).current(1,nodeEVgroup)=EV_travel(vn).energy(1,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(1,nodeEVgroup)*3600);
                EV_travel(vn).Ca(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*Cp/(EV_travel(vn).current(1,nodeEVgroup)^k);
                EV_travel(vn).consumeSOC(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*EV_travel(vn).duration(1,nodeEVgroup)/EV_travel(vn).Ca(1,nodeEVgroup);

                while EV_travel(vn).consumeSOC(1,nodeEVgroup)>0.5
                    n_EV_single=n_EV_single+1;
                    EV_travel(vn).starttime(1,nodeEVgroup)=EVtraveltime_round(singletrip(n_EV_single,1))*2; %transfer time to time interval
                    EV_travel(vn).endtime(1,nodeEVgroup)=EVtraveltime_round(singletrip(n_EV_single,2))*2;

                    if EV_travel(vn).endtime(1,nodeEVgroup)>48
                        EV_travel(vn).endtime(1,nodeEVgroup)=EV_travel(vn).endtime(1,nodeEVgroup)-48;
                    end
                    EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-singletrip(n_EV_single,4)))); %km/h
                    while EV_travel(vn).speed(1,nodeEVgroup)<=singletrip(n_EV_single,4)
```

```
                    EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-singletrip(n_EV_single,4))));% km/h
                end
                EV_travel(vn).dist(1,nodeEVgroup)=singletrip(n_EV_single,3);
                EV_travel(vn).duration(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)/EV_travel(vn).speed(1,nodeEVgroup);
                EV_travel(vn).energy(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(1,nodeEVgroup)*1000/3600)^2);
                EV_travel(vn).current(1,nodeEVgroup)=EV_travel(vn).energy(1,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(1,nodeEVgroup)*3600);
                EV_travel(vn).Ca(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*Cp/(EV_travel(vn).current(1,nodeEVgroup)^k);
                EV_travel(vn).consumeSOC(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*EV_travel(vn).duration(1,nodeEVgroup)/EV_travel(vn).Ca(1,nodeEVgroup);
            end

            EV_travel(vn).SOCp(1:48,nodeEVgroup)=zeros;
            EV_travel(vn).SOCp(EV_travel(vn).starttime(1,nodeEVgroup)-4:EV_travel(vn).starttime(1,nodeEVgroup)-1,nodeEVgroup)=ones(4,1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);

        end
        if EV_travel(vn).commute(nodeEVgroup)==2 %% double h2h trips
            n_EV_double=n_EV_double+1;
            EV_travel(vn).starttime(1,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,1))*2;

            EV_travel(vn).starttime(2,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,2))*2;

            EV_travel(vn).endtime(1,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,3))*2;
            if EV_travel(vn).endtime(1,nodeEVgroup)>48
                EV_travel(vn).endtime(1,nodeEVgroup)=EV_travel(vn).endtime(1,nodeEVgroup)-48;
            end

            EV_travel(vn).endtime(2,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,4))*2;
            if EV_travel(vn).endtime(2,nodeEVgroup)>48
                EV_travel(vn).endtime(2,nodeEVgroup)=EV_travel(vn).endtime(2,nodeEVgroup)-48;
            end


            EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,7))));% km/h
            while EV_travel(vn).speed(1,nodeEVgroup)<=doubletrip(n_EV_double,7)
                EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,7))));% km/h
            end
            EV_travel(vn).dist(1,nodeEVgroup)=doubletrip(n_EV_double,5);
            EV_travel(vn).duration(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)/EV_travel(vn).speed(1,nodeEVgroup);
            EV_travel(vn).energy(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(1,nodeEVgroup)*1000/3600)^2);
            EV_travel(vn).current(1,nodeEVgroup)=EV_travel(vn).energy(1,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(1,nodeEVgroup)*3600);
            EV_travel(vn).Ca(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*Cp/(EV_travel(vn).current(1,nodeEVgroup)^k);
            EV_travel(vn).consumeSOC(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*EV_travel(vn).duration(1,nodeEVgroup)/EV_travel(vn).Ca(1,nodeEVgroup);

            EV_travel(vn).speed(2,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,8))));% km/h
            while EV_travel(vn).speed(2,nodeEVgroup)<=doubletrip(n_EV_double,8)
                EV_travel(vn).speed(2,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,8))));% km/h
            end
            EV_travel(vn).dist(2,nodeEVgroup)=doubletrip(n_EV_double,6);
            EV_travel(vn).duration(2,nodeEVgroup)=EV_travel(vn).dist(2,nodeEVgroup)/EV_travel(vn).speed(2,nodeEVgroup);
            EV_travel(vn).energy(2,nodeEVgroup)=EV_travel(vn).dist(2,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(2,nodeEVgroup)*1000/3600)^2);
            EV_travel(vn).current(2,nodeEVgroup)=EV_travel(vn).energy(2,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(2,nodeEVgroup)*3600);
```

```
            EV_travel(vn).Ca(2,nodeEVgroup)=EV_travel(vn).current(2,nodeEVgroup)*Cp/(EV_travel(vn).current(2,nodeEVgroup)^k);
            EV_travel(vn).consumeSOC(2,nodeEVgroup)=EV_travel(vn).current(2,nodeEVgroup)*EV_travel(vn).duration(2,nodeEVgroup)/EV_travel(vn).Ca(2,nodeEVgroup);

            while EV_travel(vn).consumeSOC(1,nodeEVgroup)>0.5 || EV_travel(vn).consumeSOC(2,nodeEVgroup)>0.5
                n_EV_double=n_EV_double+1;
                EV_travel(vn).starttime(1,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,1))*2;

                EV_travel(vn).starttime(2,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,2))*2;

                EV_travel(vn).endtime(1,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,3))*2;
                if EV_travel(vn).endtime(1,nodeEVgroup)>48
                    EV_travel(vn).endtime(1,nodeEVgroup)=EV_travel(vn).endtime(1,nodeEVgroup)-48;
                end

                EV_travel(vn).endtime(2,nodeEVgroup)=EVtraveltime_round(doubletrip(n_EV_double,4))*2;
                if EV_travel(vn).endtime(2,nodeEVgroup)>48
                    EV_travel(vn).endtime(2,nodeEVgroup)=EV_travel(vn).endtime(2,nodeEVgroup)-48;
                end

                EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,7))));%km/h
                while EV_travel(vn).speed(1,nodeEVgroup)<=doubletrip(n_EV_double,7)
                    EV_travel(vn).speed(1,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,7))));%km/h
                end
                EV_travel(vn).dist(1,nodeEVgroup)=doubletrip(n_EV_double,5);
                EV_travel(vn).duration(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)/EV_travel(vn).speed(1,nodeEVgroup);
                EV_travel(vn).energy(1,nodeEVgroup)=EV_travel(vn).dist(1,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(1,nodeEVgroup)*1000/3600)^2);
                EV_travel(vn).current(1,nodeEVgroup)=EV_travel(vn).energy(1,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(1,nodeEVgroup)*3600);
                EV_travel(vn).Ca(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*Cp/(EV_travel(vn).current(1,nodeEVgroup)^k);
                EV_travel(vn).consumeSOC(1,nodeEVgroup)=EV_travel(vn).current(1,nodeEVgroup)*EV_travel(vn).duration(1,nodeEVgroup)/EV_travel(vn).Ca(1,nodeEVgroup);

                EV_travel(vn).speed(2,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,8))));%km/h
                while EV_travel(vn).speed(2,nodeEVgroup)<=doubletrip(n_EV_double,8)
                    EV_travel(vn).speed(2,nodeEVgroup)=normrnd(65,0.5*(min(113-65,65-doubletrip(n_EV_double,8))));%km/h
                end
                EV_travel(vn).dist(2,nodeEVgroup)=doubletrip(n_EV_double,6);
                EV_travel(vn).duration(2,nodeEVgroup)=EV_travel(vn).dist(2,nodeEVgroup)/EV_travel(vn).speed(2,nodeEVgroup);
                EV_travel(vn).energy(2,nodeEVgroup)=EV_travel(vn).dist(2,nodeEVgroup)*1000*(A+B*(EV_travel(vn).speed(2,nodeEVgroup)*1000/3600)^2);
                EV_travel(vn).current(2,nodeEVgroup)=EV_travel(vn).energy(2,nodeEVgroup)/efficiency/voltage/(EV_travel(vn).duration(2,nodeEVgroup)*3600);
                EV_travel(vn).Ca(2,nodeEVgroup)=EV_travel(vn).current(2,nodeEVgroup)*Cp/(EV_travel(vn).current(2,nodeEVgroup)^k);
                EV_travel(vn).consumeSOC(2,nodeEVgroup)=EV_travel(vn).current(2,nodeEVgroup)*EV_travel(vn).duration(2,nodeEVgroup)/EV_travel(vn).Ca(2,nodeEVgroup);
            end

            EV_travel(vn).SOCp(1:48,nodeEVgroup)=zeros;
            if EV_travel(vn).starttime(1,nodeEVgroup)<EV_travel(vn).endtime(2,nodeEVgroup)
                if 48-(EV_travel(vn).endtime(2,nodeEVgroup)-EV_travel(vn).starttime(1,nodeEVgroup))>4
                    if EV_travel(vn).starttime(1,nodeEVgroup)>4
                        EV_travel(vn).SOCp(EV_travel(vn).starttime(1,nodeEVgroup)-4:EV_travel(vn).starttime(1,nodeEVgroup)-1,nodeEVgroup)=ones(4,1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);
                    else
                        for t=1:4
```

```
                        if EV_travel(vn).starttime(1,nodeEVgroup)-t>0
                            EV_travel(vn).SOCp(EV_travel(vn).starttime(1,nodeEVgroup)-t,nodeEVgroup)=EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4;
                        else
                            EV_travel(vn).SOCp((EV_travel(vn).starttime(1,nodeEVgroup)-t+48),nodeEVgroup)=EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4;
                        end
                    end
                end
            else
                EV_travel(vn).SOCp(1:EV_travel(vn).starttime(1,nodeEVgroup)-1,nodeEVgroup)=ones(EV_travel(vn).starttime(1,nodeEVgroup)-1,1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);
                EV_travel(vn).SOCp(EV_travel(vn).endtime(2,nodeEVgroup):48,nodeEVgroup)=ones(48-EV_travel(vn).endtime(2,nodeEVgroup)+1,1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);
            end
        else
            if EV_travel(vn).starttime(1,nodeEVgroup)-EV_travel(vn).endtime(2,nodeEVgroup)>4
                EV_travel(vn).SOCp(EV_travel(vn).starttime(1,nodeEVgroup)-4:EV_travel(vn).starttime(1,nodeEVgroup)-1,nodeEVgroup)=ones(4,1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);
            else
                EV_travel(vn).SOCp(EV_travel(vn).endtime(2,nodeEVgroup):EV_travel(vn).starttime(1,nodeEVgroup)-1,nodeEVgroup)=ones(EV_travel(vn).starttime(1,nodeEVgroup)-EV_travel(vn).endtime(2,nodeEVgroup),1)*(EV_travel(vn).consumeSOC(1,nodeEVgroup)+0.4);

            end
        end

        if EV_travel(vn).starttime(2,nodeEVgroup)-EV_travel(vn).endtime(1,nodeEVgroup)>=4
            EV_travel(vn).SOCp(EV_travel(vn).starttime(2,nodeEVgroup)-4:EV_travel(vn).starttime(2,nodeEVgroup)-1,nodeEVgroup)=ones(4,1)*(EV_travel(vn).consumeSOC(2,nodeEVgroup)+0.4);
        else
            EV_travel(vn).SOCp(EV_travel(vn).endtime(1,nodeEVgroup):EV_travel(vn).starttime(2,nodeEVgroup)-1,nodeEVgroup)=ones((EV_travel(vn).starttime(2,nodeEVgroup)-EV_travel(vn).endtime(1,nodeEVgroup)),1)*(EV_travel(vn).consumeSOC(2,nodeEVgroup)+0.4);
        end


            end
        end
    end
end

for vn=1:12
    if nodestate(vn).calorder==1
        n1=n1+1;
        level1(n1)=vn;
    else
        if nodestate(vn).calorder==2
            n2=n2+1;
            level2(n2)=vn;
        else
            if nodestate(vn).calorder==3
                n3=n3+1;
                level3(n3)=vn;

            end
        end
    end
end

for time=1:48*day
    if mod(time,48)==0
        dt=48;
    else
        dt=mod(time,48);
    end

    memory=struct;

    chifeasible=ones(1,12);
    %%%%%%%%%%leafnode %%%%%%%%%%%%%
    for r=1:n1
        node=level1(r);

        gen=node_RG(time,node);

        cap=nodestate(node).cap;

        nodememory=mem_create(time,node,gen,EV_travel(node),cap);

        memory(node).obj=nodememory.obj;
        memory(node).pflow=nodememory.pflow;
        memory(node).Sflow=nodememory.Sflow;
        memory(node).EV=nodememory.EV;
        memory(node).RG=nodememory.RG;
        memory(node).totload=nodememory.totload;
        memory(node).totcost=nodememory.totcost;
        memory(node).soc_n=nodememory.soc_n;
        memory(node).obj_info=nodememory.obj_info;

        if isempty(memory(node).obj)
            chifeasible(node)=0;
        end
    end

    %%%%%%%%% node 2 %%%%%%%%%%%%%%%
    feasible=0;
    repeat=0;

    for r=1:n2 %nodes that have calorder=2
        node=level2(r);
        chinode=nodestate(node).chi;

        memory(node).obj=memory(chinode).obj;
        memory(node).pflow=memory(chinode).pflow;
```

```
        memory(node).Sflow=memory(chinode).Sflow;
        memory(node).EV=memory(chinode).EV;
        memory(node).RG=memory(chinode).RG;
        memory(node).totload=memory(chinode).totload;
        memory(node).totcost=memory(chinode).totcost;
        memory(node).soc_n=memory(chinode).soc_n;
        memory(node).obj_info=memory(chinode).obj_info;

        while memory(node).pflow(1,node)==0 && ~isempty(memory(node).obj) % minimum of utility sequence is at level 1
            gen=node_RG(time,node);
            cap=nodestate(node).cap;
            memory_row=mem_continue(time,node,memory(node),gen,EV_travel(node),cap,node,nodestate(node).chi);

            memory(node).obj=memory_row.obj;
            memory(node).pflow=memory_row.pflow;
            memory(node).Sflow=memory_row.Sflow;
            memory(node).EV=memory_row.EV;
            memory(node).RG=memory_row.RG;
            memory(node).totload=memory_row.totload;
            memory(node).totcost=memory_row.totcost;
            memory(node).soc_n=memory_row.soc_n;
            memory(node).obj_info=memory_row.obj_info;

        end
        if isempty(memory(node).obj)
            chifeasible(node)=0;

        end

    end
    disp('level1');

    %%%%%%%%% node 3 %%%%%%%%%%%%%%%
    for r=1:n3
        node=level3(r);
        chinode=nodestate(node).chi;
        chinum=length(chinode);
        if ~isempty(chinode) && prod(chifeasible(chinode))==1
            memory_row=chi_combinetest(memory(chinode),chinode,node);
            memory(node).obj=memory_row.obj;
            memory(node).pflow=memory_row.pflow;
            memory(node).Sflow=memory_row.Sflow;
            memory(node).EV=memory_row.EV;
            memory(node).RG=memory_row.RG;
            memory(node).totload=memory_row.totload;
            memory(node).totcost=memory_row.totcost;
            memory(node).soc_n=memory_row.soc_n;
            memory(node).obj_info=memory_row.obj_info;

            while memory(node).pflow(1,node)==0 && ~isempty(memory(node).obj)

                while prod(memory(node).pflow(1,chinode))==0
                    memory_chi=struct;
                    for c=1:chinum
                        if memory(node).pflow(1,chinode(c))==0

                            memory_chi(c).obj(1)=memory(node).obj(1);
                            memory_chi(c).pflow(1,:)=memory(node).pflow(1,:);
                            memory_chi(c).Sflow(1,:)=memory(node).Sflow(1,:);
                            memory_chi(c).soc_n(1,:,:)=memory(node).soc_n(1,:,:);
                            memory_chi(c).EV(1,:,:)=memory(node).EV(1,:,:);
                            memory_chi(c).RG(1,:)=memory(node).RG(1,:);
                            memory_chi(c).totload(1,:)=memory(node).totload(1,:);
                            memory_chi(c).totcost(1,:)=memory(node).totcost(1,:);
                            memory_chi(c).obj_info(1,:)=memory(node).obj_info(1,:);

                            while memory_chi(c).pflow(1,chinode(c))==0
                                gen=node_RG(time,node);
                                cap=nodestate(chinode(c)).cap;
                                memory_row=mem_continue(time,chinode(c),memory_chi(c),gen,EV_travel(chinode(c)),cap,level2,nodestate(chinode(c)).chi);
                                memory_chi(c)=memory_row;

                            end

                        end
                    end


                    memory_row=mem_rearrange(memory_chi,memory(node),chinode);

                    memory(node)=memory_row;



                end
                if memory(node).pflow(1,node)==0 %after extending memory_chi and adding it to memory(r) and reordering the sequence, the best state needs to be checked again
                    gen=node_RG(time,node);

                    cap=nodestate(node).cap;

                    memory_row=mem_continue(time,node,memory(node),gen,EV_travel(node),cap,node,chinode);

```

```
                        memory(node)=memory_row;

                    end
                end
                if isempty(memory(node).obj)
                    chifeasible(node)=0;

                end

            end
        end
        disp('level2');

        for node=1:12
            if nodestate(node).EV==1
                EV_travel(node).SOC(time+1,:)=memory(1).soc_n(1,node,:);
                EV_dispatch(time,node,1:3)=memory(1).EV(1,node,:);

                for EVgroup=1:3
                    if EV_travel(node).commute(EVgroup)==1
                        if dt+1==EV_travel(node).endtime(1,EVgroup)
                            EV_travel(node).SOC(time+1,EVgroup)=EV_travel(node).SOC(time,EVgroup)-EV_travel(node).consumeSOC(1,EVgroup);
                        end
                    end
                    if EV_travel(node).commute(EVgroup)==2
                        if dt+1==EV_travel(node).endtime(1,EVgroup)
                            EV_travel(node).SOC(time+1,EVgroup)=EV_travel(node).SOC(time,EVgroup)-EV_travel(node).consumeSOC(1,EVgroup);
                        end
                        if dt+1==EV_travel(node).endtime(2,EVgroup)
                            EV_travel(node).SOC(time+1,EVgroup)=EV_travel(node).SOC(time,EVgroup)-EV_travel(node).consumeSOC(2,EVgroup);
                        end
                    end
                end
            end
            if nodestate(node).DG==1
                RG_disptach(time,node)=memory(1).RG(1,node);

            end
        end
        networkload(time)=sum(memory(1).totload(1,:));
        networkcost(time)=sum(memory(1).totcost(1,:));

    end
```