Title
Fast Algorithms for Transmission Switching with High Performance Computing

Author
YANG, ZHU

Publication Date
2018

Peer reviewed|Thesis/dissertation

Fast Algorithms for Transmission Switching with High Performance Computing

by

Zhu Yang

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Industrial Engineering and Operations Research

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Shmuel S. Oren, Chair
Associate Professor Javad Lavaeiyanesi
Associate Professor Duncan Callaway

Fall 2018

Fast Algorithms for Transmission Switching with High Performance Computing

Abstract

Fast Algorithms for Transmission Switching with High Performance Computing

by

Zhu Yang

Doctor of Philosophy in Engineering – Industrial Engineering and Operations Research

University of California, Berkeley

Professor Shmuel S. Oren, Chair

There has been multiple national directives to enhance the economic operations of the power transmission system and promote the efficient use of the current grid configuration and resources. Transmission switching has been proposed as a new control method for various benefits, including improving the economic efficiency and meeting the reliability requirements. Optimal Transmission Switching model has been introduced to find an optimal generation dispatch and network topology to minimize the dispatch cost. Binary decision variables are used to denote the control of the transmission lines, which makes the model a nonlinear program. It suffers from curse of dimensionality and faces serious computational challenges.

To tackle the computational challenge of the Optimal Transmission Switching model, we propose three greedy algorithms in which only one line is switched at an iteration. In each iteration we solve a series of linear programs or smaller MIP programs, which can be implemented in parallel with the aid of high performance computing. The first algorithm enumerates all the possible line switching actions. The second algorithm produces a priority list ranking lines by a sensitivity factor based on dual criterion, and evaluate lines starting from the top of the list. The last one divides lines into small groups and consider each group at one time. We test the algorithms on the IEEE 118-Bus network and the FERC 13,867-Bus network which is representative of PJM Regional Transmission Organization. The results show that all three proposed algorithms result in cost reduction close to the best known optimal within a reasonable timeframe for IEEE network. For the FERC network which can not be solved directly by the OTS, the first two greedy algorithms are able to produce switching sequences which result in considerable cost reduction.

Furthermore we propose three machine learning based methods to produce priority lists for ranking possible line switching actions to facilitate faster searching. The algorithms take in the parameters from network status and network configuration to produce a standardized score representing the possible cost reduction of the line switching action. The numerical results on IEEE network and FERC network are presented. We evaluate the effectiveness of the priority lists based on dual criterion and machine learning methods by both regression analysis and comparison with random lists.

Based on the algorithms in literature and those we develop, we propose an algorithm selection method which selects the algorithm to optimize the cost improvement at each iteration. They show improvement in the cost reduction compared to the individual algorithms, especially for the FERC network.

To my parents,
Lin Zhang and Jian Yang

Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to express my deep gratitude to my advisor, Prof Shmuel Oren, for his guidance and encouragement through my Ph.D. journey. He has always been helpful and supportive, even through many research and life barriers. I have appreciated his expertise in many areas in the power systems, which inspired my research topics, and his patience and kindness towards his students. I also want to thank my committee members Prof Javad Lavaei, Prof Duncan Callaway for their advice and feedback to my research. I spent one great summer in Arizona State University in 2014 with Prof Kory Hedman and his research group. I want to express my gratitude for their help and thought provoking discussions, from which I have learned a lot. Dr. Rajan Deepak has assisted me obtaining the clearance for the computing resources in Lawrence Livermore National Laboratory without which I couldn't finish my thesis. He provided some deep insights on efficiently solving large scale optimization programs which I am grateful for. I also want to express appreciation to Prof Ilan Adler for his help in my first year and before the preliminary exam. He provided enormous assistance and guidance when I expressed my interest in the graduate program of IEOR during my senior year in college. I want to thank the members of my research group for their feedback and friendship: Anthony Papavasiliou, Tanachai Limpaitoon, Ruoyang Li, Chen Chen, Clay Campaigne, Jiaying Shi and Georgios Patsakis.

I want to thank my parents for their love and support along the way. I want to thank my grandmother for being an inspiration and source of encouragement. She has always been a role model for me so that I can always stay strong even in times of adversity.

Chapter 1

Introduction

1.1 Overview of Power Network

Power networks, composed of the generators, loads and transmission lines, are typically large and complexly inter-connected systems. Power dispatched from generators flow to loads to satisfy their respective demand, according to laws of physics and satisfying the power flow limits of transmission lines. In the United States, the operations of the regional power systems are coordinated by Independent System Operators (ISOs) and Regional Transmission Operators(RTOs). They are authorized by Federal Energy Regulatory Commission (FERC) to oversee the operations of power systems and relevant markets. The tasks of ISOs and RTOs include dispatching generators to satisfy the demand, providing non-discriminatory access to transmission, managing transmission congestion, acquiring and supplying ancillary services, market monitoring, transmission planning and expansion etc. For example, the primary mission of California Independent System Operator is to "operate the grid reliably and efficiently, provide fair and open transmission access, promote environmental stewardship, and facilitate effective markets and promote infrastructure development" [23].

To ensure a reliable supply of electricity, the system operators first need to create the hourly demand forecast for the electricity. Together with the availability of power resources, reserve requirements and other constraints of the network , the system operators run a unit commitment model to dispatch the power plants to meet the demand at different nodes. In the two-settlement market structure, there are two markets: the day-ahead market and the the real time market. ISOs will accept the offers from generating units and bids from loads for each hour in the next day in the day-ahead market. Then a generation commitment schedule is produced by solving the security-constrained unit commitment(SCUC) model. With the fixed generator commitment schedule, a security-constrained economic dispatch(SCED) is run to compute the Locational Marginal Price(LMP).Then the system operators run a residual unit commitment to specify the additional power plants ready to produce electricity they will need for the next day. In order to respond to contingencies and fulfill the differences between the forecast demand and the actual demand, an operating reserve is needed in the real time. North American Electric Reliability Corporation (NERC) establishes and imposes N-1 reliability standard which states that a transmission network must be able to continue to supply load uninterrupted and run the grid in a satisfactory state when the network loses a single asset (circuit, transformer, generator etc). Besides the day-ahead market, the system operators also run a real-time market in which loads buy the additional power they need but was not satisfied in the day-ahead market. The optimization and control procedure of power system operation is illustrated in Fig.1.1[16].

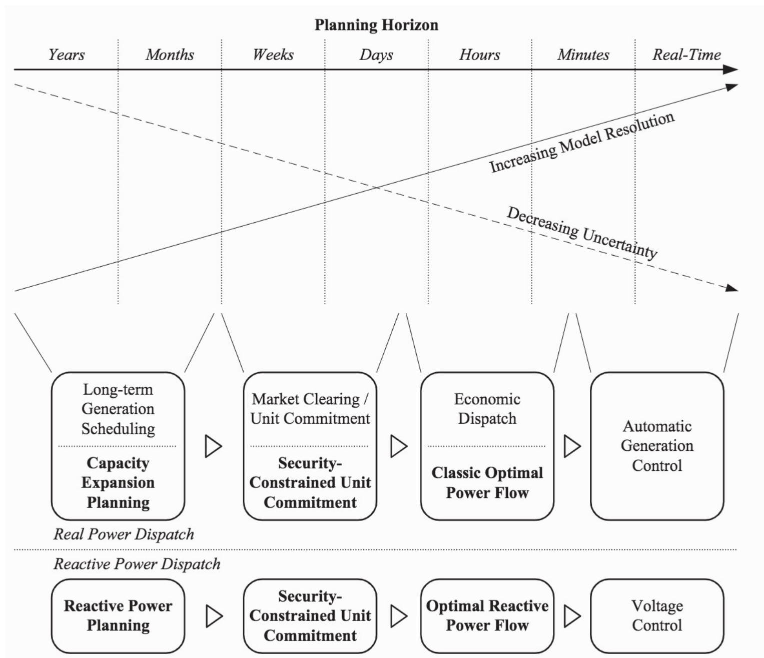

Figure 1.1: Optimization and Control Procedures for the Planning of Power System Operation

In order to efficiently run the grid and the markets, the system operators need to have a deep understanding of the network model, which examine and allocates the generation and transmission resources to find the least expensive dispatching schedule to fulfill demand. The power network is a highly interconnected system with many physical constraints for its components, including the production limits of the generators, the line ratings of the transmission lines and the limits for voltage angle differences between buses. To determine the best operating levels for the electric power plants, the researchers developed a model called Optimal Power Flow (OPF). Since its introduction 50 years ago [8], OPT has been in the spot light of power system research and innovation. It is also one of most well researched area with practical importance in constrained nonlinear optimization. The bold text in Fig.1.1 indicates processes that involves variant forms of optimal power flow. They stem from the power flow problem which seeks a solution of the following network equations.

$$P_k = \sum_{j=1}^{N} |V_k||V_j||(G_{kj}\cos(\theta_k - \theta_j) + B_{kj}\sin(\theta_k - \theta_j)) \tag{1.1}$$

$$Q_k = \sum_{j=1}^{N} |V_k||V_j||(G_{kj}\sin(\theta_k - \theta_j) - B_{kj}\cos(\theta_k - \theta_j)) \tag{1.2}$$

Where P_k and Q_k are the real and reactive power at bus k respectively

V_i is the voltage of bus i

G_{kj} and B_{kj} are the conductance and susceptance of the branch between bus k

and j respectively

θ_i is the angle at bus i

The above power flow equations, combined with an objective function such as cost minimization, forms an optimization problem. By incorporating the generator's availability, ramping rates, minimum up/down times and the reliability standards, the system operator can decide the on/off schedule by the SCUC. It also produces LMP - prices that reflects the cost of electricity at different locations. LMP is used to establish the wholesale prices at different locations, calculate transmission congestion charges and distribute compensation for holders of Financial Transmission Rights (FTR).

Optimal Power Flow problem, in its most accurate and rigorous form, is the Alternating Current Optimal Power Flow (ACOPF) problem. It's highly nonlinear and non-convex, therefore posing substantial computational challenge to the researchers and system operators. Each ISO deals with a power system with at least thousands of buses and they need to solve the problem every couple of minutes. Therefore no ISO solves the full scale ACOPF in the day ahead or real time market. Instead they rely on its direct current approximation (DCOPF), assuming all voltage magnitudes are fixed and all voltage angles are closed to zero. The system operators take full advantage of the apparent computational advantage of DCOPF, and this computational advantage is even more essential for unit commitment problems with its own binary variables to represent the on/off of the generators.

1.2 Transmission Switching

Power systems are built to be redundant so that it can work in the worst case scenario. This redundancy and the nature of the power network being a nonlinear and complex system make it possible to provide economic benefits to the system by switching transmission lines on/off, especially when there are congested lines. The average congestion cost of PJM from 2013 to

2017 is $1,252.32 million, with a spike of $2,231.2 million in 2014 [38]. California ISO reports that in 2017 the San Diego Gas and Electric area was the area that was the most affected by internal congestion within CAISO. Average day-ahead prices in this area increased by about $0.90/MWh (2.5 %) above the system average and real-time congestion increased prices by about $1.50/MWh (4 %) [23]. Therefore, there is a great opportunity for efficiency improvement through transmission switching. Energy Policy Act of 2005 promotes the use of state-of-the-art transmission technology and efficient transmission line configuration [51]. Even though the transmission switching can be done through a simple circuit breaker, the recent development of FACTS devices makes it even more reliable and realistic. They allow the system operators to control the line impedance so it can provide continuous and reliable control of power on the transmission line over a wide range.

The following simple 3-bus example can show the economic benefits that switching a line brings. There is a generator at each bus, with a unit cost of $100/MWh, $200/MWh, $50/MWh respectively. Three lines share the same impedance but different ratings. There is a demand of 250MW at node C. By solving the DCOPF model the original optimal outputs of each generator are: A: 180MW; B: 30MW; C: 40MW. The total cost is $20000. After switching off the line between bus A and B, the optimal outputs are: A: 200MW, B: 50MW, C: 0MW. The new total cost is $15000. After opening the line, Kirchhoff's law on this particular line is no longer enforced, allowing new solution space in which there is a better solution.

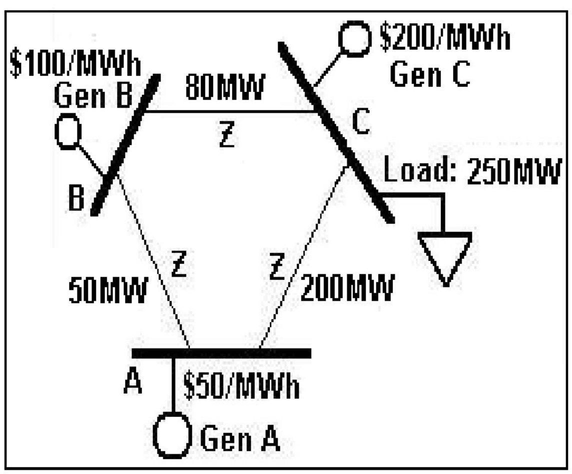

Figure 1.2: Original 3 Bus Network

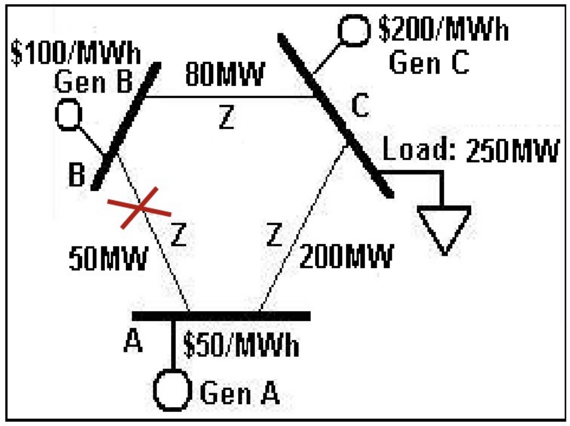

Figure 1.3: 3 Bus Network after Opening Line A-B

The existing transmission switching protocols are mostly dictated by system operators in an ad hoc manner. There are mainly four areas of application in the current industry practice [19].

Firstly, the system operators can switch some transmission lines out of service in order to improve voltage profiles. ISO New England has included transmission switching actions in the list of emergency system actions. When the documented studies have shown that a specific line switching action can relieve the existing contingency without losing the protection for another contingency, then the switching action can be implemented [24]. PJM also uses switching transmission facilities in/out of service for voltage control actions. Specifically, When high voltage conditions are expected in the PJM, the system operators use PJM Security Analysis programs to study possible actions i.e., opening an extra-high-voltage (EHV) line. Several circuits has been identified by PJM to be effective in controlling general high voltage conditions [37].

Secondly, several ISOs have listed transmission switching in their special protection schemes (SPS). Special protection scheme is a automatic protection system used to detect system contingencies and implement corrective actions accordingly. They facilitate the operational solutions of the contingency management which is much faster and less costly than building new transmission infrastructure. NERC has specified actions taken by SPS, which include system reconfiguration [48]. PJM has identified a list of potential transmission switching procedures that may assist to reduce or eliminate transmission congestion.

The transmission lines are due for maintenance from time to time due to the aging of the infrastructure and the system operators need to schedule the maintenance outages to make sure they are up to the current standards. Transmission line maintenance scheduling has traditionally been carried out with the focus on the reliability and the transmission system adequacy. However Independent System Operator of New England (ISONE) started to take dispatch efficiency into consideration and estimate a cost saving of $50 million a year [24]. The research on the economic transmission switching could provide insight and benefit to this new objective of ISOs for their line maintenance scheduling.

When the load requirements and chance of contingency change with season, it is sensible for the system operators to switch redundant transmission lines as the situation deems fit. For example in California, the demand requirements are low in winter while the chance for outages are high due to winter storms. The operators will choose to keep the redundant lines in service to maintain the system reliability. However, when summer comes with the high demand and low chances of contingencies, the system operators will find it beneficial to switch some of the redundant lines out of service, to avoid overloading concerns.

1.3 Motivation

The system operators regularly use transmission switching as a corrective mechanism but ignore its application for economic efficiency. When they run the energy market and dispatch generators the transmission elements are viewed as static components of the network. However the recent development of FACTS devices make it possible and reliable for the system operators to open or close the circuit. Furthermore, increasing common social interventions such as Not in My Backyard Phenomenon have caused problems for the transmission planning. It has been increasingly difficult to site the energy infrastructures, including transmission lines [52]. Therefore, researchers should pay more attention to the optimal use of the existing transmission system. And since the planning and construction of new transmission assets could take years, we have to rely on the existing infrastructure to serve the consumers and run the grid efficiently and reliably.

1.4 Outline

In Chapter 2, we provide a thorough literature review of transmission switching, for both corrective and economic purpose. We discuss the Optimal Transmission Switching problem and its economic benefits and computational challenge.

In Chapter 3, the applications of high performance computing in the power system are reviewed.

In Chapter 4, we first present a greedy algorithm framework to tackle the computational challenge of the OTS. Within this framework we only need to solve linear programs, not mixed integer programs. Then we discuss three heuristics that can be implemented in parallel. We present the cost reduction and computational performance of these heuristics on two networks. One is the IEEE 118-Bus network and the other is the FERC 13867-bus Network derived from the data set we obtained from PJM Regional Transmission Organization.

In Chapter 5, a machine learning based method is presented to create a priority list for the line switching candidates. The application of machine learning in the power system is first reviewed. Then we present three machine learning methods that are popular in the power system applications and suitable for line selection problem. We show the results of the line selection with machine learning on IEEE network and FERC network for the three machine learning methods. The advantages and drawbacks of different methods are then discussed.

In Chapter 6 we discuss the application of algorithm selection in transmission switching. Among the fast heuristics we developed and those in literature, the algorithm selection aims to choose the algorithm so as to maximize the economic benefit at each iteration based on the current network status. The framework and past literature of the algorithm selection are reviewed. The results of algorithm selection on IEEE network and FERC network are presented and different machine learning methods used for algorithm selection are compared.

The conclusions and the future directions are summarized in Chapter 7.

Chapter 2

Literature Review

2.1 Notation

This section lists the notations that are commonly used in this and the following chapters.

Sets and Indices:

n Bus.

N Set of all buses.

g Generator.

G_n Set of generators at bus n.

k Transmission line.

K Set of all lines.

$\hat{K}$ Set of transmission lines in service.

$\bar{K}$ Set of transmission lines out of service.

K_n^{to} Subset of lines with n as "to" bus.

K_n^{fr} Subset of lines with n as "from" bus.

n_k^{to} "To" bus for linke k.

n_k^{fr} "From" bus for linke k.

Parameters:

B_k Susceptance of line k.

c_g Unit cost of power from generator g.

P_g^M Maximum power from generator g.

P_g^m Minimum power from generator g.

P_k^M Maximum power flow on line k.

P_k^m Minimum power flow on line k.

θ^M Maximum voltage angle difference.

θ^m Minimum voltage angle difference.

P_n^{dem} Demand load at bus n.

Decision Variables:

P_g Power from generator g.

P_g Power flow on line k.

θ_n Voltage angle at bus n.

z_k Binary variable indicating the status of the line. 0 (out of service) /1 (in service).

s_k Binary decision variable to switch line k. 0 (no switch) /1 (switch).

2.2 Transmission Switching

Past literature has explored the use of transmission switching in the power network for mainly two purposes: as a corrective mechanism or for operational efficiency. Since the system operators have constantly been using the transmission switching as an ad hoc tool to react to line overloading, voltage violations, etc, it's only natural to formulate transmission switching as a formal treatment in response to contingencies. Compared to the ad hoc approach that only aims to alleviate the contingency situation, the formal formulation of transmission switching for corrective purpose can find the best action while considering other conditions such as N-1-1 reliability.

Since line switching was proposed as a control tool in 1980 [28] [53], several algorithms have been proposed to investigate the effect of line switching in the power systems. Glavitsch [18] formulated the transmission switching in a systematic way to change the load flow distribution, improving the voltage profile and enhancing the network security. He also presented a search approach using current injection to find the optimal solution. Mazi *et al* [34] presented a fast algorithm for rank the lins switching actions to relieve the overloads caused by losing a line. Instead of running a full ACOPF they used line switching to estimate the effect of lowering the power flow on a congested line by using line outage distribution factors based on DC power flow. Bacher *et al* [2] modeled transmission switching as a current injection scheme and determined the effect of such an injection on all other elements of the network by the distribution factor matrix. A linearized model was presented for corrective switching and the solution with continuous control variables and discrete control variables were discussed by Bakirtzis *et al* in [4]. The authors in [33] described a fast technique modified from existing contingency analysis program to choose line switching actions when there is a line overloading. Instead of evaluating the magnitude of the change in line flows from the overloading condition they only calculated the algebraic sign of the flow change, which saved substantial computing time.Furthermore Schnyder *et al* [43] proposed a one step method to optimize the corrective switching and maintain a N-1 secure system. They also differentiated the preventive corrective switching and post contingency corrective response. Wrubel *et al* [58] presented one of the first practical use of the corrective switching algorithm. It was implemented as part of the security analysis program at Public Service Electric and Gas Company in New Jersey. Based on the previous research efforts, Rolim *et al* [41] categorized publications on corrective switching based on several criteria, including the objective, switchable elements, reduction of search space and solution technique etc.

Transmission switching as a corrective mechanism has seen rapid development in the last decade. Shao *et al* in [45] [44] proposed an algorithm to find the best switching action for relieving line flow and voltage contingencies with fast decoupled power flow and sparse inverse technique. Korad *et al* [30] proposed a real time robust corrective switching scheme which provides several viable line switching candidates with security assessment tool. Balasubramanian *et al* [5] presented a corrective switching algorithm based on sensitivity factors to reduce the load shedding and tested it for reliability scenarios including N-1, N-m, and cascading events. In [25], the authors discussed the the use of corrective transmission switching to improve the efficiency in delivering reserve in the power network. A day-ahead SCUC model was solved first then the contingency analysis tool coupled with the topology control determined the reserves and switching actions while satisfying all post-contingency limits. Furthermore in the light of increasing renewable penetration, Korad *et al* [29] proposed a method to determine the the optimal allocation of reserves with renewable penetration and corrective transmission switching in the day-ahead time frame. Later they used a zonal approach which reduced the network model into a few interconnected zones to cut down the computational time [31]. The authors confirmed that a proper chosen transmission switching solution would be beneficial post-contingency without jeopardizing the reliability of the network compared to the case without transmission switching. They further showed that the use of corrective switching actions at N-1 stage will help obtain an N-1-1 reliable solution by reducing not only N-1 but also N-1-1 violations.

It is well known that the redundancy of the power network can cause operational inefficiency, as the 3 bus example illustrated in Chapter 1. Therefore switching out a line temporarily can enhance the economic efficiency of the network when the network remains reliable at the same time. The first formal treatment of co-optimization of generator dispatch and network topology based on Direct Current Optimal Power Flow (DCOPF) was proposed by Fisher *et al* [14]. The proposed Optimal Transmission Switching (OTS) is a mixed integer program. It results in more than 25% cost savings on IEEE 118-Bus test case, but takes a very long running time for a good solution, which prohibits it from practical application. Based on this model Hedman *et al* [21] discussed the uncertainty that the transmission switching may bring to the market participants, including nodal prices, load payment, generator revenues etc. Then they further incorporated the contingency analysis to ensure N-1 reliability standards are met [22]. The authors in [20] presented a co-optimization of unit commitment and network topology while ensuring N-1 reliability. They showed that co-optimizing the line configuration can change the optimal unit commitment schedule. In [46] the authors modeled transmission switching as a recourse action in the day-ahead unit commitment with large-scale renewable generation. They further demonstrated that the dispatch cost can be reduced by the transmission switching with or without congestion in the system.

In the next section we are going to present the mathematical model of Optimal Transmission Switching formulation, which we base our work on.

2.3 Optimal Transmission Switching

The Optimal Transmission Switching model assumes a network where all the lines are currently closed. The objective is to find the network topology and generation schedule to minimize the dispatch cost. It is built on the DCOPF model which is a linearized form of ACOPF with three simplification assumptions: voltage magnitudes are constant; voltage angles are close to zero and lossless system. The voltage angle constraint is imposed by Eq.(2.1a). Eq.(2.1b) imposes the generator output limits. Eq.(2.1c) ensures the power balance at each node. Eq.(2.1d) enforces the thermal ratings of transmission lines. Kirchhoff's law is ensured by Eq.(2.1e).

DCOPF:

$$\min_{\theta_n, P_g, P_k} \quad \sum_{n \in N} \sum_{g \in G_n} c_g P_g$$

$$\text{s.t.} \quad \theta_n^m \leq \theta \leq \theta_n^M, k \in K \tag{2.1a}$$

$$P_g^m \leq P_g \leq P_g^M, n \in N \tag{2.1b}$$

$$\sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N \tag{2.1c}$$

$$P_k \leq P_k \leq P_k^M, k \in K \tag{2.1d}$$

$$-P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) = 0, k \in K \tag{2.1e}$$

In the following OTS model, a binary variable z_k is used to represent the status of the line. When $z_k = 0$ the line is switched to be open and when $z_k = 1$ the line remains closed.

Note here M is a number greater than $B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}})$. It ensures that Kirchhoff's law holds when the line is closed and vanishes when the line is open. The use of M and z_k in Kirchhoff's law are necessary because otherwise if we open a single line, the power flow on all its parallel lines will be zero since the angle difference is forced to zero.

$$
\begin{aligned}
&\text{OTS:} && \\
&\min_{\theta_n, P_g, P_k} && \sum_{n\in N}\sum_{g\in G_n} c_g P_g \\
&\text{s.t.} && \theta_n^m \le \theta \le \theta_n^M, k \in K && \text{(2.2a)}\\
& && P_g^m \le P_g \le P_g^M, n \in N && \text{(2.2b)}\\
& && \sum_{k\in K_n^{to}} P_k - \sum_{k\in K_n^{fr}} P_k + \sum_{g\in G_n} P_g = P_n^{dem}, n \in N && \text{(2.2c)}\\
& && P_k^m z_k \le P_k \le P_k^M z_k, k \in K && \text{(2.2d)}\\
& && -P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) + (1-z_k)M \ge 0, k \in K && \text{(2.2e)}\\
& && -P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) - (1-z_k)M \le 0, k \in K && \text{(2.2f)}\\
& && \sum_k (1-z_k) \le j. && \text{(2.2g)}\\
& && z_k \text{ is binary}, k \in K. && \text{(2.2h)}
\end{aligned}
$$

Fisher *et al* [14] tested this model on IEEE 118-Bus network without the constraint (2.2g) and the dispatch cost is reduced by 25% when 38 lines were opened, which is the best know optimal solution. In terms of computation time, it grows exponentially with the increase of j and large values of j ($j \ge 13$ in this case) are impractical for CPLEX software to solve on a laptop. IEEE 118-Bus network with 10 lines switches from OTS with $j = 10$ is illustrated in Fig. 2.1.

2.4 Algorithms to Solve OTS

Several heuristics have been investigated to tackle the computational challenge caused by the inherent curse of dimensionality of the MIP formulation. There are two mainstream heuristics, corresponding to two methods for modeling opening a line. The first approach is to change the susceptance matrix to model the line removal, or in other words, make the partiticular susceptance zero. By this approach researchers consider possible line switches one at a time and solve a series of DCOPF problems, which could also be time consuming if considering all the possible line switches. The authors in [36] showed that high performance computing can be used to parallelize the computation and improve performance. A priority list can be established by using a sensitivity factor developed from the dual problem to find the desirable line switches faster [17].

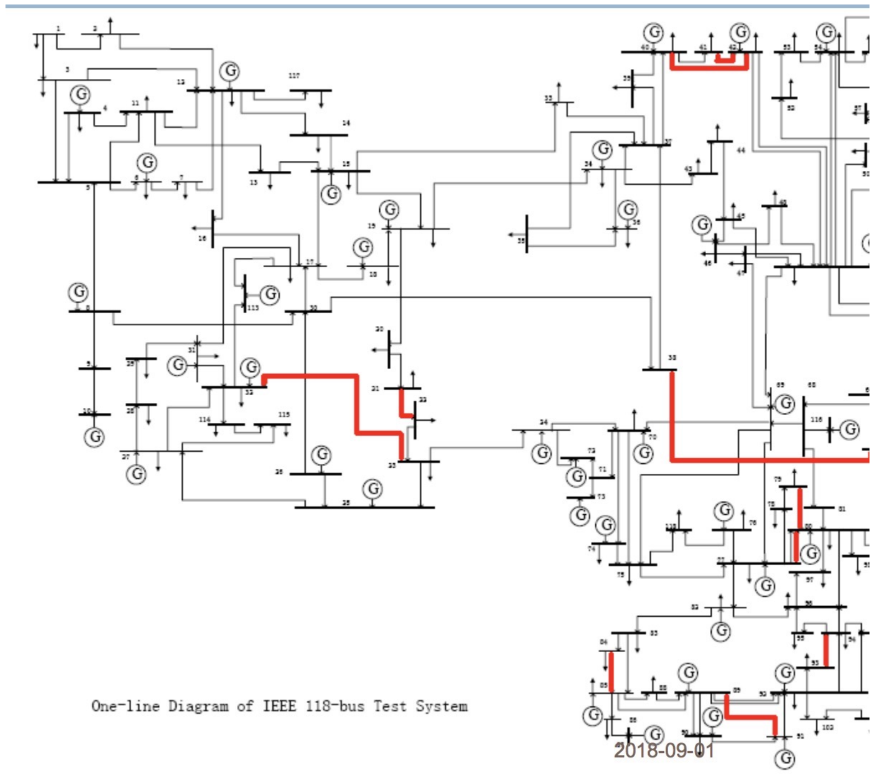

Figure 2.1: IEEE 118-Bus Network with 10 Line Opened from OTS

Alternatively, authors in [42] use the power transfer distribution factors (PTDFs) and flow canceling transactions, which is more scalable with the number of transmission lines compared to the $B\theta$ model. Instead of removing the line from the topology, this method maintains the original topology but computes a power transfer transaction that will cancel the current flow on the particular line. It will have the same effect as an actual line switch on the other parts of the power system.

NERC defines a PTDF as "in the pre-contingency configuration of a system under study, a measure of the responsiveness or change in electrical loadings on transmission system facilities due to a change in electric power transfer from one area to another, expressed in percent (up to 100%) of the change in power transfer". Simply put, PTDF measures the sensitivity of the flow on one line with respect to a power flow transfer on another line. It shows the linearized impact of a transfer of power and is independent of the size of the transfer.

Without modifying the susceptance, a line switch can be modeled by a flow cancelling transaction. To model the removal of line k between nodes m and n, let m' and n' be infinitely close to the actual nodes, as shown in Fig.2.2. f_k is the current flow on the line. Assume there is a virtual flow injection from m' to n' with magnitude ν_k. This virtual injection's effect is equivalent to the switching off line k as far as the impact on the rest of the system is considered. Assume the PTDF between node m and n respect to line l is ϕ_l^{mn}. By the definition of PTDF, the following equation must hold.

$$f_k - (1 - \phi_l^{m'n'})\nu_k = 0 \tag{2.3}$$

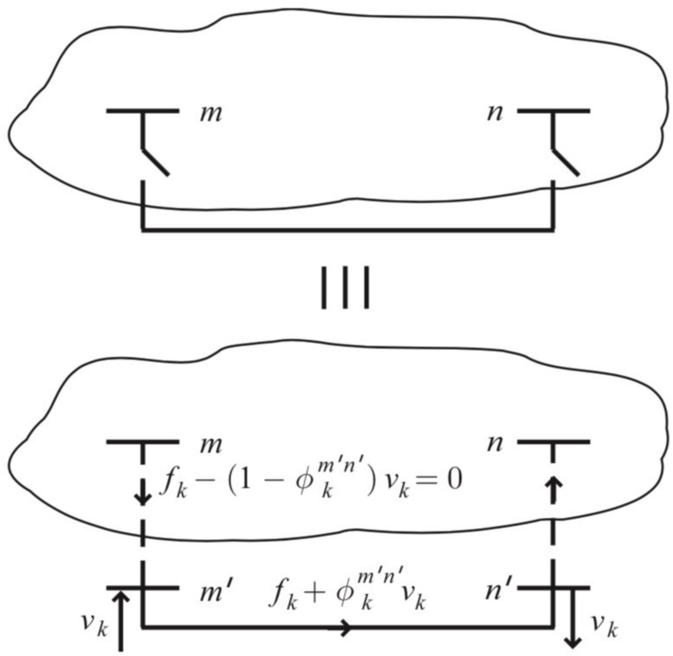

Figure 2.2: Flow Cancelling Transactions

The resulting formulation of OPF with transmission switching is:

$$\begin{aligned}
\min_{\boldsymbol{p},\boldsymbol{\nu},\boldsymbol{z}} \quad & \boldsymbol{c^T p} \\
\text{s.t.} \quad & \boldsymbol{1^T(p-1) = 0} && \text{(2.4a)} \\
& \underline{\boldsymbol{p}} \le \boldsymbol{p} \le \overline{\boldsymbol{p}} && \text{(2.4b)} \\
& \underline{\boldsymbol{f}}^{\boldsymbol{M}} \le \boldsymbol{\Psi^M(p-1) + \Psi^{MS}(p-1)\nu} \le \overline{\boldsymbol{f}} && \text{(2.4c)} \\
& \underline{\boldsymbol{F}}^{\boldsymbol{S}}\boldsymbol{z} \le \boldsymbol{\Psi^S(p-1) + (\Psi^{SS} - D)\nu} \le \overline{\boldsymbol{F}}^{\boldsymbol{S}}\boldsymbol{z} && \text{(2.4d)} \\
& -M(\boldsymbol{1-z}) \le \boldsymbol{\nu} \le M(\boldsymbol{1-z}) && \text{(2.4e)} \\
& z_k \text{ is binary} && \text{(2.4f)}
\end{aligned}$$

Where $\boldsymbol{p}$ is the vector of generation
$\boldsymbol{c}$ is the vector of generation cost
$\boldsymbol{z}$ is the vector representing the status of the transmission lines

$\underline{\boldsymbol{f}}$, $\overline{\boldsymbol{f}}$ are the vectors of thermal limits
$\underline{\boldsymbol{F}}$, $\overline{\boldsymbol{F}}$ are the diagonal matrices of thermal limits
$\boldsymbol{\Psi}^{\boldsymbol{M}}$ is the shift factor matrix associated to monitored lines
$\boldsymbol{\Psi}^{\boldsymbol{S}}$ is the shift factor matrix associated to switchable lines
$\boldsymbol{\Psi}^{\boldsymbol{SS}}$ is the PTDF matrix of switchable lines
$\boldsymbol{\Psi}^{\boldsymbol{MS}}$ is the PTDF matrix of monitored lines
$\boldsymbol{D}$ is the vector of demand

Chapter 3

High Performance Computing

3.1 Introduction

High performance computing is the concurrent use of multiple computing cores to solve a computational problem. It is especially useful when a problem is broken into discrete and independent parts that are solved simultaneously on different cores. An high level control mechanism is employed to coordinate the execution of individual parts, as illustrated in Fig.3.1. Nowadays the high performance computing clusters, including Lawrence Livermore National Laboratory (LLNL) parallel computer clusters we use, are parallel from a hardware perspective which means they have multiple functional units and multiple execution cores. The main reason we are especially interested in supercomputers is that it allows us to solve larger and more complex problems. It also saves the computation time if the algorithms can be implemented in parallel.

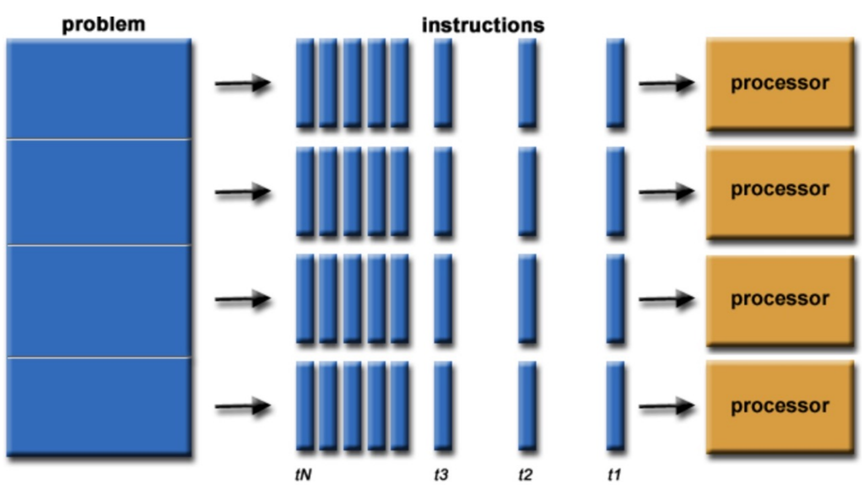

Figure 3.1: Framework for the Parallel Computing

In order to explicitly exploit the advantage of multiple cores of LLNL clusters, a distributed message passing model is required, which has the following characteristics.

1. The independent tasks only require their own local memory for its own execution. Therefore multiple tasks can be performed on the same machine and/or across different machines.

2. Since tasks reside on different cores, when the data exchange is needed , the communication is done by sending and receiving messages.

3. The communication consists of two way cooperative commands between the processes, for example, "send" and "receive" commands in Fig.3.2.

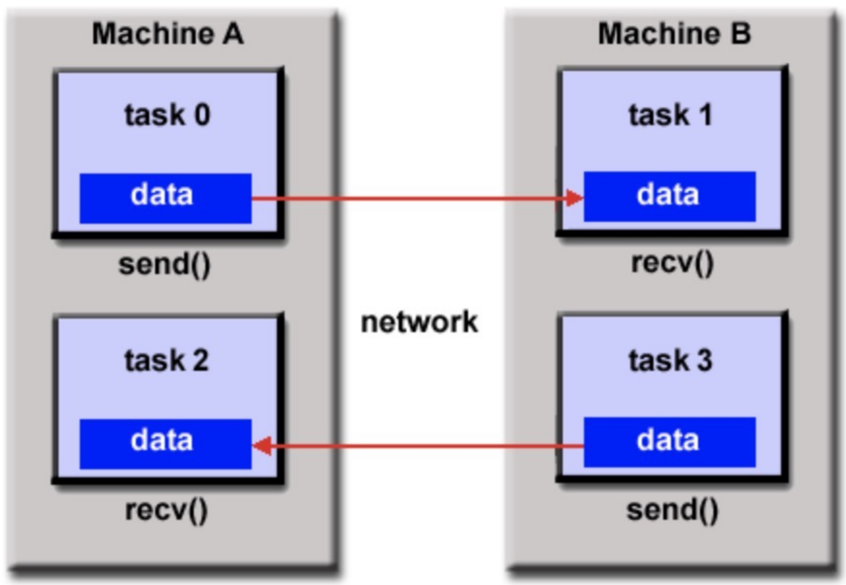

Figure 3.2: Communication between Tasks in Parallel Computing

We use Message Passing Interface (MPI), the standard message passing interface implementation for the parallelization of the algorithms in this thesis. It uses a central task manager called *MPI_COMM_WORLD* to communicate between master thread and worker threads. The master thread allocates the tasks to the worker threads and each task can do its work without requiring any information from the other tasks. After a worker finishes the task it sends the results back to the master. The master thread coordinates the results from parallel workers and determines when to stop the algorithm.

3.2 Application of High Performance Computing in Power Systems

The power system has been evolving since it began over 120 years ago with the Pear Street Station in 1885 and it is likely to see more changes in the next decade than the past century.

Distributed energy generation and storage, renewable resources, smart devices, demand response and wide spread of electric vehicles are also being employed in the power grid rapidly. The models that are used in power systems are becoming increasingly complicated and computational expensive, and system operators have more and more access to increasingly powerful computing hardware and software. The drastic changes in the size and the complexity of the power system models call for fundamental change in the tools and methods used for the operation of the power grid. First we deep to develop capabilities to accommodate the increasing complexity of smart grid data, which are generated by the power system components as well as via simulations to ensure secure and reliable power grids. The new challenge arises when the system operators need new and computational efficient software programs to accurately solve the market models. We would also need advanced research on development and implementation of algorithms for solving real-time and dynamic problems. In brief the resulting new dynamic and stochastic behaviors from power system renovations will demonstrate more complexity in models, while it is challenging to solve those models in the traditional sequential computing environment . Therefore high performance computing is promising to enhance the computational performance and though it has a rich history in power system research, it has not been fully investigated or adopted by the system operators

Monticelli *et al.* [35] described a method to solve the security-constrained economic dispatch problem with Benders decomposition. The method iterated between solution of a base case economic dispatch and individual contingency analysis with generation rescheduling. The authors in [49] discussed the implementation of concurrent programming environment in three power system applications: multi-area reliability, system analysis model and security-constrained dispatch with post-contingency corrective rescheduling. Falcao [13] presented a summary of research in the employment of distributed computing to the power system problems, especially the computing intensive ones in power system optimization and control. Kim *et al.* [26] presented an approach to implement optimal power flow in parallel by decomposing the network model into zones by duplicating the variables on the border and enforcing coupling constraints between the duplicated variables. They tested the proposed method on systems including IEEE test networks and parts of the ERCOT system, and showed potential for very large interconnected power system. Compared to OPF decoupling around border nodes in [26], Bakirtzis *et al.* [3] presented an OPF decomposition method at the tie-lines connecting the adjoining zones. They later implemented the algorithm on a network of computers using PVM software [6]. Green *et al.* discussed the development and change of power network and high performance computing in the last 15 years and explored the ways in which high performance computing will be used in the smart grid.

Chapter 4

Greedy Algorithms for Transmission Switching

4.1 Introduction

In the Chapter 2 we have discussed the computational challenge faced by Optimal Transmission Switching (OTS) problem. By the introduction of the binary variable representing the status of the line, it becomes a mixed integer program and takes more than half an hour to solve with CPLEX 11 on a four core commercial laptop for the IEEE 118-Bus network. In this network there are 186 lines considered for switching, therefore 2^{186} potential transmission network topologies. Even though it has a large solution space, it is still far too small compared to the real power networks. Normally thousands of lines will be monitored in a real power network, which makes OTS impossible to directly solve with a commercial software. Even with the aid of super computers the performance is less than satisfactory with distributed branch and bound, as we are going to show in this chapter. Therefore we develop several heuristics to achieve the goal of improving economic efficiency within a reasonable timeframe.

In this chapter we first introduce the framework for the greedy algorithms and then present three heuristics under the framework. Then we present the numerical results for IEEE 118-Bus network and FERC 13867-Bus network.

4.2 Fast Heuristics for Transmission Switching

The central idea of the fast heuristics we develop comes from the realization that only two DCOPFs need to be solved in order to compute the cost savings for a line switch. The cost saving is equal to the difference of the objective functions of the DCOPF of the original power network and the DCOPF of the network after a line switch. Therefore we develop a greedy algorithm framework in which only one line is switched at one iteration. The cost saving is then evaluated. If the cost improvement is not satisfactory we can evaluate other possible line switches before time is exhausted. Otherwise the procedure is terminated and the best line switching action is selected. This greedy algorithm gives the operator more control over the computation time. DCOPF, as shown below, is a linear program therefore quick to solve. The flowchart of the general heuristic procedure is shown in Fig.4.1.

DCOPF:

$$\min_{\theta_n, P_g, P_k} \quad \sum_{n \in N} \sum_{g \in G_n} c_g P_g$$

s.t.

$$\theta_n^m \leq \theta \leq \theta_n^M, n \in N \tag{4.1a}$$

$$P_g^m \leq P_g \leq P_g^M, g \in G_n \tag{4.1b}$$

$$\sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N \tag{4.1c}$$

$$P_k \leq P_k \leq P_k^M, k \in K \tag{4.1d}$$

$$-P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) = 0, k \in K \tag{4.1e}$$

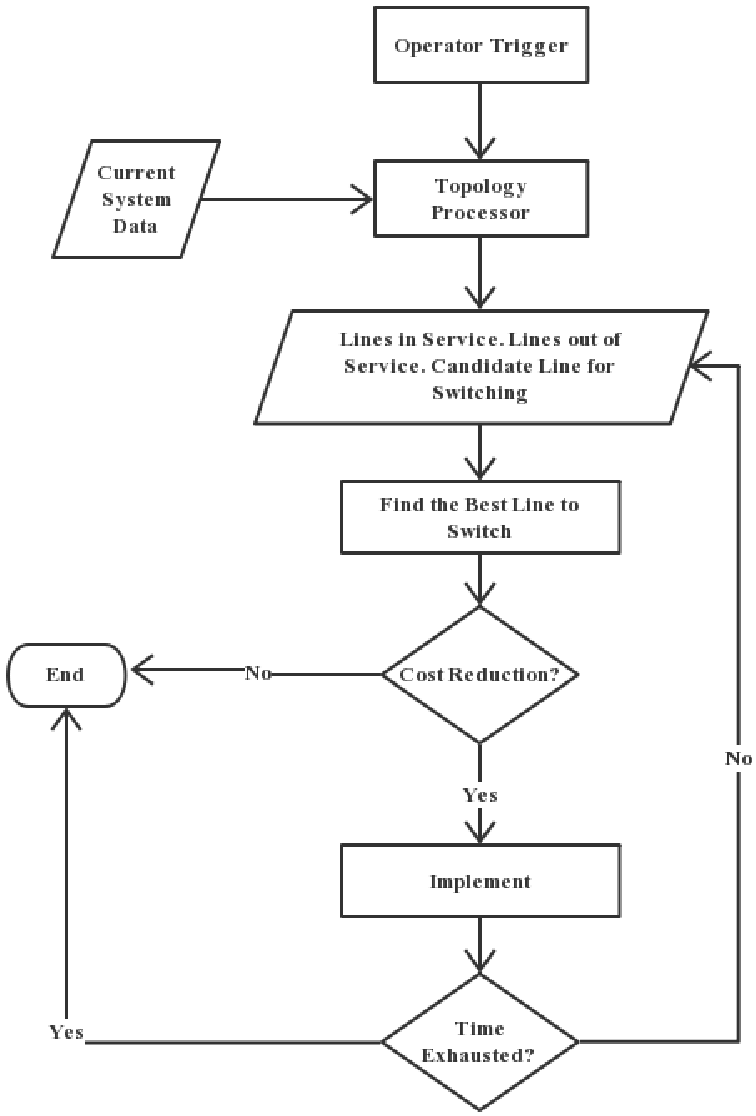

Figure 4.1: General Framework of Fast Heuristics for Transmission Switching

Line Enumeration

The first heuristic we develop is to directly enumerate all the possible line switches. For each switch a DCOPF is solved and compared with the original cost. We choose the line to switch which results in greatest cost reduction. The algorithm is iterated until the lines are exhausted. We notice that all the line switches are independent and their corresponding DCOPFs can be solved in parallel. The algorithm is shown in Fig.4.2.

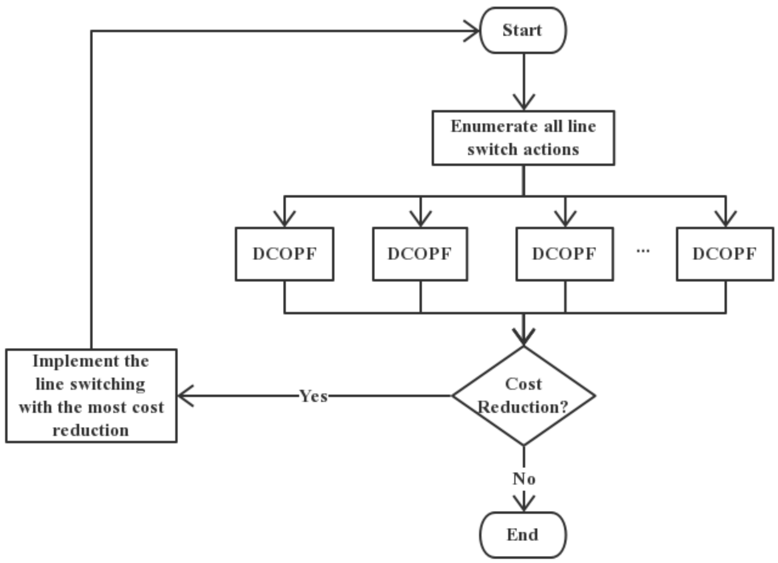

Figure 4.2: Line Enumeration

Line Selection with Priority List

Based on the Optimal Transmission Switching model in [14], we present the following single period economic dispatch model based on DCOPF. It is a mixed integer program where z_k represents the switching decision. $z_k = 1$ means the line is on and $z_k = 0$ means the line is off. The lines $K = \hat{K} \cup \bar{K}$ are divided into two sets, according to their current status: $\hat{K}$ representing the set of lines in service and $\bar{K}$ representing the set of lines out of service. The objective is to minimize the generation cost. Voltage angle limits are imposed by Eq.(4.2a) and the capacity limits on generating unites are imposed by Eq.(4.2b). Eq.(4.2c) ensures the power balance for each bus. For lines originally in service, Eq.(4.2d) makes sure the flow respects the line flow limits if it stays on and the flow is zero if it is to be switched off. M in Eqs.(4.2e) and Eqs.(4.2f) is a very large number that makes sure Kirchhoff's law holds when the line stays on. When the line is out of service at the beginning, the flow on it must be zero, as Eqs.(4.2g) and Eqs.(4.2h) state.

$$
\begin{aligned}
\min_{\theta_n, P_g, P_k} \quad & \sum_{n \in N} \sum_{g \in G_n} c_g P_g \\
\text{s.t.} \quad & \theta_n^m \le \theta \le \theta_n^M, n \in N && \text{(4.2a)} \\
& P_g^m \le P_g \le P_g^M, g \in G_n && \text{(4.2b)} \\
& \sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N && \text{(4.2c)} \\
& P_k^m z_k \le P_k \le P_k^M z_k, k \in \hat{K} && \text{(4.2d)} \\
& - P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) + (1 - z_k)M \ge 0, k \in \hat{K} && \text{(4.2e)} \\
& - P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) - (1 - z_k)M \le 0, k \in \hat{K} && \text{(4.2f)} \\
& 0 \le P_k \le 0, k \in \bar{K} && \text{(4.2g)} \\
& P_k = 0, k \in \bar{K} && \text{(4.2h)} \\
& z_k \in \{0, 1\} && \text{(4.2i)}
\end{aligned}
$$

The above model, being a mixed integer program, faces practical computational challenges. Even for the IEEE 118-Bus test case it takes more than half an hour to solve within 9e-6 optimality gap on a four processor laptop. It prompts a greedy approach which only considers one line switch at a time. The following modified DCOPF, a linear program, can be solved fast and it will give the new cost after the line switch if we move the line in consideration from set $\hat{K}$ to $\bar{K}$.

$$
\begin{aligned}
\min_{\theta_n, P_g, P_k} \quad & \sum_{n \in N} \sum_{g \in G_n} c_g P_g \\
\text{s.t.} \quad & \theta_n^m \le \theta \le \theta_n^M, n \in N && \text{(4.3a)} \\
& P_g^m \le P_g \le P_g^M, g \in G_n && \text{(4.3b)} \\
& \sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N && \text{(4.3c)} \\
& P_k^m \le P_k \le P_k^M, k \in \hat{K} && \text{(4.3d)} \\
& P_k = B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}), k \in \hat{K} && \text{(4.3e)} \\
& 0 \le P_k \le 0, k \in \bar{K} && \text{(4.3f)} \\
& P_k = 0, k \in \bar{K} && \text{(4.3g)}
\end{aligned}
$$

Following the dual criterion that Fuller [17] proposed, we would like to calculate the sensitivity of the optimal cost respective to the line switching action. In order to do that we have to express the line status explicitly as a constraint. Therefore we propose the following nonlinear program which contains a separate constraint representing the switching action.

$$
\begin{aligned}
\min_{\theta_n, P_g, P_k} \quad & \sum_{n \in N} \sum_{g \in G_n} c_g P_g \\
\text{s.t.} \quad & \text{(4.3a), (4.3b) and} \\
& \sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N, [\rho_n] && \text{(4.4a)} \\
& P_k^m (1 - s_k) \le P_k \le P_k^M (1 - s_k), k \in \hat{K}, [\lambda_k^-, \lambda_k^+] && \text{(4.4b)} \\
& P_k^m s_k \le P_k \le P_k^M s_k, k \in \bar{K}, [\lambda_k^-, \lambda_k^+] && \text{(4.4c)} \\
& P_k = B_k (1 - s_k)(\theta_{n_k^{fr}} - \theta_{n_k^{to}}), k \in \hat{K}, [\psi_k] && \text{(4.4d)} \\
& P_k = B_k s_k (\theta_{n_k^{fr}} - \theta_{n_k^{to}}), k \in \bar{K}, [\psi_k] && \text{(4.4e)} \\
& s_k = 0, k \in K, [\gamma_k] && \text{(4.4f)}
\end{aligned}
$$

The above model (4.4) is mathematically equivalent to model (4.3), but more complex by having a binary variable s_k which denotes the switching decision. For a line in service ($k \in \hat{K}$), $s = 1$ means it switches the line off and $s = 0$ means the line stays on, and vice versa. The dual variable γ measures the sensitivity of the objective function with an infinitesimal increase on the right hand side of Eq.(4.4f). Therefore it can be used as an indicator of possible cost reduction resulting from a line switch. A priority list can be produced by ranking the lines by ordering their respective γ from smallest to the largest (more negative γ indicates larger possible decrease in objective function, i.e. higher cost reduction). From KKT conditions we can derive the following fomula:

$$
\gamma_k = P_k^M \lambda_k^+ + P_k^m \lambda_k^- + B_k (\theta_{n_k^{fr}} - \theta_{n_k^{to}}) \psi_k, k \in \bar{K} \tag{4.5a}
$$

$$
\gamma_k = P_k (\rho_{n_k^{fr}} - \rho_{n_k^{to}}) \psi_k, k \in \hat{K} \tag{4.5b}
$$

We can see that all the variables on the right hand side are either parameters of the problem or the optimal primal/dual variables from model (4.3). Calculating γ doesn't necessarily require solving model (4.4), a nonlinear program. It can be obtained by just solving a linear program: model (4.3). This drastically reduces the computation time and makes producing a priority list operationally feasible.

Having computed the sensitivity factors in Eqs.(4.5a)&(4.5b), we can rank the lines according to them. In this algorithm we run DCOPF with a single switched line according to the priority list. We first evaluate the top k lines in the priority list. If a cost reduction is found, we implement the line switch with the most cost reduction and stop. Otherwise we move on to next k lines until the list is exhausted. The value of k is chosen based on the number of cores available. The algorithm is shown in Fig.4.3.

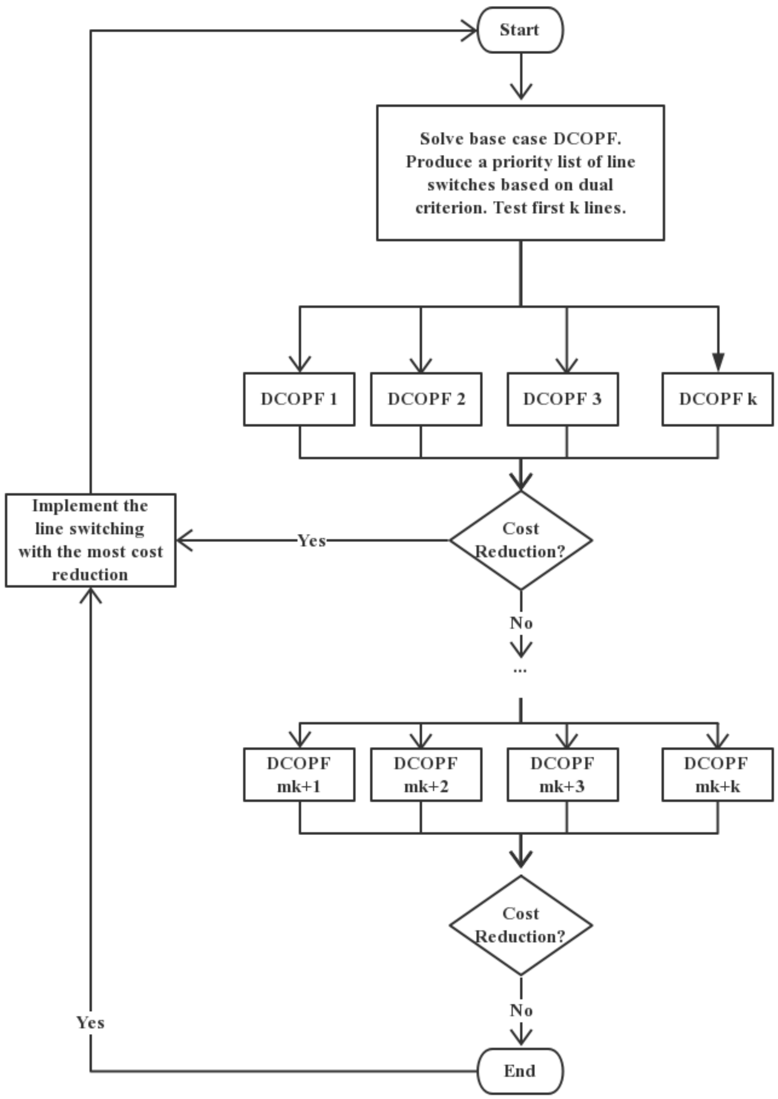

Figure 4.3: Line Selection with Priority List

Divided MIP

The Divided MIP divides the network into mutually exclusive and collectively exhaustive zones, each of which contains a group of candidate switching lines. Then a OTS with limited line switches is solved for each group in parallel. Assume the group of lines in consideration now is represented by K^*. Here we only consider switching out lines that are currently in service. Therefore $K^* \subset \hat{K}$. Eqs.(4.6d), (4.6e) and (4.6f) ensure the power flow on a line respects the thermal limits and Kirchhoff's Law holds for the lines in consideration. For the closed lines that are out of consideration, the normal thermal limit and Kirchhoff's Law constraints are enforced without binary variables, as in Eqs.(4.6g) and (4.6h).

$$\min_{\theta_n, P_g, P_k} \quad \sum_{n \in N} \sum_{g \in G_n} c_g P_g$$

$$\text{s.t.} \quad \theta_n^m \leq \theta \leq \theta_n^M, k \in K \tag{4.6a}$$

$$P_g^m \leq P_g \leq P_g^M, n \in N \tag{4.6b}$$

$$\sum_{k \in K_n^{to}} P_k - \sum_{k \in K_n^{fr}} P_k + \sum_{g \in G_n} P_g = P_n^{dem}, n \in N \tag{4.6c}$$

$$P_k^m z_k \leq P_k \leq P_k^M z_k, k \in K^* \tag{4.6d}$$

$$-P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) + (1 - z_k)M \geq 0, k \in K^* \tag{4.6e}$$

$$-P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) - (1 - z_k)M \leq 0, k \in K^* \tag{4.6f}$$

$$P_k^m \leq P_k \leq P_k^M, k \in \hat{K} \setminus K^* \tag{4.6g}$$

$$-P_k + B_k(\theta_{n_k^{fr}} - \theta_{n_k^{to}}) = 0, k \in \hat{K} \setminus K^* \tag{4.6h}$$

$$0 \leq P_k \leq 0, k \in \bar{K} \tag{4.6i}$$

$$P_k = 0, k \in \bar{K} \tag{4.6j}$$

$$z_k \in \{0, 1\}, k \in K^* \tag{4.6k}$$

Note that here we don't solve OTS for a smaller network. We solve a full power flow network model, just with a smaller set of binary variables. The program above will either produce a set of switching actions that will result in a cost improvement, or find no line switching action will further reduce the cost. The group of switching actions with the most cost reductions will then be implemented. The procedure will terminate if no cost saving switching action is found. Fig.4.4 shows the algorithm framework.

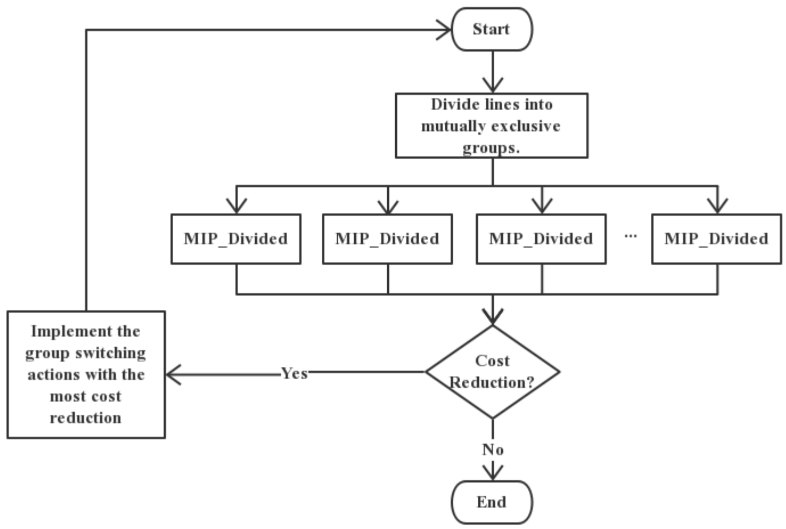

Figure 4.4: Divided MIP

4.3 Case Study Networks

IEEE 118-Bus Network

The first network we use is a the IEEE 118-Bus system with 118 buses, 19 generators, and 186 lines. We modify the thermal ratings of several lines so that they are fully loaded, or congested, thus creating conditions where the transmission switching can be used to mitigate the overloading conditions. The total demand is 4519 MW.

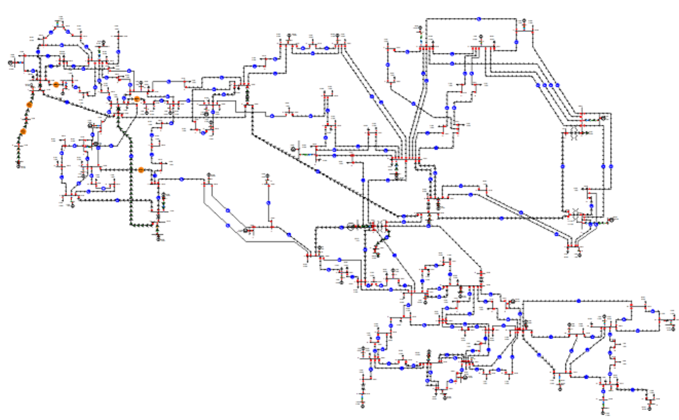

Figure 4.5: IEEE 118-Bus Network

FERC 13867-Bus Network

The second test case is FERC 13867-bus Network derived from the data set we obtained from PJM RTO, with 1,011 generating units, 18,824 lines and 13,867 buses. It is a much larger power network which is representative of realistic power grids that RTOs manage. We also obtained the loading condition of a typical summer day and a typical winter day. The average hourly load is 1,005,892MW in summer and 968,829MW in winter.

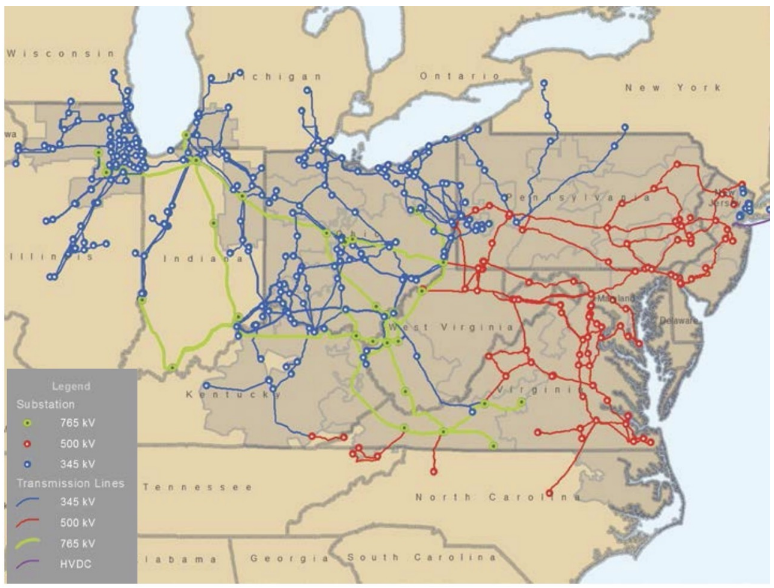

Figure 4.6: PJM Backbone Transmission System

4.4 Numerical Results

IEEE 118-Bus Network

We apply the three greedy algorithms discussed in Chapter 4.2 to the IEEE 118-Bus network. The percentage cost reduction and line switching action by each iteration is shown in Table.4.1. 12 Iterations are run for Line Enumeration and Line Selection with Priority List until there is little improvement by each iteration. For Divided MIP heuristic we partition 186 lines into 10 groups: nine with 18 lines and one with 24 lines. After 2 iterations the algorithm stops because we can no longer find the cost reduction further. L162, L153, L136 and L132 are switched by both Line Enumeration and Divided MIP. Line Selection with Priority List and Divided MIP switch L153 and L132. L132 is switched by both Line Enumeration and Line Selection with Priority List. We note that all algorithms switch two lines (L153 AND L132), which means they are critical lines for switching.

The computation time of the algorithms is listed in Table.4.2, with comparison to the best known optimal from directly solving the OTS problem. The algorithms are implemented

Table 4.1: % Cost Reduction of each Fast Heuristic (IEEE Case)

Iteration	Line Enumeration		Line Selection w/ Priority List		Divided MIP	
	Line Switched	% Cost Reduction	Line Switched	% Cost Reduction	Line Switched	% Cost Reduction
1	L153	6.30	L132	6.01	L129, L132, L126	13.42
2	L132	21.86	L163	12.49	L148, L153, L161, L162	24.59
3	L136	20.65	L133	16.52		
4	L162	21.72	L153	18.06		
5	L37	21.95	L151	21.65		
6	L122	22.09	L78	22.07		
7	L14	22.25	L85	22.27		
8	L31	22.30	L82	22.29		
9	L19	22.31	L96	22.33		
10	L54	22.32	L45	22.33		
11	L60	22.32	L48	22.33		
12	L68	22.32	L59	22.33		

on a laptop with 3 cores in parallel. In this toy test case, the computation time for the OTS with limited line switching choices is comparable to the time of solving a DCOPF. Therefore Line Enumeration and Line Selection with Priority List take much longer than Divided MIP. They have more models to run and need more iterations as well. From Fig.4.7, we can see that in terms of cost improvement, Divided MIP also outperforms the other two, only 0.5% less cost reduction than the best known optimal, with a computation time much less than directly solving the OTS. By the end of 12th iteration, Line Enumeration and Line Selection with Priority List achieve almost the same cost reduction, around 2.5% less than the best known optimal. For IEEE 118-Bus network, Divided MIP performs the best among three, both in terms of computation time and cost performance.

The convergence of the cost saving from Line Selection and Priority List show that to gain most of economic benefits from transmission switching, only a small number of lines need to be switched. This proves to be true in the original OTS model as well. However OTS model doesn't benefit from this knowledge since the operator still needs to run a full network MIP model. With the greedy algorithm, the operator can choose to stop after a few iterations after he sees the convergence of the cost savings.

Table 4.2: Final % Cost Reduction & Computation Time of each Heuristic

Alogorithm	% Cost Reduction	Computation Time (in seconds)
Line Enumeration	22.32	1314
Line Selection with Priority List	21.86	300
Divided MIP	22.33	17
Best Known Optimal	24.88	1680

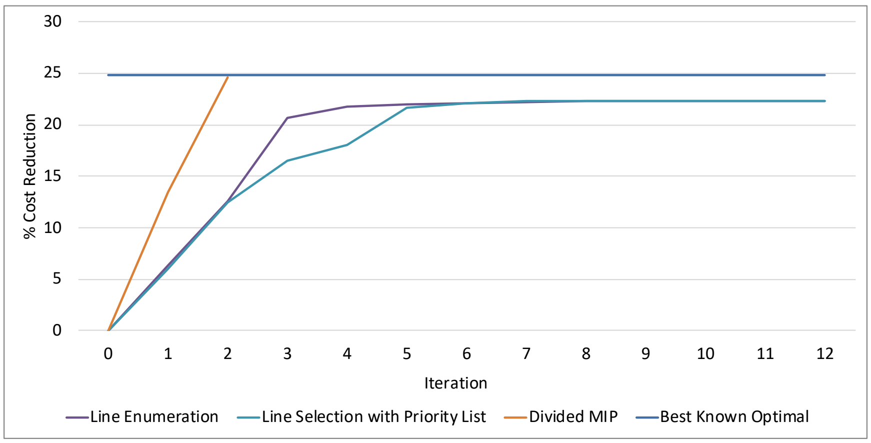

Figure 4.7: % Cost Reduction of each Heuristic compared to the Best Known Optimal (IEEE Case)

FERC 13867-Bus Network

For the FERC model, we implement the algorithms on the high performance computing cluster in LLNL. The demand data is from a typical summer day. Due to the scale of the model, Divided MIP, the best performer for the IEEE case, fails to be applicable for the FERC case. The MIP program, even with only a very small number of switching choices, cannot be solved within a reasonable timeframe. Therefore we only present the results of Line Enumeration and Line Selection with Priority List in Table.4.3, including the cost reduction and switching sequence.

Table 4.3: % Cost Reduction by each Heuristic(FERC Case)

Iteration	Line Enumeration		Line Selection with Priority List	
	Line Switched	% Cost Reduction	Line Switched	% Cost Reduction
1	L17230	0.193	L2813	0.098
2	L2913	0.502	L1831	0.200
3	L8731	0.792	L11231	0.226
4	L12031	0.991	L103	0.441
5	L7031	1.404	L7482	0.605
6	L721	1.420	L2310	0.893
7	L293	1.556	L14823	1.030
8	L7981	1.652	L5567	1.059
9	L10002	1.762	L787	1.255
10	L8310	1.860	L8313	1.268

Due to running time constraints, we only performed 10 iterations. With both algorithms implemented on 500 cores, Line Enumeration takes 18 hours to run while Line Selection with Priority List takes only 5.5 hours. Both algorithm leads to a cost reduction less than 2% after 10 iterations. However the cost saving has the potential to increase further when we switch more lines. Compared to the large number of lines in the network, 10 lines switched are far too small for the cost saving to converge. No line is switched by both algorithms. Line Enumeration is better than Line Selection with Priority List in the cost improvement. However the running time of Line Selection with Priority List is much less than the running time of Line Selection since Line Selection with Priority List only examines a small part of the lines at each iteration.

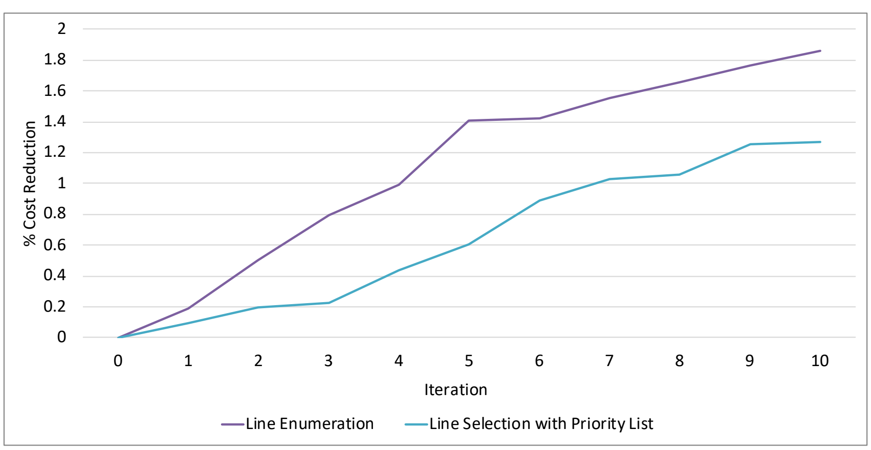

Figure 4.8: % Cost Reduction by each Heuristic (FERC Case)

4.5 Results Discussion

Demand Profile

In the last section we test the greedy algorithms on the base case of power networks. Obviously different patterns of loading conditions will produce different optimal network topology. The magnitude of cost benefits that the transmission switching is going to bring will also differ for different loading conditions. The loading condition used in the last section already causes congestion in several lines. If the demands are increased by 10% more, the problem becomes infeasible. Therefore we refer to the original load profile as High Demand. We decreased the demand by 10% to create the Medium Demand case and 20% to create the Low Demand case to simulate different loading scenarios. In the Low Demand profile there is nearly no congestion in the network.

The % cost reduction for the High demand, Median Demand and Low Demand are shown in Fig.4.9, 4.10 and 4.11. Several observations can be made. Firstly, Divided MIP heuristic constantly performs better than others, with more cost reduction and fewer iterations. Secondly due to the low congestion in Low Demand case, there is little potential for the transmission switching and the problem becomes merit-order dispatch. In contrast there are considerable cost reductions in the High Demand and Median Demand cases. Thirdly in terms of cost improvement the performances of Line Enumeration and Line Selection with Priority List are very close in all three loading conditions.

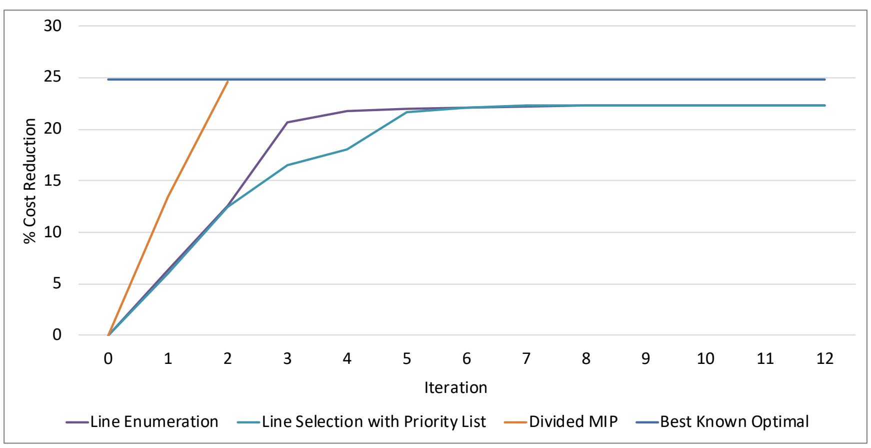

Figure 4.9: % Cost Reduction by each Heuristic (High Load)

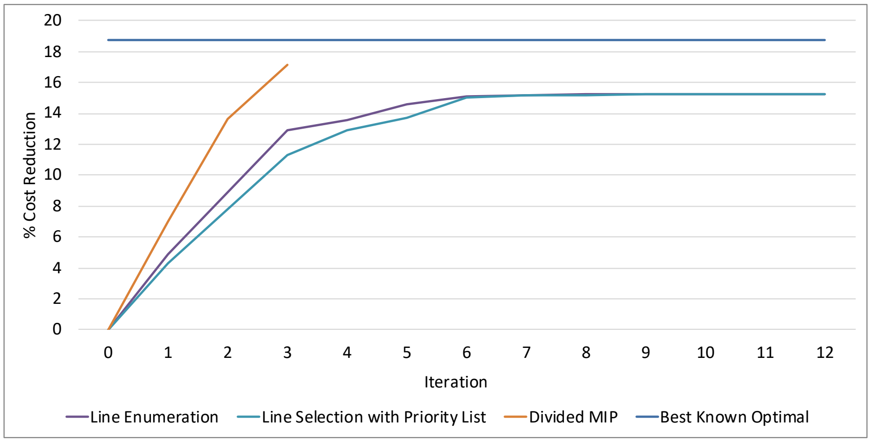

Figure 4.10: % Cost Reduction by each Heuristic (Median Load)

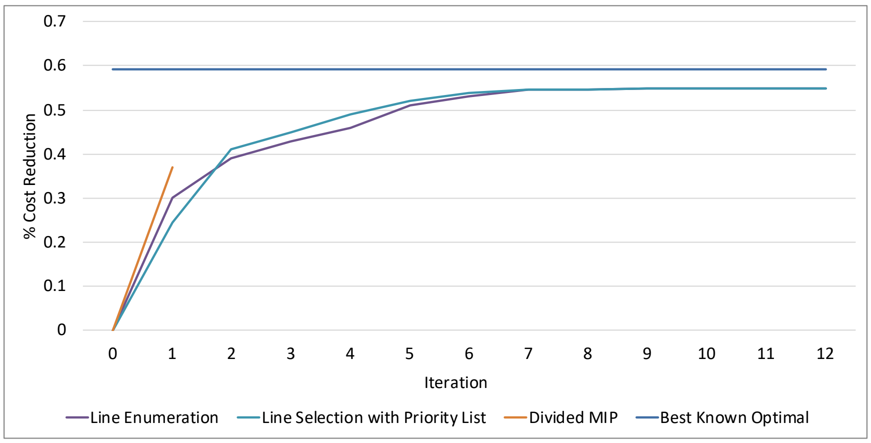

Figure 4.11: % Cost Reduction by each Heuristic (Low Load)

Table 4.4: Computation Time of each Heuristic in High/Median/Low Demand(in seconds)

Alogorithm	High Demand	Medium Demand	Low Demand
Line Enumeration	1314	1278	1264
Line Selection with Priority List	300	260	592
Divided MIP	17	24	14
Best Known Optimal	1680	1530	1280

The computation time of different loading conditions are summarized in the Table.4.4. The elapsed times for Line Enumeration under three loading conditions are relatively stable. This is expected since in Line Enumeration we examine every single possible line switch at every iteration. Line Selection with Priority List under Low Demand takes much longer than the other two because it has to search more lines for the cost improvement.

Table 4.5: R^2 of Regression Analysis between Actual Cost Reduction and γ

Alogorithm	R^2
High Demand	0.093
Medium Demand	0.082
Low Demand	0.010

Accuracy of γ as Predictor of Cost Reduction

In Line Selection with Priority List, the priority list is based on γ which is a dual variable indicating possible cost reduction. Ideally if γ is a good predictor then the priority list would be beneficial. More indicative is γ, less lines we need to search in each iteration. So we would like to find about how "accurate" γ is in terms of cost reduction prediction.

The most basic and intuitive measure for the above task is the regression analysis. We iterate over all possible line switching actions and compute the cost reduction corresponding to each line switch. Then we use ordinary least squares to understand whether actual cost reduction and γ are correlated. The R^2 measures the percentage of cost reduction explained by γ. $R^2 = 100\%$ means the two variables are perfectly correlated and therefore one is the perfect indicator of the other. $R^2 = 0\%$ indicates a list of lines in random order is as good as using γ. Table.4.5 shows the results from the regression analysis.

We can see that the performance of γ is very poor in terms of regression analysis. This is understandable because the dual variable just computes the cost reduction when there is an infinitesimal change towards opening the line. OPF is highly nonlinear and non-convex so when the binary variable actually changes from 1 to 0, the cost reduction can be very different than the sensitivity measure. Luckily we do not require γ to predict the magnitude of cost savings nor to produce a list with the perfect order of cost savings from line switches. With parallel computing techniques we are going to examine a number of lines as a batch and choose the best line switch within the batch. As long as the line with largest cost savings is within the first k lines in the priority list, it would be chosen as the next switching actions. Obviously larger number of k will give us a higher chance to choose the best possible line. For different k values, we calculate the average cost improvement difference between the best possible line switch in all lines and the best line switch in the first k lines over 5 iterations, for the IEEE 118-Bus Network. The results are shown in Fig.4.12, 4.13, and 4.14. Note that sometimes we cannot find the cost reduction among the first k lines. In the actual algorithm we move to the next k lines. But for this graphs, we treat the line switch with least cost increase as the one with largest cost reduction when there is no cost improving switching action in first k lines.

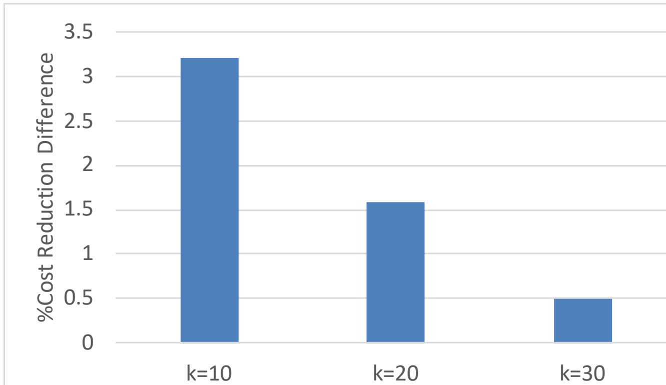

Figure 4.12: % Cost Reduction Difference between Overall Best Line and Best Line among First k Lines in the Priority List (High Demand)

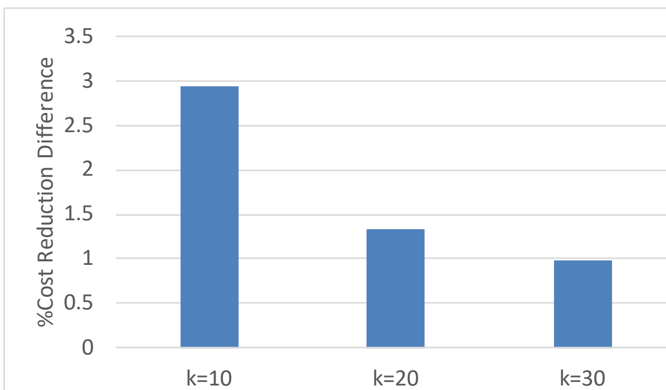

Figure 4.13: % Cost Reduction Difference between Overall Best Line and Best Line among First k Lines in the Priority List (Medium Demand)

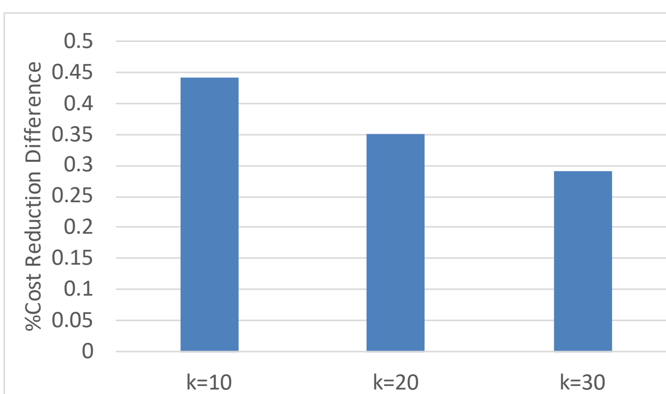

Figure 4.14: % Cost Reduction Difference between Overall Best Line and Best Line among First k Lines in the Priority List (Low Demand)

We can see that increasing number of parallel cores (k) can effectively reduce the percentage cost reduction differences, which means high k will lead to a more robust algorithm. With $k = 30$, the best line switching action among first 30 lines in the priority list only produce 1% less cost reduction on average than the best switching action among all lines, for all loading profiles. Even though γ is not the perfect indicator of cost saving, it provides good enough guidance for the line selection. It is also worth noticing that in the Low Demand case, the performance of γ is poor since the percentage cost difference shown in Fig.4.14 is relatively large compared to the actual cost reduction in Fig.4.11. It seems that when the network is more congested, the performance of γ is better.

Line Grouping of Divided MIP

For Divided MIP algorithm, the lines are assigned to their groups based on line numbers in IEEE 118-Bus model. Lines close in number are more likely to connect to the same bus or close to each other. Therefore, when we solve a Divided MIP model, the line switching actions considered are the lines in the same region in the network. In order to remove this effect, we randomly divided lines into 10 groups and investigate if different grouping could affect the result of this algorithm. The results are shown in Fig.4.15 and 4.16 for High Demand and Medium Demand. We don't include the results for Low Demand case here since the cost savings are not significant by Divided MIP in Low Demand.

We can see that there is no significant impact of different line groups. Therefore, line grouping mechanism is not an important consideration when we use Divided MIP algorithm. However we do notice that different line groups will result in different number of iterations used. This may be worth investigating when the network size becomes much larger and each iteration takes much longer. Except for Line 132 and L153, there are no other common switched lines from different groups. This suggests that there are multiple line switching combinations that can achieve satisfactory cost reduction.

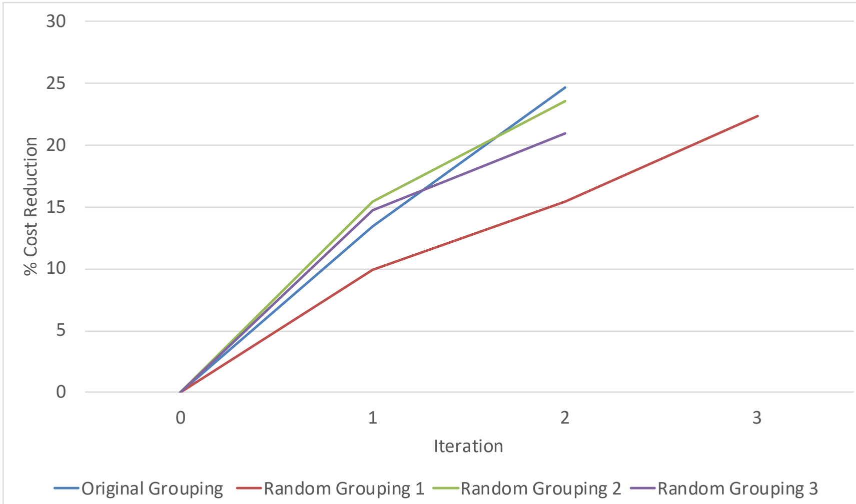

Figure 4.15: % Cost Reduction by Divided MIP of Different Line Groupings (High Demand)

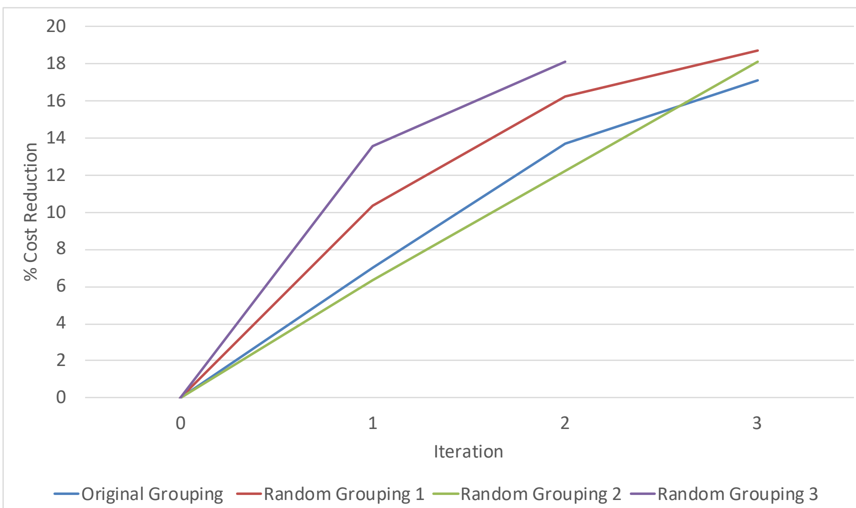

Figure 4.16: % Cost Reduction by Divided MIP of Different Line Groupings (Medium Demand)

Warm Starts for FERC Network

If the optimal solution of a linear program has been found and minor changes to the model need to be made, we can use a warm start in which the previous solution is initially used to restart the program. This is extremely useful for transmission switching algorithms since each iteration the network only differs by one line switch. For FERC network, the majority of the line flows and generator outputs will remain the same. Essentially only the area with the line switch needs to be re-evaluated. This property is especially useful since the model size of FERC network is so large that a DCOPF takes more than 5 minutes to solve on with

CPLEX 11. With the warm start technique, the solution time is reduced to seconds. Warm start may prove it even more beneficial for the Divided MIP algorithm, which is very hard to solve directly.

Chapter 5

Application of Machine Learning to Transmission Switching Line Selection

5.1 Introduction

The power system industry has been dealing with big data in many areas and constantly striving to develop new big data analytic tools to run the grid reliably and efficiently. Supervisory control and data acquisition (SCADA) has been implemented in power system to oversee, control and optimize the generation, transmission and distribution systems. SCADA improves the efficiency as well as enhance the reliability and stability of the integrated system operation. System operators depend on computer aided tools such as Energy management systems (EMS) to control and run the grid efficiently.

Machine Learning is an important branch of big data analytics which has been the center of research in the recent years. It is known for its superior performance for both classification and regression problems with large complex datasets, and the insights it can provide to the big data. Utilities have realized the potential that the machine learning will bring to the delivery of more reliable energy to their costumers. A survey [50] by SAS has revealed that utilities already use or plan to use machine learning in their key management and operation areas, as shown in Fig.5.1 and 5.2. The survey also revealed that one of the primary ways the utilities use machine learning is to better fit demand response events. The utilities used to determine the customers' baselines and find out how they actually performed by a manual process. Now they use machine learning to come up with a better baseline and operate in a real-time fashion. There is also great potential for risk management where neural network is used to predict the probability of outages of certain network components. The utilities have also used machine learning for the contingency response: identifying what the outage is, where it is and how fast they need to respond to it.

In the recent years there has also been rich literature of machine learning applications in the power system. Wehenkel [57] developed a framework that used machine learning to assess power system security. Firstly they examined diverse simulation scenarios which

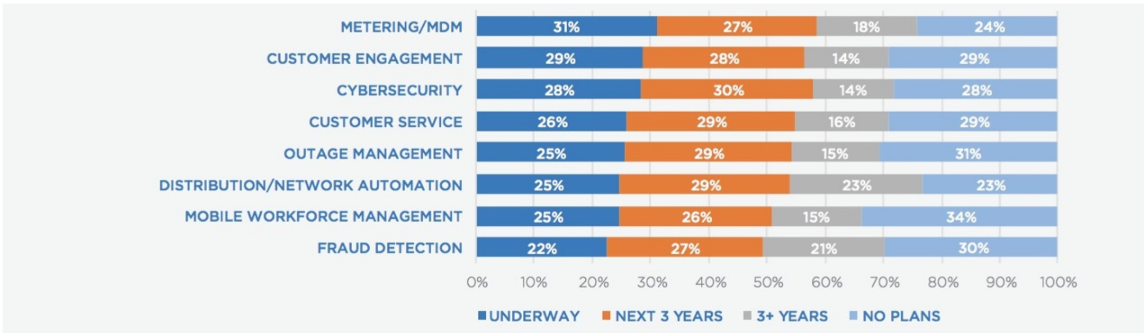

Figure 5.1: Use of Machine Learning for Key Utility Areas

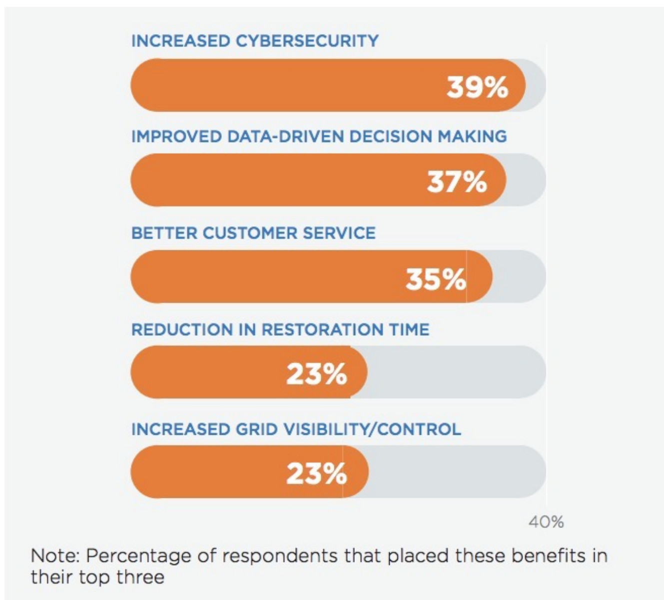

Figure 5.2: Top Five Machine Learning Benefits identified by Utilities

produced a large database. Then they extracted the main features of the system by applying machine learning techniques to the scenarios in the database. Based on this work, Ernst *et al.* [11] further explored how a computational reinforcement learning approach can be used for power system control. They discussed two reinforcement learning methods: the online mode and the offline model. In the online mode the machine learning agents were used in the realistic power system. In the offline mode the machine learning agents were used in a simulation model of the realistic power system. The authors in [10] reviewed the applications of unsupervised machine learning to the dynamic security assessment, including optimal load shedding and optimal power flow with security constraints. Canyasse *et al.* [7] designed and compared various supervised learning algorithms to compute the cost of ACOPF. They tested the algorithms on two IEEE networks and showed less than 1% error for both cost regression and feasibility classification. By reducing the dimensionality of the phasor measurement unit data, Xie *et al.* [59] proposed an early event detection algorithm. They implemented a dimensionality reduction algorithm based on PCA with an adaptive training procedure. Chen *et al.* [9] tackled the scenario generation problem by a data driven approach with high renewable penetration using two interconnected neural networks.

Due to the nature of the power network as a large and complex system and the fact that most properties of the network remain unchanged after a line switch, machine learning seems a natural choice for the line selection. In the next section we are going to discuss the general machine learning framework and the machine learning techniques used for this purpose.

5.2 Machine Learning Techniques

The general framework for machine learning consists of the following three components:

- Knowledge Base Generation
- Machine Learning Model Creation
- Test and Implementation

Knowledge Base Generation For the training of a machine-learning method, a Knowledge Base is necessary. A Knowledge Base contains a substantial amount of network states or instances. Each instance is described by a vector of variables called features or attributes, which are either directly given or measured (network configuration, generation limits etc) or indirectly calculated quantities (outputs from DCOPF, cost improvement by each line switch etc). A Knowledge Base is generated offline either with historical data or by random sampling. For IEEE toy test cases, the Knowledge Base is generated from random variables drawn from independent uniform distributions. For real networks, sufficient historical data of demand would be the best. If they are not available, the next best solution is to generate samples based on one instance of historical demand.

The Knowledge Base is then divided into a learning set for training the machine learning algorithms and a test set used for evaluating the performance of the developed algorithm.

Machine Learning Model Creation The model creation is constructed and trained using the learning set. It is used to extract and integrate hidden information and present it in a way useful for decision making. Normally the model creation will contain five steps: model type selection, data preprocessing, feature selection, model parameters fine tuning and the actual learning procedure.

The efficiency and effectiveness of the method depend on the selection of proper machine learning model. Feature selection techniques - algorithms that select a subset of relevant variables, are applied in order to cut down the dimensionality of the Knowledge Base. Feature selection is essential for shorter training times and it also makes the model easier to interpret by users. The noise of the training patterns is reduced by means of data preprocessing such as normalization and scaling of the data. The parameters of the model itself, such as the value of k or k nearest neighbor, are also tuned, in order to provide an optimized model.

Machine Learning model creation is the heart of the framework. A large number of machine learning methods may be used in a toolbox fashion, in accordance with the nature of data they have access to and type of prediction they need to make. Machine learning methods are organized into three main classifications: supervised learning methods, unsupervised learning methods, and reinforcement learning methods. In our work we use supervised learning methods since every line switch is tagged with a label: cost reduction. And we want to predict the switching action with the most cost reduction.

Test and Implementation During the test and implementation stage, the machine learning model takes the current network status as inputs, in order to predict the best line switch for the cost improvement.

Based on past literature, we select three common machine learning techniques that prove useful for the nonlinear and complex physical nature of the power systems: k nearest neighbor, artificial neural network and decision tree.

k Nearest Neighbor

The k nearest neighbor algorithm is the most basic of the instance reasoning method. This method view every instance as a point in the n-dimensional space R^n, where n is the dimension of the feature vector. The standard Euclidean distance is used to calculate the nearest neighbor of an instance. More precisely, let the feature vector of an instance x be

$$\{a_1(x), a_2(x), ..., a_n(x)\}$$

Where $a_r(x)$ represents the value of the rth feature of instance x

Then the distance between instances x_i and x_j is defined as $d(x_i, x_j)$:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{n}(a_r(x_i) - a_r(x_j))^2)} \tag{5.1}$$

Fig.5.3 shows how k nearest neighbor algorithms work where every instance is described by only two features. The label or the prediction one tries to make with the algorithm is a boolean variable. With a 5 nearest neighbor algorithm, the query instance x_q shown in the figure should be predicted with a negative value since the majority of its 5 nearest neighbor is labeled with a negative value.

The steps of k-nearest neighbor algorithm with a continuous function label are shown below.

Step 1: Add every training instance $(x, f(x))$ the Knowledge Base.

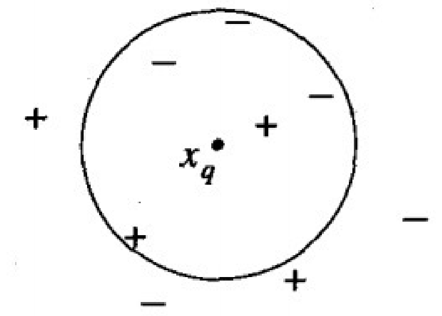

Figure 5.3: K-Nearest Neighbor

Step 2: For a query instance x_q, let $x_1, ..., x_k$ be the k instances in the Knowledge Base that are nearest to x_q in Euclidean distance. Return $\hat{f}(x_q) = \frac{\sum_{i=1}^{k} f(x_i)}{k}$.

The k nearest neighbor algorithm is a very effective inductive inference algorithm especially when the Knowledge Base contains adequately large amount of training data. It also can handle noisy training data ver well. However feature selection is extremely important for this algorithm since the algorithm solely relies on the Euclidean distance which is based on all features. Therefore all attributes are considered equal. To avoid the distance dominated by many irrelevant features, the features have to be chosen very carefully.

Decision Tree

The decision tree is a tree built from the root node at the top covering all the features in the Knowledge Base to ensure it is representative of the system. Each node examines a single feature, each branch represents a decision rule and each leaf contains an outcome. At each iteration, a leaf is examined and the algorithm determines whether it will be a terminal leaf node or it will be further splitted. An example decision tree predicting passenger's survival rate based on sex, age and number of relatives aboard on the Titanic is shown in Fig.5.4.

To further grow the tree based on a node, a proper feature is chosen first. Then a dichotomy test based on the feature's value is identified. The test T_0 is defined as:

$$T_0 : X \geq t^* \tag{5.2}$$

To determine what is the next feature to split and what is the optimal test, the additional information gained from the split is maximized. The additional information gain is calculated from the entropy (represented by "H") of each subset resulting from the split. The entropy is defined as:

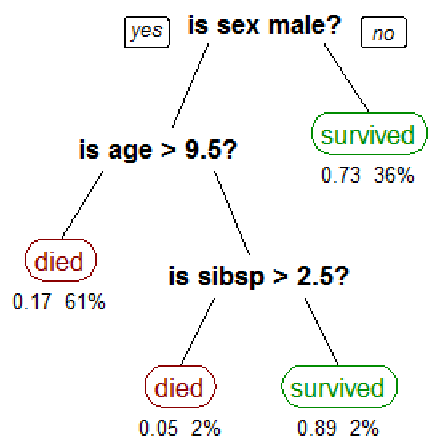

Figure 5.4: The Decision Tree: Probability of Survival on Titanic

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{5.3}$$

Where X is the split

n is the number of different classes in the child node

p_i is the percentage of class i in the child node

The information gain is the decrease in entropy after a split of a node. The algorithm will find the feature and its split with the maximum information gain. The construction of a decision tree takes the following five steps:

Step 1: Calculate the entropy of existing tree.

Step 2: Consider splitting current node based on different attributes. Normally two branches will be added to the tree and entropy for each branch is calculated. The sum of entropies from two branches are the total entropy after the split. The difference in entropies before and after the split is the information gain("IG") , defined as:

$$IG(T, X) = H(T) - H(T, X) \tag{5.4}$$

Where $IG(T, X)$ is the information gain of decision tree T with a child node X

$H(T)$ is the entropy of decision tree T before the spitting

$H(T, X)$ is the entropy of the decision tree T with a child node X

Step 3: Choose the feature and the test to maximize the information gain as the new branches and the child nodes.

Step 4a: If there is a branch with entropy 0, then it is a terminal node.

Step 4b: If the entropy is more than 0 for some branches, Step 1-3 is run recursively on those branches, until all branches' entropy are 0.

Artificial Neural Network

Inspired by biological neural systems composed of complicated layers of interconnect neurons, neural network learning is a robust approach to approximate real valued functions. The backpropagation algorithm has been proven to be a robust method and shown surprisingly successful performance in many practical problems such as recognition of handwritten characters [32], speech recognition [56], and facial recognition [15].

Capable of approximating different types of functions, multilayer networks with backpropagation algorithm is extremely useful when a nonlinear decision surface is considered. The network consists of one input layer whose number of neurons equal to the number of features, one output layer whose number of neurons equal to the number of outputs, and some hidden layers. Fig.5.5 shows a simple graphical illustration with 2 hidden layers. The neural network with backpropagation algorithm will learn the weights for neurons with an attempt to minimize the squared error between the network output values and the target values by gradient descent.

$$E(\overrightarrow{w}) = \frac{1}{2} \sum_{d \in D, k \in \text{outputs}} (t_{kd} - o_{kd})^2$$

Where D is the training dataset

t_{kd} and o_{kd} are the target and real output values for kth output unit of instance d

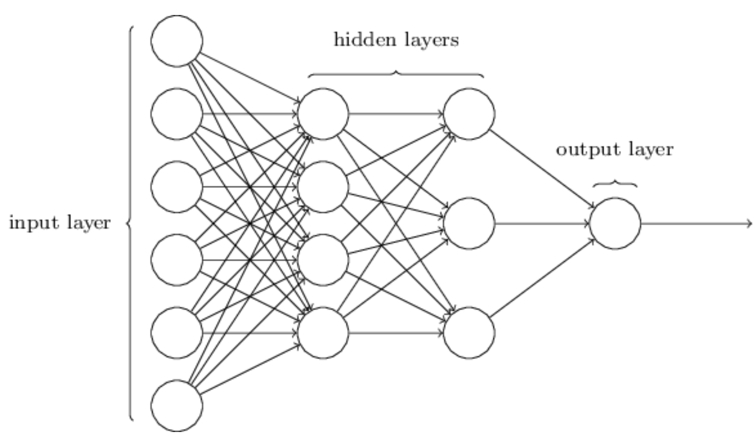

Figure 5.5: A Multi-layer Artificial Neural Network with 2 Hidden Layers

The steps for neural network algorithm with back propagation are as follows:

Each training instance consists of two parts: $(\overrightarrow{x}, \overrightarrow{y})$, where $(\overrightarrow{x}$ is the vector of features to input in the network, and $\overrightarrow{y}$ is the function value we want to approximate, i.e. the target output values.

Other parameters are defined as:

η: the learning rate.
n_{in}: the number of network features (inputs).
n_{hidden}: the number of neurons in the hidden layer.
n_{out}: the number of output neurons.
x_{ji}: the input from unit i to j.
w_{ji}: the weight from unit i to j.

Step 1: Create a network with n_{in} neurons in the input layer, n_{hidden} neurons in the hidden layer, and n_{out} neurons in the output layer.

Step 2: Randomly sets the initial value of weights.

Step 3: Until time is exhausted or target accuracy is acheved, do
For each $(\overrightarrow{x}, \overrightarrow{t})$ in Knowledge Base, do
Forward Propagation:
1). For instance $\overrightarrow{x}$ as input, calculate the output o_u of every neuron u in the network
For each neuron we use a sigmoid activation function to calculate the output o from the input x:

$$o = \frac{1}{1 + e^{-\overrightarrow{w}\overrightarrow{x}}}$$

Backward Propagation:
2). For each neuron k in the output layer, calculate its error term δ_k

$$\delta_k = o_k(1 - o_k)(y_k - o_k)$$

3). For each neuron h in the input layer, calculate its error term δ_h

$$\delta_h = o_k(1 - o_k) \sum_{k \in \text{outputs}} w_{kh}\delta_k$$

4). Update each network $w_j i$

$$\delta_{ji} = \delta_{ji} + \Delta\delta_{ji}$$

where

$$\Delta\delta_{ji} = \eta\delta_j x_{ji}$$

5.3 Algorithm Framework

We used three established machine learning algorithm in Weka machine learning software which are 10 nearest neighbor, artificial neural network and decision tree, as introduced in the last section. These are the most used regression methods that can take in the the parameters and loading conditions of the power network and produce a list of high priority line switches. The inputs are network status including the line configuration and DCOPF solutions and the output is a standardized score indicating the possible cost reduction. The process of this algorithm is very similar to the line selection with priority listing and shown in Fig.5.6. The only difference is that here we use machine learning to produce the list of lines which are worth evaluating more than others.

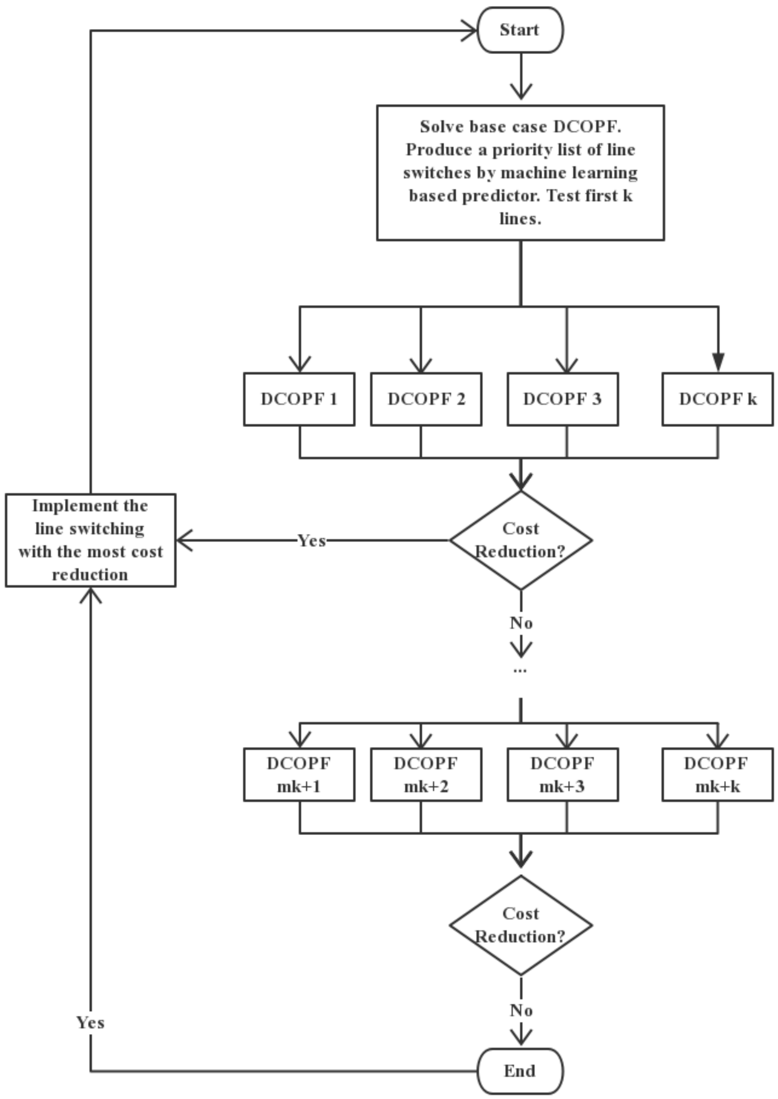

Figure 5.6: Line Selection with Machine Learning

5.4 Numerical Results

IEEE 118-Bus Network

Different network states were randomly generated representing various loading conditions in order to train and test the algorithm. In each network state, the loads were scaled by independent random variables drawn from uniform distributions. 30000 test cases are created for IEEE 118-Bus network are created for training, tuning and validation. First we

Table 5.1: % Cost Reduction by each Machine Learning Method (IEEE Case)

Iteration	10 Nearest Neighbor	Decision Tree	Neural Network
1	5.46	3.80	5.80
2	10.75	7.90	14.50
3	15.62	9.45	16.79
4	17.33	12.53	18.35
5	19.50	14.78	20.50
6	20.88	15.32	21.87
7	21.56	16.77	22.53
8	21.99	16.96	23.62
9	22.39	17.22	23.69
10	22.50	17.80	23.79

run the line enumeration algorithm on all the test cases so we have the complete information on the performances of every line switch which we can label the lines with. Then another 10000 test cases of IEEE network are used to assess the performance of the machine learning algorithm. The average percentage cost reduction by each iteration of Line Selection with Machine Learning is shown in Table.5.1, for three machine learning techniques. For each algorithm we run 10 iterations.

The average cost reduction of 10000 test cases by Line Selection with Machine Learning is shown in Figure.5.7. We can see that neural network outperforms the 10 nearest neighbor, which outperforms the decision tree. After 10 iterations, neural network and 10 nearest neighbor based line selection methods produce approximately 25% more cost reduction than decision tree method.

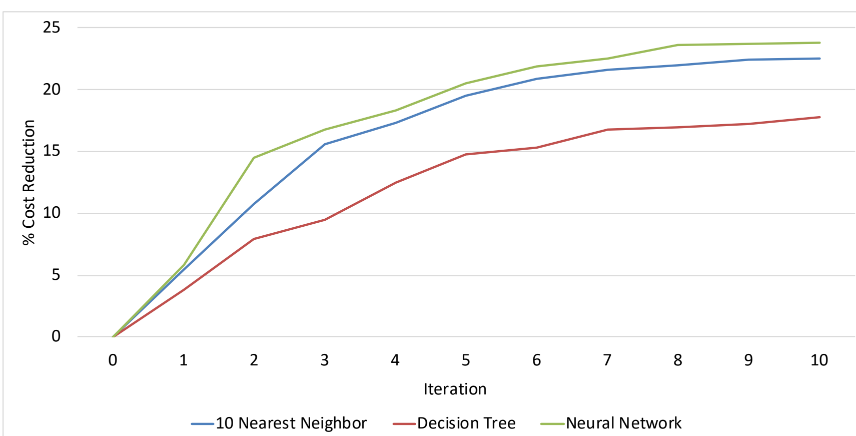

Figure 5.7: Average % Cost Reduction of each Machine Learning Method (IEEE Case)

In order to compare with the algorithms discussed in the last chapter, we apply the three machine learning based line selection algorithm to the base case of IEEE 118-Bus Network. This base case isn't in the Knowledge Base so it is considered as out of sample testing. The results are shown in Fig.5.8 and average percentage cost reduction after 5 and 10 iterations of all algorithms are illustrated in Fig.5.9. The computation time of each algorithm is shown in Table.5.2. Divided MIP is still the best algorithm in terms of cost improvement and computational performance. Line Enumeration and Line Selection with Priority List converge fast as the cost reduction after 5 iterations is very close to the cost reduction after 10 iterations. Machine learning based line selection generally converge slower than them. However after 10 iterations, neural network based line selection is able to achieve a higher cost reduction (1.5% more). In terms of computation time, machine learning based line selection methods are generally faster than the line selection based on dual criterion. With machine learning based line selection, no model is needed to be solved to develop the dual criterion so it takes less time to develop a priority list. But most of the solution time is dedicated to solve DCOPFs therefore the reduced solution time by the machine learning based line selection means that machine learning based methods are able to find a cost reduction switching action faster. The performance of the priority list produced by machine learning based methods will be discussed further in the Chapter 5.5.

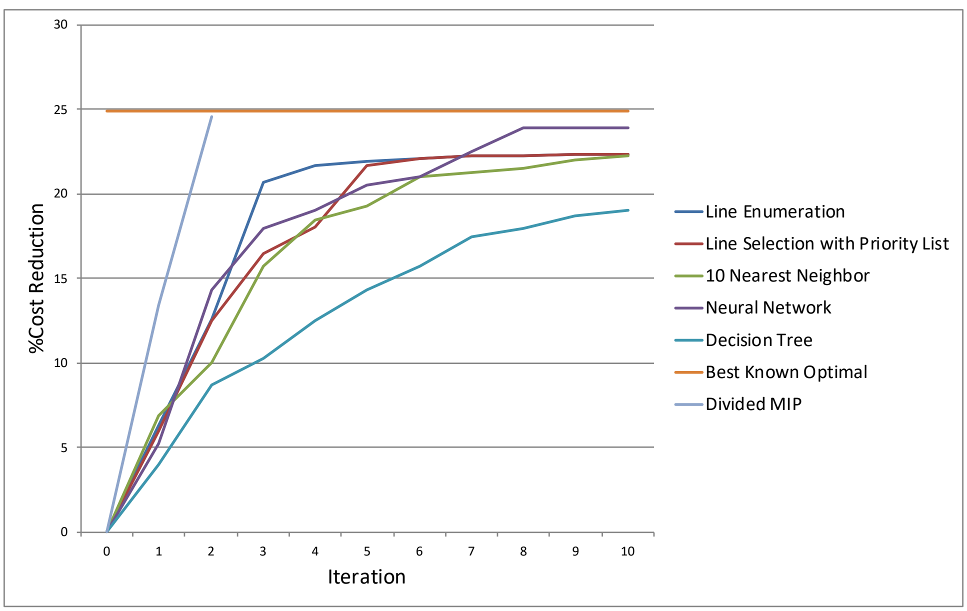

Figure 5.8: Average % Cost Reduction of Algorithms in IEEE Base Case

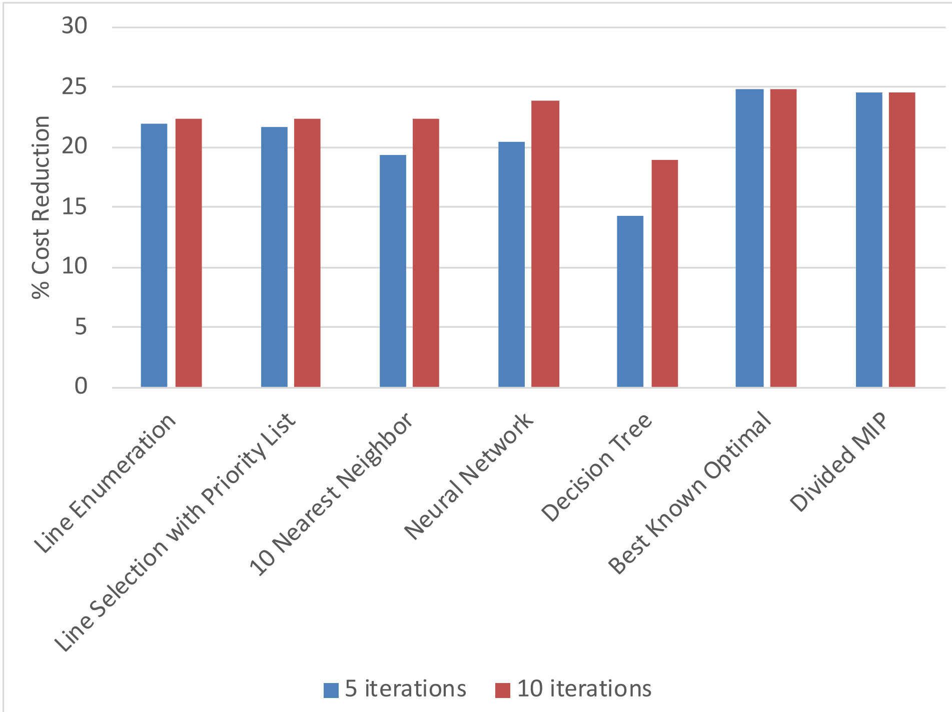

Figure 5.9: Average % Cost Reduction after 5 and 10 Iterations

Table 5.2: Computation Time of each Algorithm

Alogorithm	Computation Time (in seconds)
Line Enumeration	1314
Line Selection with Priority List	300
Divided MIP	17
10 Nearest Neighbor	263
Neural Network	278
Decision Tree	235
Best Known Optimal	1680

FERC 13867-Bus Network

Since we only have limited demand data for the FERC network, which is not enough to build a Knowledge Base, random sampling based on a hourly demand is used to generate instances. Similarly to IEEE 118-Bus network, we generate 4000 instances of network states drawn from independent uniform distributions, 3000 for training and 1000 for evaluation. The average percentage cost reductions by each iteration of line selection with machine learning for FERC network are shown in Table.5.3, for three machine learning techniques. 10 iterations are run for each algorithm. The best performer is still the neural network and it almost triples the cost reduction by decision tree after 10 iterations. The cost reduction hasn't converged by

Table 5.3: % Cost Reduction by each Machine Learning Method (FERC Case)

Iteration	10 Nearest Neighbor	Decision Tree	Neural Network
1	0.21	0.14	0.24
2	0.60	0.35	0.55
3	0.94	0.51	1.01
4	1.31	0.76	1.43
5	1.76	0.87	1.83
6	2.01	0.93	2.21
7	2.13	0.99	2.45
8	2.29	1.09	2.65
9	2.38	1.19	2.79
10	2.50	1.21	2.95

the end of 10 iterations and conducting more line switching actions have the potential to further reduce the cost.

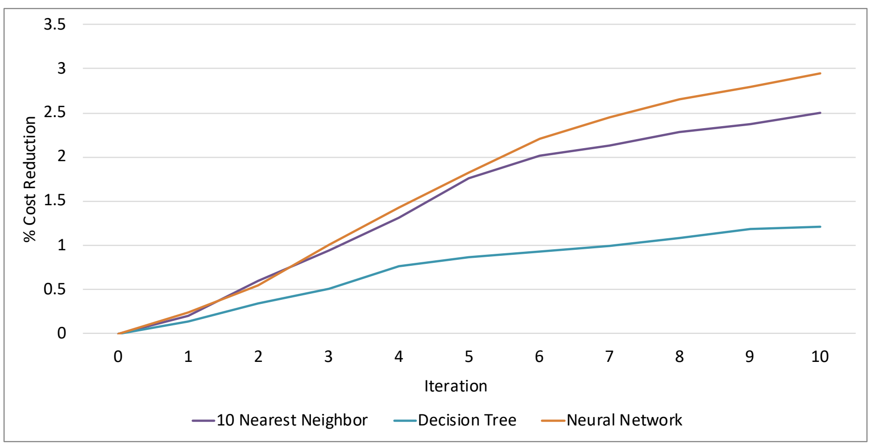

Figure 5.10: Average % Cost Reduction of each Machine Learning Method (FERC Case)

Compare to the number of lines in the network, this seems a small Knowledge Base due to the computation power constraint. However in reality operators only monitor part of the transmission assets and even a smaller set of lines are in consideration for switching. With the development of High Performance Computing and analysis on the switchable lines, through reliability analysis for example, a larger Knowledge Base and a smaller switchable set could enhance the accuracy and efficiency of the model training.

The results from applying machine learning based line selection methods on the base demand of FERC network are shown in Fig.5.11 and Fig.5.12. After 10 iterations, neural network and 10 nearest neighbor based line selection achieve similar cost reduction with Line Enumeration, and they outperform dual criteria based line selection. Decision tree method is the worst performer, which is the same with the IEEE network. Due to the small number of iterations performed compared to the large size of switchable lines, there is no convergence to a best known optimal. Almost every method used double the cost reduction from 5 iterations to 10 iterations. If more iterations are performed, it is highly likely that there will continue to be significant increase in the cost reduction.

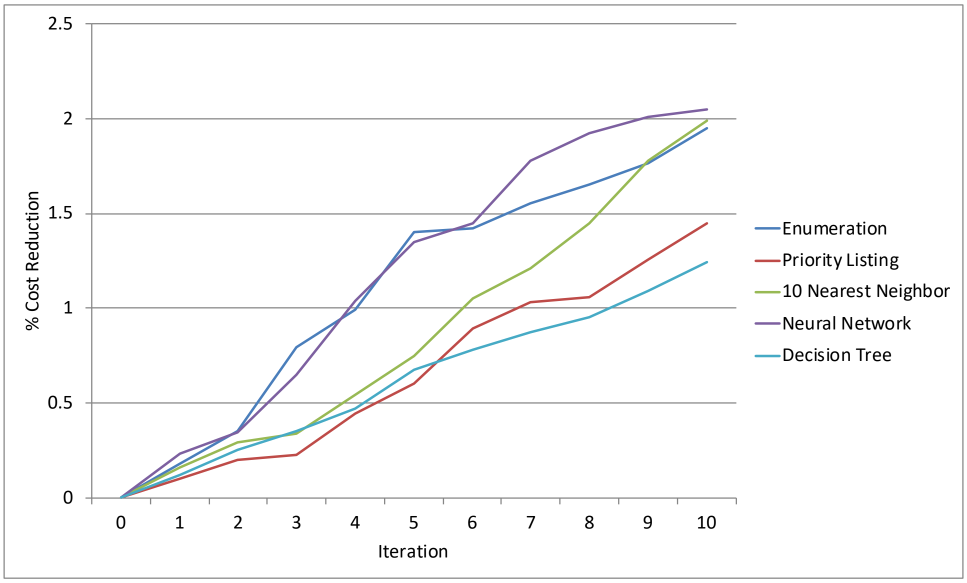

Figure 5.11: Average % Cost Reduction of Algorithms in FERC Base Case

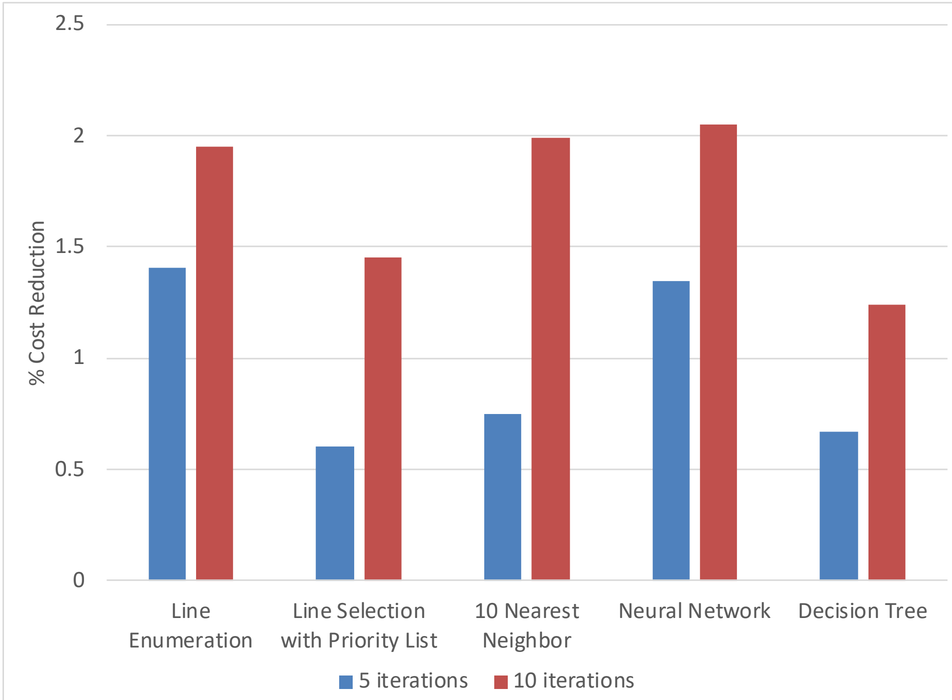

Figure 5.12: Average % Cost Reduction after 5 and 10 Iterations

5.5 Results Discussion

In this section, we would like to first summarize the results of machine learning based algorithms for IEEE network and FERC network and then discuss some important issues related to the machine learning algorithms and the results. After 10 switches the cost reduction gets very close to best know optimal for the 118-bus IEEE network, but not so for FERC network, understandably due to the size of the FERC model. 10 nearest neighbor and artificial neural network based line selection's performances are at least as good as Line Enumeration and Line Selection with Priority List. The best performer is neural network with little surprise for its consistent impressive performance with continuous-valued inputs and robustness to noisy data. The idea behind K nearest neighbor is simple but here it well captures the property of the power network that the switching action of a similar network state can be highly relevant and suggestive. The decision tree approach performs worse than others. The possible explanation is that it is most suitable for linear separable classes which goes against the nature of power systems. And data for transmission switching problem contains a lot of noise and outlier which decision tree approach is sensitive to.

A significant advantage of the machine learning based line selection algorithm, besides its superior performance in cost reduction for FERC network, is that once it completes the training, the runtime to select a line switch with cost reduction is negligible. In the previous algorithms, solving DCOPF and line selection are done online, which means for a practical

Table 5.4: R^2 of Regression Analysis between Actual Cost Reduction and γ

Alogorithm	R^2
Dual Criterion	0.093
10 Nearest Neighbor	0.219
Decision Tree	0.050
Neural Network	0.235

power network such as FERC network, it can take hours on a commercial laptop. It is especially critical for line enumeration algorithm, where a DCOPF optimization problem has to be solved at each step and for every branch. However if we use the machine learning approach, at each step we only need to examine a few switches that the algorithm suggests, which takes a few seconds, therefore saving computation power and time.

The Effectiveness of the Priority List

So far we have developed four methods to create a priority list which ranks the line according to the possible cost reductions. They are: dual criterion discussed in Chapter 4 and three machine learning based methods in this chapter. We would like to analyze and compare the effectiveness of the priority lists produced by different methods. First we show the result of regression analysis which gives the relationship between the actual cost saving and scores output by the machine learning methods or the γ produced by dual criterion, as in Table.5.4. The results are based on IEEE 118-Bus network. We can see that the results are still not very good but R^2 for 10 nearest neighbor and neural network has improved a lot compared to the dual criterion. Decision tree method may experience severe over fit or excess generalization error. The potential problem comes from the sequential nature of the decision tree.

To further analyze the effectiveness of the priority list, we consider a control group where the priority lists are randomly generated. The algorithm will search the first k lines and implement the line switching action with the most cost reduction. If the algorithm cannot find the cost reduction in first k lines it will search the next k lines until all the lines are exhausted. We compare the results from the four line selection algorithms and three randomized list, as shown in Fig.5.13, 5.14 and 5.15. Note that the random lists used for each case are different even though they may have the same name. (Random List 1 in Fig.5.13 is not the same list as the Random List 2 in Fig.5.14).

From these three figures we can see in general the four priority list based methods perform better than the random lists, especially for small k. When $k = 10$, the % cost reduction achieved by priority lists almost doubled the cost reduction of Random List 1 and Random List 3. Random List 2 is an exception. Random lists seem to be "hit or miss" while priority lists provide more reliable and consistent cost savings.

We notice in Fig.5.13 when $k = 10$, Random List 2 shows superior performance than the other two and even some random lists for $k = 20$ or 30. We take a closer look into the details of Random List 2 and find out that the first 10 lines on the list contains L153, which is a critical line switched in all three greedy algorithms, as discussed in Chapter 4. Therefore at the first iteration, L153 is switched, which leads to good performance in cost reduction in this case.

Another observation we make is that in general as the value of k increases, the performances of all algorithms will become better. This effect is more significant for the random lists both in terms of cost reduction and rate of convergence, which means that the priority lists are more robust against small k. When the number of parallel cores are limited (which is often so in reality), priority lists prove to be more beneficial than the random lists.

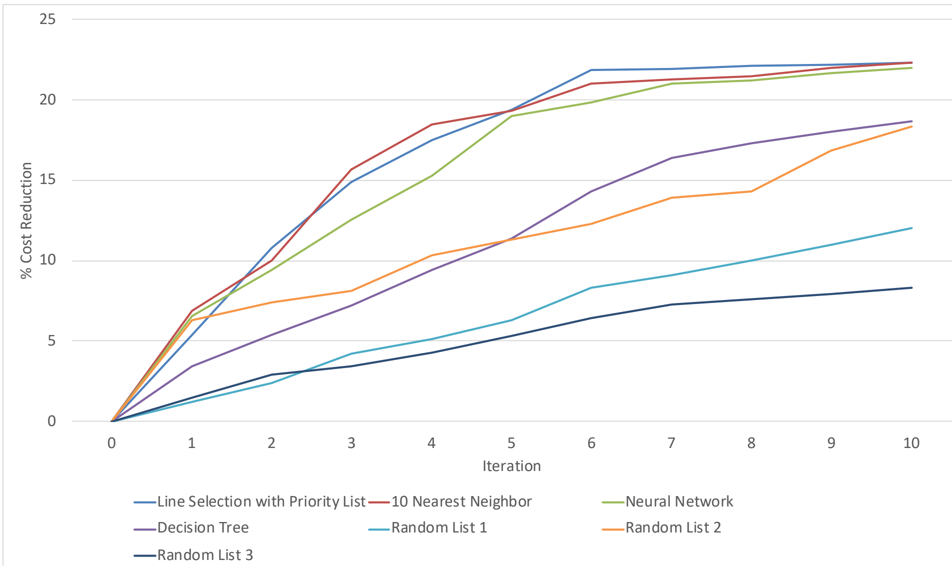

Figure 5.13: Average % Cost Reduction of Line Selection Algorithms And Random Lists ($k = 10$)

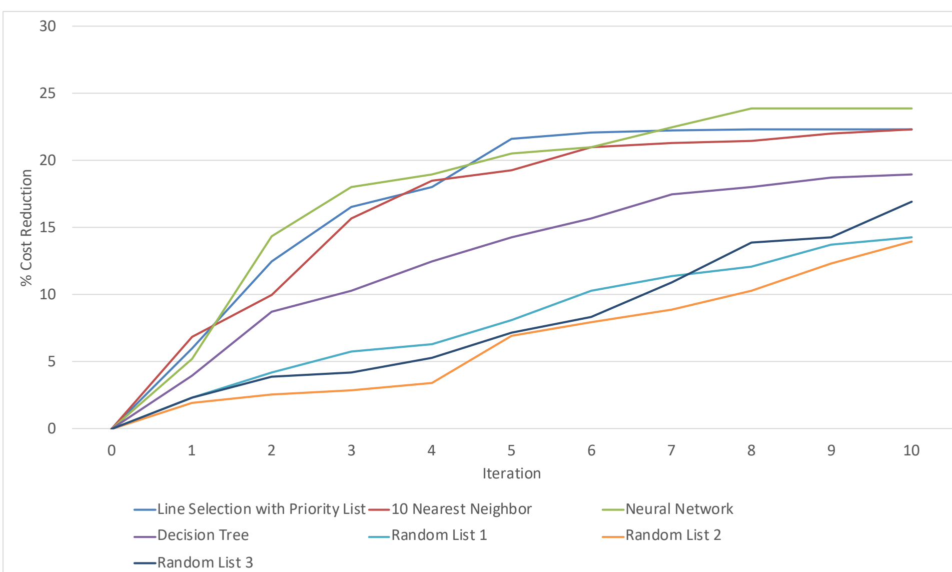

Figure 5.14: Average % Cost Reduction of Line Selection Algorithms And Random Lists ($k = 20$)

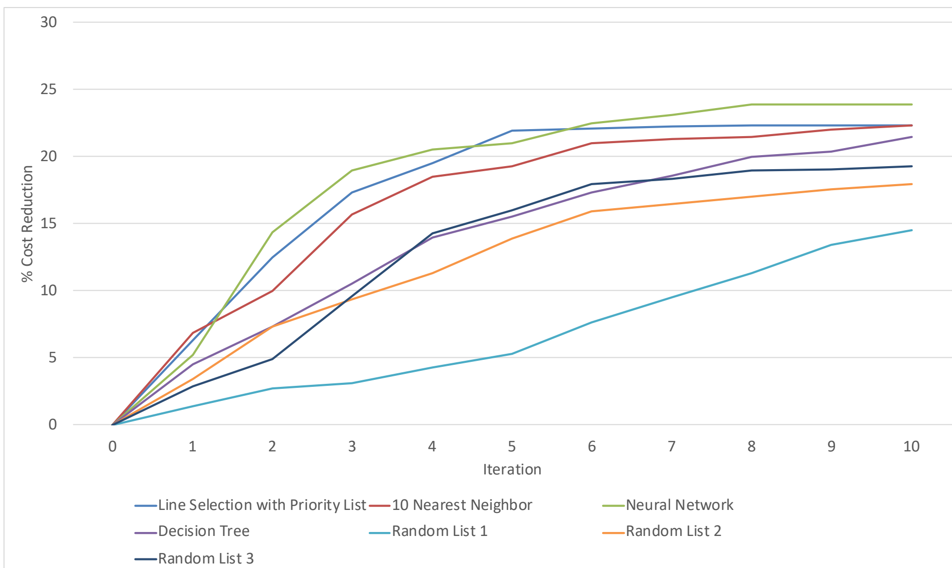

Figure 5.15: Average % Cost Reduction of Line Selection Algorithms And Random Lists ($k = 30$)

Chapter 6

Algorithm Selection

6.1 Introduction

Many research explored how to select the optimal algorithm for a given problem based on the varying properties of the problem. Rice [39] first propose the formal formulation of the algorithm selection problem, which has attracted a lot of attention since then. With the development of various algorithms for a single application, researchers have come to realize it is very difficult to find one best algorithm for the application with different properties or data inputs. The algorithm selection problem seeks to find the relationship between the data inputs of the model and the performances of different algorithms for the model. Once the relationship is established, it can be used to predict the performances of different algorithms for new data inputs. Machine learning has a rich history in algorithm selection for various applications. It is a classification problem which should be trained by supervised learning methods with label attached to each training example representing its optimal algorithm. The selector is trained to pick the best algorithm with inputs being parameters of the system's status. Many machine learning methods have been applied to the algorithm selection problem. A meta-learning inspired framework for analyzing the performance of heuristics for optimization problems by neural networks was proposed by [47]. The effectiveness of an integrated algorithm selection method was demonstrated in simulation systems with decision trees when the underlying algorithms and their implementations were unclear to the users in [12]. The authors of [55] showed the performance of support vector machine based automatic tuning system for computational kernels. The authors of [27] presented algorithm selectors for the power flow management based on network states and show performance benefit based on IEEE networks and a real power grid.

Algorithm Selection Problem

Following the framework developed by Rice [39] and Vanschoren [54], we characterize the Algorithm Selection Problem by the following four elements:

- The Problem Space (P), characterized by all the inputs x in the dataset used for the study. In this paper each x represents a different network state.
- The Feature Space (F), characterized by the key characteristics produced by a feature extraction process $f(x)$, that can be used to represent the problem. In transmission switching features it could be the line status or the loading conditions of each network state.
- The algorithm Space (A), containing the set of algorithms used to solve a given problem. In this paper it contains heuristics proposed to solve the transmission switching problem.
- The performance measures space (Y), containing ranges of measures that describes the performance of the algorithms. In this paper the performance is measured by the cost reduction from the line switches.

Solving the algorithm selection problem can be illustrated as follows: [1]

Definition: For a given problem instance $x \in P$, with features $f(x) \in F$, there is a selection mapping S(f(x)) into the algorithm space A, such that the selected algorithm $\alpha \in A$ maximize the performance mapping $y(\alpha \in A) \in Y$.

In the algorithm selection problem, we use machine learning methods to try to find the mapping S, which is from problem space to algorithm space, which will produce a prediction on the performances of algorithms based on the features of x.

The framework of the algorithm selection is shown in Fig.6.1 [1].

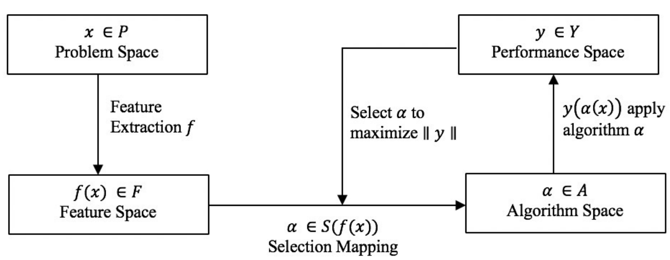

Figure 6.1: Framework for Algorithm Selection

Algorithm Selection Methodologies

In an ideal situation, we would have adequate information of the algorithms and dataset to choose the most suited algorithm based on some characteristics of the problem. However in reality, the systems like power grids are too vast and complex for such analysis. Due to the fact that a large amount of data is involved in the various applications including power flow problems, machine learning is a natural choice to derive selectors. Several most prevailing machine learning methodologies are discussed here.

Case-based Reasoning

As first introduced in [40], case- based reasoning chooses algorithm for the existing problem with knowledge of past problems. Instead of trying to learn what characteristics affect the performance, it just used the performances of past known problems to infer performance on new problems.

The most intuitive and commonly used case-based reasoning algorithm is nearest neighbor classifier. The case base contains the solved problem instances, labeled with the optimal algorithms to solve them. The nearest neighbors are determined by calculating the Euclidean distance. The advantage of case-based reasoning is that when the case base (a.k.a Knowledge Base) contains instances representative of all the possible problem states, it usually achieves very good performance. Weka IBk nearest neighbour classifier with 10 nearest neighbors is used in this thesis.

Classification

Conceptually, algorithm selection is a straightforward classification problem - each problem instance comes with a label on the best algorithm to solve this particular instance. This classification problem is usually solved by building a model that differentiates he algorithms based on the input dataset of the problem. Many of the classifier machine learning methods have been applied to the algorithm selection problem, most popular two being artificial neural networks and decision trees. The input layer contains input neurons that send information to the hidden layer. For neural networks, we use MultilayerPerceptron ANN implementation from Weka machine learning software. For decision tree learning, Weka's J48 implementation of classification tree is used in this paper.

Regression

On the other hand, researchers have been using regressors to predict the performances of each algorithm. For example, when runtime is considered as a factor in the performance space, regression is usually used to predict the computation time. Instead of labeling each problem instance with the best algorithm to use, the regression model predicts the performance of all algorithms for a given problem and gives the users more information about how algorithms compare to each other for different input datasets.

6.2 Generation of Algorithm Selector

As discussed in Chapter 2, there are two mainstream transmission switching formulations: one based on DCOPF and the other with PTDF and flow canceling transactions. Our work propose a method to create algorithm selectors for transmission switching. In the algorithm space, three algorithms are considered: line enumeration, line selection with priority listing and PTDF based method. The first two algorithms are discussed in Chapter 3.. We use the PTDF based algorithm as illustrated in [42], which is reviewed in Chapter 2. We train classifiers which take in the parameters and solutions of DCOPF as inputs and predict which is the best algorithm to use for a specific loading profile. To create an algorithm selector, the following five steps are followed:

Step 1: Generate the training dataset to be used for building selectors. The problem instance is the transmission switching problem, whose purpose is to find the best switching line in terms of cost improvement. For IEEE 118-Bus network, the training dataset is constructed by testing each algorithm on 30000 loading profiles generated for each network. This dataset and the loading profiles with in it are separate from the 10000 test cases that later are used to test the selectors. For FERC network, the procedure is similar except for the fact that only 3000 test cases are used for training and 1000 are used for evaluation.

Step 2: Split the algorithm performance dataset into equal parts for training, tuning and validation.

Step 3: Iterate over possible selection sets and create selectors. In this step first we train a selector with training and tuning parts of the dataset then evaluate the selectors with the validation dataset.

Step 4: Re-split algorithm performance dataset into equal parts for training and tuning.

Step 5: Take the most effective selector built in Step 3 and re-train it.

We still use the same three machine learning algorithms: k nearest neighbor, artificial neural network and decision tree.

6.3 Results

IEEE 118-Bus Network

We test the three machine learning based algorithm selectors on test set which contains 10000 instances. The results are shown in Fig.6.2. It also shows the cost reduction achieved by an "oracle" that has perfect a priori knowledge about which algorithm will be the best. Specifically, this is the result when the algorithm with the most cost reduction is used at each iteration to select the line switching action. Note that it does not mean the line selection is the optimal. We can see in general that neural network performs better than nearest neighbor and decision tree, though the differences in final cost reductions are relatively small. Even the worst performing decision tree is only around 10% worse than the oracle selection.

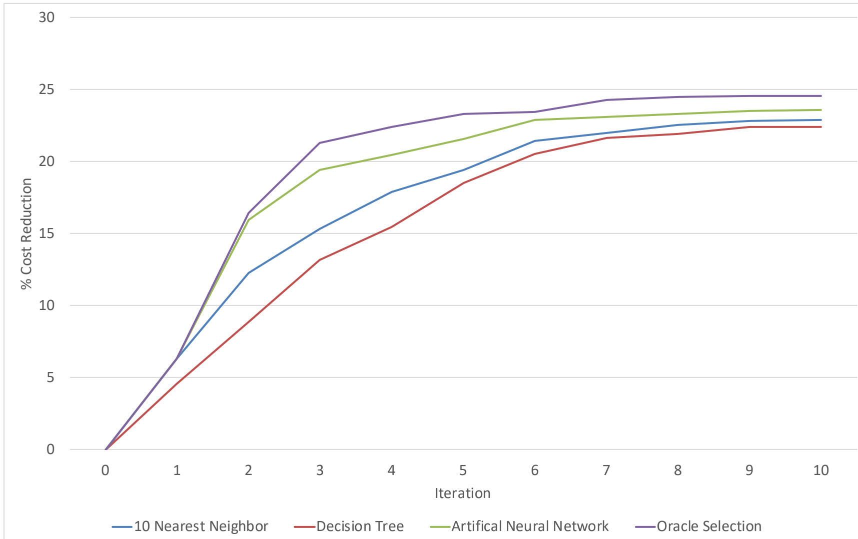

Figure 6.2: Average % Cost Reduction of Algorithm Selectors (IEEE Case)

Table 6.1: % Cost Reductions of Transmission Switching Algorithms and Selectors (10 Iterations) in IEEE Base Case

Algorithm	% Cost Reduction
Line Enumeration	22.31%
Line Selection with Priority Listing	22.33%
PTDF Method	21.04%
10 Nearest Neighbor Based Selector	22.12%
ANN Based Selector	23.15%
DT Based Selector	21.45%
Oracle Selections	23.51%

Fig.6.3 and Table.6.1 show the performances of individual transmission switching algorithms available to the selectors and the selectors themselves. For IEEE 118-Bus base test case, two selectors show performance improvement from the best performer of individual transmission switching algorithm - Line Enumeration. All three selectors show improvement from the worst performer - PTDF method. The cost reduction improvement is small, less than 2% in the best case. This is expected since the individual algorithm's solution gets very close to the best known optimal: 24.88%. Even with optimal algorithm selection at every step the cost reduction is 23.51%, only half percent higher than our best selector performer - ANN based selector.

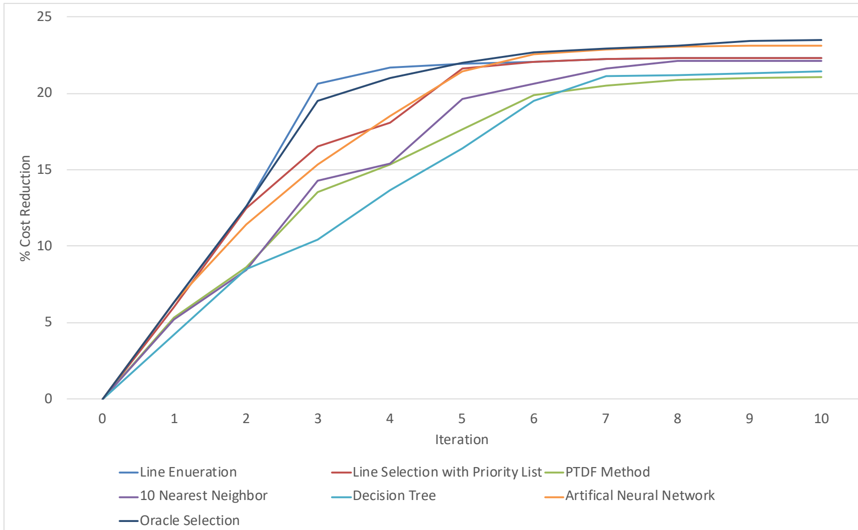

Figure 6.3: % Cost Reduction of Algorithm Selectors in IEEE Base Case

FERC 13867-Bus Network

We test the three machine learning based algorithm selectors on test set which contains 10000 instances. The results are shown in Fig.6.2, with comparison to oracle selection. We can see for FERC network, the neural network outperforms decision tree which outperforms the 10 nearest neighbor. The differences in performance are significant that neural network is almost as good as oracle selection whereas 10 nearest neighbor is only half as good as oracle selection after 10 iterations.

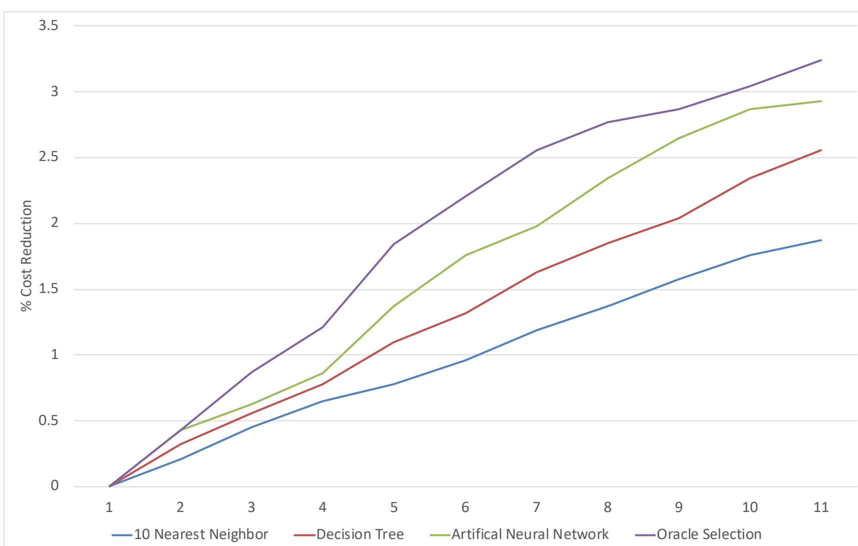

Figure 6.4: Average % Cost Reduction of Algorithm Selectors (FERC Case)

The individual algorithm and selectors' performances are summarized in Fig.6.5 and Table.6.2. We can see that all three machine learning based selectors improve on the individual algorithms. The improvement is much more significant in FERC network then in IEEE network. Even though the nearest neighbor based selector performs worst than the line selection, the best performer - ANN based selector results in around 50% more cost reduction than the best performer in the individual algorithm, and doubles the cost reduction of the other two. It also gets very close to the oracle selection, which means that the selector wisely chooses the algorithms so that it captures most of the benefits brought by the ability to select algorithms. Decision tree based selector shows considerable performance improvement from the individual algorithms as well. In terms of computation time, the selector itself adds negligible time to the computation since it just performs a linear calculation to select the algorithm for use.

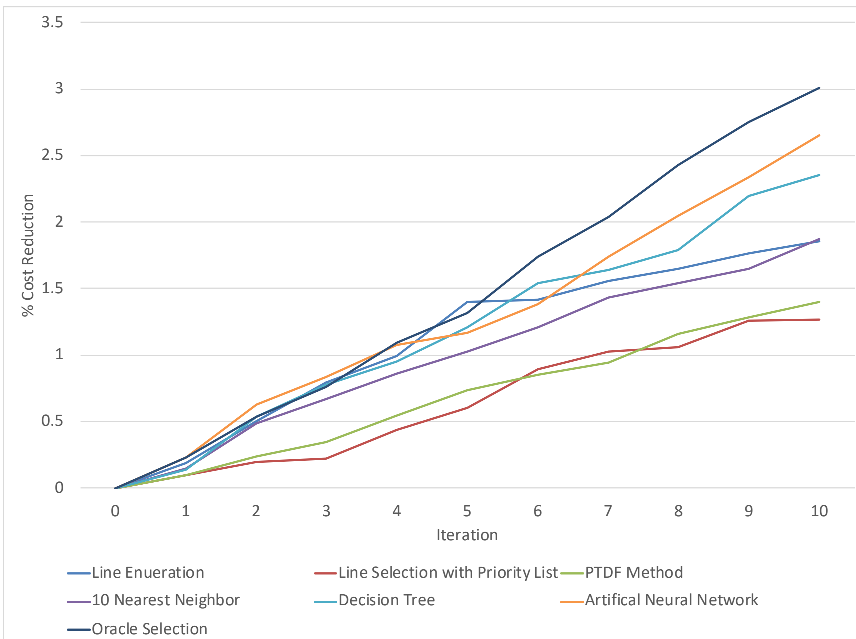

Figure 6.5: % Cost Reduction of Algorithm Selectors in FERC Base Case

Table 6.2: % Cost Reductions of Transmission Switching Algorithms and Selectors (10 Iterations) in FERC Base Case

Algorithm	% Cost Reduction
Line Enumeration	1.86%
Line Selection with Priority Listing	1.268%
PTDF Method	1.4%
10 Nearest Neighbor Based Selector	1.87%
ANN Based Selector	2.65%
DT Based Selector	2.35%
Oracle Selections	3.01%

All three machine learning based selectors prove effective in terms of cost reduction for both test cases. It shows considerable improvement especially for the realistic FERC test case. Due to the high complexity of the FERC network, it is expected that no algorithm will always be the most effective, therefore rendering algorithm selector useful. Even though with the machine learning algorithm we tested, the selector gets very close to oracle selection, it isn't necessarily the best selector there is. Although we tested it on two networks that are drastically different in their scales and characteristics, it doesn't guarantee the performance improvement for all power networks.

Machine learning based algorithm selection requires significant amount of computational power and time to train the model. Without proper parallelization techniques for the machine learning methods, it takes more than 10 hours to train the selector for FERC case. The effective parallelization of the machine learning methods and the increasing computational power within power systems are the two critical factors that will speed up the process of creating selectors.

Chapter 7

Conclusions and Perspectives

In order to tackle the computational challenge of the mixed integer Optimal Transmission Switching program, we present several fast greedy algorithms implemented in parallel to find the transmission switching actions for economic benefits. Instead of solving the OTS problem directly, the algorithms we use only seek to switch one line per each iteration. A series of DCOPF, a linear program, are solved in sequence. We presented three algorithms: Line Enumeration, Line Selection with Transmission Switching and Divided MIP. Furthermore we explore the use of machine learning methods for the line selection, which proves useful both in cost performance and computation time. Lastly, based on the algorithms in literature and algorithms we develop, we apply the algorithm selection to the transmission switching problem. The algorithm selection, based on machine learning classification methods, re-selects the algorithm at each iteration. We tested these algorithms on IEEE 118-Bus network and a realistic FERC 13867-Bus network and present the results.

In this chapter first we present a summary of conclusions based on our current work. Then possible further directions that can contribute to the research on transmission switching are discussed.

7.1 Conclusions

Greedy algorithms can achieve near optimal cost reduction with significant saving in computation time. The IEEE network results show that all three greedy algorithms achieve near optimal cost reduction within a few iterations. The numerical results also show that for small sized IEEE network, Divided MIP performs better than Line Enumeration and Line Selection with Priority List. However for the practical size network, Divided MIP fails to work due to the large size of the model. The other two algorithms show their strength for practical size models for they only need to deal with linear programs. Line Selection with Priority List is especially promising since at each iteration we only need to examine limited number of lines on the top of the priority list, therefore reducing computation time. A good priority list is essential for the performance of the algorithm.

High performance computing resources are essential for the greedy algorithms' superior performance in computation time. The advantage of the greedy algorithms is further strengthened by adoption of high performance computers. With the parallel implementation of the greedy algorithms, the computational time can be greatly reduced. Increasing the number of cores available will directly reduce the computation time. When the number of lines are far more than number of cores, doubling number of cores will halve the computation time.

The appropriate machine learning improves the performance of line selection. We test three machine learning methods for the line selection. With input being the network's configuration and DCOPF outputs, the machine learning based algorithm is trying to learn a score representing the possible benefits that line switching actions can bring. The test results of two networks show that the neural networks and 10 nearest neighbor improve the cost performance of the original dual criteria, while the decision tree fails. Machine learning based line selection also enjoys the advantage of quick online priority list production, even though the offline training time can be significant.

Algorithm selection effectively selects the algorithm for both test networks The three machine learning based algorithm selector all outperform the individual switching algorithms. With the artificial neural network based selector the cost reduction gets very close to the oracle selection. The performance improvement is especially significant for the FERC case, showing great benefit potential of algorithm selectors for real life power networks.

7.2 Future Area of Research

The Divided MIP produce the best result for IEEE network. However it does't scale well for the real size power network with the current parallelization method. Further research in the parallelization of MIP programs with large sets of continuous variables and constraints but a small set of integer variables can be useful to improve the performance of Divided MIP algorithm. A promising direction is distributed parallel mixed integer programming, which modifies the branch and bound algorithm to be implemented in parallel. It facilitates solving a MIP problem with distributed memory and possibly across different machines. It takes advantage of the fast development of high performance computing resources and can be potentially very useful for the transmission switching problem with limited set of line switching candidates.

When we train the machine learning line selection for the FERC network, a Knowledge Base of only 3000 instances is used due to time and computational power constraints. Even though only part of the lines are monitored and considered for the FERC network, it still suffers the risk of being under trained. Further research will be useful to confirm the benefit of machine learning based line selection a with larger Knowledge Base and more efficient machine learning algorithms, potentially implemented in parallel as well.

Another further research direction is to incorporate the requirement of reliability standards. NERC has imposed a 'safety net' minimum reliability standard of N-1 for contingencies which means that the system is planned such that for any one credible contingency event, the system can move to a satisfactory operating state. Reliability and stability check after switching actions are necessary in practice to ensure the smooth operation of the power grid. This is especially important if the operators wish to switch multiple lines at once for economic benefits.

Bibliography

[1] Salisu Mamman Abdulrahman et al. “An Overview of the Algorithm Selection Problem”. In: *International Journal of Computer (IJC)* 26.1 (2017), pp. 89–98.

[2] R Bacher and H Glavitsch. “Network topology optimization with security constraints”. In: *IEEE Transactions on Power Systems* 1.4 (1986), pp. 103–111.

[3] Anastasios G Bakirtzis and Pandelis N Biskas. “A decentralized solution to the DC-OPF of interconnected power systems”. In: *IEEE Transactions on Power Systems* 18.3 (2003), pp. 1007–1013.

[4] Anastasios G Bakirtzis and AP Sakis Meliopoulos. “Incorporation of switching operations in power system corrective control computations”. In: *IEEE Transactions on Power Systems* 2.3 (1987), pp. 669–675.

[5] Pranavamoorthy Balasubramanian and Kory W Hedman. “Real-time corrective switching in response to simultaneous contingencies”. In: *Journal of Energy Engineering* 141.1 (2014), B4014003.

[6] Pandelis N Biskas et al. “A decentralized implementation of DC optimal power flow on a network of computers”. In: *IEEE Transactions on Power Systems* 20.1 (2005), pp. 25–33.

[7] Raphaël Canyasse, Gal Dalal, and Shie Mannor. “Supervised learning for optimal power flow as a real-time proxy”. In: *Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), 2017 IEEE*. IEEE. 2017, pp. 1–5.

[8] J Carpentier. “Contribution to the economic dispatch problem”. In: *Bulletin de la Societe Francoise des Electriciens* 3.8 (1962), pp. 431–447.

[9] Yize Chen et al. “Model-free renewable scenario generation using generative adversarial networks”. In: *IEEE Transactions on Power Systems* 33.3 (2018), pp. 3265–3275.

[10] Mircea Eremia, Chen-Ching Liu, and Abdel-Aty Edris. *Advanced solutions in power systems: HVDC, FACTS, and artificial intelligence*. Vol. 52. John Wiley & Sons, 2016.

[11] Damien Ernst, Mevludin Glavic, and Louis Wehenkel. “Power systems stability control: reinforcement learning framework”. In: *IEEE transactions on power systems* 19.1 (2004), pp. 427–435.

[12] Roland Ewald, Jan Himmelspach, and Adelinde M Uhrmacher. “An algorithm selection approach for simulation systems”. In: *Principles of Advanced and Distributed Simulation, 2008. PADS’08. 22nd Workshop on*. IEEE. 2008, pp. 91–98.

[13] Djalma M Falcão. “High performance computing in power system applications”. In: *International Conference on Vector and Parallel Processing*. Springer. 1996, pp. 1–23.

[14] Emily B Fisher, Richard P O’Neill, and Michael C Ferris. “Optimal transmission switching”. In: *IEEE Transactions on Power Systems* 23.3 (2008), pp. 1346–1355.

[15] Michael K Fleming and Garrison W Cottrell. “Categorization of faces using unsupervised feature extraction”. In: *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*. IEEE. 1990, pp. 65–70.

[16] Stephen Frank and Steffen Rebennack. “An introduction to optimal power flow: Theory, formulation, and examples”. In: *IIE Transactions* 48.12 (2016), pp. 1172–1197.

[17] J David Fuller, Raynier Ramasra, and Amanda Cha. “Fast heuristics for transmission-line switching”. In: *IEEE Transactions on Power Systems* 27.3 (2012), pp. 1377–1386.

[18] Hans Glavitsch. “Switching as means of control in the power system”. In: *International Journal of Electrical Power & Energy Systems* 7.2 (1985), pp. 92–100.

[19] Kory W Hedman, Shmuel S Oren, and Richard P O’Neill. “A review of transmission switching and network topology optimization”. In: *Power and Energy Society General Meeting, 2011 IEEE*. IEEE. 2011, pp. 1–7.

[20] Kory W Hedman et al. “Co-optimization of generation unit commitment and transmission switching with N-1 reliability”. In: *IEEE Transactions on Power Systems* 25.2 (2010), pp. 1052–1063.

[21] Kory W Hedman et al. “Optimal transmission switching: sensitivity analysis and extensions”. In: *IEEE Transactions on Power Systems* 23.3 (2008), pp. 1469–1479.

[22] Kory W Hedman et al. “Optimal transmission switching with contingency analysis”. In: *IEEE Transactions on Power Systems* 24.3 (2009), pp. 1577–1586.

[23] California ISO. *Our commitment*. 2008. URL: `http://www.caiso.com/about/Pages/OurCommitment/default.aspx` (visited on 11/20/2018).

[24] *ISO New England Outlook: Smart Grid is About Consumers*. 2009. URL: `http://www.iso-ne.com/nwsiss/nwltrs/outlook/2009/outlook_may_2009_final.pdf` (visited on 11/20/2018).

[25] Mojdeh A Khorsand and Kory W Hedman. “Day-ahead corrective transmission topology control”. In: *PES General Meeting— Conference & Exposition, 2014 IEEE*. IEEE. 2014, pp. 1–5.

[26] Balho H Kim and Ross Baldick. “Coarse-grained distributed optimal power flow”. In: *IEEE Transactions on Power Systems* 12.2 (1997), pp. 932–939.

[27] James E King, Samuel CE Jupe, and Philip C Taylor. "Network state-based algorithm selection for power flow management using machine learning". In: *IEEE Transactions on Power Systems* 30.5 (2015), pp. 2657–2664.

[28] Hans-Jürgen Koglin and H Muller. "Overload reduction through corrective switching actions". In: *Int. Conf. Power Sys. Monitor. & Contr.* Vol. 24. 26. 1980.

[29] Akshay S Korad and Kory W Hedman. "Enhancement of do-not-exceed limits with robust corrective topology control". In: *IEEE Transactions on Power Systems* 31.3 (2016), pp. 1889–1899.

[30] Akshay S Korad and Kory W Hedman. "Robust corrective topology control for system reliability". In: *IEEE Transactions on Power Systems* 28.4 (2013), pp. 4042–4051.

[31] Akshay S Korad and Kory W Hedman. "Zonal do-not-exceed limits with robust corrective topology control". In: *Electric Power Systems Research* 129 (2015), pp. 235–242.

[32] Yann LeCun et al. "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4 (1989), pp. 541–551.

[33] Elham B Makram, Katherine P Thorton, and Homer E Brown. "Selection of lines to be switched to eliminate overloaded lines using a Z-matrix method". In: *IEEE Transactions on Power Systems* 4.2 (1989), pp. 653–661.

[34] Abdulhalem A Mazi, Bruce F Wollenberg, and Morten H Hesse. "Corrective control of power system flows by line and bus-bar switching". In: *IEEE Transactions on Power Systems* 1.3 (1986), pp. 258–264.

[35] A Monticelli, MVF Pereira, and S Granville. "Security-constrained optimal power flow with post-contingency corrective rescheduling". In: *IEEE Transactions on Power Systems* 2.1 (1987), pp. 175–180.

[36] Anthony Papavasiliou et al. "An application of high performance computing to transmission switching". In: *IREP bulk power system dynamics and control symposium, Rethymnon, Greece*. 2013.

[37] *PJM Manual 03: Transmission Operations*. 2018. URL: `https://www.pjm.com/-/media/documents/manuals/m03.ashx` (visited on 11/20/2018).

[38] *PJM State of the Market*. 2017. URL: `http://www.monitoringanalytics.com/reports/PJM_State_of_the_Market/2017.shtml` (visited on 11/20/2018).

[39] John R Rice. "The algorithm selection problem". In: *Advances in computers*. Vol. 15. Elsevier, 1976, pp. 65–118.

[40] Christopher K Riesbeck and Roger C Schank. *Inside case-based reasoning*. Psychology Press, 2013.

[41] Jacqueline G Rolim and Luiz Jairo B Machado. "A study of the use of corrective switching in transmission systems". In: *IEEE Transactions on Power Systems* 14.1 (1999), pp. 336–341.

[42] Pablo A Ruiz et al. "Reduced MIP formulation for transmission topology control". In: *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE. 2012, pp. 1073–1079.

[43] Gilbert Schnyder and Hans Glavitsch. "Security enhancement using an optimal switching power flow". In: *IEEE Transactions on Power Systems* 5.2 (1990), pp. 674–681.

[44] Wei Shao and Vijay Vittal. "BIP-based OPF for line and bus-bar switching to relieve overloads and voltage violations". In: *Power Systems Conference and Exposition, 2006. PSCE'06. 2006 IEEE PES*. IEEE. 2006, pp. 2090–2095.

[45] Wei Shao and Vijay Vittal. "Corrective switching algorithm for relieving overloads and voltage violations". In: *IEEE Transactions on Power Systems* 20.4 (2005), pp. 1877–1885.

[46] Jiaying Shi and Shmuel S Oren. "Stochastic unit commitment with topology control recourse for power systems with large-scale renewable integration". In: *IEEE Transactions on Power Systems* 33.3 (2018), pp. 3315–3324.

[47] Kate A Smith-Miles. "Towards insightful algorithm selection for optimisation using meta-learning concepts". In: *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE. 2008, pp. 4118–4124.

[48] *Special Protection Systems (SPS) / Remedial Action Schemes (RAS): Assessment of Definition, Regional Practices, and Application of Related Standards*. 2012. URL: `https://www.nerc.com/docs/pc/Agenda%203.7_SPS%20Assessment%20Report_20120910.pdf` (visited on 11/20/2018).

[49] MJ Teixeira et al. "Developing concurrent processing applications to power system planning and operations". In: *IEEE Transactions on Power Systems* 5.2 (1990), pp. 659–664.

[50] *THE AUTONOMOUS GRID: Machine Learning and IoT for Utilities*. 2016. URL: `http://www.mcrockcapital.com/uploads/1/0/9/6/10961847/autonomous-grid-machine-learning-iot-utilities-may_2016.pdf` (visited on 11/20/2018).

[51] *The USA Energy Policy Act of 2005 (Section 1223), PUBLIC LAW 109-58?AUG. 8, 2005*. 2005. URL: `https://www.gpo.gov/fdsys/pkg/PLAW-109publ58/pdf/PLAW-109publ58.pdf` (visited on 11/20/2018).

[52] Shalini P Vajjhala and Paul S Fischbeck. "Quantifying siting difficulty: A case study of US transmission line siting". In: *Energy Policy* 35.1 (2007), pp. 650–671.

[53] RAM Van Amerongen and HP Van Meeteren. "Security control by real power rescheduling network switching and load shedding". In: *Proc. 1980 CTGRE Report 32-02* (1980).

[54] Joaquin Vanschoren. "Understanding machine learning performance with experiment databases". In: *lirias. kuleuven. be, no. May* (2010).

[55] Richard Vuduc, James W Demmel, and Jeff A Bilmes. "Statistical models for empirical search-based performance tuning". In: *The International Journal of High Performance Computing Applications* 18.1 (2004), pp. 65–94.

[56] Alexander Waibel et al. "Phoneme recognition using time-delay neural networks". In: *Readings in speech recognition*. Elsevier, 1990, pp. 393–404.

[57] Louis Wehenkel. "Machine learning approaches to power-system security assessment". In: *IEEE Expert* 12.5 (1997), pp. 60–72.

[58] Joseph N Wrubel et al. "Practical experience with corrective switching algorithm for on-line applications". In: *Power Industry Computer Application Conference, 1995. Conference Proceedings., 1995 IEEE*. IEEE. 1995, pp. 365–371.

[59] Le Xie, Yang Chen, and P Roshan Kumar. "Dimensionality reduction of synchrophasor data for early event detection: Linearized analysis". In: *IEEE Transactions on Power Systems* 29.6 (2014), pp. 2784–2794.