Accuracies of Optimal Transmission Switching Heuristics Based on Exact and Approximate Power Flow Equations

by

Milad Soroush

A thesis

presented to the University of Waterloo

in fulfillment of the

thesis requirement for the degree of

Master of Applied Science

in

Management Sciences

Waterloo, Ontario, Canada, 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Optimal transmission switching (OTS) enables us to remove selected transmission lines from service as a cost reduction method. A mixed integer programming (MIP) model has been proposed to solve the OTS problem based on the direct current optimal power flow (DCOPF) approximation. Previous studies indicated computational issues regarding the OTS problem and the need for a more accurate model. In order to resolve computational issues, especially in large real systems, the MIP model has been followed by some heuristics to find good, near optimal, solutions in a reasonable time. The line removal recommendations based on DCOPF approximations may result in poor choices to remove from service. We assess the quality of line removal recommendations that rely on DCOPF-based heuristics, by estimating actual cost reduction with the exact alternating current optimal power flow (ACOPF) model, using the IEEE 118-bus test system. We also define an ACOPF-based line-ranking procedure and compare the quality of its recommendations to those of a previously published DCOPF-based procedure. For the 118-bus system, the DCOPF-based line ranking produces poor quality results, especially when demand and congestion are very high, while the ACOPF-based heuristic produces very good quality recommendations for line removals, at the expense of much longer computation times. There is a need for approximations to the ACOPF that are accurate enough to produce good results for OTS heuristics, but fast enough for practical use for OTS decisions.

Acknowledgements

I would like to express my great gratitude to my supervisor, Professor David Fuller. He continuingly guided and mentored me for doing research. I wish to thank him for his constructive guidance and great support.

My sincere thanks also go to Mehrdad Pirnia for valuable insights and knowledge he has shared with me in the process of this study.

Besides, I would like to thank Mohammadreza Fakhari Moghaddam who supported me by his helpful information in the preparation of the study.

My thanks also go to Professor James Bookbinder and Professor Osman Ozaltin for their feedbacks and suggestions.

The last but most important, I would like to express the most appreciation to my dear family, especially my parents, which I cannot find words to describe it.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction[1]

1.1 Overview of power systems terminology

The power network is responsible to transfer electrical energy and can be decomposed into different levels. The highest voltage scale is called a transmission network or power grid. Each network has various elements. Transmission lines are interconnected with each other at the network nodes. A network node is referred to as a "bus" in power systems language. In addition to connecting lines, some buses are connected to generators to generate electrical power and some are connected to "loads" to serve electricity consumers [1]. The power flowing on a line is divided into two components of real power "P," which is what is purchased by consumers, and reactive power "Q," which is a necessary part of the physical description of power flows in alternating current (AC) systems. In power systems analyses, both types of power are represented by a single complex number having P as the real part and Q as the imaginary part. In an AC system, voltage levels vary sinusoidally at a frequency that is held constant throughout the system, e.g., 60 cycles per second. Each bus has a voltage magnitude "V" and a voltage phase angle "δ" which describes the position in the sine wave at time t=0 [1]. For more detailed explanations, see Chapter 3.

Each transmission line is associated with the electrical admittance "Y," a complex number. It is a measure of how easily current flows through a circuit. Electrical conductance "G" is the real

[1] The thesis is an extended version of a submitted journal paper: M. Soroush, J.D. Fuller, "Accuracies of optimal transmission switching heuristics based on DCOPF and ACOPF," submitted at *IEEE Trans. Power Syst*, Apr 2013.

part of admittance that measures the ability of an element to pass electric current. Electrical conductance is calculated from the reciprocal quantity of electrical resistance "R", which measures the line's opposition to the passage of electric current. On the other hand, susceptance "B" is the imaginary part of the admittance, which measures how easily the alternating current passes through the line. In other words, it is a property of a line, which shows how easily a circuit passes a changing current.

The power flow or load flow problem is a numerical study associated with an electric power system to find the steady-state point of it. In other words, given the demand at load buses and the supply of power generation, the problem finds bus voltage magnitudes and phase angles, and complex power in the network [2]. The *optimal* power flow (OPF) problem usually uses a power flow problem as constraints, with the objective function of minimizing the generation cost of supplying all the power demand [2].

The equations in a power flow and optimal power flow problem (see Chapter 3) are based on the properties of power grid elements such as transmission lines and generators, and Kirchhoff's laws. We can divide the equations into various sets. The network constraints are based on Kirchhoff's laws at nodes; they are called balance constraints because they require that the amount of power flowing into a node equals the amount flowing out of the node. They consist of 2n nodal equations for real and reactive power in a network of n buses [2]. Other inequalities are operational constraints such as bus voltage limits and line power flow limits, or limits on control variables such as real and reactive power at generators.

In a power flow problem, "P" and "Q" line flow variables are nonlinear functions of "V" and "δ" variables at the two ends of the line, and the problem is called an alternating current (AC) load flow problem. However, providing a good linear approximation of the dependence of "P" and "Q" variables on "V" and "δ" variables is possible, and is called a direct current (DC) load flow problem because of its mathematical similarity to the equations that describe power flows in direct current networks. To derive the linear approximation, all voltage values V are approximated to 1, in special units scaled by the target voltages at the buses, which makes the nonlinear functions much simpler. The other key part of the linear approximation is to recognize that the difference in voltage phase angles at the two ends of a line is controlled to be small enough that a linear approximation of the sine and cosine of the phase angle differences is reasonably accurate.

1.2 Introduction to Optimal Transmission Switching

The optimal transmission switching (OTS) problem was recently proposed as a cost reduction method by temporarily removing transmission lines from service [3] when there is congestion, i.e., when some lines are carrying power at the allowed limits. The OTS problem is a mixed integer program based on OPF that is a complex continuous optimization problem. The accurate OPF model is called the alternating current optimal power flow (ACOPF) model. There are different approximations regarding the OPF model and the most common approximation in the literature for OPF problems is the direct current optimal power flow (DCOPF) problem. The accuracy of the OTS is directly related to the OPF problem that generated the OTS. Accordingly, the ACOPF model is a nonlinear model that is the most accurate and the most complex OPF model. However, the DCOPF model is a linear approximation of the ACOPF model.

Previous studies in OTS method are mostly based on DCOPF approximations; however, they still indicated the long computational time of the problem [3]. They linearized the ACOPF problem in order to track cost improvement in the network, which is a reason for the reliability of the model to be in question and requiring an assessment of the approximations accuracy in OTS. This study provides an assessment of the reliability of the DCOPF-based model for the OTS problem and its corresponding heuristics. We assess the reliability of the DCOPF-based line ranking of [4] in reducing costs, by estimating the actual cost reduction with the exact ACOPF model. We also extend the heuristic line ranking idea of [4] and [5] to a ranking based on the ACOPF model, and we assess its reliability in reducing costs. Finally, we modify one heuristic of [4] to be based on the ACOPF, for the removal of several lines from service.

The remainder of the paper is organized as follows. Chapter 2 provides a review of studies on the topic. Chapter 3 presents all mathematical models required for the assessments, which includes both of the DCOPF and ACOPF models. Chapter 4 outlines the methods to rank lines for removal, including the new extension based on the ACOPF model. Tests of reliability of DCOPF and ACOPF based rankings are described in Chapter 5 while Chapter 6 presents computational results of the tests. Chapter 7 summarizes the conclusions and suggests directions for future research. Appendix-A defines all mathematical symbols. Appendices-B to D provide detailed data of the IEEE-118 bus test system. Finally, Appendices-E and F present the GAMS codes for the calculations of Chapter 6.

Chapter 2
Literature Review

Transmission line switching has been proposed and discussed in the literature as a method of controlling the grid for regular problems such as voltage level, line flow overloads, and network security. It has also been used as a way of reducing power losses of the network [3].

Recently the optimal transmission switching problem has been used as a method of cost reduction [3]. It is developed based on a mixed integer linear programming (MIP) model to minimize the cost of generation in a transmission grid, with the linear DCOPF approximation of power flows. [3], [6], and [7] reported significant cost reduction by removing from one to several lines suggested by the OTS problem. The MIP problem is generated based on using binary variables assigned to the lines of the network, which represent the lines being in service or out of service. [3] reported achieving about 25% cost saving by solving the MIP and obtaining a few line removals in the IEEE 118-bus test system. They also reported computational time issue; regarding the solving 118-bus test system, especially when seeking several line removals. Even though it is not a large system, the potential solutions make it a complex problem to provide optimal solution. [8] tested the MIP model on Independent System Operator of New England (ISONE) test cases with 6652 transmission lines. By providing feasible solutions, they reached 4%, 5.5% and 6% with respectively 1, 2 and 4 line removals without guaranteeing optimality. Computational times are respectively 2, 50 and 82 hours that is too long to be a practical solution. [8] also used a sequential method to remove lines one by one to improve the computational times for more than one line removal. They removed 20 lines and achieved a 12% percent cost saving with the solving execution time of 6.3 hours. The time was not still short enough to be a practical solution, though it's better than the MIP problem with even 2 line removals at the same time [8].

The OTS problem is established based on the direct current optimal power flow (DCOPF) problem, which is a linear program (LP) [3]. The exact ACOPF model is not commonly used for OPF problems [8] in market operations. Correspondingly, all the heuristics created for the OTS problem have been based on the DCOPF model.

The only reason for using a less accurate model is due to the computational difficulties. [7] indicated the necessity of using the ACOPF model on OTS problem; since the DCOPF model ignores many features of the network such as voltage differences and reactive power. [9] compared the previous results of DCOPF-based optimal solution for line removal with the ACOPF-based model and showed that the optimal solution of OTS based on DCOPF model are not good solutions in an exact ACOPF model.

Recently [4] introduced two fast heuristics that significantly improved computational times for solving the problem; a similar idea is presented in [5]. The heuristics rely on a ranking of lines for removal from service, based on optimal primal and dual variables of the DCOPF model, that are combined in parameters designated as α_k (one for each line k) [4]. The short total run time of the heuristic methods could make it possible for the OTS problem to be solved in a reasonable time in practice. The first heuristic in [4] sequentially solves the DCOPF, LP model, and the second one is a sequential heuristic based on the MIP model for the OTS problem [4], but with the binary variables restricted to a small set of lines for which the Alpha parameters predict good cost reductions. They tested IEEE 118-bus and IEEE 662-bus test systems and achieved about 12% cost savings for both systems. The first heuristic, the sequential LP heuristic, showed better results in practice [4]. The total run time of the heuristic methods could make it possible for the OTS problem to be solved in a reasonable time, i.e., that is a few minutes. It can be used for real-time decision making in transmission switching [4]. Accordingly, [9] used the heuristic idea for 1-line removal. Although the heuristic idea is not based on a DCOPF formulation, they used it for an ACOPF formulation and achieved good cost reductions based on it. They obtained up to 6% cost saving in IEEE 118-bus test system with different demand levels. In addition, [10] reported the need for more exact model for the heuristic method in some cases, especially when resistance loss is included in the model. In this study, we also regenerate the heuristic idea based on an AC formulation model to extend it to an ACOPF-based model.

Chapter 3
Models and Formulations

In this chapter, we present the DCOPF and ACOPF problems as two optimization models. We can create the OTS problem based on each of the models. Previous works (see [3] and [4]) used DCOPF, an LP model, to create the OTS problem as an MIP model and to devise heuristics. We presented the ACOPF model, a nonconvex nonlinear problem for use in Chapter 4 and 5, where we extend the heuristic ideas in [4] based on it.

3.1 DCOPF Formulation for OTS problem

The optimal power flow problem is often formulated based on the DCOPF formulation [8]. Recent papers on the OTS problem are also based on the DCOPF model. A typical DCOPF model is in (1) as below. A "bus" is a term for nodes (indexed by "n" or "m") that are connected by transmission lines in a transmission network. Variables in the mathematical model for OPF are P_g, Q_g, P_{knm}, Q_{knm}, V_n, and δ_n (see Appendix-A: Nomenclature). P_g is the "active" power generation amount by generator "g", which is the commodity that is sold to consumers. The "reactive" power variables, Q_g, and Q_{knm} are set to zero in DCOPF models. Voltage magnitudes V_n are controlled in real systems to be close to 1 in special, scaled units; in the DCOPF model they are fixed at the value 1. The phase of the steady state, sinusoidal voltage differs from one bus to another, as measured by the phase angle, δ_n, in radians. We use the notation P_{knm} for the power flow on line *k*, in the direction from bus *n* to bus *m*; a negative value for P_{knm} means that the flow is from m to n. Other symbols are all parameters, e.g. B_{nm} is the element nm of the susceptance series (for detailed electrical engineering explanations, see [2]). Note that models are for one short period of the network system, e.g. 1 hour.

$$\min \sum_{g} c_g P_g \tag{1a}$$

subject to :

$$\sum_{g \in G_n} P_g - \sum_{m} B_{nm}(\delta_n - \delta_m) = P_n^{dem}, \ \forall n \tag{1b}$$

$$P_g^{\min} \le P_g \le P_g^{Max}, \ \forall g \tag{1c}$$

$$P_{knm} = B_{nm}(\delta_n - \delta_m), \ \forall k \tag{1d}$$

$$P_k^{\min} \le P_{knm} \le P_k^{Max}, \ \forall k \tag{1e}$$

$$\delta^{\min} \le \delta_n - \delta_m \le \delta^{Max}, \ \forall n,m \text{ connected by a line} \tag{1f}$$

The objective function (1a) used in the DCOPF model is generation cost that is being minimized. Constraints are generated from Kirchhoff's laws. There is a power flow balance in each node represented by (1b). The summation over m in (1b) is a sum over buses m such that there exists a line k connecting buses n and m. The limit of voltage angle differences is in (1f); this is imposed because if the phase angle differences are too large, then the system becomes unstable. Constraints (1c) and (1e) represent power generation limit in each generator and power flow limit on each line respectively. The approximated power flow for each line in the system is in

(1d). Please note that (1d) and (1e) can be combined together as one constraint. It does not have an effect on the solution. Constraint (1d) is separated in order to develop an MIP model of OTS in (2). Many approximations are made to linearize the mathematical model of AC power flows on a transmission network. Therefore, there is no guarantee that using a DCOPF model for OTS heuristic line ranking would lead to the same choice of line removal as using the much more accurate ACOPF model. The OTS problem first is generated and solved in [3] based on the DCOPF model (1). Model (2) is an MIP first introduced by [3] to make the OTS problem.

$$\min \sum_{g} c_g P_g \tag{2a}$$

subject to :

$$\sum_{g \in G_n} P_g - \sum_{m} B_{nm}(\delta_n - \delta_m) = P_n^{dem}, \ \forall n \tag{2b}$$

$$P_g^{\min} \le P_g \le P_g^{Max}, \ \forall g \tag{2c}$$

$$B_{nm}(\delta_n - \delta_m) - P_{knm} + (1 - z_k)M \ge 0, \ \forall k \tag{2d}$$

$$B_{nm}(\delta_n - \delta_m) - P_{knm} - (1 - z_k)M \le 0, \ \forall k \tag{2e}$$

$$P_k^{\min} z_k \le P_{knm} \le P_k^{Max} z_k, \ \forall k \tag{2f}$$

$$\delta^{\min} \le \delta_n - \delta_m \le \delta^{Max}, \ \forall n,m \text{ connected by a line} \tag{2g}$$

$$\sum_{k} (1 - z_k) \le j. \tag{2h}$$

The formulation includes z_k as a binary variable to represent the state of a line and a new parameter *M*, which is a large positive number. The circuit breaker for the line is closed if z_k is equal to one and is open if z_k is equal to zero. Constraint (2f) forces a line to be out of the system by making the line flow of line k zero if z_k is equal to zero. Constraint (1d) breaks to two constraints (2d) and (2f) to apply of using the binary variable as mentioned; if $z_k = 0$, then (2d) and (2e) ensure that the model does not relate the voltage angles at the two ends of line k which is not in service. The maximum number of removed lines from the system, *j*, is also considered by (2h) [3].

3.2 ACOPF Formulation for OTS problem

The ACOPF formulation includes the voltage magnitude variables that were approximated to 1 in the DCOPF model. It also includes reactive power that shows a different pattern than the active power and is not considered in DCOPF models. The reactive power appears in balance and limit constraints. The constraints of the following ACOPF formulation are based on [2]. The results provided in the rest of the study are also based on model (1) in comparison to the AC model below in (3). δ_{nm} is a common shorthand symbol for $\delta_n - \delta_m$.

$$\min \sum_g c_g P_g \tag{3a}$$

subject to :

$$\sum_{g \in G_n} P_g - \sum_m V_n V_m (G_{nm} \cos \delta_{nm} + B_{nm} \sin \delta_{nm}) = P_n^{dem}, \forall n \tag{3b}$$

$$\sum_{g \in \mathcal{G}_n} Q_g - \sum_m V_n V_m (G_{nm} \sin \delta_{nm} - B_{nm} \cos \delta_{nm}) = Q_n^{dem}, \forall n \tag{3c}$$

$$P_g^{\min} \le P_g \le P_g^{Max}, \ \forall g \tag{3d}$$

$$Q_g^{\min} \le Q_g \le Q_g^{Max}, \ \forall g \tag{3e}$$

$$P^2{}_{knm} + Q^2{}_{knm} \le S_k^{2\ Max}, \forall k \tag{3f}$$

$$P_{knm} = V_n V_m (G_{nm} \cos \delta_{nm} + B_{nm} \sin \delta_{nm}) - G_{nm} V_n^2), \forall k \tag{3g}$$

$$Q_{knm} = V_n V_m (G_{nm} \sin \delta_{nm} - B_{nm} \cos \delta_{nm}) + V_n^2 (B_{nm} - b_{nm}^p), \ \forall k \tag{3h}$$

$$\delta^{\min} \le \delta_{nm} \le \delta^{Max}, \ \forall n, m \text{ connected by a line} \tag{3i}$$

$$V_n^{\min} \le V_n \le V_n^{Max}, \ \forall n \tag{3j}$$

The summation over *m* in (3b) and (3c) are sums over all buses *m* connected by a line *k* to bus *n*, and also terms for m=n. Constraints (3f), (3g) and (3h) can be written as a single constraint; this presentation clarifies the line flow in (3f). (3g) and (3h) represent active and reactive power flows. The right side expression of (3g) becomes the right side of (1d) when $V_n = 1, \forall n$ and the cosine and sine functions are approximated to first order in δ_{nm}, in an approximated DCOPF model. Constraints (3c) and (3e) consider balance and power generation regarding reactive power in the network. Constraint (3j) represents voltage magnitude limits which do not appear in DCOPF models. When the linear DCOPF is used to formulate the OTS, the computations are impractically long. Therefore, heuristics have been used for the OTS as discussed in the next chapter.

Chapter 4
Heuristic Ideas

In previous studies, [3] introduced an MIP problem to provide optimal solutions of the OTS problem. The result of the studies shows really good cost savings up to 25% for different test systems. In larger systems, the computational time of the problem was not good enough. Thus, they also performed a sequential method, at each step removing the best single line (by setting j=1 in (2h)), as a heuristic idea to reduce the computational time in search of a good solution. However, computation times were still too long for large, real systems. The sequential method removes one line in each of the iterations and finds the next best line based on the new structure of the system. The method shows good result though for very large systems it couldn't solve the computational time issue. In response to the issue of computational difficulties, two heuristic ideas were proposed by [4] leading to much faster computations and good cost reductions for standard test models. A line ranking parameter named α_k was introduced and interpreted as the key idea of the heuristics. This ranking parameter is based on the optimal primal and dual variables of the model (1), as in (5) below; π_n is the dual variable associated with the balance constraint (1b) of the DCOPF model [4]. The reformulation of model (1) that [4] used to derive the heuristic idea is as follows: Symbols in square brackets represent the dual variables of constraints.

$$\min \sum_g c_g P_g \qquad \text{(4a)}$$

subject to:

$$\sum_{g \in G_n} P_g \; - \sum_m B_{nm}(\delta_n \text{-} \delta_m) = P_n^{dem} [\pi_n], \; \forall n \tag{4b}$$

$$P_k^{\min}(1\text{-}\lambda_k) \le P_{knm} \le P_k^{Max}(1\text{-}\lambda_k) \; [\gamma_k^L, \gamma_k^R], \; \forall k \tag{4c}$$

$$P_{knm} = B_{nm}(\delta_n \text{-} \delta_m)\,(1\text{-}\lambda_k)\; [\sigma_k], \; \forall k \tag{4d}$$

$$P_g^{\min} \le P_g \le P_g^{Max}, \; \forall g \tag{4e}$$

$$\delta^{\min} \le \delta_n \text{-} \delta_m \le \delta^{Max}, \; \forall n,m \text{ connected by a line} \tag{4f}$$

$$\lambda_k = 0 \; [\alpha_k], \; \forall k \tag{4h}$$

λ_k variables in model (4) represent the fraction of line k which is removed from the system. If $\lambda_k = 0$ line k is fully in use and if $\lambda_k = 1$ line k is removed from the system [4]. (The only possible values in reality are $\lambda_k = 0$ and $\lambda_k = 1$, but λ_k is treated as a continuous variable to derive the line ranking parameters α_k.) Therefore, setting all the values to zero in (4h) means that the reformulation is equivalent to the DCOPF in model (2).

The most important part of the method is based on α_k values and the way they are interpreted and calculated in the model. The dual variables α_k associated with constraint (4h) are the key idea to the method. α_k represents the rate of change in the value of the objective function (4a) with an ε amount of change in the right side of constraint (4e); in particular, with $\varepsilon = 1$, α_k estimates the change in optimal cost if line k is removed, i.e., $\lambda_k = \varepsilon = 1$. Alpha values are calculated from the Karush-Kuhn-Tucker (KKT) conditions of model (4). The α_k values can be calculated from the optimal dual and primal variables of the standard DCOPF model (2) [4].

Equation (5) below is used to calculate Alpha values for DC model. π_n is the dual variable associated with balance constraint (1b) of DCOPF model [4]; π_n is normally interpreted as the market price at bus *n*, when the DCOPF model is used to run electricity market. Considering equation (5) for line *k*, π_n is associated with the "from" bus and π_m is associated with the "to" bus in line *knm*. In the right side of the equation, the first factor shows the difference between the "to" and "from" nodes prices. The second factor is the power flow on the line.

$$\alpha_k = (\pi_m - \pi_n) P_{knm}, \forall k \tag{5}$$

The idea in [4] suggested that the most negative α_k is likely to show the most promising line removal for the largest cost reduction. Thus, solving the LP model (1) results in having all lines ranked from most to least promising. Calculations with two test systems showed a good agreement between the α_k predictors and actual good line removals of the DCOPF model [4]. However, by solving the MIP model including loss approximations, [10] reported poor choices by the predictors and proposed the need for a more accurate model of power flows.

4.1 Extending the fast heuristic methods based on AC model

In this study, we generate new α_k values derived from the ACOPF model. Following the ideas in [4], but replacing the DCOPF with the ACOPF, we first reformulate the ACOPF (3) as in model (6) below. To ease the derivation of the new α_k expressions, we define the following functions that give the real and reactive power flows on line *k*, as measured at bus *n* in the direction towards bus *m*, in terms of voltage magnitudes and voltage angles:

$$P_{knm}(V,\delta) = (V_n V_m (G_{nm}\cos\delta_{nm} + B_{nm}\sin\delta_{nm}) - G_{nm}V_n^2) \ ,\forall k$$

$$Q_{knm}(V,\delta) = (V_n V_m (G_{nm}\sin\delta_{nm} - B_{nm}\cos\delta_{nm}) + V_n^2(B_{nm} - b_{nm}^p)), \forall k$$

As in [4], the λ_k in the following model appears as the fraction of line *k* which is removed from service, leaving the fraction $(1-\lambda_k)$ in service, which is modeled as changing the values of the line parameters susceptance B_{nm}, conductance G_{nm}, and line flow limit S_k^{Max} to $(1-\lambda_k)$ times of their previous amount. The reformulation (6) of the ACOPF (3) includes the requirement $\lambda_k = 0$, i.e., the line is fully in service. Square brackets indicate the dual variable associated with the corresponding constraint.

$$\min_i \sum_i c_i P_i \tag{6a}$$

subject to: (3d), (3e), (3i), (3j) and

$$\sum_{g\in G_n} P_g - \sum_m P_{knm}(V,\delta)\times(1-\lambda_k) = P_n^{dem} \ [\pi_n^P], \ \forall n \tag{6b}$$

$$\sum_{g\in G_n} Q_g - \sum_m Q_{knm}(V,\delta)\times(1-\lambda_k) = Q_n^{dem} \ [\pi_n^Q], \ \forall n \tag{6c}$$

$$(P_{knm}(V,\delta)\times\ (1-\lambda_k))^{\,2}+(Q_{knm}(V,\delta)\times\ (1-\lambda_k))^{\,2}\le(S_k^{Max}\times\ (1-\lambda_k))^2\quad[\gamma_k],\ \forall k \tag{6d}$$

$$\lambda_k=0\ \ [\alpha_k],\forall k \tag{6e}$$

The sums over *m* in (6b) and (6c) are for all $m\neq n$ such that there is a line *k* connecting *n* to *m*. *The* Alpha values are the key to the heuristic ideas. They are calculated from the solution of the ACOPF model in the following way. First, note that an optimal solution to (3) is also optimal in (6), and also that the Karush-Kuhn-Tucker (KKT) conditions of (6) at its optimal solution are satisfied by the optimal primal and dual variables of (3), together with $\lambda_k=0$ and α_k computed as follows. The KKT condition for the derivative of the Lagrangian of (6) with respect to λ_k, and evaluated at $\lambda_k=0$ is

$$\begin{aligned}&\alpha_k+\ \gamma_k\times(-2P_{knm}^2(V,\delta)-2Q_{knm}^2(V,\delta)+2S_k^{Max})\ +\pi_n^P\ P_{knm}(V,\delta)+\pi_m^P\ P_{kmn}(V,\delta)\\&+\pi_n^Q\ Q_{knm}(V,\delta)+\pi_m^Q\ Q_{kmn}(V,\delta)\ =0\quad[\gamma_k],\ \forall k\end{aligned} \tag{7}$$

The second term of (7) simplifies to

$$2\gamma_k\times(S_k^{Max}-P_{knm}^2(V,\delta)-Q_{knm}^2(V,\delta))\,,\forall k$$

Using the complementary slackness condition of line limit constraint (6d), either $S_k^{Max}-P_{knm}^2(V,\delta)-Q_{knm}^2(V,\delta)=0$ or γ_k =0. Thus, the condition indicates

$$\gamma_k\times(S_k^{Max}-P_{knm}^2(V,\delta)-Q_{knm}^2(V,\delta))=0\,,\forall k$$

which simplifies equation (7) to

$$\alpha_k + \pi_n^P \; P_{knm}(V,\delta) + \pi_m^P \; P_{kmn}(V,\delta) + \pi_n^Q \; Q_{knm}(V,\delta) + \pi_m^Q \; Q_{kmn}(V,\delta) \; = 0, \; \forall k \qquad (8)$$

4.2 Alpha values based on DC model

$P_{knm}(V,\delta)$ function is approximated by the P_{knm} variable for a DC model in constraint (1d). Considering the DC approximation, power and reactive power flow functions of a line in the DC model are as follow:

$$P_{kmn}(V,\delta) = -P_{knm}(V,\delta) = -B_{nm}(\delta_n - \delta_m), \forall k \qquad (10)$$

$$Q_{kmn}(V,\delta) = -Q_{knm}(V,\delta) = 0, \forall k \qquad (11)$$

Applying (10) and (11) in equation (8) leads to the same equation as in (5) in [4] to calculate Alpha values for a DC model, which leads to:

$$\alpha_k = (\pi_m^P - \pi_n^P) \times P_{knm} \;, \forall k \qquad (12)$$

4.3 Alpha values based on AC model

We can use equation (8) to calculate α_k values from a solution of the ACOPF model:

$$\alpha_k = -\pi_n^P \; P_{knm}(V,\delta) - \pi_m^P \; P_{kmn}(V,\delta) - \pi_n^Q \; Q_{knm}(V,\delta) - \pi_m^Q \; Q_{kmn}(V,\delta), \; \forall k$$

This formula for α_k can be related to the DC-based calculation in (5) or (12) by using the relation among power losses P_k^{loss} and Q_k^{loss}, and power flows at the two end of a line:

$$P_{kmn}(V,\delta) = -P_{knm}(V,\delta) + P_k^{loss} \qquad (13)$$

$$Q_{kmn}(V,\delta) = -Q_{knm}(V,\delta) + Q_k^{loss} \qquad (14)$$

We apply (13) and (14) in equation (8), which leads to a final formula for calculating α_k values as follows:

$$\begin{aligned}\alpha_k &= (\pi_m^P - \pi_n^P) \times P_{knm}(V,\delta) - P_k^{loss}\pi_m^P \\ &+ (\pi_m^Q - \pi_n^Q) \times Q_{knm}(V,\delta) - Q_k^{loss}\pi_m^Q\end{aligned} \qquad (15)$$

The first term of (15) is the same as the expression generated for the DCOPF model. In addition, there are three other terms, one of which is an expression associated with the reactive power similar to that for the active power. The other terms are associated with losses of active and reactive power in lines and are also not considered in the DCOPF problem.

Chapter 5
Tests of DCOPF- and ACOPF-based Line Rankings

In this chapter, we describe the methods to assess the reliability of recommendations for line removal that are based on (5) after solving the DCOPF (1), and recommendations that are based on (15) after solving the ACOPF (3). We also describe the modification to one of the DCOPF-based heuristics of [4], to substitute ACOPF calculations in the heuristic.

5.1 The 118-bus test system and software used

We used the IEEE 118-bus test system [11] for the studies described below, with additional data from [12]. Small variations in the data are explained in [4] and [3]. The tests are done with three different load levels obtained by multiplying load levels in [12] by 0.8, 1.0 and 1.2 to create low, high and very high demand respectively. The value $\delta^{Max} = -\delta^{\min}$ = 0.6 radians is used in (1f) and (3i) to limit voltage angle differences.

We used CPLEX to solve the DCOPF LP (1), and CONOPT to solve the ACOPF (3), in the GAMS environment on a Windows server. We chose CONOPT because in preliminary testing, it gave the same or better solutions than other solvers such as MINOS, KNITRO and COIN-ipopt. Note that CONOPT is not guaranteed to find a global optimal solution, because the ACOPF (3) is a non-convex nonlinear program. However, in preliminary testing at the three demand levels, we tried different initial points and different solvers with various options, and found that the solution found by CONOPT with standard option settings was always the best, which makes us reasonably confident that we are getting global optimal solutions.

5.2 Determination of exact line ranking using ACOPF

The following procedure was carried out for each of the three demand levels to determine the exact ranking of lines to remove. Model (3) was first solved with all lines in service, and the optimal cost was noted. Then model (3) was solved repeatedly again, each time with one of the lines removed from service; for each single line removed, the optimal cost was compared with the optimal cost when all lines are in service, and the cost reduction was expressed as a percentage of the cost with all lines in service. The largest cost reduction was assigned a rank of 1, the second largest was given a rank of 2, etc.

This exact ranking of lines is compared with rankings based on α_k calculated from the DCOPF and ACOPF solutions, as described in the next section.

5.3 Assessment of accuracy of line rankings based on α_k from DCOPF and ACOPF

For each of the three demand levels, the DCOPF (1) was solved with all lines in service, and α_k was calculated from (5) for each line *k*. Since $-\alpha_k$ can be interpreted as an estimate of the reduction in optimal cost if line *k* is removed from service (see [4] and the discussion in Chapter 4), we assigned a rank of 1 to the line with the largest $-\alpha_k$, a rank of 2 to the line with second largest $-\alpha_k$, etc. We compared this ranking to the exact ranking described previously, by observing which of the top ten exact rankings were among the top ten ranked by $-\alpha_k$.

Similarly, for each of the three demand levels, the ACOPF (3) was solved and α_k was calculated from (15) for each line *k*. Rankings were based on the ACOPF-derived $-\alpha_k$ and compared to the exact rankings in the same manner as for the DCOPF-based rankings.

5.4 Modification of DCOPF-based heuristic to use ACOPF

We also tested the sequential LP heuristic in [4] but with the ACOPF replacing the DCOPF. Here, we provide a brief description; details may be found in [4]. The heuristic has three control parameters, integers *L, I* and *m*. It removes one line at each iteration, up to the maximum number of lines to remove, *L* (or it stops when no cost-reducing line removals can be found). The first line is removed by: solving the ACOPF (3) with all lines in service; ranking the lines using α_k calculated from the ACOPF solution; testing up to *I* top-ranked line removals by re-solving the ACOPF without each such line, until *m* candidate lines have been found that reduce the cost, or *I* have been tested; and removing the candidate line which produces the largest cost reduction. Later iterations follow the same procedure, but starting from the current configuration of the system, with lines removed from service as decided at previous iterations.

Chapter 6
Results of Tests

In this chapter, we present the result of line rankings of single line removals. We also examine the performances of Alpha values rankings based on the DCOPF and ACOPF models in providing reliable predictions. Finally, we run the ACOPF-based heuristic to present the result of running the heuristic in different demand levels. In addition, we also compare the cost savings opportunities and computational times of the heuristic based on the DCOPF and ACOPF models.

6.1 Exact ranking of lines using ACOPF

In this section, we present some of the results of single line removals based on the ACOPF model for the three different demand levels.

TABLE I
COST REDUCTIONS IN LOW DEMAND

Lines Rankings	Removed Line	Cost Reduction (%)
1	98	1.91
2	95	1.46
3	100	0.98
4	74	0.94
5	70	0.58
6	69	0.46
7	79	0.37
8	78	0.30
9	81	0.16
10	67	0.10

By comparing ACOPF and DCOPF results, recently [9] reported inconsistencies between the best line removals from DCOPF model comparing with the ACOPF model and vice versa. In this chapter, we examine the results of single line removals estimated based on the ACOPF model in different demand levels.

First, we operate single line removals and report the best results in Tables I to III for different demand levels. High demand level has regular real power demands in the dataset; however, low and very high demand levels have regular demands multiplied by 0.8 and 1.2 respectively. The high demand level represents a high congestion situation and a very high demand level represents a very high congested network that is a critical situation to be assessed in OTS problem.

TABLE II
COST REDUCTIONS IN HIGH DEMAND

Lines Rankings	Removed Line	Cost Reduction (%)
1	74	2.60
2	98	2.11
3	95	1.79
4	76	0.17
5	61	0.15
6	117	0.13
7	77	0.09
8	41	0.04
9	42	0.04
10	59	0.04

TABLE III
COST REDUCTIONS IN VERY HIGH DEMAND

Lines Rankings	Removed Line	Cost Reduction (%)
1	117	16.58
2	123	8.11
3	74	4.06
4	103	3.68
5	41	2.97
6	121	2.92
7	122	2.92
8	102	2.17
9	101	2.07
10	27	1.61

Figure 1 represents the cost savings that can be achieved by removing one single line for different demand levels. Table I to III show the rankings of the lines based on the percentage of cost savings. Table I and Table II show achieving up to almost 2.5 percent cost reduction in low and high demands by removing just a single line from the system. On the other hand, Table III shows significant cost savings for very high demand level. The possibilities of a significant cost saving is notably higher in high congestion. For example, the best line to remove, under very high demand, would reduce generation cost by over 16%, while the second best line would reduce the cost by about 8%. Therefore, the results show some large cost savings opportunities with the higher congestions under very high demand, but much smaller opportunities at the lower demand levels.

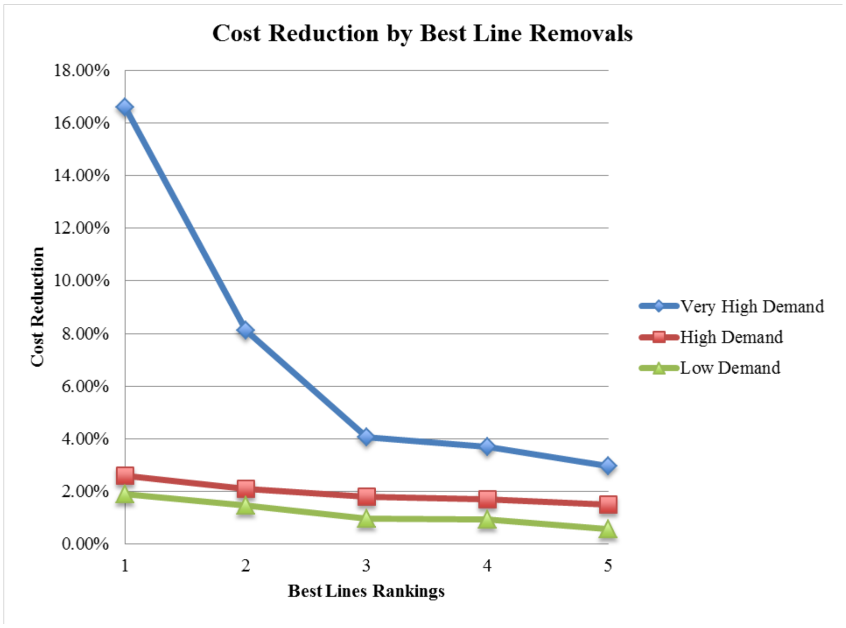

FIGURE 1: BEST LINE RANKINGS FOR DIFFERENT LOAD LEVELS BY 1-LINE REMOVAL

6.2 Accuracies of line rankings based on α_k from ACOPF and DCOPF

Two kinds of comparison are presented in this section. First, we compare the consistency of ranking lines by α_k based on the DCOPF versus the ACOPF models. Second, we note their accuracy in predicting good lines to remove in comparison with the results of the exact cost reduction estimates. The ACOPF model generated Alpha values by the expression we created in previous chapter including all terms in equation (15). The DCOPF model also generates Alpha values by using equation (12) which is the same as the equation used for a DC-based model in [4].

TABLE IV
PREDICTIONS IN LOW DEMAND BY ACOPF ALPHAS

Recommended Line Rankings	Line	ACOPF Alpha Value
1	98	-70.38
2	74	-44.37
3	103	-24.81
4	70	-18.05
5	100	-10.27
6	79	-9.78
7	66	-8.10
8	69	-6.67
9	78	-5.66
10	67	-3.70

Table IV shows a ranking of the lines in low demand level based on the predictions of Alpha values from (15) based on the ACOPF model. The most negative Alpha values are considered to be the best recommendations for the line switching method. Therefore, we ranked ten best lines as long as their corresponding Alpha values are negative.

Tables IV to IX represent the result of calculating new Alpha values for ACOPF and comparing the performances of ACOPF and DCOPF model predictions based on their line rankings. Tables IV, VI, and VIII represent the line ranking based on Alpha Values predictions. We also created same columns for DCOPF-based model to compare the result of both models with exact line rankings. Accordingly, Tables V, VII, and IX show a shaded box when one of the top ten lines (exact ranking) appears among the top ten lines as ranked by α_k from the ACOPF and the

DCOPF. In Table V, for low demand, both α_k rankings perform well. They predict eight (ACOPF) and seven (DCOPF) of the top ten cost reductions. The result for high demand in Table VII shows a poor performance for both α_k rankings – only two of the top ten are predicted by each, and both miss the number one ranked line; however, the ACOPF ranking is slightly better than the DCOPF ranking because the ACOPF ranking includes the sixth best line, but the DCOPF ranking includes the eighth best.

TABLE V
COMPARING THE PREDICTIONS IN LOW DEMAND

Lines Rankings	Cost Reduction (%)	ACOPF	DCOPF
1	1.91	▨	▨
2	1.46		
3	0.98	▨	▨
4	0.94	▨	▨
5	0.58	▨	▨
6	0.46	▨	▨
7	0.37	▨	▨
8	0.3	▨	
9	0.16		
10	0.1	▨	▨

▨ The line is predicted by Alpha values generated of the model
☐ The line is NOT predicted by Alpha values generated of the model

TABLE VI

PREDICTIONS IN HIGH DEMAND BY ACOPF ALPHAS

Recommended Line Rankings	Line	ACOPF Alpha Value
1	98	-43.27
2	123	-38.45
3	103	-34.26
4	31	-8.57
5	122	-7.36
6	117	-4.91
7	100	-3.99
8	121	-3.77
9	73	-3.40
10	75	-3.22

TABLE VII

COMPARING THE PREDICTIONS IN HIGH DEMAND

Lines Rankings	Cost Reduction (%)	ACOPF	DCOPF
1	2.6		
2	2.11		
3	1.79		
4	0.17		
5	0.15		
6	0.13		
7	0.09		
8	0.04		
9	0.04		
10	0.04		

The line is predicted by Alpha values generated of the model

The line is NOT predicted by Alpha values generated of the model

Table VIII shows the predictions in very high congestion situation and the magnitude of the values is significantly bigger than the Alpha values in the other congestion levels (high or low demand situations).

TABLE VIII
PREDICTIONS IN VERY HIGH DEMAND BY ACOPF ALPHAS

Recommended Line Rankings	Line	ACOPF Alpha Value
1	123	-5782.68
2	103	-2572.28
3	121	-1830.08
4	117	-836.62
5	101	-635.91
6	102	-611.39
7	126	-291.58
8	54	-289.51
9	41	-264.93
10	113	-198.51

Table IX shows very poor accuracy of the DCOPF-based α_k ranking in the very high demand case – only one of the top ten lines is among the top ten predicted. In contrast, the ACOPF-based α_k ranking has seven out of the top ten lines among its top ten predictions. If this is generally true of most systems, then it is crucial to use the more accurate ACOPF-based α_k under highly congested conditions, which is when the largest cost reductions are expected for OTS.

Therefore, for the first comparison, we note that the α_k rankings by the ACOPF and DCOPF are very similar under low and high demand conditions, but under very high demand, the ACOPF-based ranking is much better than the DCOPF-based ranking. If this observation is true of most systems, then it suggests that the DCOPF, which is much faster to solve than the ACOPF, is sufficient for line ranking under moderate demand and congestion, but for highly congested systems, rankings should be based on the ACOPF. In other words, ACOPF Alpha predictors can really make reliable recommendation in very high demand levels.

TABLE IX
COMPARING THE PREDICTIONS IN VERY HIGH DEMAND

Lines Rankings	Cost Reduction (%)	ACOPF	DCOPF
1	16.58	▨	
2	8.11	▨	
3	4.06		
4	3.68	▨	▨
5	2.97	▨	
6	2.92	▨	
7	2.92		
8	2.17	▨	
9	2.07	▨	
10	1.61		

▨ The line is predicted by Alpha values generated of the model
□ The line is NOT predicted by Alpha values generated of the model

To summarize the result of Tables IV to IX, ACOPF Alphas are always better predictors for line switching especially with very high load level.

We also compared the performance of ACOPF-based α_k rankings for which α_k is calculated using just the first term of (15), i.e., $(\pi_m^P - \pi_n^P) \times P_{knm}(V, \delta)$. For different demand levels, this variant has very similar results to those displayed in the ACOPF columns of Tables I to Table III, which used the whole of expression (15).

6.3 Heuristic using ACOPF-based α_k

For the ACOPF-based heuristic described in the previous chapter, we set the maximum number of lines to remove at L=4, the maximum number of line removals to test per iteration at I=12, and the maximum number of candidate lines per iteration at m=6. The value for m differs from that used in [4] (m=2) because after some experimentation, we found that the larger candidate list produced noticeably larger cost reductions. Considering 6, instead of 2, lines that produce cost reductions has a better chance of finding the larger cost reductions. Furthermore, if a good line removal is missed at the first iteration, due to a small value of m, there is no guarantee that the heuristic is capable of finding a similar suggestion in the next iterations since the topology of the network is changed.

The heuristic in [4] is constantly using DCOPF to estimate the cost reduction of each recommendation. It also uses the recommendations of Alpha values generated by DCOPF, which is not reliable as shown in the previous analysis of the results. Therefore, we expect that the heuristic based on the ACOPF model to be able to perform better than the DCOPF-based heuristics of [4].

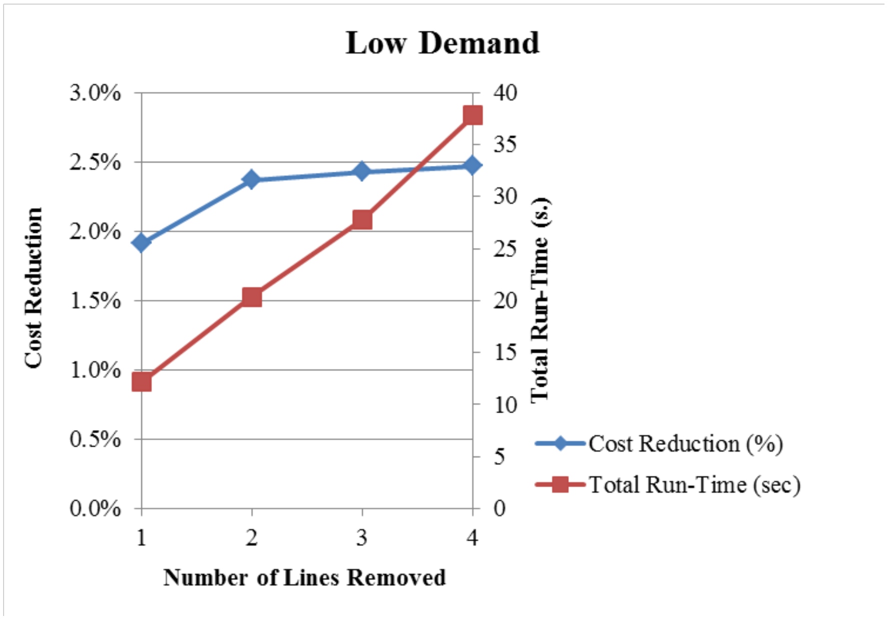

FIGURE 2: COST REDUCTION OF ACOPF-BASED HEURISTIC AND ITS COMPUTATIONAL TIME IN SECONDS BY DIFFERENT NUMBER OF LINE REMOVALS IN LOW DEMAND LEVEL.

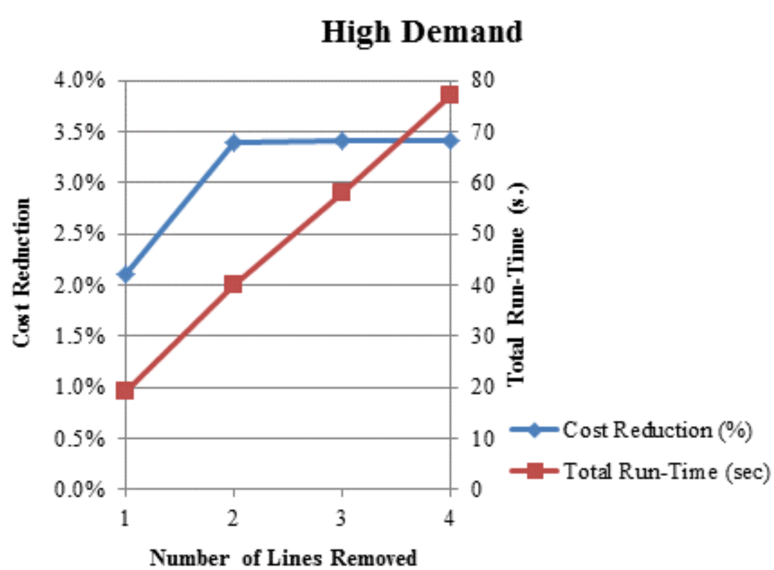

FIGURE 3: COST REDUCTION OF ACOPF-BASED HEURISTIC AND ITS COMPUTATIONAL TIME IN SECONDS BY DIFFERENT NUMBER OF LINE REMOVALS IN HIGH DEMAND LEVEL.

Figures 2 and 3 show about 2.5% and 3.5% cumulative cost reductions with four line removals, for low and high demand level respectively, using the ACOPF-based heuristic. In both cases, most of the cost reduction is found with the first two line removals.

Figure 4 shows a very high cost saving – over 16% -- for the very high demand case, which is consistent with Table IX results, i.e., that the ACOPF-based α_k ranking finds the best line to remove at the first iteration, among the ten best lines ranked by α_k. The heuristic stopped at iteration 2 because no cost-reducing lines were found among the *I*=12 that were tested, with the new topology of the network that was created by the line removal at the first iteration. A variant of the heuristic, discussed in the next subsection, avoids this problem, and finds further cost-reducing line removals after the first.

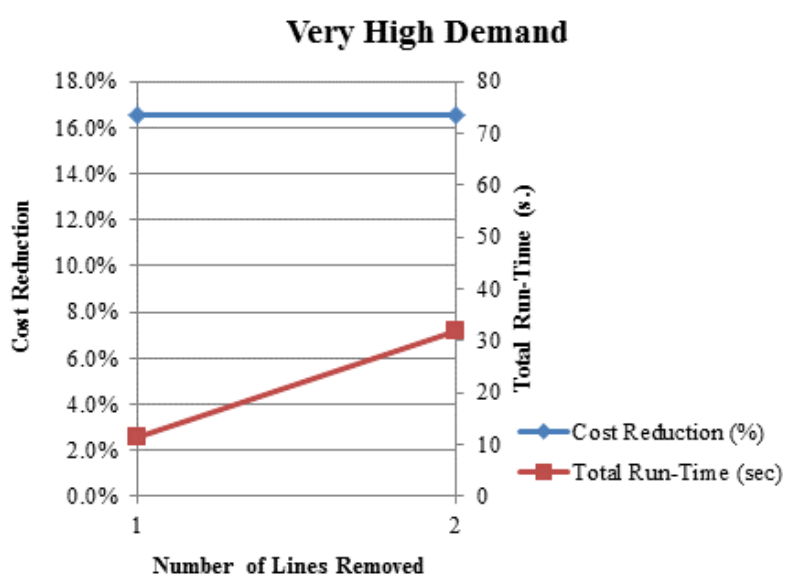

FIGURE 4: COST REDUCTION OF ACOPF-BASED HEURISTIC AND ITS COMPUTATIONAL TIME IN SECONDS BY DIFFERENT NUMBER OF LINE REMOVALS IN VERY HIGH DEMAND LEVEL.

We also compared the total run time and cost saving of the heuristic based on DCOPF and ACOPF. We ran the DCOPF with the same values of *L*, *I*, and *m* as in the first runs for the ACOPF above. The ACOPF computational time is much higher, but still in a reasonable amount of time which is less than a few minutes. Table X shows the difference between computations times based on different models and load levels. The total run time for low and high demand levels are considerably higher, for the ACOPF-based heuristic, than the very high demand level because of running more number of iterations. In general, the differences of run times between DCOPF-based and ACOPF-based heuristics are notable. We expect to have a considerable difference in very large systems, which is a critical issue to consider for practical implementations.

We compared the actual cost improvements of the heuristic based on DCOPF with cost improvements represented in Figure 2 to 4. Cost improvements in high congestion are about the same, but the ACOPF based heuristic performed better for low and very high demand levels. After a detailed analysis, we found that the DCOPF heuristic could not predict the best lines to remove in very high congestion in any of the iterations. It also performed poorly by suggesting a line that actually increases cost in an iteration in low demand level. Therefore, it actually increases the cost after the cumulative cost changes after 4 iterations. As also reported in [9], the DCOPF model doesn't show consistent results, in which a line removal that is actually increasing cost showed as a cost saving option.

TABLE X
TOTAL RUN TIME AND COST SAVING OF HEURISTIC COMPARISON

Demand Levels	DCOPF Run-Time	DCOPF-based Actual Cost Saving	ACOPF Run-Time	ACOPF-based Cost Saving
Low	7.83 sec	-4.51%	67.28 sec	2.42%
High	8.03 sec	3.03%	77.02 sec	3.65%
Very High	7.11 sec	4.27%	31.72 sec	16.58%

Figures 2 to 4 show that computing times for different demand levels increase almost linearly with respect to the number of iterations, which is due to approximately the same number of ACOPF calculations done per iteration. Yet, most cost improvements are for the first two line removals. Thus, if computing time is limited for the practical reason of needing to make a decision in time to implement it, then not much is lost by limiting the iterations to two, for the system tested.

6.4 Variation in Calculating Alpha Values

We also calculated α_k from a modified version of (15) using just the first term, i.e., $\alpha_k = (\pi_m^P - \pi_n^P) \times P_{knm}(V, \delta)$, and ran the heuristic based on it. The performance of the method in the first iteration, for each demand level, is almost the same as reported above; however, this variation shows a better performance in cost savings for very high demand. Whereas, the heuristic based on (15) finds only one line to remove, the variation finds four lines to remove, for greater cumulative total cost savings, about 20%.; see Figure 5.

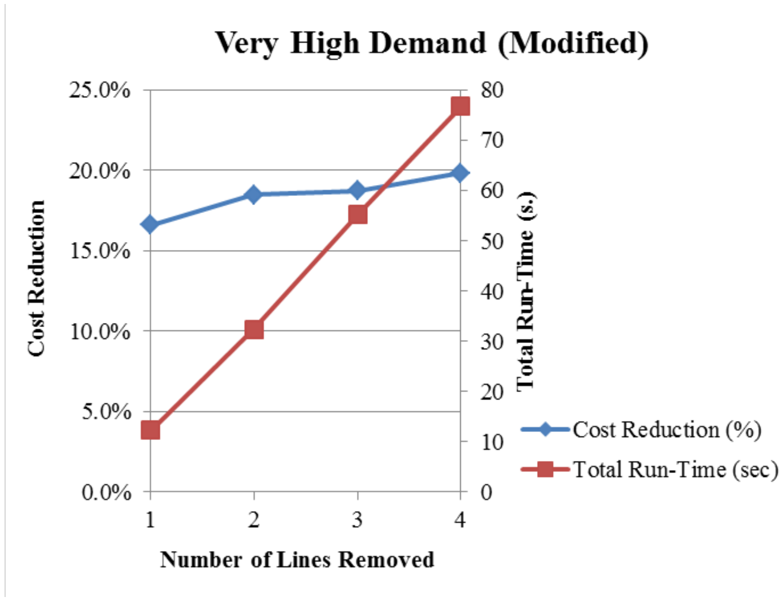

FIGURE 5: COST REDUCTION OF PERFORMING HEURISTIC AND ITS COMPUTATIONAL TIME IN SECONDS BY DIFFERENT NUMBER OF LINE REMOVALS IN VERY HIGH DEMAND LEVEL.

Chapter 7
Conclusions and Directions for Future Research

OTS calculations – exact and heuristic – have relied on the DCOPF approximation of the optimal power flow problem. This paper presents evidence that reliance on the DCOPF can give poor choices for removal of lines from service in various demand levels, especially at high demand levels, when congestion and the need for transmission switching are greatest. Therefore, the much more accurate ACOPF is highly recommended as the basis for OTS heuristics, especially for highly congested conditions.

This thesis modifies one such heuristic -- the sequential LP heuristic of [4] – to rely on repeated ACOPF calculations instead of DCOPF calculations. The main effect of the cost savings is for the first term of expression (15) in this system. The key idea in the heuristic, a line-ranking procedure based on the solution of the OPF problem, has been extended from the DCOPF case of [4] to reliance on an ACOPF solution.

However, substituting ACOPF calculations for DCOPF increases computing times a great deal, which may be impractical for real applications. Therefore, faster methods such as heuristic and metaheuristic methods (see, e.g., [13] and [14]), or approximations to the ACOPF that are better than the DCOPF, should be investigated for OTS in future research.

References

[1] O. I. Elgerd, Basic Electric Power Engineering. Reading, MA: Addison-Wesley, 1978.

[2] A. Gómez-Expósito, A. J. Conejo, and C. A. Cañizares, Electric Energy Systems Analysis and Operation. Boca Raton, FL: CRC, 2008.

[3] E. B. Fisher, R. P. O'Neill, and M. C. Ferris, "Optimal transmission switching," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1346–1355, Aug. 2008.

[4] J.D. Fuller, R. Ramasra, and A. Cha, "Fast heuristics for transmission-line switching," *IEEE Trans. Power Syst.*, vol. 27, no. 3, pp. 1377 - 1386, Aug 2012.

[5] P. A. Ruiz, J. M. Foster, A. Rudkevich, and M. C. Caramanis, "Tractable transmission topology control using sensitivity analysis," *IEEE Trans. Power Syst.*, vol. 27, no. 3, pp. 1550-1559, 2012.

[6] K. W. Hedman, R. P. O'Neill, E. B. Fisher, and S. S. Oren, "Optimal transmission switching with contingency analysis," *IEEE Trans. Power Syst.*, vol. 24, no. 3, pp. 1577–1586, Aug. 2009.

[7] K. W. Hedman, R. P. O'Neill, E. B. Fisher, and S. S. Oren, "Optimal transmission switching—Sensitivity analysis and extensions," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1469–1479, Aug. 2008.

[8] K.W. Hedman, R.P. O'Neill, E.B. Fisher, and S.S. Oren, "Smart Flexible Just-in-Time Transmission and Flowgate Bidding ," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 93 - 102 , Feb 2011.

[9] T. Potluri and K. W. Hedman, "Impacts of topology control on the ACOPF" presented at Power and Energy Society General Meeting , San Diego, 2012, pp. 1-7.

[10] J.D. Fuller, "Fast heuristics for optimal transmission switching," presented at the GERAD Seminar, Montreal, 2011.

[11] "Power systems test case archive," Dept. Electr. Eng., Univ. Washington , accessed Mar. 21, 2011. [Online]. Available: https://www.ee.washington.edu/research/pstca/index.html

[12] S. Blumsack, "Network topologies and transmission investment under electric-industry restructuring," Ph.D. dissertation, Dept. Eng. and Public Policy, Carnegie Mellon University, Pittsburgh, PA, 2006.

[13] R. C. Bansal, "Optimization methods for electric power systems: An overview," *Int. J. Emerg. Elect. Power Syst.*, vol. 2, no. 1, Mar. 2005.

[14] S. Frank, L Steponavice, and S. Rebennak, "Optimal power flow: a bibliographic survey II, non-deterministic and hybrid methods," *Int. J. EnergySystem* (Springer-Verlag), 2012

Appendix-A

Nomenclature

Sets and Indices

n, m	bus
g	generator
G_n	set of generators at bus *n*
k	transmission line
knm	transmission line with a flow measured at node *n* going toward *m*

Parameters

B_{nm}	element *nm* of the susceptance matrix
b^p_{nm}	shunt susceptance of line *knm*
G_{nm}	element *nm* of the conductance matrix
c_g	unit cost power from generator *g*
V^{Max}	maximum bus voltage
$V^{\min}$	minimum bus voltage

P_n^{dem} real power demand at bus n

Q_n^{dem} reactive power demand at bus n

P_g^{Max} maximum real power from generator g

$P_g^{\min}$ minimum real power from generator g

Q_g^{Max} maximum reactive power from generator g

$Q_g^{\min}$ minimum reactive power from generator g

S_k^{Max} maximum flow of apparent power on line k

δ^{Max} maximum voltage angle difference

$\delta^{\min}$ minimum voltage angle difference

Variables

P_g real power from generator g

Q_g reactive power from generator g

P_{knm} real power flow on line k

Q_{knm} reactive power flow on line k

δ_n	voltage angle at bus *n*
δ_{nm}	voltage angle difference, $(\delta_n - \delta_m)$
V_n	voltage at bus *n*
λ_k	fraction of line *k* out of service
α_k, γ_k	dual variables
π_n^P, π_n^Q, π_n	dual variables
P_k^{loss}	active power loss on line *k*
Q_k^{loss}	reactive power loss on line *k*

Appendix-B
Bus Data

The bus data of the IEEE-118 bus test system is shown in Table B-1. [2]

TABLE XI
The Bus Data of the IEEE-118 Test System

Bus Number	Real Power Demand (MW)	Reactive Power Demand (MVAR)	Maximum Real Power Output	Minimum Reactive Power Output	Max Reactive Power Output	Cost Of Power Generation	Shunt Susceptance
1	0.51	0.27	0	0	0	0	0
2	0.2	0.09	0	0	0	0	0
3	0.39	0.1	0	0	0	0	0
4	0.3	0.12	0	0	0	0	0
5	0	0	0	0	0	0	-0.4
6	0.52	0.22	0	0	0	0	0
7	0.19	0.02	0	0	0	0	0
8	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0
10	0	0	5.5	-1.47	2	217	0
11	0.7	0.23	0	0	0	0	0
12	0.47	0.1	1.85	-0.35	1.2	1052	0
13	0.34	0.16	0	0	0	0	0
14	0.14	0.01	0	0	0	0	0
15	0.9	0.3	0	0	0	0	0
16	0.25	0.1	0	0	0	0	0
17	0.11	0.03	0	0	0	0	0
18	0.6	0.34	0	0	0	0	0
19	0.45	0.25	0	0	0	0	0
20	0.18	0.03	0	0	0	0	0
21	0.14	0.08	0	0	0	0	0
22	0.1	0.05	0	0	0	0	0

[2] Bus and branch data are basically from IEEE-118 test system and are modified based on [12].

23	0.07	0.03	0	0	0	0	0
24	0	0	0	0	0	0	0
25	0	0	3.2	-0.47	1.4	434	0
26	0	0	4.14	-10	10	308	0
27	0.62	0.13	0	0	0	0	0
28	0.17	0.07	0	0	0	0	0
29	0.24	0.04	0	0	0	0	0
30	0	0	0	0	0	0	0
31	0.43	0.27	1.07	-3	3	5882	0
32	0.59	0.23	0	0	0	0	0
33	0.23	0.09	0	0	0	0	0
34	0.59	0.26	0	0	0	0	0.14
35	0.33	0.09	0	0	0	0	0
36	0.31	0.17	0	0	0	0	0
37	0	0	0	0	0	0	-0.25
38	0	0	0	0	0	0	0
39	0.27	0.11	0	0	0	0	0
40	0.2	0.23	0	0	0	0	0
41	0.37	0.1	0	0	0	0	0
42	0.37	0.23	0	0	0	0	0
43	0.18	0.07	0	0	0	0	0
44	0.16	0.08	0	0	0	0	0.1
45	0.53	0.22	0	0	0	0	0.1
46	0.28	0.1	1.19	-1	1	3448	0.1
47	0.34	0	0	0	0	0	0
48	0.2	0.11	0	0	0	0	0.15
49	0.87	0.3	3.04	-0.85	2.1	467	0
50	0.17	0.04	0	0	0	0	0
51	0.17	0.08	0	0	0	0	0
52	0.18	0.05	0	0	0	0	0
53	0.23	0.11	0	0	0	0	0
54	1.13	0.32	1.48	-3	3	1724	0
55	0.63	0.22	0	0	0	0	0
56	0.84	0.18	0	0	0	0	0
57	0.12	0.03	0	0	0	0	0
58	0.12	0.03	0	0	0	0	0

59	2.77	1.13	2.55	-0.6	1.8	606	0
60	0.78	0.03	0	0	0	0	0
61	0	0	2.6	-1	3	588	0
62	0.77	0.14	0	0	0	0	0
63	0	0	0	0	0	0	0
64	0	0	0	0	0	0	0
65	0	0	4.91	-0.67	2	249.3	0
66	0.39	0.18	4.92	-0.67	2	248.7	0
67	0.28	0.07	0	0	0	0	0
68	0	0	0	0	0	0	0
69	0	0	8.05	-3	3	189.7	0
70	0.66	0.2	0	0	0	0	0
71	0	0	0	0	0	0	0
72	0	0	0	0	0	0	0
73	0	0	0	0	0	0	0
74	0.68	0.27	0	0	0	0	0.12
75	0.47	0.11	0	0	0	0	0
76	0.68	0.36	0	0	0	0	0
77	0.61	0.28	0	0	0	0	0
78	0.71	0.26	0	0	0	0	0
79	0.39	0.32	0	0	0	0	0.2
80	1.3	0.26	5.77	-1.65	2.8	205	0
81	0	0	0	0	0	0	0
82	0.54	0.27	0	0	0	0	0.2
83	0.2	0.1	0	0	0	0	0.1
84	0.11	0.07	0	0	0	0	0
85	0.24	0.15	0	0	0	0	0
86	0.21	0.1	0	0	0	0	0
87	0	0	1.04	-1	10	7142	0
88	0.48	0.1	0	0	0	0	0
89	0	0	0	0	0	0	0
90	0.78	0.42	0	0	0	0	0
91	0	0	0	0	0	0	0
92	0.65	0.1	1	-0.5	1.55	10000	0
93	0.12	0.07	0	0	0	0	0
94	0.3	0.16	0	0	0	0	0

95	0.42	0.31	0	0	0	0	0
96	0.38	0.15	0	0	0	0	0
97	0.15	0.09	0	0	0	0	0
98	0.34	0.08	0	0	0	0	0
99	0	0	0	0	0	0	0
100	0.37	0.18	3.52	-0.03	0.09	381	0
101	0.22	0.15	0	0	0	0	0
102	0.05	0.03	0	0	0	0	0
103	0.23	0.16	1.4	-0.15	0.4	2000	0
104	0.38	0.25	0	0	0	0	0
105	0.31	0.26	0	0	0	0	0.2
106	0.43	0.16	0	0	0	0	0
107	0.28	0.12	0	0	0	0	0.06
108	0.02	0.01	0	0	0	0	0
109	0.08	0.03	0	0	0	0	0
110	0.39	0.3	0	0	0	0	0.06
111	0	0	1.36	-1	10	2173	0
112	0.25	0.13	0	0	0	0	0
113	0	0	0	0	0	0	0
114	0.08	0.03	0	0	0	0	0
115	0.22	0.07	0	0	0	0	0
116	0	0	0	0	0	0	0
117	0.2	0.08	0	0	0	0	0
118	0.33	0.15	0	0	0	0	0

Appendix-C

Branch Data

The Branch Data of the IEEE-118 bus test system is shown in Table C-2. [3]

TABLE XII
The Branch Data of the IEEE-118 Test System

Line Number	Tap Bus Number (From)	Z Bus Number (To)	Branch Resistance (R) , per unit	Branch Reactance (X) , per unit	Line Charging (ch)	Transformer Final Turns Ratio	Line Limit (MVA)
1	1	2	0.0303	0.0999	0.0254	0	2.2
2	1	3	0.0129	0.0424	0.01082	0	2.2
3	4	5	0.00176	0.00798	0.0021	0	4.4
4	3	5	0.0241	0.108	0.0284	0	2.2
5	5	6	0.0119	0.054	0.01426	0	2.2
6	6	7	0.00459	0.0208	0.0055	0	2.2
7	8	9	0.00244	0.0305	1.162	0	11
8	8	5	0	0.0267	0	0.985	8.8
9	9	10	0.00258	0.0322	1.23	0	11
10	4	11	0.0209	0.0688	0.01748	0	2.2
11	5	11	0.0203	0.0682	0.01738	0	2.2
12	11	12	0.00595	0.0196	0.00502	0	2.2
13	2	12	0.0187	0.0616	0.01572	0	2.2
14	3	12	0.0484	0.16	0.0406	0	2.2
15	7	12	0.00862	0.034	0.00874	0	2.2
16	11	13	0.02225	0.0731	0.01876	0	2.2
17	12	14	0.0215	0.0707	0.01816	0	2.2
18	13	15	0.0744	0.2444	0.06268	0	2.2
19	14	15	0.0595	0.195	0.0502	0	2.2
20	12	16	0.0212	0.0834	0.0214	0	2.2
21	15	17	0.0132	0.0437	0.0444	0	4.4
22	16	17	0.0454	0.1801	0.0466	0	2.2

[3] The data also is modified by combining parallel lines. It can also be tested with the original data with separate lines; however, the code should be changed based on it.

23	17	18	0.0123	0.0505	0.01298	0	2.2
24	18	19	0.01119	0.0493	0.01142	0	2.2
25	19	20	0.0252	0.117	0.0298	0	2.2
26	15	19	0.012	0.0394	0.0101	0	2.2
27	20	21	0.0183	0.0849	0.0216	0	2.2
28	21	22	0.0209	0.097	0.0246	0	2.2
29	22	23	0.0342	0.159	0.0404	0	2.2
30	23	24	0.0135	0.0492	0.0498	0	2.2
31	23	25	0.0156	0.08	0.0864	0	4.4
32	26	25	0	0.0382	0	0.96	2.2
33	25	27	0.0318	0.163	0.1764	0	4.4
34	27	28	0.01913	0.0855	0.0216	0	2.2
35	28	29	0.0237	0.0943	0.0238	0	2.2
36	30	17	0	0.0388	0	0.96	6.6
37	8	30	0.00431	0.0504	0.514	0	2.2
38	26	30	0.00799	0.086	0.908	0	6.6
39	17	31	0.0474	0.1563	0.0399	0	2.2
40	29	31	0.0108	0.0331	0.0083	0	2.2
41	23	32	0.0317	0.1153	0.1173	0	2.2
42	31	32	0.0298	0.0985	0.0251	0	2.2
43	27	32	0.0229	0.0755	0.01926	0	2.2
44	15	33	0.038	0.1244	0.03194	0	2.2
45	19	34	0.0752	0.247	0.0632	0	2.2
46	35	36	0.00224	0.0102	0.00268	0	2.2
47	35	37	0.011	0.0497	0.01318	0	2.2
48	33	37	0.0415	0.142	0.0366	0	2.2
49	34	36	0.00871	0.0268	0.00568	0	2.2
50	34	37	0.00256	0.0094	0.00984	0	4.4
51	38	37	0	0.0375	0	0.935	6.6
52	37	39	0.0321	0.106	0.027	0	2.2
53	37	40	0.0593	0.168	0.042	0	2.2
54	30	38	0.00464	0.054	0.422	0	2.2
55	39	40	0.0184	0.0605	0.01552	0	2.2
56	40	41	0.0145	0.0487	0.01222	0	2.2
57	40	42	0.0555	0.183	0.0466	0	2.2
58	41	42	0.041	0.135	0.0344	0	2.2

59	43	44	0.0608	0.2454	0.06068	0	2.2
60	34	43	0.0413	0.1681	0.04226	0	2.2
61	44	45	0.0224	0.0901	0.0224	0	2.2
62	45	46	0.04	0.1356	0.0332	0	2.2
63	46	47	0.038	0.127	0.0316	0	2.2
64	46	48	0.0601	0.189	0.0472	0	2.2
65	47	49	0.0191	0.0625	0.01604	0	2.2
66	42	49	0.03575	0.1615	0.172	0	1.1
67	45	49	0.0684	0.186	0.0444	0	2.2
68	48	49	0.0179	0.0505	0.01258	0	2.2
69	49	50	0.0267	0.0752	0.01874	0	2.2
70	49	51	0.0486	0.137	0.0342	0	2.2
71	51	52	0.0203	0.0588	0.01396	0	2.2
72	52	53	0.0405	0.1635	0.04058	0	2.2
73	53	54	0.0263	0.122	0.031	0	2.2
74	49	54	0.039672921	0.144998276	0.1468	0	1.1
75	54	55	0.0169	0.0707	0.0202	0	2.2
76	54	56	0.00275	0.00955	0.00732	0	2.2
77	55	56	0.00488	0.0151	0.00374	0	2.2
78	56	57	0.0343	0.0966	0.0242	0	2.2
79	50	57	0.0474	0.134	0.0332	0	2.2
80	56	58	0.0343	0.0966	0.0242	0	2.2
81	51	58	0.0255	0.0719	0.01788	0	2.2
82	54	59	0.0503	0.2293	0.0598	0	2.2
83	56	59	0.040692568	0.122426531	0.1105	0	1.1
84	55	59	0.04739	0.2158	0.05646	0	2.2
85	59	60	0.0317	0.145	0.0376	0	2.2
86	59	61	0.0328	0.15	0.0388	0	2.2
87	60	61	0.00264	0.0135	0.01456	0	4.4
88	60	62	0.0123	0.0561	0.01468	0	2.2
89	61	62	0.00824	0.0376	0.0098	0	2.2
90	63	59	0	0.0386	0	0.96	4.4
91	63	64	0.00172	0.02	0.216	0	4.4
92	64	61	0	0.0268	0	0.985	2.2
93	38	65	0.00901	0.0986	1.046	0	4.4
94	64	65	0.00269	0.0302	0.38	0	4.4

95	49	66	0.009	0.04595	0.0496	0	2.2
96	62	66	0.0482	0.218	0.0578	0	2.2
97	62	67	0.0258	0.117	0.031	0	2.2
98	65	66	0	0.037	0	0.935	2.2
99	66	67	0.0224	0.1015	0.02682	0	2.2
100	65	68	0.00138	0.016	0.638	0	2.2
101	47	69	0.0844	0.2778	0.07092	0	2.2
102	49	69	0.0985	0.324	0.0828	0	2.2
103	68	69	0	0.037	0	0.935	4.4
104	69	70	0.03	0.127	0.122	0	4.4
105	24	70	0.00221	0.4115	0.10198	0	2.2
106	70	71	0.00882	0.0355	0.00878	0	2.2
107	24	72	0.0488	0.196	0.0488	0	2.2
108	71	72	0.0446	0.18	0.04444	0	2.2
109	71	73	0.00866	0.0454	0.01178	0	2.2
110	70	74	0.0401	0.1323	0.03368	0	2.2
111	70	75	0.0428	0.141	0.036	0	2.2
112	69	75	0.0405	0.122	0.124	0	4.4
113	74	75	0.0123	0.0406	0.01034	0	2.2
114	76	77	0.0444	0.148	0.0368	0	2.2
115	69	77	0.0309	0.101	0.1038	0	2.2
116	75	77	0.0601	0.1999	0.04978	0	2.2
117	77	78	0.00376	0.0124	0.01264	0	2.2
118	78	79	0.00546	0.0244	0.00648	0	2.2
119	77	80	0.010771552	0.033175896	0.07	0	1.466666667
120	79	80	0.0156	0.0704	0.0187	0	2.2
121	68	81	0.00175	0.0202	0.808	0	2.2
122	81	80	0	0.037	0	0.935	2.2
123	77	82	0.0298	0.0853	0.08174	0	2.2
124	82	83	0.0112	0.03665	0.03796	0	2.2
125	83	84	0.0625	0.132	0.0258	0	2.2
126	83	85	0.043	0.148	0.0348	0	2.2
127	84	85	0.0302	0.0641	0.01234	0	2.2
128	85	86	0.035	0.123	0.0276	0	2.2
129	86	87	0.02828	0.2074	0.0445	0	2.2
130	85	88	0.02	0.102	0.0276	0	2.2

131	85	89	0.0239	0.173	0.047	0	2.2
132	88	89	0.0139	0.0712	0.01934	0	4.4
133	89	90	0.016307407	0.065149809	0.1588	0	1.65
134	90	91	0.0254	0.0836	0.0214	0	6.6
135	89	92	0.007907927	0.038274449	0.0962	0	1.1
136	91	92	0.0387	0.1272	0.03268	0	2.2
137	92	93	0.0258	0.0848	0.0218	0	2.2
138	92	94	0.0481	0.158	0.0406	0	2.2
139	93	94	0.0223	0.0732	0.01876	0	2.2
140	94	95	0.0132	0.0434	0.0111	0	2.2
141	80	96	0.0356	0.182	0.0494	0	2.2
142	82	96	0.0162	0.053	0.0544	0	2.2
143	94	96	0.0269	0.0869	0.023	0	2.2
144	80	97	0.0183	0.0934	0.0254	0	2.2
145	80	98	0.0238	0.108	0.0286	0	2.2
146	80	99	0.0454	0.206	0.0546	0	2.2
147	92	100	0.0648	0.295	0.0472	0	2.2
148	94	100	0.0178	0.058	0.0604	0	2.2
149	95	96	0.0171	0.0547	0.01474	0	2.2
150	96	97	0.0173	0.0885	0.024	0	2.2
151	98	100	0.0397	0.179	0.0476	0	2.2
152	99	100	0.018	0.0813	0.0216	0	2.2
153	100	101	0.0277	0.1262		0	2.2
154	92	102	0.0123	0.0559	0.01464	0	2.2
155	101	102	0.0246	0.112	0.0294	0	2.2
156	100	103	0.016	0.0525	0.0536	0	4.4
157	100	104	0.0451	0.204	0.0541	0	2.2
158	103	104	0.0466	0.1584	0.0407	0	2.2
159	103	105	0.0535	0.1625	0.0408	0	2.2
160	100	106	0.0605	0.229	0.062	0	2.2
161	104	105	0.00994	0.0378	0.00986	0	2.2
162	105	106	0.014	0.0547	0.01434	0	2.2
163	105	107	0.053	0.183	0.0472	0	2.2
164	105	108	0.0261	0.0703	0.01844	0	2.2
165	106	107	0.053	0.183	0.0472	0	2.2
166	108	109	0.0105	0.0288	0.0076	0	2.2

167	103	110	0.03906	0.1813	0.0461	0	2.2
168	109	110	0.0278	0.0762	0.0202	0	2.2
169	110	111	0.022	0.0755	0.02	0	2.2
170	110	112	0.0247	0.064	0.062	0	2.2
171	17	113	0.00913	0.0301	0.00768	0	2.2
172	32	113	0.0615	0.203	0.0518	0	2.2
173	32	114	0.0135	0.0612	0.01628	0	2.2
174	27	115	0.0164	0.0741	0.01972	0	2.2
175	114	115	0.0023	0.0104	0.00276	0	2.2
176	68	116	0.00034	0.00405	0.164	0	4.4
177	12	117	0.0329	0.14	0.0358	0	2.2
178	75	118	0.0145	0.0481	0.01198	0	2.2
179	76	118	0.0164	0.0544	0.01356	0	2.2

Appendix-D

IEEE 118-bus test system diagram[4]

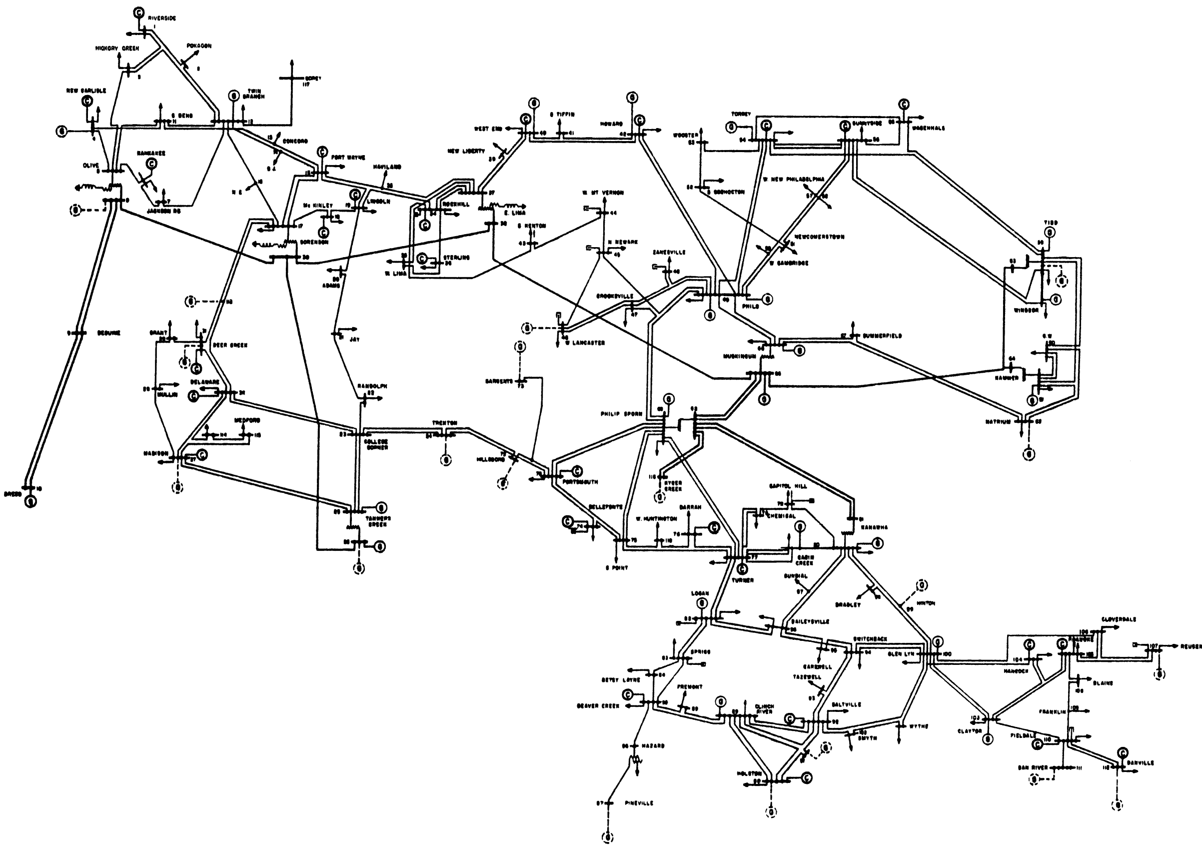

FIGURE 6: IEEE 118-BUS TEST SYSTEM DIAGRAM

[4] The diagram of the test system is as follow in [11]

Appendix-E

The Alpha Values Calculation GAMS Code

```
* Milad Soroush - 2012
* This code is written by Milad Soroush, using same code fragment from the 6-bus example by
K. Battacharya in class for notes from ECE-666, and from OPF code by M. Pirnia.


scalar phi   /3.141592654 /
Var /1.2/  Demand Multipliers;
Parameters AngleMax limit on absolute value of all voltage angles /.6/
*        LoadSupVar Global scaling for effect of QdSup /0/
         QdVar % of maximum demand as required(min) demand /1/
*        rpow /0.0/
*        qpow /0.0/
;

$onecho > input6.txt
set=i rng=type!a2 rdim=1 cdim=0 values=nodata
par=type rng=type!a2 cdim=0 rdim=1
set=k rng=resis!a2 rdim=1 cdim=0 values=nodata
set=lines rng=resis!a2 rdim=3 cdim=0 values=nodata
par=resis rng=resis!a2 cdim=0 rdim=3
par=react rng=react!a2 cdim=0 rdim=3
par=Pdem rng=Pdem!a2 cdim=0 rdim=1
par=Qdem rng=Qdem!a2 cdim=0 rdim=1
par=c rng=c!a2 cdim=0 rdim=1
par=ch rng=ch!a2 cdim=0 rdim=3
par=M rng=M!a2 cdim=0 rdim=1
par=U rng=U!a2 cdim=0 rdim=3
par=Qmin rng=Qmin!a2 cdim=0 rdim=1
par=Qmax rng=Qmax!a2 cdim=0 rdim=1
par=btype rng=btype!a2 rdim=3 cdim=0
par=transformer rng=transformer!a2 rdim=3 cdim=0
par=shuntB rng=shuntB!a2 rdim=1 cdim=0
par=vlevel rng=vlevel!a2 cdim=0 rdim=1
par=pinitial rng=pinitial!a2 rdim=1 cdim=0
par=qinitial rng=qinitial!a2 rdim=1 cdim=0
par=vinitial rng=vinitial!a2 rdim=1 cdim=0
par=tinitial rng=vinitial!a2 rdim=1 cdim=0
```

```
par=vfx rng=vfx!a2 rdim=1 cdim=0
$offecho

$CALL GDXXRW.EXE input=118bus.xls @input6.txt
$GDXin 118bus.gdx

set      i   buses
$load i
alias (i,j);

set  k  lines
$load k
alias(k,k1);

set kk(k) /170*179/;
display k;

parameter transformer(k,i,j) bus type index
$load transformer

parameter shuntB(i) bus shunt susceptance index
$load shuntB

parameter vlevel (i) voltage desired level
$load vlevel

parameter btype(k,i,j) bus type index
$load btype

parameter type(i) bus type index
$load type

set      is(i)  supply buses ;
is(i)=yes$(type(i)=2);



parameter resis(k,i,j) resistance between bus i and j
$load resis
display resis;

parameter react(k,i,j) reactance between bus i and j
```

```
$load react

parameter ch(k,i,j) Line Charging (per unit)
$load ch

parameter Pdem(i) demanded real power (per unit)
$load Pdem

parameter Qdem(i) demanded reactive power (per unit)
$load Qdem

parameter Pinitial(i) initial real power (per unit)
$load Pinitial

parameter Qinitial(i) initial reactive power (per unit)
$load Qinitial

parameter  Vinitial(i)   initial
$load Vinitial

parameter  tinitial(i)   initial
$load tinitial

Parameter  vfx(i)   desired voltage
$load vfx

Scalar k2 /0/;
Set  lines(k,i,j)  mapping of line numbers to from & to nodes;
$load lines

Set  isb(k,i,j) transformer branches;
isb(k,i,j)$(lines(k,i,j))= yes$(btype(k,i,j)=1);

Parameter lines2(k,i,j);
Parameter Z(k,i,j)  is X2+R2 = magnitude of impedence power two;
Parameter ZA(i,j) absolute value of Z;
Parameter BB(k,i,j),GG(k,i,j),Bx(k,i,j),Gx(k,i,j) small susceptance and conductance  ;
Parameter B(i,j), G(i,j), YCL(i) susceptance and conductance ;
Parameter Y(i,j), Theta(i,j) admittance magnitude and its angle;
Parameters
       U(k,i,j)    line capacity
       c(i)  supply cost coefficient
```

```
        M(i)   supply capacity parameter
        qd(i)   quantity demanded at i  (Mw)
        QdSup(i) random percentage increases in demand
        Qmax    reactive power max
        Qmin reactive power min   ;


QdSup(i)=0;

Parameter ppdem(i)  augmented demand ;
*        ppdem(i) = Pdem(i)*(QdVar + LoadSupVar*QdSup(i)) ;

$load c
$load M
$load U
$load Qmin
$load Qmax

$gdxin

*********** keeping initial value of lines  **********

parameter  isb2(k,i,j), btype2(k,i,j),transformer2(k,i,j) ;
parameter line(k,i,j),ch2(k,i,j),resis2(k,i,j),react2(k,i,j);
parameter GG2(i,j), BB2(i,j),GG3(i,j), BB3(i,j),trans(i,j),charg(i,j);

transformer(k,i,j)$(transformer(k,i,j)=0 and (lines(k,i,j)))=1.0;
lines2(k,i,j)= lines(k,i,j);
ch2(k,i,j)= ch(k,i,j);
resis2(k,i,j)= resis(k,i,j);
react2(k,i,j)= react(k,i,j);
isb2(k,i,j)= isb(k,i,j);
transformer2(k,i,j)= transformer(k,i,j);

********  conductances and suseptances ***************
Z(k,i,j)= resis(k,i,j)*resis(k,i,j) + react(k,i,j)*react(k,i,j);
GG(k,i,j)$(Z(k,i,j) ne 0.0000)= resis(k,i,j)/Z(k,i,j) ;
BB(k,i,j)$(Z(k,i,j) ne 0.0000)= -react(k,i,j)/Z(k,i,j);

************* Just losing K *************************
GG2(i,j) = sum((k)$lines(k,i,j), GG(k,i,j));
BB2(i,j) = sum((k)$lines(k,i,j), BB(k,i,j));
```

```
trans(i,j)= sum((k)$lines(k,i,j),transformer(k,i,j));
charg(i,j)= sum((k)$lines(k,i,j),ch(k,i,j));

************* making G and B matrix ******************
G(i,j)$(trans(i,j) ne 0.00)= -GG2(i,j)/trans(i,j);
G(j,i)$(G(i,j) ne 0.00)= G(i,j);

B(i,j)$(trans(i,j) ne 0.00) = -BB2(i,j)/trans(i,j);
B(j,i)$(B(i,j) ne 0.00)= B(i,j);

GG3(i,j)$(trans(i,j) ne 0.00)= GG2(i,j)/sqr(trans(i,j));
BB3(i,j)$(trans(i,j) ne 0.00)= BB2(i,j)/sqr(trans(i,j));

G(i,i)= sum(j,GG3(i,j))+ sum(j,GG2(j,i)) ;
B(i,i)= sum(j,BB3(i,j)) + sum(j,BB2(j,i))+  sum(j,charg(j,i)/2)+sum(j,charg(i,j)/2) + shuntB(i);

************* making admittance matrix ***************
Y(i,j) = sqrt(G(i,j)*G(i,j) + B(i,j)*B(i,j));
ZA(i,j)$(G(i,j) ne 0.00)  = abs(B(i,j))/abs(G(i,j)) ;

****************************************************
display resis, react, Z, GG2, BB2, lines2, trans, charg,transformer,vfx;


Theta(i,j) = arctan(ZA(i,j));

Theta(i,j)$((B(i,j) eq 0) and (G(i,j) gt 0)) = 0.0 ;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) lt 0)) =  -0.5*phi  ;
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) gt 0)) = Theta(i,j) ;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) gt 0)) = 2*phi - Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) lt 0)) = phi - Theta(i,j);
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) lt 0)) = phi + Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) eq 0)) = 0.5*phi;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) eq 0)) = -0.5*phi;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) eq 0)) = 0.0 ;
***************************************************
display Z,GG,BB,G,B,Y,Theta;
display c,M,U,lines,react,resis,is,ch,Pdem,Qdem,isb;
**************** various demand levels **************
Pdem(i)= Var * Pdem(i);
***************************************************
parameter rstat(k);
```

```
Variables
        lineload(k)
        cost    total generation cost ($)
        t(i)    theta at bus i (voltage angle in radians)
        V(i)    voltage magnitude at bus i
        Qs(i)   reactive power supplied at bus i;

Positive variables
        q(i)   quantity supplied at i  (Mw);
*********** initialize voltage and angles ***********
*V.l(i)  = 1.00 ;
V.l(i)  = Vinitial(i) ;
*t.l(i)  =  0.00 ;
t.l(i)  = tinitial(i) ;
*q.fx(i)  = Pinitial(i);
*Qs.l(i) = Qinitial(i);

**************************************************

Equations
        GenCost         define objective function
        powerbal(i)     power balance at node i
        repowerbal(i)    reactive power balance at node i
        limit(k,i,j)     line limit power (for poth active and reactive power)
        angleLimUp(k,i,j)   upper limit on voltage angle difference
        angleLimLo(k,i,j)   lower limit on voltage angle difference
        gencap(i)      generation capacity  ;

GenCost..  cost =e= sum(i, c(i)*q(i));

powerbal(i).. q(i)$is(i) - Pdem(i)- sum((j),Y(i,j)*V(i)*V(j)*cos(theta(i,j) + t(j) - t(i))) =e= 0;
repowerbal(i).. Qs(i)$is(i) -Qdem(i)+ sum((j),Y(i,j)*V(j)*V(i)*sin(theta(j,i) + t(j) - t(i))) =e= 0;

*powerbal(i).. q(i)$is(i) - Pdem(i)- sum((j),V(i)*V(j)*(G(i,j)*cos(t(i) - t(j))+B(i,j)*sin(t(i) - t(j))))
=e= 0;
*repowerbal(i..  Qs(i)$is(i)-Qdem(i)-  sum((j),V(i)*V(j)*(G(i,j)*sin(t(i)  -  t(j))-B(i,j)*cos(t(i)  -
t(j)))) =e= 0;

limit(k,i,j)$(lines(k,i,j))..   sqr(-V(i)*V(i)*G(i,j)+  V(i)*V(j)*(G(i,j)*cos(t(i)-t(j))+  B(i,j)*sin(t(i)-
t(j)))))+     sqr(V(i)*V(i)*B(i,j)+      V(i)*V(j)*(G(i,j)*sin(t(i)-t(j))-     B(i,j)*cos(t(i)-t(j))))     =l=
sqr(U(k,i,j));
```

```
angleLimUp(k,i,j)$(lines(k,i,j)).. t(i)-t(j) =l= AngleMax;
angleLimLo(k,i,j)$(lines(k,i,j)).. t(i)-t(j) =g= -AngleMax;

gencap(i)$is(i)..  q(i)=l=M(i);

Qs.Up(i)  = Qmax(i);
Qs.Lo(i)  = Qmin(i);

V.UP(i) = 1.06;
V.Lo(i) = 0.94;
*V.fx(i)$(vfx(i) ne 0)= vfx(i);

model network /all/;

**************************************************

*option nlp=coinCouenne
*option nlp=coinipopt;
*option nlp=minos;
option nlp=conopt;
*option nlp=pathNLP

solve network using NLP Minimizing cost;

parameter costNLP;
costNLP=cost.l;
display costNLP;

Parameter rcost(k);
rcost(k)=0;
rstat(k)=0;


display rstat,rcost;
**************************************************

file Line118out2
put Line118out2;

*** calculating real and reactive power of lines ****
parameter linepower(k,i,j), linerepower(k,i,j);
```

```
linepower(k,i,j)=   -V.l(i)*V.l(i)*G(i,j)+   V.l(i)*V.l(j)*(G(i,j)*cos(t.l(i)-t.l(j))+   B(i,j)*sin(t.l(i)-
t.l(j)));
linerepower(k,i,j)=   V.l(i)*V.l(i)*B(i,j)+   V.l(i)*V.l(j)*(G(i,j)*sin(t.l(i)-t.l(j))-   B(i,j)*cos(t.l(i)-
t.l(j)));

*** calculating Alpha based on the formula **********

parameter alpha(k,i,j),alpha2(k,i,j);

alpha2(k,i,j)$(lines(k,i,j)) = -linepower(k,i,j)* powerbal.m(i)-linerepower(k,i,j)*repowerbal.m(i);
alpha2(k,i,j)$(lines(k,j,i)) = -linepower(k,i,j)* powerbal.m(i)-linerepower(k,i,j)*repowerbal.m(i);
alpha(k,i,j)$(lines(k,i,j)) = alpha2(k,i,j)+ alpha2(k,j,i);

parameter Lalph(k);
Lalph(k)=sum((i,j)$lines(k,i,j),alpha(k,i,j));

**************************************************
$ontext
Parameter proportion(k);
proportion(k)= sum((i,j)$lines(k,i,j),linepower(k,i,j)/linerepower(k,i,j));

Parameter freeBand(k,i,j),freeBand2(k,i,j),totalfree,totalfree2,perc;

freeBand(k,i,j)$lines(k,i,j)=                    U(k,i,j)-                   sqrt(sqr(-V.l(i)*V.l(i)*G(i,j)+
V.l(i)*V.l(j)*(G(i,j)*cos(t.l(i)-t.l(j))+         B(i,j)*sin(t.l(i)-t.l(j))))+         sqr(V.l(i)*V.l(i)*B(i,j)+
V.l(i)*V.l(j)*(G(i,j)*sin(t.l(i)-t.l(j))- B(i,j)*cos(t.l(i)-t.l(j))))));
freeBand2(k,i,j)= U(k,i,j)**2-limit.l(k,i,j);

totalfree=sum((k,i,j)$lines(k,i,j),freeBand(k,i,j));
perc=sum((k,i,j)$lines(k,i,j),sqrt(linepower(k,i,j)*linepower(k,i,j)+linerepower(k,i,j)*linerepower
(k,i,j)));
perc=sum((k,i,j)$lines(k,i,j),U(k,i,j));
perc=100*(perc-totalfree)/perc;
totalfree2=sum((k,i,j)$lines(k,i,j),freeBand2(k,i,j));

Display Lalph,totalfree,totalfree2,perc, freeBand;
$offtext
**************************************************
loop(k,
put Lalph(k)/;
);
**************************************************
```

```
parameter percP(k,i,j),percQ(k,i,j),percLOSS(k,i,j),percSUM(k,i,j);

percP(k,i,j)$(lines(k,i,j)) = (linepower(k,i,j))*(powerbal.m(j)-powerbal.m(i)) ;
percQ(k,i,j)$(lines(k,i,j)) =  (linerepower(k,i,j))* (repowerbal.m(j)-repowerbal.m(i));
percSUM(k,i,j)$(lines(k,i,j)) = percP(k,i,j)+ percQ(k,i,j);
percLOSS(k,i,j)$(lines(k,i,j)) =  alpha(k,i,j)- percSUM(k,i,j)

parameter perP(k),perQ(k),perLOSS(k),perSUM(k);
Display percP,percQ,percLOSS,percSUM;
```

Appendix-F

The Heuristic Method GAMS Code

```
* Milad Soroush - 2012
* This code is written by Milad Soroush, using same code fragment from the 6-bus example by
K. Battacharya in class for notes from ECE-666, and from OPF code by M. Pirnia.

scalar phi   /3.141592654 /
Var /1.2/  Demand Multipliers;
Parameters AngleMax limit on absolute value of all voltage angles /.6/
*        LoadSupVar Global scaling for effect of QdSup /0/
 *       QdVar % of Demand Multipliers  /1.2/
        Jnum maximum number of lines to remove (iterations) /4/
*        rpow /0.0/
*        qpow /0.0/
;

$onecho > input6.txt
set=i rng=type!a2 rdim=1 cdim=0 values=nodata
par=type rng=type!a2 cdim=0 rdim=1
set=k rng=resis!a2 rdim=1 cdim=0 values=nodata
set=lines rng=resis!a2 rdim=3 cdim=0 values=nodata
par=resis rng=resis!a2 cdim=0 rdim=3
par=react rng=react!a2 cdim=0 rdim=3
par=Pdem rng=Pdem!a2 cdim=0 rdim=1
par=Qdem rng=Qdem!a2 cdim=0 rdim=1
par=c rng=c!a2 cdim=0 rdim=1
par=ch rng=ch!a2 cdim=0 rdim=3
par=M rng=M!a2 cdim=0 rdim=1
par=U rng=U!a2 cdim=0 rdim=3
par=Qmin rng=Qmin!a2 cdim=0 rdim=1
par=Qmax rng=Qmax!a2 cdim=0 rdim=1
par=btype rng=btype!a2 rdim=3 cdim=0
par=transformer rng=transformer!a2 rdim=3 cdim=0
par=shuntB rng=shuntB!a2 rdim=1 cdim=0
par=vlevel rng=vlevel!a2 cdim=0 rdim=1
par=pinitial rng=pinitial!a2 rdim=1 cdim=0
par=qinitial rng=qinitial!a2 rdim=1 cdim=0
par=vinitial rng=vinitial!a2 rdim=1 cdim=0
par=tinitial rng=vinitial!a2 rdim=1 cdim=0
```

```
par=vfx rng=vfx!a2 rdim=1 cdim=0
$offecho

$CALL GDXXRW.EXE input=118bus.xls @input6.txt
$GDXin 118bus.gdx

set      i   buses
$load i
alias (i,j);

Sets
l   maximum # heuristic alpha solves / 1*12 /;
alias(l,n);

Scalar starttime;
starttime = jnow;
scalar elapsed;
set itn /1*20/;
parameter CostRedPC(itn);
parameter Time(itn);
Parameters
MaxBest # of cost improvements before selecting best / 6 /;

set  k  lines
$load k
alias(k,k1);

set kk(k) /170*179/;
display k;

parameter transformer(k,i,j) bus type index
$load transformer

parameter shuntB(i) shunt susceptance index
index
$load shuntB

parameter vlevel (i) voltage desired level
$load vlevel

parameter btype(k,i,j) bus type index
$load btype
```

```
parameter type(i) bus type index
$load type

set      is(i)  supply buses ;
is(i)=yes$(type(i)=2);



parameter resis(k,i,j) resistance between bus i and j
$load resis
display resis;

parameter react(k,i,j) reactance between bus i and j
$load react

parameter ch(k,i,j) Line Charging (per unit)
$load ch

parameter Pdem(i) demanded real power (per unit)
$load Pdem

parameter Qdem(i) demanded reactive power (per unit)
$load Qdem

parameter Pinitial(i) initial real power (per unit)
$load Pinitial

parameter Qinitial(i) initial reactive power (per unit)
$load Qinitial

parameter  Vinitial(i)   initial
$load Vinitial

parameter  tinitial(i)   initial
$load tinitial

parameter  vfx(i)   desired voltage
$load vfx


*Parameter NumLines;
```

```
*Parameter NumBuses;

*NumLines = card(k);
*NumBuses = card(i);

Scalar k2 /0/;


*display NumLines, NumBuses;

Set  lines(k,i,j)  mapping of line numbers to from & to nodes;
$load lines

Set  isb(k,i,j) transformer branches;
isb(k,i,j)$(lines(k,i,j))=yes$(btype(k,i,j)=1);

parameter lines2(k,i,j);
Parameter Z(k,i,j)  is X2+R2 = magnitude of impedence power two;
Parameter ZA(i,j) absolute value of Z;
Parameter BB(k,i,j),GG(k,i,j),Bx(k,i,j),Gx(k,i,j) small susceptance and conductance  ;
Parameter B(i,j), G(i,j), YCL(i) susceptance and conductance ;
Parameter Y(i,j), Theta(i,j) admittance magnitude and its angle;

parameters
      U(k,i,j)    line capacity
      c(i)  supply cost coefficient
      M(i)   supply capacity parameter
      qd(i)   quantity demanded at i  (Mw)
      QdSup(i) random percentage increases in demand
      Qmax    reactive power max
      Qmin reactive power min   ;


QdSup(i)=0;

Parameter ppdem(i)  augmented demand ;
*        ppdem(i) = Pdem(i)*(QdVar + LoadSupVar*QdSup(i)) ;

$load c
$load M
$load U
$load Qmin
```

```
$load Qmax

$gdxin

*********** keeping initial value of lines  ************

parameter  isb2(k,i,j), btype2(k,i,j),transformer2(k,i,j) ;
parameter line(k,i,j),ch2(k,i,j),resis2(k,i,j),react2(k,i,j);
parameter GG2(i,j), BB2(i,j),GG3(i,j), BB3(i,j),trans(i,j),charg(i,j);

transformer(k,i,j)$(transformer(k,i,j)=0 and (lines(k,i,j)))=1.0;
lines2(k,i,j)= lines(k,i,j);
ch2(k,i,j)= ch(k,i,j);
resis2(k,i,j)= resis(k,i,j);
react2(k,i,j)= react(k,i,j);
isb2(k,i,j)= isb(k,i,j);
transformer2(k,i,j)= transformer(k,i,j);

********  conductances and suseptances *****************
Z(k,i,j)= resis(k,i,j)*resis(k,i,j) + react(k,i,j)*react(k,i,j);

GG(k,i,j)$(Z(k,i,j) ne 0.0000)= resis(k,i,j)/Z(k,i,j) ;
BB(k,i,j)$(Z(k,i,j) ne 0.0000)= -react(k,i,j)/Z(k,i,j);

************* Just losing K *************************
GG2(i,j) = sum((k)$lines(k,i,j), GG(k,i,j));
BB2(i,j) = sum((k)$lines(k,i,j), BB(k,i,j));
trans(i,j)= sum((k)$lines(k,i,j),transformer(k,i,j));
charg(i,j)= sum((k)$lines(k,i,j),ch(k,i,j));

************* making G and B matrix *****************
G(i,j)$(trans(i,j) ne 0.00)= -GG2(i,j)/trans(i,j);
G(j,i)$(G(i,j) ne 0.00)= G(i,j);

B(i,j)$(trans(i,j) ne 0.00) = -BB2(i,j)/trans(i,j);
B(j,i)$(B(i,j) ne 0.00)= B(i,j);

GG3(i,j)$(trans(i,j) ne 0.00)= GG2(i,j)/sqr(trans(i,j));
BB3(i,j)$(trans(i,j) ne 0.00)= BB2(i,j)/sqr(trans(i,j));

G(i,i)= sum(j,GG3(i,j))+ sum(j,GG2(j,i)) ;
B(i,i)= sum(j,BB3(i,j)) + sum(j,BB2(j,i))+  sum(j,charg(j,i)/2)+sum(j,charg(i,j)/2) + shuntB(i);
```

```
************* making admittance matrix ***************
Y(i,j) = sqrt(G(i,j)*G(i,j) + B(i,j)*B(i,j));
ZA(i,j)$(G(i,j) ne 0.00)  = abs(B(i,j))/abs(G(i,j)) ;

***************************************************
display resis, react, Z, GG2, BB2, lines2, trans, charg,transformer,vfx;


Theta(i,j) = arctan(ZA(i,j));

Theta(i,j)$((B(i,j) eq 0) and (G(i,j) gt 0)) = 0.0 ;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) lt 0)) =  -0.5*phi  ;
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) gt 0)) = Theta(i,j) ;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) gt 0)) = 2*phi - Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) lt 0)) = phi - Theta(i,j);
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) lt 0)) = phi + Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) eq 0)) = 0.5*phi;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) eq 0)) = -0.5*phi;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) eq 0)) = 0.0 ;
**************************************************
display Z,GG,BB,G,B,Y,Theta;
display c,M,U,lines,react,resis,is,ch,Pdem,Qdem,isb;
**************** various demand levels **************
Pdem(i)= Var * Pdem(i);
**************************************************
parameter rstat(k);

*Pdem(i)= Qdvar * Pdem(i);
*U(k,i,j)=inf;

Variables
        lineload(k)
        cost    total generation cost ($)
        t(i)    theta at bus i (voltage angle in radians)
        V(i)    voltage magnitude at bus i
        Qs(i)   reactive power supplied at bus i;

Positive variables
        q(i)   quantity supplied at i  (Mw);
*********** initialize voltage and angles ***********
*V.l(i)  = 1.00 ;
```

```
V.l(i)  = Vinitial(i) ;
*t.l(i)  =  0.00 ;
t.l(i)  = tinitial(i) ;
*q.fx(i)  = Pinitial(i);
*Qs.l(i) = Qinitial(i);

*************************************************

Equations
        GenCost         define objective function
        powerbal(i)     power balance at node i
        repowerbal(i)    reactive power balance at node i
        limit(k,i,j)     line limit power (for poth active and reactive power)
        angleLimUp(k,i,j)   upper limit on voltage angle difference
        angleLimLo(k,i,j)   lower limit on voltage angle difference
        gencap(i)      generation capacity  ;

GenCost..  cost =e= sum(i, c(i)*q(i));

powerbal(i).. q(i)$is(i) - Pdem(i)- sum((j),Y(i,j)*V(i)*V(j)*cos(theta(i,j) + t(j) - t(i))) =e= 0;
repowerbal(i).. Qs(i)$is(i) -Qdem(i)+ sum((j),Y(i,j)*V(j)*V(i)*sin(theta(j,i) + t(j) - t(i))) =e= 0;

*powerbal(i).. q(i)$is(i) - Pdem(i)- sum((j),V(i)*V(j)*(G(i,j)*cos(t(i) - t(j))+B(i,j)*sin(t(i) - t(j))))
=e= 0;
*repowerbal(i)..  Qs(i)$is(i)-Qdem(i)-  sum((j),V(i)*V(j)*(G(i,j)*sin(t(i)  -  t(j))-B(i,j)*cos(t(i)  -
t(j)))) =e= 0;

limit(k,i,j)$(lines(k,i,j))..   sqr(-V(i)*V(i)*G(i,j)+  V(i)*V(j)*(G(i,j)*cos(t(i)-t(j))+  B(i,j)*sin(t(i)-
t(j))))+     sqr(V(i)*V(i)*B(i,j)+      V(i)*V(j)*(G(i,j)*sin(t(i)-t(j))-      B(i,j)*cos(t(i)-t(j))))      =l=
sqr(U(k,i,j));

angleLimUp(k,i,j)$(lines(k,i,j)).. t(i)-t(j) =l= AngleMax;
angleLimLo(k,i,j)$(lines(k,i,j)).. t(i)-t(j) =g= -AngleMax;

gencap(i)$is(i)..  q(i)=l=M(i);

Qs.Up(i)  = Qmax(i);
Qs.Lo(i)  = Qmin(i);

V.UP(i) = 1.06;
V.Lo(i) = 0.94;
*V.fx(i)$(vfx(i) ne 0)= vfx(i);
```

```
model network /all/;

**************************************************************

*option nlp=coinCouenne
*option nlp=coinipopt;
*option nlp=minos;
option nlp=conopt;
*option nlp=pathNLP

solve network using NLP Minimizing cost;


Scalar SolveTime;
SolveTime = network.ETsolve;


parameter costNLP;
costNLP=cost.l;
display costNLP;

Parameter rcost(k);
rcost(k)=0;
rstat(k)=0;

**************************************************************

file Line118out2
put Line118out2;

*** calculating real and reactive power of lines ***************
parameter linepower(k,i,j), linerepower(k,i,j);
linepower(k,i,j)=   -V.l(i)*V.l(i)*G(i,j)+   V.l(i)*V.l(j)*(G(i,j)*cos(t.l(i)-t.l(j))+   B(i,j)*sin(t.l(i)-
t.l(j)));
linerepower(k,i,j)=   V.l(i)*V.l(i)*B(i,j)+   V.l(i)*V.l(j)*(G(i,j)*sin(t.l(i)-t.l(j))-   B(i,j)*cos(t.l(i)-
t.l(j)));

*** calculating Alpha based on the formula ********************

parameter alpha(k,i,j),alpha2(k,i,j);
```

```
alpha2(k,i,j)$(lines(k,i,j)) = -linepower(k,i,j)* powerbal.m(i)-linerepower(k,i,j)*repowerbal.m(i);
alpha2(k,i,j)$(lines(k,j,i)) = -linepower(k,i,j)* powerbal.m(i)-linerepower(k,i,j)*repowerbal.m(i);
alpha(k,i,j)$(lines(k,i,j)) = alpha2(k,i,j)+ alpha2(k,j,i);

parameter Lalph(k);
Lalph(k)=sum((i,j)$lines(k,i,j),alpha(k,i,j));

*************************************************************

Display Lalph;

Parameter TotDem;
Parameter TotSup;
Parameter TotCap;
Parameter TotLoss;
Parameter TotLossPC;
Parameter VangDiff(k,i,j);
Parameter MaxVangDiff;
Parameter NumLines;

*************************************************************
TotDem = sum(i,qdem(i));
TotSup = sum(i,qs.l(i));
TotLoss = TotSup - TotDem;
TotLossPC = 100*TotLoss/TotSup;
VangDiff(k,i,j)$lines(k,i,j) = abs(t.l(i)-t.l(j));
MaxVangDiff = 0;

*************************************************************
loop(lines(k,i,j),
        if(VangDiff(k,i,j)>MaxVangDiff, MaxVangDiff=VangDiff(k,i,j));
);
*************************************************************

TotCap = sum(is,M(is));
NumLines = card(lines);
Parameter NumZeroPrices;
NumZeroPrices = sum(i$(powerbal.m(i) = 0),1);
parameter line3(k,i,j),ch3(k,i,j),resis3(k,i,j),react3(k,i,j), isb3(k,i,j),
btype3(k,i,j),transformer3(k,i,j);
```

```
****************** Heurstic Loop **********************************

Parameter SMalpha smallest alpha value among lines;
Parameters
        x optimality flag /1/
        yy Jnum counter /0/
        zzz counter for cost reductions / 0 /
        w flag to only remove 1 line if ties / 0 /
        s counter for number of alpha tries / 0 /

        Bsave(k,i,j) susceptance from previous iteration
        Gsave(k,i,j) conductance from previous iteration

        Balpha(l,i,j) susceptance without alpha line
        Galpha(l,i,j) conductance without alpha line

        newcost(l) with largest alpha line removed
        stat(l) model status with alpha line removed
        NLPtime(l) keeping executing time of the method
        lastcost last smallest cost
        firstcost first NLP cost solution
        NetStat Solver status of the last best solve /1/
        Loss(l) loss  during each alpha solve
        MaxVDiff(l) maximum voltage angle difference for each l
        t_alpha(l,i) variables required for computing alpha for each l.
        V_alpha(l,i) variables required for computing alpha for each l
        t_save(l,i)
        V_save(l,i)
        lines_alpha(l,k,i,j)
        powerbal_alpha(l,i)
        repowerbal_alpha(l,i)
        limit_alpha(l,k,i,j);

lastcost = cost.l;
firstcost = cost.l;

*******************************************************************

put 'NLP-118Bus-Heuristic-RR-v1' /;
put 'Time stamp: ' @15 Jnow::4 /;
put 'Numlines: ' @10 Numlines @25 ' Jnum: ' @35 Jnum /;
put 'LoadSupVar: ' @10 LoadSupVar @25 ' QdVar: ' @35 QdVar /;
```

```
put 'Number of Alpha Solves per iteration: ' @40 card(l) /;
put 'Number of cost reductions before best MaxBest: ' @50 MaxBest /;
put 'Initial cost: ' @15 cost.l @30 network.Tmodstat /;
put 'Initial loss %: ' TotLossPC /;
put 'MaxVangDiff: ' MaxVangDiff / /;



***** main loop: each iteration for one removal *****************

while ((x=1 and (NetStat=1 OR NetStat=2) and yy<Jnum),
        x=0;
        yy=yy+1;
        put 'Iteration: ' yy /;
        zzz=0;
        s=0;
        newcost(l)=lastcost + 10000;
        NLPtime(l) = 0;

***** the loop for alpha tries in each iteration ****************

        loop(l,
******* picking smallest alpha **********************************

                SMalpha = smin((k,i,j)$lines(k,i,j),alpha(k,i,j));

******* alpha should be bigger than zero to be picked ***********
******* required at most msxbest=4 nbr of improvements **********

                if (SMalpha < 0 AND zzz<MaxBest,
                        w=0;
                        s=s+1;
                        Lalph(k)=sum((i,j)$lines(k,i,j),alpha(k,i,j));

******* restructuring the values after the removal **************

                        loop(k1,
                                if (Lalph(k1)=SMalpha AND w=0,

                                  YCL(i)=0;
                                  GG(k,i,j)=0;
                                  GG2(i,j)=0;
```

```
GG3(i,j)=0;
BB(k,i,j)=0;
BB2(i,j)=0;
BB3(i,j)=0;
Z(k,i,j)=0;
transformer(k,i,j)=0;
trans(i,j)= 0;
BB(k,i,j)=0;
G(i,j)=0;
B(i,j)=0;
Y(i,j)=0;
theta(i,j)=0;
ZA(i,j)=0;
ch(k,i,j)=0;
lines(k,i,j)= no;
isb(k,i,j)=no;

lines(k,i,j)= lines2(k,i,j);
lines(k1,i,j)= no;

isb(k,i,j)$(lines2(k,i,j))= isb2(k,i,j);
transformer(k,i,j)$(lines2(k,i,j))= transformer2(k,i,j);
ch(k,i,j)$(lines2(k,i,j))= ch2(k,i,j);
resis(k,i,j)$(lines2(k,i,j))= resis2(k,i,j);
react(k,i,j)$(lines2(k,i,j))= react2(k,i,j);

ch(k,i,j)$(not lines(k,i,j))= 0;
resis(k,i,j)$(not lines(k,i,j))= 0;
react(k,i,j)$(not lines(k,i,j))= 0;
isb(k,i,j)$(not lines(k,i,j))= no;
btype(k,i,j)$(not lines(k,i,j))= 0;
transformer(k,i,j)$(not lines(k,i,j))= 0;

Z(k,i,j)= resis(k,i,j)*resis(k,i,j) + react(k,i,j)*react(k,i,j);
GG(k,i,j)$(Z(k,i,j) ne 0.0000)= resis(k,i,j)/Z(k,i,j) ;
BB(k,i,j)$(Z(k,i,j) ne 0.0000)= -react(k,i,j)/Z(k,i,j);

*********************** Just losing K ************************

GG2(i,j) = sum((k)$lines(k,i,j), GG(k,i,j));
BB2(i,j) = sum((k)$lines(k,i,j), BB(k,i,j));
trans(i,j)= sum((k)$lines(k,i,j),transformer(k,i,j));
```

```
charg(i,j)= sum((k)$lines(k,i,j),ch(k,i,j));


G(i,j)$(trans(i,j) ne 0.00)= -GG2(i,j)/trans(i,j);
G(j,i)$(G(i,j) ne 0.00)= G(i,j);

B(i,j)$(trans(i,j) ne 0.00) = -BB2(i,j)/trans(i,j);
B(j,i)$(B(i,j) ne 0.00)= B(i,j);

GG3(i,j)$(trans(i,j) ne 0.00)= GG2(i,j)/sqr(trans(i,j));
BB3(i,j)$(trans(i,j) ne 0.00)= BB2(i,j)/sqr(trans(i,j));

G(i,i)= sum(j,GG3(i,j))+ sum(j,GG2(j,i)) ;
B(i,i)= sum(j,BB3(i,j)) + sum(j,BB2(j,i))+
sum(j,charg(j,i)/2)+sum(j,charg(i,j)/2) + shuntB(i);

Y(i,j) = sqrt(G(i,j)*G(i,j) + B(i,j)*B(i,j));
ZA(i,j)$(G(i,j) ne 0.00)  = abs(B(i,j))/abs(G(i,j)) ;

display resis, react, Z, GG2, BB2, lines2, trans, charg,transformer;

Theta(i,j) = arctan(ZA(i,j));

Theta(i,j)$((B(i,j) eq 0) and (G(i,j) gt 0)) = 0.0 ;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) lt 0)) =  -0.5*phi  ;
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) gt 0)) = Theta(i,j) ;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) gt 0)) = 2*phi - Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) lt 0)) = phi - Theta(i,j);
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) lt 0)) = phi + Theta(i,j);
Theta(i,j)$((B(i,j) gt 0) and (G(i,j) eq 0)) = 0.5*phi;
Theta(i,j)$((B(i,j) lt 0) and (G(i,j) eq 0)) = -0.5*phi;
Theta(i,j)$((B(i,j) eq 0) and (G(i,j) eq 0)) = 0.0 ;

display Z,GG,BB,YCL,G,B,Y, Theta, isb;

display c, M, U, lines, react, resis, is, ch, Pdem,Qdem;

V.l(i) = 1.0 ;
t.l(i) =  0 ;
q.l(i) = 0 ;
Qs.l(i)= 0 ;
cost.l = 0;
```

```
        w=1;
        alpha(k1,i,j) = 0;
        Lalph(k1)=0;
        lines_alpha(l,k,i,j)=lines(k,i,j);
        display lines_alpha;
      );
);


display lines_alpha;
solve network using nlp minimizing cost;
NLPtime(l) = network.ETsolver;

t_alpha(l,i) = t.l(i);
V_alpha(l,i) = V.l(i);

powerbal_alpha(l,i) = powerbal.m(i);
repowerbal_alpha(l,i) = repowerbal.m(i);
limit_alpha(l,k,i,j) = limit.m(k,i,j);

stat(l) = network.Modelstat;
TotDem = sum(i,Pdem(i));
TotSup = sum(i,q.l(i));
TotLoss = TotSup - TotDem;
Loss(l) = 100*TotLoss/TotSup;
newcost(l) = cost.l;
put 'Alpha 'l.val ' Cost ' cost.l /;
newcost(l)$(stat(l)<>1 AND stat(l)<>2 )= lastcost + 10000;
if (newcost(l) < lastcost,
      zzz=zzz+1
);

 Balpha(l,i,j) = B(i,j);
 Galpha(l,i,j) = G(i,j);
 Balpha(l,i,i) = B(i,i);
 Galpha(l,i,i) = G(i,i);


 alpha(k,i,j)$(alpha(k,i,j)$(lines(k,i,j)) = SMalpha) = 0;
```

```
                VangDiff(k,i,j)$lines(k,i,j) = abs(t.l(i)-t.l(j));
                MaxVDiff(l) = smax((k,i,j)$lines(k,i,j),VangDiff(k,i,j));
        );
    );
    put 'Alpha solves: '  s /;

************** recalculating alphas and updating the costs *******

    loop(l,
        if (((newcost(l) = smin(n, newcost(n))) and (newcost(l) < lastcost) and ((stat(l) =
1)OR(stat(l) = 2) )),


                display lines, lines2, lines_alpha;

                lines2(k,i,j)$(lines(k,i,j))= lines_alpha(l,k,i,j);
                B(i,j) = Balpha(l,i,j);
                G(i,j) = Galpha(l,i,j);
                loop((k,i,j),
                    linepower(k,i,j)$(lines(k,i,j))=        -V_alpha(l,i)*V_alpha(l,i)*Galpha(l,i,j)+
V_alpha(l,i)*V_alpha(l,j)*(Galpha(l,i,j)*cos(t_alpha(l,i)-t_alpha(l,j))+
Balpha(l,i,j)*sin(t_alpha(l,i)-t_alpha(l,j)));
                    linerepower(k,i,j)$(lines(k,i,j))=       V_alpha(l,i)*V_alpha(l,i)*Balpha(l,i,j)+
V_alpha(l,i)*V_alpha(l,j)*(Galpha(l,i,j)*sin(t_alpha(l,i)-t_alpha(l,j))-
Balpha(l,i,j)*cos(t_alpha(l,i)-t_alpha(l,j)));
                );
                display lines2;

                alpha2(k,i,j)$(lines2(k,i,j))     =     -linepower(k,i,j)*     powerbal_alpha(l,i)-
linerepower(k,i,j)*repowerbal_alpha(l,i);
                alpha2(k,i,j)$(lines2(k,i,j))     =     -linepower(k,i,j)*     powerbal_alpha(l,i)-
linerepower(k,i,j)*repowerbal_alpha(l,i);

                alpha(k,i,j)$(lines2(k,i,j)) = alpha2(k,i,j)+ alpha2(k,j,i);

                display lines2, alpha;
                alpha(k,i,j)$(not lines2(k,i,j)) = 0;
                display lines2, alpha;
                put 'New cost: ' @10 newcost(l) /;
                put 'Cost %: ' @10 (100*(firstcost-newcost(l))/firstcost) /;
                put 'Loss %: ' @10 Loss(l) /;
                put 'MaxVangDiff: ' MaxVDiff(l) /;
```

```
                    lastcost = newcost(l);

                    NetStat=stat(l);
                    x=1;
               );
          );

******** if we have no further improvement x=0 to end the main loop ********

          if (x=0,
               put 'No Further Cost Reduction - Heuristic Termination' / /;
               put 'Optimal Cost: ' @15 lastcost /;
               put 'Cost %: ' @10 (100*(firstcost-lastcost)/firstcost) /;
          );

*********************** calculating the run-time *************************

SolveTime = SolveTime + sum(n,NLPtime(n));
elapsed = (jnow - starttime)*24*3600;
put 'Solve Time: ' SolveTime::4 /;
put 'Total Code Run Time: ' elapsed::4 / / ;
Time(itn)$(ord(itn)=yy)=elapsed;

);

*********************** ending the main loop (when) **********************
if (yy=jnum and x<>0,
          put / 'Jnum Lines Removed - Heuristic Termination' /
          put 'Last Best Cost: ' @15 lastcost /;
          put 'Cost %: ' @10 (100*(firstcost-lastcost)/firstcost) /;
);

display firstcost,lastcost;


put /  ;
put 'Time in Sec,' ;
loop(itn$(ord(itn) <= Jnum), put Time(itn) ',') ;
**********************************************************************
```