MATHEMATICAL OPTIMIZATION TECHNIQUES FOR MANAGING SELECTIVE CATALYTIC REDUCTION FOR COAL-FIRED POWER PLANTS

by

PASSAKORN PHANANIRAMAI

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
May 2011

To my mother, Dr. Mathana, my father, Vichai Phananiramai,

and to the loving memory of my grandmother, Kao-Si Chen.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervising professor, Dr. Jay Rosenberger for his invaluable advice, support, and motivation throughout the course of my graduate study at UTA. I have been extremely fortunate to have worked with such a great mentor who not only dedicated the time for me, he goes above and beyond what is needed of him. He introduced me to a very interesting research topic and patiently guided me through the challenges. He also constantly instilled me with great tools and experiences to become a better researcher and programmer. I am also grateful for his help with funding especially during the final semesters. I greatly appreciated his time, patience, understanding, and dedication that he has provided me over the course of my graduate study, and I will cherish this feeling forever. My deep gratitude also goes to Dr. Victoria Chen, who is also one of my committee members, for her tremendous assistance throughout the course of my Ph.D. program. I am extremely grateful for her help with recommendations for funding programs and also the GRA position which allowed me to work on a great research project. I would like to especially thank her for the advice she has given me to join the Ph.D. program and COSMOS. In addition, I am extremely grateful for all her concerns and assistance that she has provided me and also the comments and suggestions she has given on my research. My deep gratitude also goes to the rest of my committee members, Dr. H.W. Corley and Dr. Melanie Sattler, for their comments, suggestions, and reviews of my research. I would also like to thank Luminant for funding this research. I am especially grateful to Mr. Jay Snow for providing the data and also verifying the results.

My thanks also goes to former and present COSMOS students for the helpful discussions and friendships they have provided. I would like to thank the following: Bancha, Surachai, John, Narakorn, Wei-Che, Pin, Aera, Piyush, Diana, Nadia, Poovich, Chatabush, Subrat, Goh, Panita, Chingfeng, Panitarn, Weerawat, Huiyuan, Thuntee, Prashant, Chivalai, Siriwat, Prattana, Duraikannan. I am sorry if I have missed anyone.

Finally, I would like to thank all my family members for all their sacrifices and the motivation they have provided me throughout the years of my study. I would like to thank Niki for her support and motivation that she has given me. Thank you so much for taking great care of me and being there for me no matter what. I appreciate all her love and supports she has given me. I would like to thank my sister, Thaweeporn for always cheering me up and motivation. She is more than a sibling, she is my best friend who always provided me care and concern.

Last but certainly not least, I would like to express my deepest gratitude to my parents for everything they have done for me. Words cannot describe how grateful I am to them, and it is also impossible for me to ever pay them back what they have given me. I would like to thank them for all the sacrifice they have made for me. They always strive to provide me with the best education possible regardless of the costs. I would like to thank them for raising me up with great love and care and for providing me with the best in everything regardless the sacrifices they need to make. Thank you for all the advice, understanding, and motivation. Most importantly, thank you for the unconditional love that they have provided me.

April 15, 2011

ABSTRACT

MATHEMATICAL OPTIMIZATION TECHNIQUES FOR MANAGING
SELECTIVE CATALYTIC REDUCTION FOR
COAL-FIRED POWER PLANTS

PASSAKORN PHANANIRAMAI, Ph.D.
The University of Texas at Arlington, 2011

Supervising Professor: Jay Rosenberger

Coal is one of the most important energy sources in the U.S. However, it is also one of the biggest air polluters where the major emissions generated from coal combustion are oxides of nitrogen (NO_x). NO_x leads to ozone formation and make people more susceptible to respiratory illness. The US Environmental Protection Agency (EPA) has steadily tightened the regulation for NO_x emissions that can be discharged into the atmosphere. Many techniques and technologies can all assist with NO_x removal. However, to meet upcoming EPA mandates, more aggressive technique such as Selective Catalytic Reduction (SCR) is highly recommended. SCR is an emissions control technique that primarily reduces harmful emissions of NO_x. To maintain SCR performance, catalyst layers may be added or replaced to improve NO_x reduction efficiency. To make these changes, power plants must be temporarily shut down, and SCR maintenance during scheduled power plant outages can be very expensive. Consequently, developing fleet-wide SCR management plans that are both efficient at reducing NO_x and limiting operating costs would be extremely de-

sirable. In this dissertation, we propose an SCR management framework that finds an optimal SCR management plan for the fleet number of plants that minimizes NO_x emissions or total operating costs using mathematical optimization techniques. In the first part of this dissertation, we propose an SCR schedule generation and optimization algorithm (SGO) to solve the fleet SCR management problem. SCR schedule generation enumerates the set of possible outage schedules by recursion. An optimal set of these generated schedules are then selected by a 0-1 large scale integer program. The main approaches for SGO are recursion, branch-and-bound, and Pareto efficient frontiers. Although SGO is very effective and can yield a good result within a reasonable amount of time, the problem size can get larger and the computational time can increase exponentially. In the second part of this dissertation, we address this limitation by replacing SGO with a multi-commodity network flow problem (MCFP). We first formulated the MCFP as a relaxed problem to solve the fleet SCR management problem without a constraint on average daily NO_x. Edges are generated instead of schedules to represent the flow of all SCR catalyst layers for the fleet. The MCFP relaxed problem is solved by a 0-1 integer program. We then address the average daily NO_x constraint limitation by introducing MCFP with schedule elimination constraints (MCFPwSEC). The MCFPwSEC algorithm uses a single cut per iteration to incorporate an average daily NO_x constraint into the model. We then reduce the computational time further with the introduction of a multi-cut MCFPwSEC. Multi-cut MCFPwSEC similarly eliminate infeasible solutions per iteration based on a heuristic algorithm. Then, we further explore additional ways to reduce the computational time further with discussions on a reactor potential (RP) constraint. Finally, we discuss future extensions of this research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Coal is considered as one of the most important energy sources in the U.S. It is accountable for roughly half of the electricity we use and one fourth of our total energy [3]. However, it is also one of the biggest industrial air polluters in the U.S [4]. One of the major emissions generated from coal combustion is *oxides of nitrogen* (NO_x). In a year a typical coal plant generate around 10,200 tons of NO_x [4]. NO_x leads to the formation of ozone that could potentially inflames the lungs, burning through lung tissue and making people more susceptible to respiratory illness [5]. NO_x also leads to fine particle formation and acid precipitation. In addition, NO_x causes health impacts in and of itself, and thus is one of the six regulated criteria pollutants. The US Environmental Protection Agency (EPA) has steadily tightened regulations for allowable amounts of pollutants that can be discharged into the atmosphere [5]. For fossil fueled power plants, regulations are now in place for the amounts of sulfur dioxide (SO_2), carbon monoxide (CO), volatile organic compounds (VOCs), particulate matter (PM), and NO_x that can be released into the atmosphere. Equipment and operating modifications can reduce NO_x emissions. Technologies such as low NO_x burners, staged combustion, gas recirculation and low excess air firing can all assist with NO_x removal. However, to meet upcoming EPA mandates, more aggressive reduction techniques, such as *Selective Catalytic Reduction* (SCR), need to be used [5].

SCR is an emissions control technique with the primary purpose of converting NO_x in exhaust gases into harmless nitrogen gas and water. The SCR process consists of injecting ammonia (NH_3) into boiler flue gas and passing the flue gas through a catalyst bed where the NO_x and NH_3 react to form nitrogen and water vapor [4]. As the SCR reduces NO_x, its performance deteriorates. To maintain the performance, SCR catalyst layers may be added, removed, or replaced to improve NO_x reduction efficiency. However, to make these changes, the power plant must be temporarily shut down, so SCR maintenance occurs during scheduled power plant outages, which are expensive. Consequently, developing fleet-wide SCR management plans that are both efficient at reducing NO_x and limiting operating costs would be extremely desirable.

1.2 Electricity Generation Process

The basic process of a coal-fired power plant is to convert the chemical energy in coal into thermal energy or heat, thermal energy into mechanical energy of a turbine, and mechanical energy into electrical energy. Figure 1.1 displays an overview of a coal-fired power plant. Coal from the mine is pulverized and delivered by a conveyor belt to the boiler where a mixture of coal and air ignites. Intense heat from the burning coal converts a large amount of water in the boiler into steam that spins the turbine to generate electricity. In addition, there are numerous sub-processes associated with the boiler (not shown in Figure 1.1). Burning coal produces harmful emissions (e.g., NO_x, PM, and SO_2) that are vented from the process (not directly from the boiler). NO_x emissions are formed when molecular nitrogen and oxygen naturally occurring in the air combine at the high temperatures present in the boiler (thermal NO_x), and when nitrogen in the coal is oxidized during the combustion process (fuel NO_x). Many of these harmful emissions can be decreased using a variety of emissions control technologies. For example, NO_x emissions at the stack can be significantly reduced

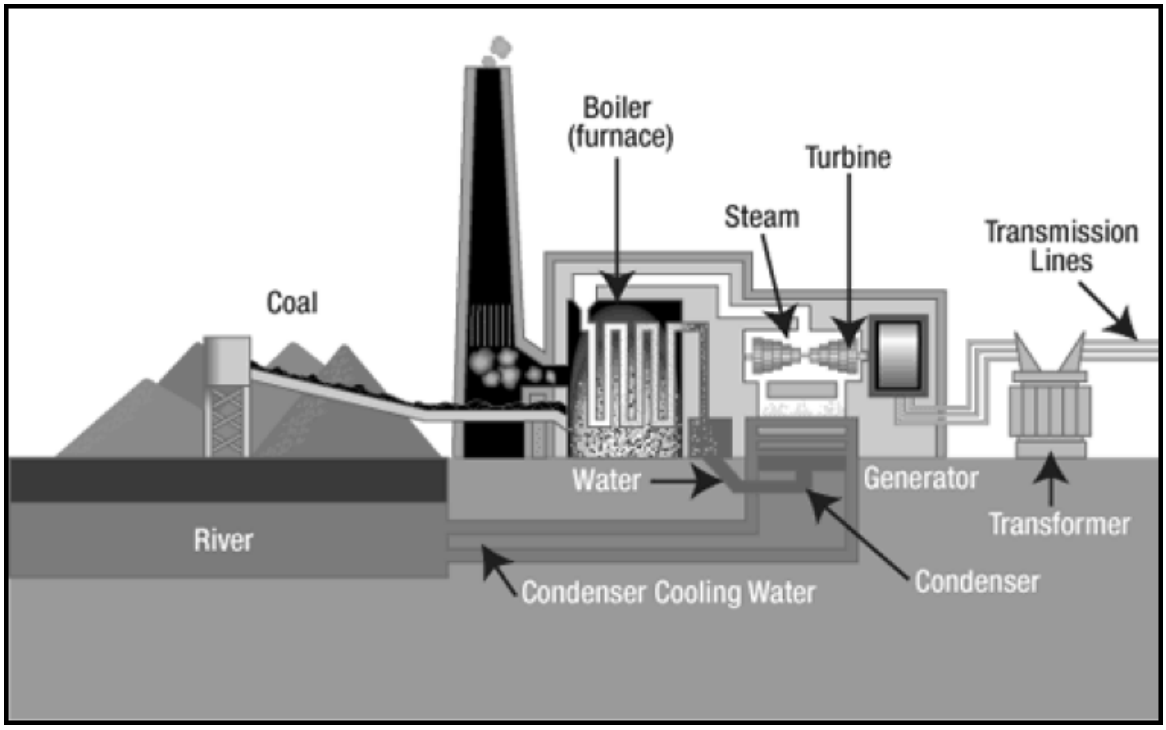


Figure 1.1. Overview of a coal-fired power plant [1].

through SCR technology. In Figure 1.1, the SCR would be implemented between the boiler and the stack. In the SCR, NO_x emissions travel through a series of catalyst layers to react with ammonia (NH_3). This reaction can convert NO_x and NH_3 into harmless byproducts, mainly nitrogen and water. We refer to the NO_x that comes from the boiler as *inlet* NO_x, and the NH_3 injected into the SCR as *NH_3 injection*. Not all of the inlet NO_x and NH_3 injection reacts in the SCR, and the remaining NO_x and NH_3 are referred to as *outlet* NO_x and *NH_3 slip*, respectively.

The amount of outlet NO_x and NH_3 slip depend upon the *reactor potential* (RP) of the catalyst in the SCR that degrades over time. As the SCR degrades, NH_3 injection is ramped up to maintain reasonable levels of outlet NO_x, but this increases NH_3 slip as well. Since high levels of NH_3 exposure is hazardous to humans, it is necessary to maintain low levels of NH_3 slip. In addition, increased NH_3 injection

in turn affects the long-term performance of the SCR. NH_3 slip can form particles that could potentially corrode downstream equipment and can cause plugging in the SCR that can be expensive to maintain. Overuse of NH_3 injection is also expensive. Therefore, optimal SCR management involves adding and/or removing catalyst layers during scheduled outages, so as to maintain high levels of SCR potential reactivity and thereby control NO_x emissions, NH_3 slip, and operating costs across multiple plants. Specifically, the problem is to find an optimal SCR plan that minimizes NO_x emissions or costs given a scheduled outage plan for each plant where during these outages catalyst layers maybe added or removed and replaced to improve NO_x reduction efficiency.

1.3 Research Overview and Contributions

In this dissertation, we propose an SCR management framework that primarily focuses on SCR catalyst management using mathematical optimization techniques. Since SCR catalyst layers degrade over time, in order to maintain their performances, SCR catalyst must be maintained. In order to do this, catalyst layers may be added or replaced to improve NO_x reduction efficiency. To make these changes, power plants must be temporary shut down for the maintenance, which is expensive. Therefore, an effective SCR management plan that is both efficient in reducing NO_x emissions and limit operating costs would be extremely beneficial to SCR catalysts users. Consequently, the main objective of the SCR management framework is to find an optimal SCR management plan for a fleet of plants that would maximize NO_x reduction or minimize the total operating cost given a schedule of outages for the fleet. Although SCR has been implemented for over 50 years, only recently has there been a focus on the optimization of SCR management. The main focus of most literature on SCR management has been on optimizing the process design. There is also commercial

software provided by SCR catalyst vendors to address SCR catalyst management on an individual plant basis. Even though the focus on SCR catalyst management on a single plant does yield a good result, there are fleet-wide considerations that when applied would yield a better solution. Therefore, in this dissertation, we propose an SCR management framework that maximizes NO_x reduction or minimizes the total operating costs over a fleet of plants given a scheduled outage plan using mathematical optimization techniques.

In Chapter 3, we describe an *SCR schedule generation and optimization* (SGO) algorithm. The SGO algorithm consists of two main modules, *SCR schedule generation module* and *SCR optimization module*. The SCR schedule generation module enumerates a set of possible outage schedules for all plants in the fleet using recursion. From these generated schedules from the SCR schedule generation module, the SCR optimization module that uses COIN-OR CBC as the solver, finds an optimal set of outage schedules for the fleet. The underlying optimization model is a 0-1 large scale integer programming model. We demonstrate the effectiveness of the algorithm by providing computational experiments based upon two different types of NO_x reduction policies, the fixed NH_3 slip policy and the fixed NO_x policy. For the fixed NH_3 slip policy, there are computational experiments that consider different objectives and also Pareto optimal efficient frontiers. Then, for the fixed NO_x policy, we discuss computational experiments based upon a modified version of a real-world problem instance with six plants over a five-year time horizon. This input is used consistently throughout the dissertation for comparison and analysis.

From the proposed model in Chapter 3, we have found that although the enumeration of all possible outage schedules would yield a good set of schedules within a reasonable computation time, the number of outages, the time horizon, and the number of plants increases the computation time exponentially. Therefore, in Chapter

4, we propose a *multi-commodity flow model* (MCFP) to replace the SGO algorithm described in Chapter 3. Instead of generating schedules, we generate edges that represent all SCR catalyst layers flowing from the start of the time horizon and through outages until the end of the time horizon. Given a scheduled outage plan, the MCFP are solved by a 0-1 large scale integer programming model using the CPLEX 12.1 callable library. We first formulate a relaxed MCFP while ignoring the average daily NO_x constraint that was incorporated in Chapter 3. From costs and schedule comparison, we observe that the average daily NO_x constraint is extremely crucial to the model. Therefore, we would like to incorporate the average daily NO_x constraint in the MCFP. In order to incorporate the average daily NO_x constraint to the model, we introduce the *MCFP with schedule elimination constraints* (MCFPwSEC). The MCFPwSEC algorithm uses a single cut per iteration to remove infeasible solutions that do not meet the minimum average daily NO_x reduction requirement from the model. From the explorations of the cuts, we observe that each cut only marginally improves the solution. Therefore, in order to speed up to algorithm performance, we introduce multi-cut MCFPwSEC. Instead of cutting off infeasible solutions one by one, we cut off multiple infeasible solutions by a heuristic method. We then explore for ways to reduce the computation time further. From the solutions, we found that the weighted reactor potential (RP) over the time horizon derived from MCFP is an upper bound to the RP from the solution schedule. From this relationship, we can introduce an RP constraint to the optimization model that represents this relationship. However, since RP values are fractional, we also introduce fractional coefficients to the problem. In future research, we can potentially improve the algorithm performance by introducing independent set constraints based upon the RP constraint. In summary, we have found MCFP to be significantly faster than SGO. While MCFP has been studied extensively in the literature, their application in SCR catalysts man-

agement has not yet been studied. Finally, Chapter 5 discuss conclusions and future directions.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we discuss a general overview of selected literature related to SCR management and multi-commodity flow problems. The selected literature does not contemplate the comprehensiveness of the scope but merely a discussion of an overview of such related papers in the literature. In Section 2.1, we discuss an overview of literature related to SCR management and existing applications that can be found in industry. Furthermore, in Section 2.2, we discuss an overview of algorithms and applications based upon multi-commodity flow problems.

2.1 Overview of SCR Management

Staudt and Engelmeyer [2] stated that in order to optimize the catalyst consumption, the catalyst cost and operation of the facility are optimized simultaneously to achieve the lowest cost to produce power. The stated objective has led to many trade-offs such as catalyst consumption, the frequency and duration of outages, NH_3 slip, NO_x reduction, and baseline NO_x. Furthermore, catalyst activity is defined as the ability to facilitate the NO_x reduction reactions, which is cased by the impurities in the gas stream that over time will deposit on the catalyst and block exhaust gas from reaching active sites within catalyst.

Based on Muzio, Quartucy, and Cichanowicz [6] and Pritchard et al. [7], there are three major factors that affect the catalyst deactivation. The first factor is sintering of the catalyst due to high temperatures. The second factor is due to alkaline

metals, earth metal masking, and/or arsenic oxide. The last factor is catalyst plugging.

From Staudt and Engelmeyer [2], "Most SCR reactors are designed with up to four available levels of catalyst. When the system is new, with fresh catalyst, at least one level is typically empty as shown in Figure [2.1]." When the SCR performance drops to an unacceptable level that occurs around two years, a new catalyst level is added to the spare layer. Later, when SCR reactor potential drops again, an old catalyst level that generated the lowest activity is replaced with new catalyst. Thus, this will increase total catalyst activity.

According to Cichanowicz and Muzio [8], there are three options to maintain SCR performance. The first option is to install a new catalyst into a spare layer or replace the old layer with new catalyst. The second option is to install regenerated catalyst. This option has lower cost than the first option, but SCR performance will be lower when using regenerated catalyst than when using new catalyst. The last option is in-situ cleaning or regeneration. Cleaning is when a catalyst is taken out and cleaned thoroughly with chemical while regeneration is when catalyst is restored to a like new condition. This option can be done in a shorter period of time such as two to three days compared with other options that require two to three weeks. Although, it can be done in a shorter period of time, the reactor potential will not be restored as with the other two options.

Cichanowicz, Smith, and Muzio [9] estimated that the capital cost of SCR is \$125 per kW depending on the level of NO_x reduction, type of catalyst, and complexity of the retrofit. Due to capital cost of catalyst replacement, the maintenance outage schedule, and the SCR performance, Cichanowicz, Smith, Muzio, and Marchetti [10] provided five options as follows. The first option is to replace catalyst as planned. The second option is to delay catalyst exchange and increase NH_3 injection. The

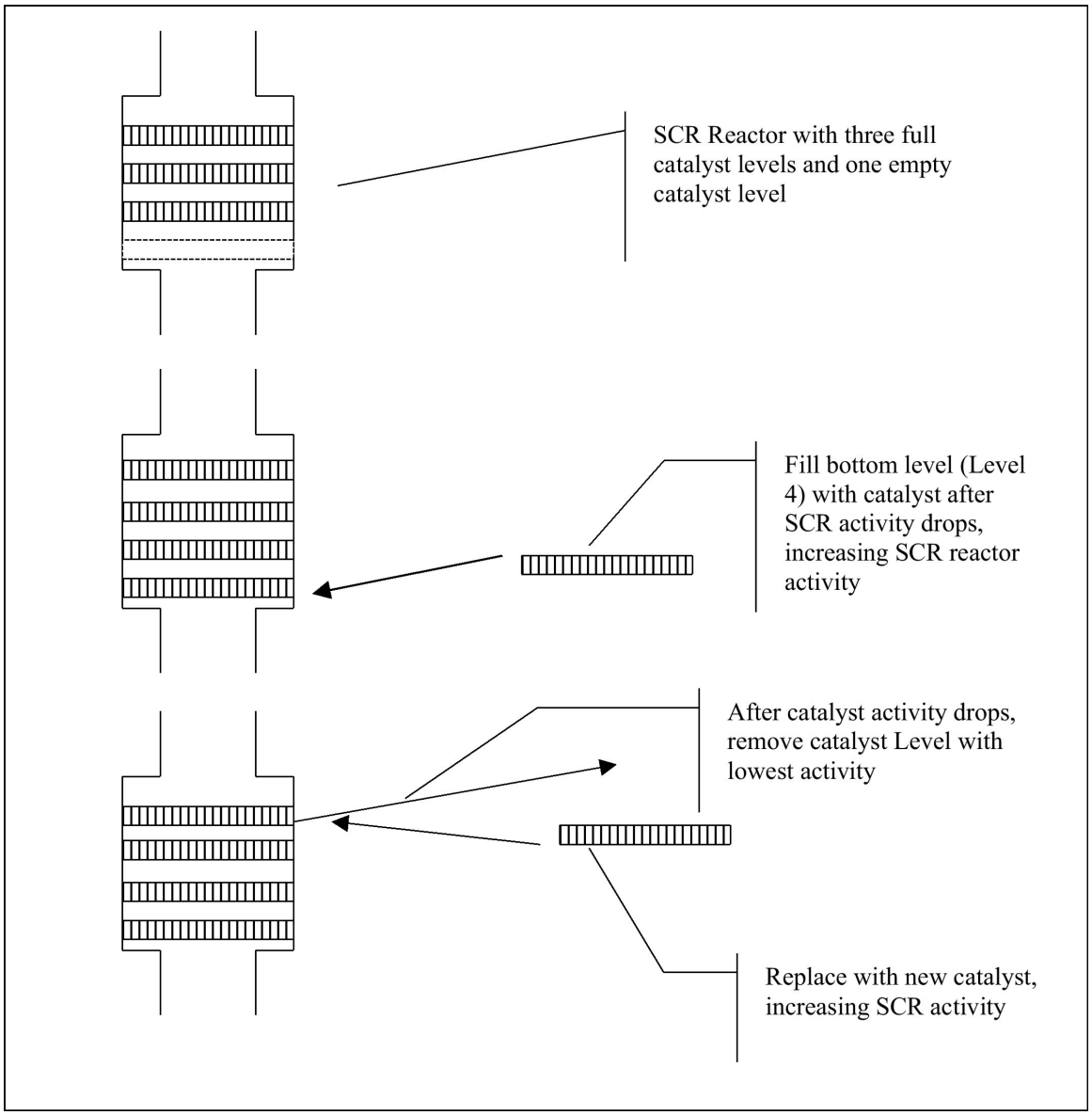


Figure 2.1. Example of add and replace sequence of SCR catalyst layers [2].

next option is to delay catalyst exchange and maintain NH_3 injection. The fourth option is to accelerate an outage for early replacement. The last option is to perform on-line cleaning. All of the above have pros and cons that cause the catalyst to be challenging to manage.

Rubin, Salmento, and Frey [11] discussed an effective emissions control for coal-fired power plants using integrated environmental control (IEC) concepts involving combined SO_2/NO_x removal processes in combination with pre-combustion and combustion control methods.

Pritchard and DiFrancesco [12] described SCR catalyst management with goals of NO_x reduction, Hg oxidation, SO_3 emissions, and operation flexibility by physical inspection of the plants, collection of data, and tests to predict the future performance. Mi [13] optimized NO_x emissions by using Computational Fluid Dynamics (CFD) for SCR coal-fired steam power plants to improve the chemical process of the system. Another way to improve the effectiveness of the chemical process is by using the NH_3 injection grid (AIG) that is essentially an optimization of the chemical process of NH_3 injection by adjusting the AIG [14].

Chen and Frey [15] optimized NO_x and operating costs of the SCR by process design using stochastic optimization and programming. They described methods for optimization of process technologies by considering the distinction between variability and uncertainty. These methods are developed and applied to case studies of NO_x control for integrated gasification combined cycle systems. Another related method described by Rubin, Diwekar and Frey [16] also optimized the process design using deterministic and stochastic optimization.

Grabitech [17] provided a NO_x optimization software, multisimplex, that is based on three different theories: evolutionary operation, simplex algorithms, and fuzzy set theory. The basic idea in evolutionary operation is to replace the static operation of a process by a continuous systematic scheme of slight perturbations in the control variables. The effect of these perturbations is evaluated, and the process is shifted in the direction of improvement similar to the Simplex algorithm. The fuzzy set theory allows several goals to be handled at the same time. Multisimplex calculates

new settings for the next trial on the journey towards the optimum. Another widely used SCR management tool is called CatReact [18] that was developed as part of an EPRI-sponsored collaboration between FERCo and JEC Inc.

From current literature, there have not been studies done on applying mathematical optimization techniques on SCR management. While the improvement of the process design is also important, once the process design has improved the process to its maximum capabilities, the need to optimize the SCR management schedule is greatly desired. Since SCR catalyst layers are expensive in themselves, the costs to maintain them are also very expensive. The total cost that plants spend on SCR management can be greatly reduced while limiting NO_x emissions to a satisfactory level by applying mathematical optimization techniques to SCR management. In Chapter 3, we propose an SGO algorithm that finds an SCR management plan that results in lowest costs or highest NO_x reduction. The proposed algorithm is a model that can address a fleet of plants, which also addresses the single plant limitation that is currently commercially available.

2.2 Overview of Multi-Commodity Flow Problem

A network flow problem is a mathematical programming problem that determines an optimal path flow through the nodes from a source to a sink. A path consists of a set of edges (arcs) that must satisfy some restrictions in order to flow from one node to another. Network flow is a directed graph, $G(N, E)$ where N is a set of nodes and E is a set directed edges in the network [19] [20] [21].

A type of network flow problem that is widely used is the minimum cost flow problem. The problem is used to determine the lowest cost of shipment of a commodity through a network while satisfying demands at each node. A multi-commodity flow problem (MCFP) occurs when several commodities use the same underlying net-

work [22] [21] [23]. There are numerous real-world applications such as transportation problems, shortest path problems, assignment problems, and routing problems [24]. Hu [25] suggested many practical applications, for example, communication networks, traffic problems, and transportation systems. Consequently, large numbers of real-world applications are modeled as MCFP.

2.2.1 MCFP Algorithms

Research on network flow problems has been around before linear programming. Earlier work on network flow problems was conducted by Kantorovich [26], Hitchcock [27], Koopmans [28][29], and Dantzig [30]. Ford and Fulkerson [31], who proved the maximum flow and minimum cut theorem, presented a minimum cut and minimum path problem approach for the maximum flow problem for the multi-commodity flow problem. The method created two new special nodes that are now called source and sink.

Ford and Fulkerson [22] proposed a new algorithm approach for a general case of maximum flow problems by means of a large linear program. They introduced the simplex computation for an arc-chain formulation of the maximum flow multi-commodity flow problem. In order to deal with the large number of variables, the algorithm treated non-basic variables implicitly by also introducing the shortest chain algorithm to join points in a network.

Gomory and Hu [32] considered connected networks consisting of nodes N_i and arcs b_{ij} connecting the ith and jth nodes. Assuming that $b_{ij} = b_{ji}$, the multi-terminal network flow problem finds maximum flow between all pairs of nodes. For the linear programming approach see Gomory and Hu [33].

Hu [25] considered a generalization analogue of the maximum flow and minimum cut theorem for the multi-commodity flow problem. The work is an extension to the

traditional maximum flow and minimum cut theorem. It considers many kinds of flows simultaneously in a network and decides whether a set of flow requirements is feasible or not in a given network.

Tang [34] presented a special class of multi-commodity flow problems. The special class of bi-path networks was defined. The result can be extended to solve cases with time-varying requirements using linear programming.

Tomlim [35] formulated the minimum cost multi-commodity flow problem in both node-arc and arc-chain forms. For problems with directed edges, both formulations are similar. However, the advantage on the arc-chain formulation is its ability to deal with both directed and undirected edges.

Kennington [36] presented a survey of known results and algorithms for linear multi-commodity flow problems. The work contains extensive lists of detailed discussions and references of existing results, algorithms, and applications of linear multi-commodity flow problems.

Bixby and Cunningham [37] presented an algorithm that converts linear programming to a network problem. If A is in a standard form, the computational effort is bounded by O(number of rows * number of nonzeros) of A.

Fontes, Hadjiconstantinou, and Christofides [38] presented a branch-and-bound approach to solve a single source uncapacitated minimum cost network flow problem where the cost function is a concave function. They concluded that the branch-and-bound algorithm has shown a better performance than available alternative methods for same type of problems.

2.2.2 MCFP Applications

MCFP problems have been used to solve many real-world applications. Bellmore and Ratliff [39] discussed a multi-commodity flow problem in the area of military

logistics. The problem is to find an optimal allocation of resources that is similar to the maximum flow problem. The algorithm finds an optimal strategy for reinforcing arcs and nodes in a multi-commodity network.

Baker [40] presented a network flow problem for project selection. The algorithm requires no prior knowledge of a network simplex algorithm. The author provided a simple example and approach by a simple activity-on-node diagram.

Farvolden et.al. [41] considered a transportation problem involving the routing of customers. Each customer is considered as one commodity. The problem is to route the customers over a given network of transportation services to minimize total costs, subject to the capacity constraints. They presented a new solution approach based on primal partitioning and decomposition techniques.

Hane et.al. [42] presented a fleet assignment model for airline fleet assignment. They modeled the problem as a multi-commodity flow problem with side constraints. They also increased the computational efficiency by introducing methods such as an interior-point algorithm, dual steepest edge simplex, cost perturbation, model aggregation, branching on set-partitioning constraints, and prioritizing the order of branching.

Barnhart et.al. [43] presented a string based fleet assignment and routing models for airline fleet assignment. Connectivity constraints that are a type of flow constraint were introduced where it requires the aircraft to fly in a sequence. The problem was solved with branch and bound, and it includes column and cut generation.

McBride [44] discussed advances over the years in solving the multi-commodity flow problem. The work provided and discussed improvements in both hardware and algorithmic performance to solve ever increasing problem sizes. Extremely large network flow problems are solved by using a primal basis-partitioning algorithm.

Rosenberger et.al. [45] presented a robust fleet assignment model. Their model considered the possibility of disruptions and accounted for them during the planning phase to reduce unplanned costs.

Pilla et.al. [46] provided a fleet assignment model that employs a two-stage stochastic program and a design and analysis of computer experiments approach.

While the multi-commodity network flow model has been studied extensively in the current literature and has been applied to numerous real-world applications, the work on applying the multi-commodity network flow model as well as mathematical optimization techniques in general for SCR management has not yet been studied in any literature. In Chapter 4, we propose the MCFP to solve the fleet SCR management problem. The model is introduced to replace the SGO algorithm in Chapter 3 to improve computational efficiency.

CHAPTER 3

SCR SCHEDULE GENERATION AND OPTIMIZATION ALGORITHM

3.1 Introduction

In this chapter, we describe the SGO algorithm that finds an optimal SCR management plan for a fleet of plants that minimizes NO_x emissions or total operating cost given a scheduled outage plan. The SGO algorithm consists of two main modules: SCR schedule generation and SCR optimization. The SCR schedule generation module enumerates a set of possible outage schedules for all plants in the fleet using recursion with two main algorithms: *generate timelines* and *find timelines*. The SCR optimization module uses COIN-OR CBC to find an optimal set of outage schedules in the fleet from these generated schedules. The underlying optimization model is a 0-1 large scale integer programming model. In addition, we employ two NO_x reduction policies that consist of the fixed NH_3 slip policy and the fixed NO_x policy. We demonstrate the effectiveness of the algorithm by providing computational experiments based upon these two types of NO_x reduction policies. For the fixed NH_3 slip policy, there were computational experiments on optimization problems with different objectives, and we consider Pareto optimal efficient frontiers. Then, for the fixed NO_x policy, we discuss computational experiment based upon an example with six plants over a five-year time horizon. This six-plant five-year example is a modified version of a real-world problem instance. From this computational experiment, the input is then kept consistent throughout the dissertation for comparison and analysis.

In this section, we describe the input information and an overview of the architecture of the SGO algorithm. The input information includes three text files—a

plant file that includes information on the power plants, an outage file that includes information on the currently scheduled outages, and a parameter file that includes the fleet-wide information. These outages span over a predetermined time horizon that is usually about five years. The output is an optimal outage plan in the same format as the inputted outage file with outages that span the same time horizon. An overview of the architecture of the SGO algorithm is shown in Figure 3.1.

The input information can be categorized into four main categories that consist of plant information, layer information, outage information, and global information. The summary of the input data information is shown in Table 3.1.

Table 3.1. Summary of input data information

Plant Information	Layer Information	Outage Information	Global Information
Plant number	Degradation rate	Start date	Fleet-wide constraints
Number of installed SCR layers	Blockage rate	End date	Fleet-wide parameters
Power generation plan	Volume	Plant number	High budget
NH_3 slip	Surface area	Action	Low budget
Minimum NO_x reduction	Catalyst activity	Layer number	Number of Pareto points
Maximum operating costs	Flue gas flow rate		
Inlet NO_x	Dates of last activity		

Plant information includes characteristics of each plant. The number of installed SCR layers is the number of layers installed at the beginning of the time horizon. This number is used to determine the RP as well as the change and add sequences as shown in Figure 2.1. The position of these layers that are filled from the bottom up, also affect the degradation rates and blockage rates as shown in Figure 1.1, where the closer the catalyst layers to the boiler, the higher the degradation and blockage rates. A power generation plan is used as a constraint where the plants would need to meet the minimum fleet-wide power generation plan that is pre-specified. NH_3 slip

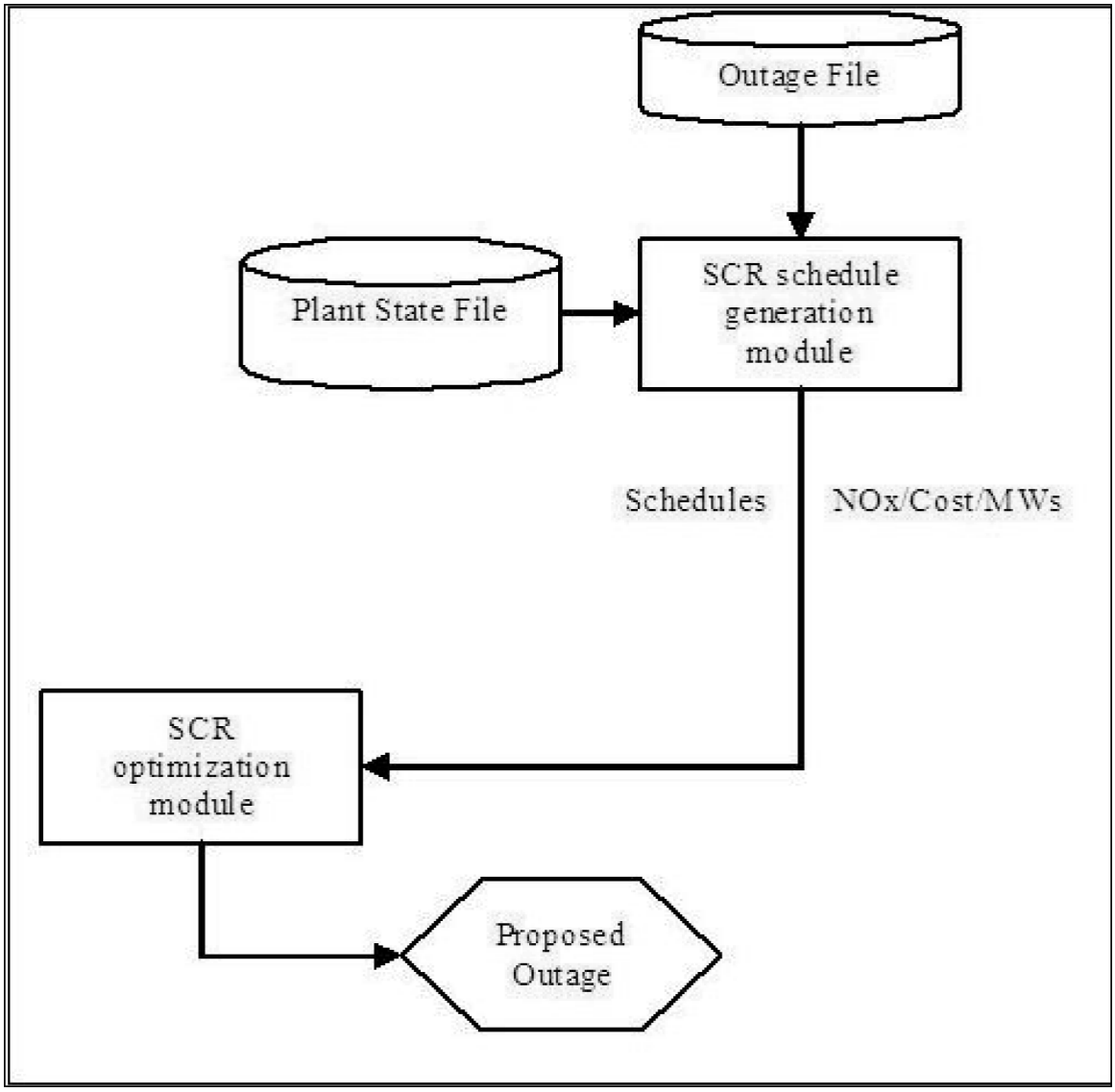


Figure 3.1. Overview of SGO algorithm.

values are inputted in two levels; the level in which is the assigned plant is currently using and the maximum level allowed by the plant. These levels are then used to generate schedules for the fixed NH_3 slip policy that will be described in Section 3.4. Minimum average daily NO_x reduction is a constraint where each plant has to meet this required minimum. Maximum operating costs is the budget allowed per plant

that consists mainly of catalyst, labor, reagent, fan power, and electricity costs. These costs values are determined by the plant, the layer, the outage, and the pre-specified information. Inlet NO_x, generally measured in pounds per hour, is the amount of NO_x entering the SCR that is used to find the amount of outlet NO_x leaving the SCR by taking into consideration the layer information. Layer information includes specific characteristics of each layer that can be different for each layer as well as in different plants. There are typically four available layer slots per plant where two slots are typically filled at the beginning of the time horizon. A degradation rate is typically given in guarantee usage hours by the catalyst manufacturer. The number of actual hours can vary for each layer depending upon its operating conditions. Blockage rate is also a factor that affects the degradation rate of a catalyst layer. Over time, blockage is caused by excess NH_3 slip or ashes from coal combustion that form particles inside the catalyst layer. Consequently, it gradually degrades the SCR catalyst performance that reduces the SCR catalyst reactor potential. Volume, surface area, and catalyst activity are general catalyst layer characteristics where the higher their values, the higher the reactor potential of the layer. Flue gas flow rate is inversely proportional to the reactor potential so the higher the flow rate, the more layers need to be changed. Dates of last activity of each layer will determine how long the layer has been in the system. As a general rule, where all of the layer slots are filled, the layer that has been in the system the longest will be changed first. If there is a tie, generally the layer closest to the boiler is the one that will be changed. Furthermore, the layer characteristics depend upon the position slots and the distance away from the boiler that will directly impact the SCR reactor potential, where the higher the reactor potential, the higher NO_x reduction it will achieve. Outage information includes what is planned to be done for all outages in the time horizon. This information includes the start and end date for each scheduled

outage, the plant it is associated with, the layer action, and the layer position in the SCR. Layer actions include catalyst additions or changes where the layers being added or changed can be new, regenerated, or cleaned. A new layer is a brand new layer that has not been used before, a regenerated layer is a used layer that has been restored very close to new layer conditions, and finally a cleaned layer is also a used layer that has been taken out and cleaned thoroughly with catalyst. These three types of layer actions usually have different operating costs and NO_x reduction efficiency implications with the efficiency of a new layer being greater than that of a regenerated layer, and that of a regenerated layer being greater than that of a cleaned layer. This outage information is used to generate timelines that will be described in Section 3.3. After the timelines are generated the SCR optimization model that will be described in Section 3.5, will then select an optimal schedule for each plant during time horizon.

Global information includes all fleet-wide constraints and parameters that are associated with the whole fleet. These include minimum power generation, minimum NO_x reduction, maximum operating costs, and the time horizon. High budget, low budget, and the number of Pareto points are used to determine the Pareto optimal efficient frontiers that will be discussed in Section 3.6.3.

3.2 SCR Reactor Potential and NO_x Reduction

In this section, we describe equations for NO_x reduction and RP that are used throughout the dissertation. The policy used for NO_x reduction depends upon what is specified prior to the start of the algorithm. The policy consists of either the fixed NO_x policy or the fixed NH_3 slip policy that will be described in Section 3.4. The RP, a measure of the overall potential of the reactor to reduce NO_x for a given catalyst layer at time t, can be calculated as follows.

$$RP_{\text{layer}}(t) = \frac{K_0 e^{\frac{-t}{T}}}{A_v}, \tag{3.1}$$

where K_0 is the initial catalyst activity value (in $\frac{m}{hr}$), T is the degradation rate of the catalyst (hr), and A_v is the area velocity ($\frac{m}{hr}$).

In addition, the RP for a given plant at time t can be calculated as follows.

$$RP_{\text{ plant}}(t) = \sum_{\text{layer}\in\text{plant}} RP_{\text{layer}}(t). \tag{3.2}$$

The percentage of NO_xreduction is an increasing function of NH_3 slip and $RP(t)$ and is given by

$$DNOX_{\%}(t) = f(RP_{\text{plant}}(t), NH_3slip(t)), \tag{3.3}$$

where $NH_3slip(t)$ is the NH_3 slip at time t. As mentioned previously, we assumed that $NH_3slip(t)$ is set at either of two constant values, usually two ppm or four ppm for all values of t. The formula for f is based upon the manufacturer of the catalyst, and the one used is this research is considered proprietary by the sponsor of the research. To find an anticipated daily NO_x reduction and average RP for a schedule s that will be discussed in Section 3.3, we integrate over the time horizon as follows:

$$\overline{DNOX}_s = \int_{\underline{t}}^{\overline{t}} \frac{InletNO_x \times DNOX_{\%}(t)}{\overline{t} - \underline{t}}\, dt, \tag{3.4}$$

$$\overline{RP}_s = \int_{\underline{t}}^{\overline{t}} \frac{RP_s(t)}{\overline{t} - \underline{t}}\, dt, \tag{3.5}$$

where $\underline{t}$ is the start of the time horizon, $\overline{t}$ is the end of time horizon, and $InletNO_x$ is the amount of inlet NO_x into the SCR. Observe that between two outages, average RP is given by

$$RP_{\text{ plant}} = \sum_{\text{layer}\in\text{plant}} \frac{T_{\text{layer}}\left(RP(\underline{t}) - RP(\overline{t})\right)}{\overline{t} - \underline{t}}. \tag{3.6}$$

Consequently, average RP for a schedule can be derived by taking a weighted average of (3.6) between the outages. However, calculating average NO_x reduction requires numerical integration.

3.3 SCR Schedule Generation Module

The primary purpose of the SCR schedule generation module is to enumerate a set of possible outage schedules where each generated schedule includes the following characteristics:

1. One power plant to which the schedule pertains.
2. A set of outages from the inputted outage file that maintain the following qualities:
   - The first outage in the schedule is the same as the first outage of the currently scheduled outage of the power plant.
   - The last outage in the schedule is the last outage of one of the plants in the currently scheduled outages.
   - One SCR catalyst layer from the power plant will be either added or changed in each outage.
   - All consecutive outages in the schedule will be within a predetermined window of days apart, usually around 270 to 450 days.
3. The maximum level of NH_3 slip.
4. An anticipated average daily RP of the schedule.
5. An anticipated average daily NO_x reduction of the schedule. (The NO_x equation is a function of allowable NH_3 slip and RP that is an increasing function where the higher the value of either or both of NH_3 slip and RP, the higher the NO_x reduction. The actual function will be different based upon different

manufacturers where the NO_x equation used in this dissertation is considered proprietary and cannot be describe here)

6. An anticipated operating cost of the schedule.
7. An anticipated power generation of the schedule.

The first step in the SCR schedule generation module is to read in a set of original schedules. Each original schedule includes an original timeline and a set of parameters, like the amount of NH_3 slip. Each original timeline represents a sequence of outages assigned to the same plant. Each outage includes a starting time, an ending time, an originally assigned plant, and a catalyst layer decision. The potential catalyst layer decisions are: add a new layer (*ADDnew*), add a regenerated layer (*ADDregen*), add a cleaned layer (*ADDclean*), change the oldest layer with a new layer (*CHANGEnew*), change the oldest layer with a regenerated layer (*CHANGEregen*), or clean the oldest layer (*CHANGEclean*). The second step in the SCR schedule generation module is to enumerate a set of new timelines, using the two algorithms, generate timelines and find timelines described as Algorithms 1 and 2, respectively. Like the original timelines, each new timeline is a sequence of outages assigned to the same plant. However, the outages may have a newly assigned plant and a new catalyst layer decision. Using these new timelines, the SCR schedule generation module creates two schedules for each timeline. For each timeline, the method used to generate schedules depends on two policies, which are the fixed NO_x policy and the fixed NH_3 slip policy. The preference of the policy will depend upon the emphasis on whether we would like to control either NO_x or NH_3 . These policies will be discussed in Section 3.4. Finally, the SCR schedule generation module calculates average NO_x and operating costs and returns these schedules. Table 3.2 shows the list of symbols used in Algorithms 1 and 2 and Figure 3.2 shows an overview of the find timelines and the generate timelines algorithms.

Table 3.2. List of sets and indices symbols in Algorithms 1 and 2

Sets and indices symbols	Descriptions
P	Set of all plants information fleet-wide as shown in Table 3.1
O	Set of all outages information fleet-wide as shown in Table 3.1
T	Set of generated timelines.
O^+	Set of next outage event in the timelines.
o	each outage in the set O.
o^1	first outage event in the timelines.
o^+	next outage event in the timelines.
o^n	outage event n in the timelines.
p	each plant in the Set P
$o^1(p)$	first outage in the schedule assigned to plant p
τ	each timeline in the Set T

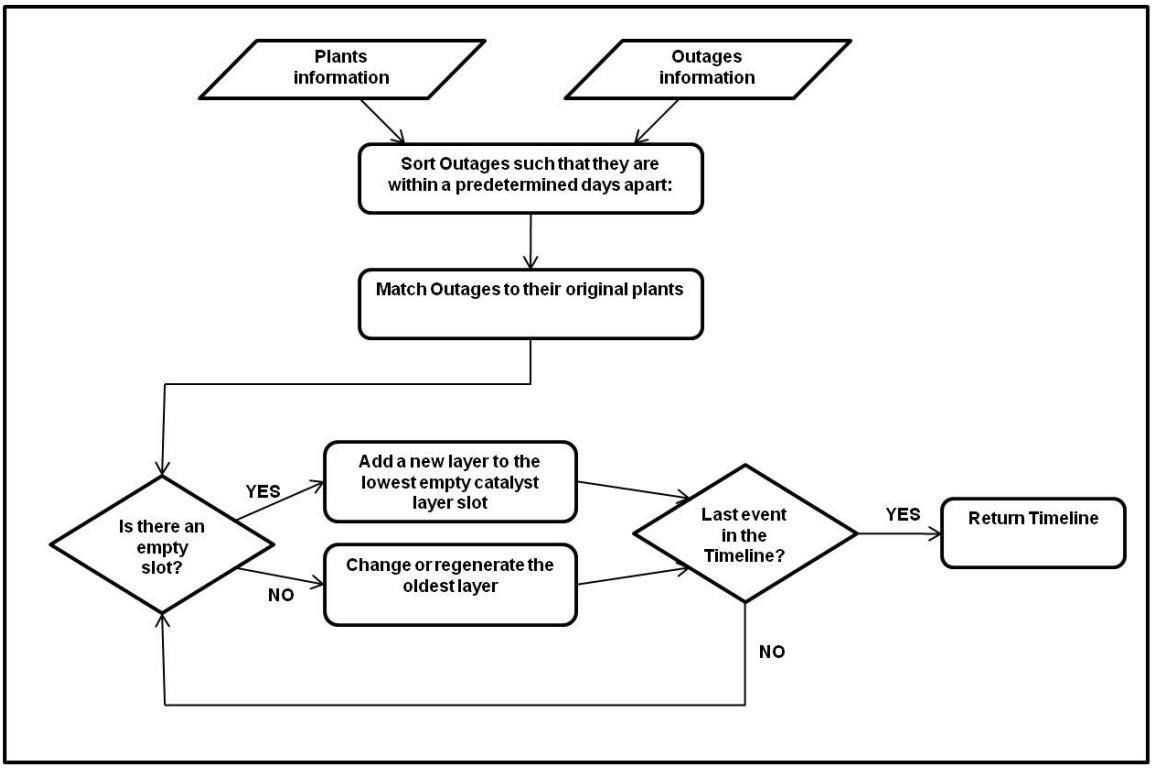


Figure 3.2. Overview of generate timelines and find timelines algorithms.

Algorithm 1 Generate timelines algorithm

Let P and O be the sets of plants and the outages.

Let timeline set $T = \emptyset$ be the set of generated timelines.

for each outage $o \in O$ **do**

if outage o is the last outage of one of the original timelines **then**

Let outage set $O^+(o) = \emptyset$

else

Let outage set $O^+(o)$ be such that each outage $o^+ \in O^+(o)$ is such that the ending time of outage o and the starting time of outage o^+ are within a predetermined window of days apart, usually around 270 to 450 days.

end if

end for

for each plant $p \in P$ **do**

Let outage $o^1(p)$ be the first outage in the schedule originally assigned to plant p.

Let timeline $\tau = \langle o^1(p) \rangle$

Let timeline set $T = T \bigcup$ find timelines(τ).

end for

Return timeline set T.

To better illustrate the SCR schedule generation module, consider a single plant with three scheduled outages. Assume that currently there is only one installed catalyst layer. The following Table 3.3 display the timelines and number of generated schedules.

From the Table 3.3, observe that seven timelines were generated for this particular example that considered the set of all possibilities from three outages. For each

Algorithm 2 Find timelines(τ) algorithm

Let outage $o^n(\tau)$ and plant $p(\tau)$ be the last outage and the assigned plant of timeline τ.

Let timeline set $T(\tau) = \emptyset$ be the set of generated timelines that extend from timeline τ.

if outage set $O^+(o^n(\tau)) = \emptyset$ **then**

Return the set of timelines $\{\tau\}$.

else

for each outage $o^+ \in O^+(o^n(\tau))$ **do**

Let outage $\bar{o}^+$ be a copy of outage o^+ in which the assigned plant of outage $\bar{o}^+$ is plant $p(\tau)$, and the catalyst layer decision is to change or regenerate the oldest catalyst layer at plant $p(\tau)$ based upon timeline τ.

Let timeline $\bar{\tau} = \langle \tau, \bar{o}^+ \rangle$.

Let timeline set $T(\tau) = T(\tau) \bigcup$ find timelines($\bar{\tau}$).

if there is an empty catalyst layer slot at plant $p(\tau)$ based upon the timeline τ **then**

Let outage $\hat{o}^+$ be a copy of outage o^+ in which the assigned plant of outage $\hat{o}^+$ is plant $p(\tau)$ and the catalyst layer decision is to add a new layer to the lowest empty catalyst layer slot at plant $p(\tau)$ based upon the timeline τ.

Let timeline $\hat{\tau} = \langle \tau, \hat{o}^+ \rangle$.

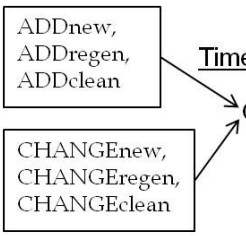
Let timeline set $T(\tau) = T(\tau) \bigcup$ find timelines($\hat{\tau}$).

end if

end for

Return timeline set $T(\tau)$.

end if



	Timeline 1	Timeline 2	Timeline 3	Timeline 4	Timeline 5	Timeline 6	Timeline 7
	O_1	O_2	O_3	O_1	O_1	O_2	O_1
				O_2	O_3	O_3	O_2
							O_3
Number of Schedules	6	6	6	12	12	12	18
Fixed NO_x and fixed NH_3 slip policy	12	12	12	24	24	24	36

Figure 3.3. Example of generate timelines and find timelines algorithms.

outage, potential actions could be either ADDnew, ADDregen, ADDclean, CHANGEnew, CHANGEregen, or CHANGEclean. Therefore the number of schedules for each outage would be six. Then either the policy is the fixed NO_x policy or the fixed NH_3 slip policy, which will also double the number of schedules. Consequently, the total number of schedules generated is 144. Schedules are then removed if they do not meet NO_x reduction or NH_3 slip constraints. The rest are potential candidate schedules for the SCR optimization module.

3.4 SCR Management Policy

After timelines are generated that was described previously in Section 3.3, two schedules are then created per each timeline. The method used to generate the schedules can be either the fixed NH_3 slip policy or the fixed NO_x policy. The decision to choose which policy to employ would depend upon which parameter we would like to control.

3.4.1 Fixed NH_3 Slip Policy

For the fixed NH_3 slip policy, each timeline will generate two schedules based on two pre-specified levels of target NH_3 slip values. The target levels of NH_3 slip would usually be the minimum allowable NH_3 slip and maximum allowable NH_3 slip usually at two parts per million (ppm) and four ppm. The higher the NH_3 slip value, the higher NO_x reduction. However, increasing NH_3 slip will also increase the reagent costs and potentially become be harmful to people. These two levels are then fixed throughout the schedule. At each level, we then calculate the corresponding NO_x reduction, cost, and RP values. For each generated schedule, if the corresponding NO_x reduction is less than a pre-specified value, usually at 75%, the schedule will be removed as a potential candidate for the optimization. Consequently, in terms of the optimization that will be discussed in Section 3.5, if the objective function is to minimize costs, then the schedules with two ppm will most likely be selected first provided that they have met the minimum NO_x reduction requirement due to lower costs. In contrast, if the objective function is to maximize the NO_x reduction, the optimizer will likely select the four ppm schedules first. Consequently, there exists a tradeoff here, that depends upon the policy and requirements of the user.

3.4.2 Fixed NO_x Policy

For the fixed NO_x policy, each timeline will again generate two schedules based on two pre-specified levels of target NO_x reduction percentage. The two levels are usually set at the minimum and maximum NO_x reduction target that we would like to achieve. The two levels are generally at 75% for minimum NO_x reduction and 85% for maximum NO_x reduction. In this case, the higher the NO_x reduction, the higher NH_3 slip value, which will also have implications on cost. At each level, NO_x reduction is fixed at the pre-specified value throughout the schedule while allowing the NH_3

slip to increase to compensate until the NH_3 slip reaches a pre-defined maximum. If the maximum allowable NH_3 slip is reached, the NH_3 slip will be fixed at that value while NO_x reduction will decrease to compensate for this. If the NO_x reduction goes below the pre-specified minimum, then this schedule is eliminated from the potential candidate for the optimizer. The following Figure 3.4, illustrates an example plot of NH_3 slip over time that employed the fixed NO_x policy.

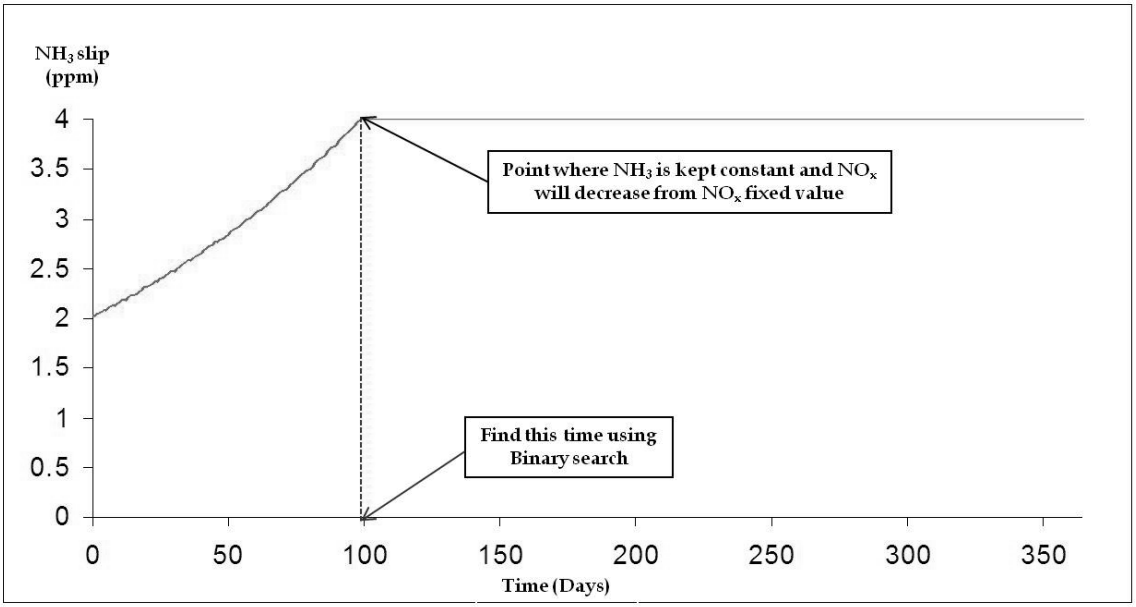


Figure 3.4. Example plot of NH_3 slip over time.

From the Figure 3.4, the NH_3 slip started out at two ppm and over time it increases to compensate for the constant NO_x reduction of 75%. Once the NH_3 slip reaches the maximum allowable value, which is four ppm in this case, the NH_3 slip is kept constant while NO_x reduction will go down to compensate. The time in which the NH_3 slip reaches the maximum value can be easily found by binary search. Regarding the optimization, this would be similar to the fixed NH_3 slip case since

NO_x reduction and NH_3 slip are closely related. The tradeoff and the policy chosen will depend on whether costs or NO_x reduction is more important and whether we would like to closely monitor NH_3 slip or NO_x reduction.

3.5 SCR Optimization Module

The primary purpose of the SCR optimization module is to find an optimal set of schedules based upon the schedules generated from the SCR schedule generation module described in Section 3.3. The mathematical optimization technique applied by the SCR optimization module is integer linear program, while the solver uses Computational Infrastructure for Operations Research branch and cut (COIN-OR CBC [47]) that is described in this section.

Integer linear programming is a linear optimization technique that some or all the variables are restricted to integer values. An integer programming problem in which all variables are required to be integer is called a *pure integer programming problem*. If some variables are restricted to be integer and some are not, then the problem is a *mixed integer programming problem*. Problems that the integer variables are restricted to be zero or one are called pure (mixed) *0-1 programming problems* or pure (mixed) *binary integer programming problems*. In this case, the problem is a pure 0-1 programming or pure binary integer programming problem because all variables are required to be zero or one [48].

The SCR optimization module finds a set of schedules that maximizes NO_x emissions reduction subject to a total operating cost and power generation plan. The SCR optimization module uses an integer linear programming problem with the following:

1. Decision Variables: For each generated schedule, a 0-1 decision variable determines whether the schedule is used in the optimal plan.

2. Objective: The objective function is to maximize the anticipated average daily NO_x reduction over all power plants. (The objective can be seamlessly changed to maximize the anticipated average daily reactor potential. This will increase computationally efficiency of the algorithm, because average daily reactor potential has a closed-form equation, while average daily NO_x reduction usually requires numerical integration. The NO_x reduction equation is a concave function and has no closed-form equation.)
3. Constraints: The constraints of the model ensure that the optimal SCR maintenance plan maintains the following conditions:
    - The total anticipated operating cost of the plan is less than or equal to a predetermined budget.
    - The total anticipated power production is greater than or equal to a predetermined minimum production.
    - Each plant will be assigned to exactly one schedule in the plan.
    - Each outage is included in at most one schedule in the plan.

Let S be the set of all schedules from the SCR schedule generation module, let P be the set of plants, and let O be the set of outages. For each schedule $s \in S$, let $DNOX_s$ be the NO_x reduction of s, let c_s be the operating costs, let g_s be the power generation, and let

$$x_s = \begin{cases} 1 & \text{if schedule } s \text{ is selected for the outage,} \\ 0 & \text{otherwise.} \end{cases} \tag{3.7}$$

For each outage $o \in O$, let $S(o)$ be the schedules that include outage o, and for each plant $p \in P$, let $S(p)$ be the set of schedules that can be assigned to plant p. Let

C be the maximum operating costs of the fleet, and let G be the minimum power generation. The integer linear programming problem is given by the following:

$$\max \sum_{s \in S} DNOX_s x_s \tag{3.8}$$

$$\text{s.t.} \sum_{s \in S(p)} x_s = 1 \qquad \forall p \in P \tag{3.9}$$

$$\sum_{s \in S(o)} x_s \leq 1 \qquad \forall o \in O \tag{3.10}$$

$$\sum_{s \in S} c_s x_s \leq C \tag{3.11}$$

$$\sum_{s \in S} g_s x_s \geq G \tag{3.12}$$

$$x \in \{0,1\}^{|S|} \tag{3.13}$$

The solver within the SCR optimization module uses COIN-OR branch and cut (COIN-OR CBC). Branch and cut first achieved success in solving large instances of the traveling salesman problem [49]. It is the core of the fastest commercial general purpose integer programming packages. It is like branch and bound, except that in addition, the algorithm may generate cutting planes [48]. These cutting planes are constraints that, when added to the problem at a search node, result in a tighter LP polytope (while not cutting off the optimal integer solution) and thus generate a higher lower bound. The higher lower bound in turn can cause earlier termination of the search path, and thus yields smaller search trees. COIN-OR CBC is an open-source mixed integer programming solver also written in C++. The following are some basic parameter settings within CBC that could potentially increase the speed of the solver.

1. Cutoff: cutoff all nodes with objective greater than or less than a specified value.

2. IntegerTolerance: treat variables as integer if close enough.
3. Seconds: treat as maximum nodes after this time.
4. CutDepth: only generate cuts at multiples of this.
5. MaxNodes: stop after this many nodes.
6. PassCuts: number of cut passes at root.
7. StrongBranching: number of candidates for strong branching.

COIN-OR CBC uses strong branching where the algorithm performs a one-step look ahead for each variable that is non-integral in the LP at the node. The one-step look ahead computations solve the LP relaxation for each of the children.

The SCR management framework employs the SCR optimization module multiple times to determine an efficient Pareto optimal frontier. This Pareto optimal model provides a tradeoff between NO_x reduction and operating costs. The model calculates multiple outage plans, or *Pareto points*, that are not dominated by any other outage plan in operating costs and NO_x reduction. The inputs needed are a minimum and a maximum budget interval, and the number of Pareto points to be determined.

3.6 Computational Experiments

In this section, we provide computer experiments on the SGO algorithm described earlier. The computational experiments are divided into two parts, which are the fixed NH_3 slip policy and the fixed NO_x policy. Three different problem instances were applied on these three different computational experiments. The first problem instance was simulated and were applied in Section 3.6.2. The goal of this experiment is to test the effectiveness of the SGO algorithm to find out whether it would work accordingly. After we have made sure that the algorithm works as planned, we tested them with a new problem instance to test the effectiveness of the SCR optimization

module with the Pareto efficient frontiers described in Section 3.6.3. We then obtained a modified version of a real-world problem instance and discovered that the fixed NO_x policy is more applicable, and the objective that users' most strive for is to minimize cost. Therefore, we apply the problem instance to Section 3.6.4. This problem instance will then be used for the rest of the dissertation.

3.6.1 Fixed NH_3 Slip Policy

In this section, we discuss two example problem instances that employed the fixed NH_3 slip policy. The instance in Section 3.6.2 has seven power plants, and the instance in Section 3.6.3 includes six power plants. Both cases use a five-year planning horizon with approximately one layer either being added or changed each year, which is a typical maintenance interval. The values are based upon default settings in CatReact [18]. In these two instances, we have decided to maximize the NO_x reduction. Although from global information in Table 3.1, the fleet-wide constraints include both minimum NO_x reduction and minimum costs. Therefore, the objective function can also be easily changed to cost or used as a second objective.

3.6.2 Optimization Results

Using the SCR management framework, we obtained three additional outage plans with different objectives and constraints as shown in Table 3.3. The original plan is the current plan originally inputted in the outage file. The max DNO_x plan is an outage plan that maximizes NO_x reduction without a constraint on the operating costs. The min cost plan is one that minimizes operating costs while ignoring NO_x reduction. Finally, the optimal plan was found by maximizing NO_x reduction subject to an operating budget no greater than that of the original plan. In all four plans, we maintained the same power generation plan. From Table 3.3, the optimal plan

Table 3.3. NO_x reduction and operating costs of four outage plans

Outage Plan	Plant Number	NO_x Reduction (%)	Operating Costs ($million)
Original Plan	Plant 1	78.06	50.61
	Plant 2	92.53	56.46
	Plant 3	77.79	50.62
	Plant 4	84.50	53.70
	Plant 5	79.29	51.16
	Plant 6	78.40	50.61
	Plant 7	74.29	53.33
	Total	**80.69**	**366.49**
Max DNO_x Plan	Plant 1	96.32	53.79
	Plant 2	97.10	56.46
	Plant 3	96.29	53.81
	Plant 4	96.37	53.81
	Plant 5	98.05	54.32
	Plant 6	96.50	53.80
	Plant 7	90.73	56.50
	Total	**95.91**	**382.49**
Min Cost Plan	Plant 1	77.26	50.58
	Plant 2	81.35	53.37
	Plant 3	75.93	50.58
	Plant 4	74.94	50.58
	Plant 5	73.83	38.26
	Plant 6	72.15	51.15
	Plant 7	67.23	40.48
	Total	**74.69**	**335.01**
Optimal Plan	Plant 1	96.11	50.58
	Plant 2	97.11	56.64
	Plant 3	96.06	53.81
	Plant 4	96.17	53.81
	Plant 5	97.77	54.26
	Plant 6	96.35	53.80
	Plant 7	89.34	43.04
	Total	**95.56**	**365.94**

has a substantially higher NO_x reduction rate and slightly smaller operating costs than the original plan. Observe that the NO_x reduction of the optimal plan is almost

identical to the unconstrained max DNO_x plan. The primary reason for this dramatic improvement in NO_x reduction is likely due to elevated levels of NH_3 slip from two ppm to four ppm at each of the plants.

3.6.3 Pareto Optimal Efficient Frontiers

We obtained a six-plant example that included outages of five years. The low budget used was \$67 million while the high budget was \$117 million. We did a preliminary analysis before hand and found those intervals were appropriate because the lowest possible operating cost is \$67.37 million, which is why the first point does not show any results. Similarly for the high budget, we found that no matter how much the budget is increased, the best solution that is possible to obtain is an outlet NO_x reduction of 2840.28 lbs/hr. The number of Pareto points was set to 20. The results are summarized Table 3.4. A plot of the Pareto optimal efficient frontier from Table 3.4 is given in Figure 3.5. From Figure 3.5, we can observe a large change in NO_x reduction from 2626.52 lbs/hr to 2767.1 lbs/hr with an increase in operating costs of roughly \$4 million. The reason for this significant improvement is due to the fact that an extra SCR catalyst layer was added to the fleet. This graph is very helpful since it would allow users to find the trade off of increasing operating costs to further reduce NO_x.

3.6.4 Fixed NO_x policy

In this section, we provide a problem instance that employed the fixed NO_x policy and changed the objective to minimize cost. There are total of six plants, and the planning horizon is five years. The input is a modified version of a real-world problem instance that we have obtained and will be kept consistent throughout the

Table 3.4. Pareto optimal plans

Pareto Points	Budget ($)	NO_x Reduction (lbs/hr)	Operating Costs ($)
1	67000000		
2	69631600	2594.94	67367800
3	72263200	2607.11	71942200
4	74894700	2618.3	73657600
5	77526300	2618.3	73657600
6	80157900	2626.52	79870800
7	82789500	2626.52	79870800
8	85421100	2767.1	85149000
9	88052600	2767.1	85149000
10	90684200	2779.27	89723400
11	93315800	2790.46	91438800
12	95947400	2790.46	91438800
13	98578900	2798.68	97652000
14	101211000	2798.68	98704200
15	103842000	2808.7	102930000
16	106474000	2808.7	104760000
17	109105000	2820.87	107505000
18	111737000	2832.06	109220000
19	114368000	2832.06	109220000
20	117000000	2840.28	116485000

rest of this dissertation. The following Table 3.5 displays an optimal outage plan for this example.

From Table 3.5, the optimal SCR management schedule were found after 610.67 minutes and the total costs were $137.26 million. As mentioned earlier, the input information is based on a modified version of a real-world problem instance for a six-plant and five-year problem.

3.7 Conclusion

In this chapter, we proposed an SGO algorithm that finds an optimal SCR outage plan. As mentioned earlier in Section 2, there is currently no fleet-wide SCR

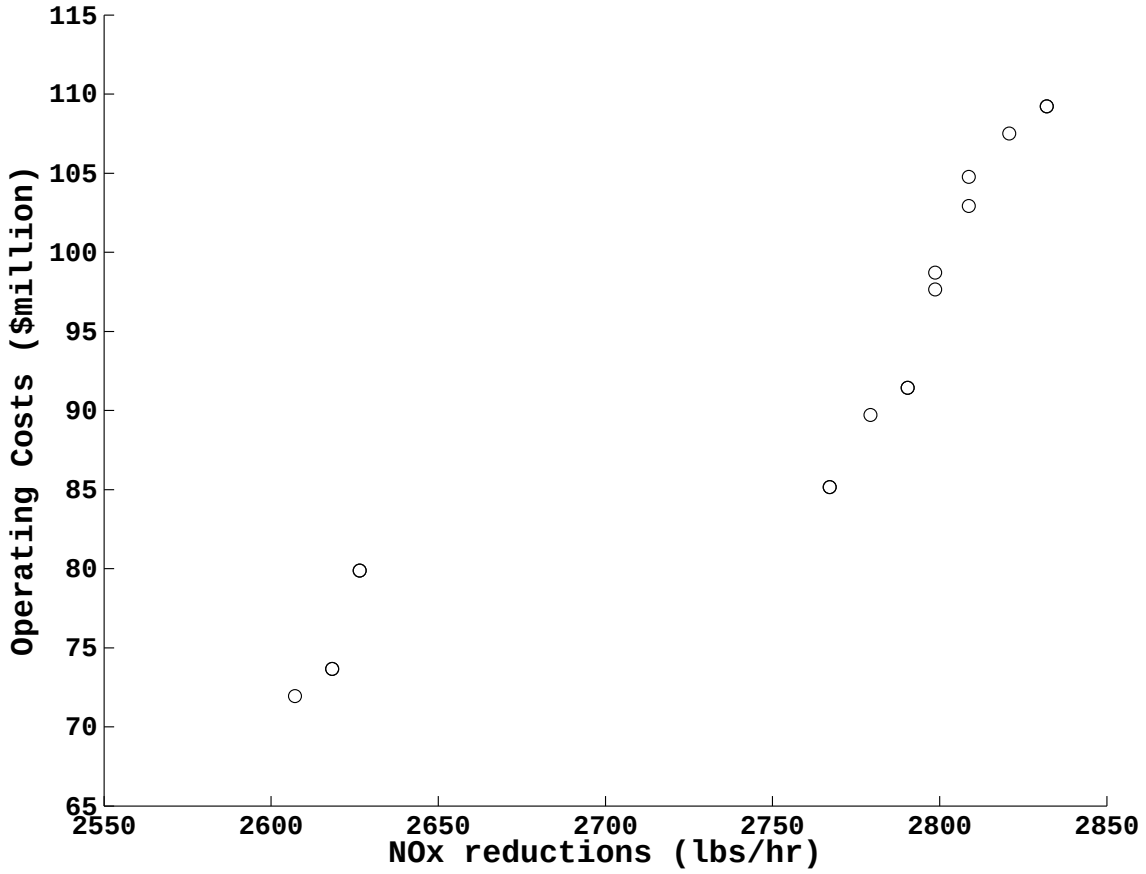


Figure 3.5. Pareto optimal efficient frontiers plot of operating costs vs. NO_x reductions.

management tool that is commercially available. We address this limitation by introducing the ability to solve the fleet SCR management problem. In this chapter, we started out by discussing the input information as well as key equations. We then described the two main modules of the SGO algorithm that are SCR schedule generation and SCR optimization. We then discussed three problem instances to test the effectiveness of the algorithm. From the final computational experiment, we found based on a modified version of a real-world problem instance as well as a more applicable policy that the computational time was 610.67 minutes. While this is a

Table 3.5. Optimal fleet-wide SCR management schedule

Start Date	End Date	Plant	Action	Layer
03/15/2010	03/29/2010	1	ADDregen	2
11/16/2011	11/24/2011	1	ADDregen	1
04/06/2013	05/04/2013	1	CHANGEregen	4
10/30/2010	11/07/2010	2	ADDclean	2
11/08/2011	11/20/2011	2	ADDclean	1
11/02/2012	11/10/2012	2	CHANGEclean	4
10/18/2014	11/15/2014	2	CHANGEclean	3
11/10/2012	11/20/2012	3	ADDclean	2
11/15/2014	11/25/2014	3	ADDclean	1
03/15/2010	03/29/2010	4	ADDregen	2
11/16/2011	11/24/2011	4	ADDregen	1
04/06/2013	05/04/2013	4	CHANGEregen	4
10/30/2010	11/07/2010	5	ADDclean	2
11/08/2011	11/20/2011	5	ADDclean	1
11/02/2012	11/10/2012	5	CHANGEclean	4
10/18/2014	11/15/2014	5	CHANGEclean	3
11/10/2012	11/20/2012	6	ADDclean	2
11/15/2014	11/25/2014	6	ADDclean	1

reasonable computational time, we have tested many problem instances where the computational time reaches several weeks. Therefore, Chapter 4 will discuss ways to improve the computational efficiency.

CHAPTER 4

MULTI-COMMODITY NETWORK FLOW MODEL

4.1 Introduction

From the proposed model in Chapter 3, we have found that although the enumeration of all possible outage schedules would yield a good set of schedules within a reasonable computational time, a problem arises when a combination of the number of outages, the time horizon, and the number of plants increases. In such a case, the computational time also increases exponentially. There are instances that we have faced that would require running the SGO algorithm for up to several weeks. In this chapter, we propose a multi-commodity flow model (MCFP) to replace the SGO algorithm described in Chapter 3. Instead of generating schedules, we generate edges that represent all SCR catalyst layers flowing from the start of the time horizon and through outages until the end of the time horizon. Given a set of scheduled outage plans with the same input information as in Chapter 3.6.4, these edges are then optimized to find a set of edges with the lowest operating cost while limiting the NO_x reduction to a pre-specified amount.

In this chapter, we begin with the problem formulation where variable creation and the optimization model will be discussed. Then, we formulate a relaxed MCFP while ignoring the average daily NO_x constraint that was incorporated in Chapter 3. We provide a computational experiment with the results and computational time comparison with the SGO algorithm that was described in Chapter 3 for a single plant. We expand the problem further by applying it to an example with six plants and a five-year planning horizon. Since input information is the same as in Section 3.6.4,

we make the comparison between the SGO algorithm and the MCFP model. From costs and schedule comparison, we observe that the average daily NO_x constraint is extremely crucial to the model. Therefore, we would like to incorporate the average daily NO_x constraint in the MCFP. However, there is no closed-form equation to calculate average daily NO_x reduction. The only way the average daily NO_x constraint can be obtained is through the schedule that can only be found after we obtain the solution. Therefore, in order to incorporate the average daily NO_x constraint to the model, we introduce an MCFP with schedule elimination constraints (MCFPwSEC). The MCFPwSEC algorithm uses a single cut per iteration to remove infeasible solutions that do not meet the minimum average daily NO_x reduction requirement. We then compare the computational efficiency of MCFPwSEC with the SGO algorithm that employed the fixed NO_x policy discussed in Chapter 3.6.4. We further reduce the computational time by introducing a multi-cut MCFPwSEC. Instead of cutting the solution one by one, we apply a heuristic method that when the solution is found, we find the NO_x reduction value on the schedule. We compare the solution found with the minimum NO_x reduction requirement. If the minimum NO_x reduction requirement is satisfied, then the solution found is an optimal solution. However, if it does not, we investigate the layer actions. As mentioned earlier in Section 3.1, the set of layer actions consists of ADDnew, ADDregen, ADDclean, CHANGEnew, CHANGEregen, and CHANGEclean. For instance, consider the following solution sequence, CHANGEclean-CHANGEclean-CHANGEclean. First, we determine whether the minimum NO_x reduction requirement is satisfied. Then, if it is unsatisfied, we replace the solution with the sequence CHANGEnew-CHANGEnew-CHANGEnew. The new solution sequence just replaces the layer actions while keeping the outage dates the same. We calculate the average daily NO_x reduction for the new sequence. Recall from Section 3.1, new layers are the most effective method for NO_x reduction.

Therefore, if this new sequence still does not meet the minimum NO_x reduction requirement, we can remove a total of 27 solution sequence combinations that consist of CHANGEnew, CHANGEregen, and CHANGEclean layer actions. The heuristic method has provided positive results and further reduces computational time. Finally, from observation of the algorithm, we observe that the average RP associated with the average daily NO_x reduction of the solution is an upper bound to the RP value associated with the average daily NO_x reduction of the schedule. Consequently, in order to meet the requirement, these values must be higher than the RP value associated with the minimum NO_x reduction requirement. From this relationship, we can incorporate an additional RP constraint to the MCFP that requires the average RP of the solution to be greater than the RP value associated with the minimum NO_x reduction requirement. Therefore, with an added RP constraint to the MCFP, we could potentially reduce the number of schedule elimination constraints generated early in the algorithm execution.

4.2 Problem Formulation

In this section, we describe the formulation of the relaxed MCFP and the optimization model to solve the fleet SCR management problem. The relaxed MCFP problem ignores the average daily NO_x reduction requirement that was incorporated in Chapter 3. Consequently, for the relaxed MCFP, in order to control the NO_x reduction, we use constraints to limit instantaneous NO_x. We formulate edges to represent the flow of SCR catalyst layers that can be up to four layers per plant. Essentially we would like to find a path from the start of time horizon (source) to the end of time horizon (sink) for each layer and each plant throughout the time horizon. Like time, the flow of the edges can only be forward. A path represents a sequence of outages used for that particular layer. The edges in the path also determine the actions

used in the outages. Layer actions consist of adding of a new layer (ADDnew), a regenerated layer (ADDregen), or a cleaned layer (ADDclean) and replacing the layer with a new layer (CHANGEnew), a regenerated layer (CHANGEregen), or a cleaned layer (CHANGEclean). As mentioned earlier, only one action can be taken during each outage due to time and cost restrictions. First, we define the variable vector x that represents edges that flow from one outage to the next as well as from the start (source node) of the time horizon and to the end (sink node) of the time horizon. The edge information also includes which layer it represents and the layer action that was taken at the previous outage. After each edge is generated, we calculate the corresponding RP and cost associated with that particular edge. Consequently, for each edge, a 0-1 decision variable determines whether the edge is used in the solution plan. The following formulation describes the MCFP variables.

Let

$$x_{ij}^{la} = \begin{cases} 1 & \text{if two consecutive outages } i \text{ and } j \\ & \text{are used for layer } l \text{ and action } a \text{ is taken in outage } i, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

Given a set of scheduled outages, consider a set of SCR catalyst layers, where up to four layers can be filled at the start of the time horizon. Edges are generated based upon any pair of consecutive outages i and j and include the start of the time horizon (source node) and the end of time horizon (sink node). Edges are also generated for the slots that are empty at the start of the time horizon for potential additions. If a layer at a given slot is already filled prior to the start of the time horizon, all subsequent outages for that particular slot can only consist of changes that can be either a new layer, a regenerated layer, or a cleaned layer. Conversely, if a given slot is unfilled prior to the start of time horizon, subsequent layer actions can only be additions that can also be either a new layer, a regenerated layer, or a

cleaned layer. Furthermore, after an addition for a particular layer has been made, the following actions can only consist of changes. Similarly, at each outage (node), we determine whether that particular slot has been filled or not, which determines what set of actions that can be applied at the particular outage. Consequently, after each edge is generated, we calculate its corresponding RP and cost for that particular edge, where RP_{ij}^{la} is the reactor potential between two consecutive outages i and j for layer l where action a is taken in outage i, and C_{ij}^{la} is the total cost incurred between the two consecutive outages i and j for layer l where action a is taken in outage i. To illustrate the algorithm, consider a simple example of a layer from a single plant where there are two outages, o_1 and o_2. The following Figure 4.1 demonstrates the flow of a catalyst layer from the start of the time horizon (S) to the end of time horizon (T).

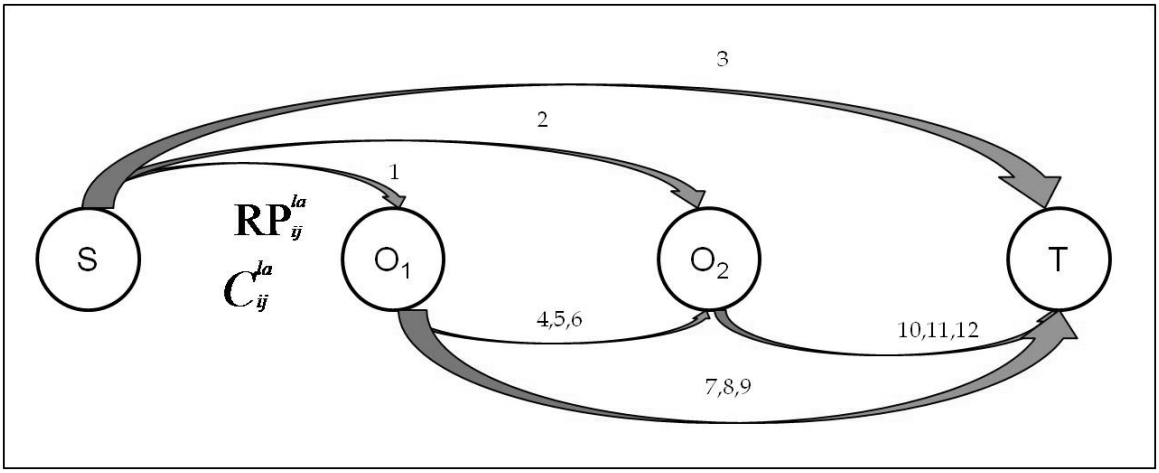


Figure 4.1. Example of the relaxed MCFP variable creation.

From Figure 4.1, we can observe that there were 12 edges created that are denoted by their indices with their corresponding x_{ij}^{la}, RP_{ij}^{la}, and C_{ij}^{la}. Note that

edges must originate from the source to a future outage. Therefore, edge 1 goes from source to o_1, edge 2 goes from source to o_2, and edge 3 goes from source to sink, which means that no action was taken for this particular edge. We refer these types of edges as *from source* and *source to sink* edges. Next, edges must end in the sink node (T) so there are edges 3, 7, 8, 9, 10, 11, 12 that go to the sink, and we refer to these types of edges as *source to sink* and *to sink* edges. Observe that from o_1 to T there were three edges to sink. These three edges represent changing either a new layer, a regenerated layer, or a cleaned layer in outage o_1. Whether it was an addition or a change depends upon whether that particular layer slot was filled or not at the end of o_1. Figure 4.2 summarizes an example of having a filled layer at the source node S. Edges 4, 5, 6 are referred to as *intermediate* edges. These are edges that flow between two consecutive outages that do not include the source node and or the sink node as an endpoint. Similarly, there are three possible actions that can be done during previous outage.

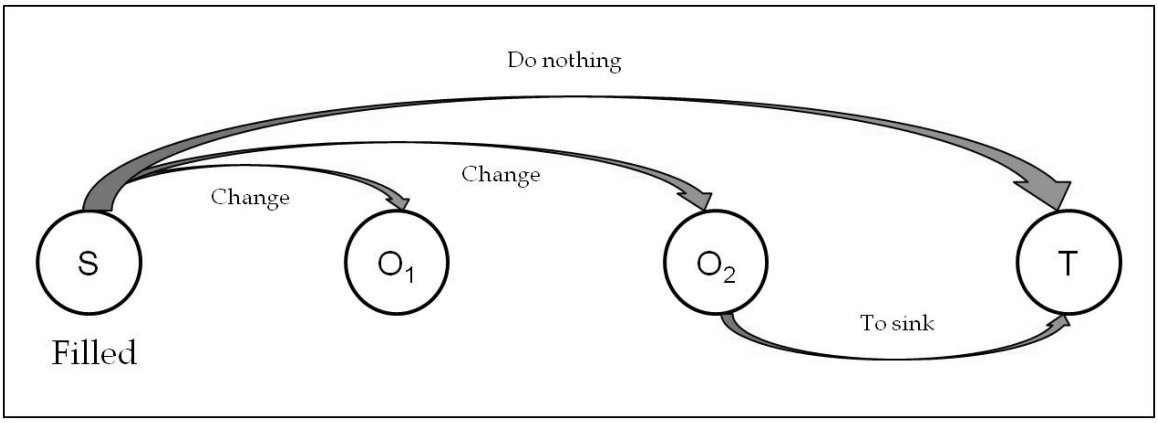


Figure 4.2. Example with filled layer at source.

In Figure 4.2, we assumed that the layer was originally filled at the start of the time horizon. Observe that not all edges are drawn in Figure 4.2. For a complete list of edges, refer to Figure 4.1. Since the layer is already filled in the particular slot, it cannot be added, so the only option is to change layers. Consequently, the decision is to either do a change in o_1, a change in o_2, or do nothing. Changes in o_1 and o_2 can be either with a new layer, a regenerated layer, or a cleaned layer. Suppose the edge that flows from S to o_2 was chosen. There are three edges actions from o_2 to T representing the three possible change actions in outage o_2. Therefore in this example, the layer was filled at the start of time horizon, then a change was made at o_2.

Figure 4.3 illustrates the case where the layer slot was empty at the start of time horizon.

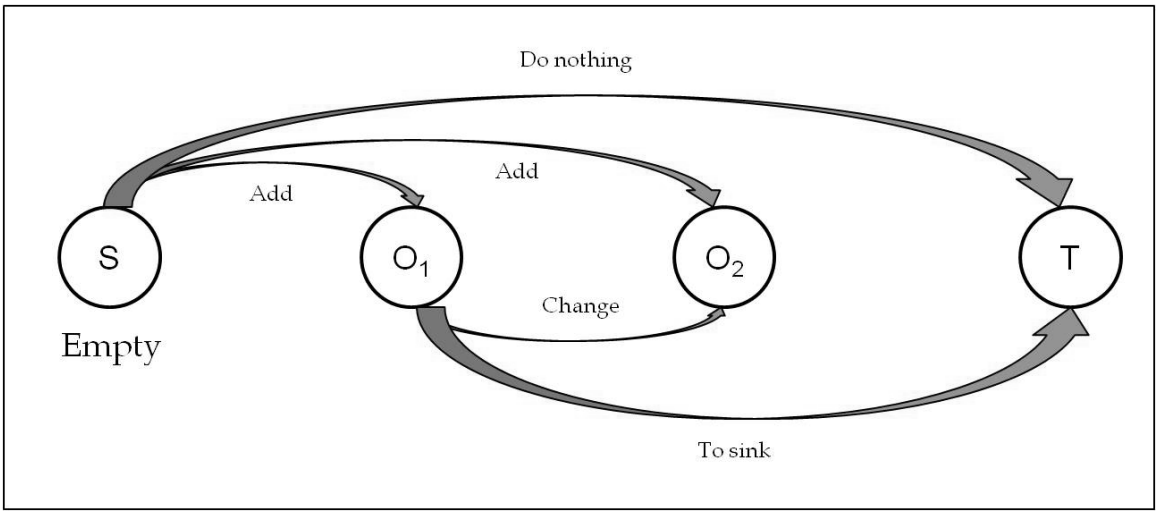


Figure 4.3. Example with empty layer at source.

In Figure 4.3, we assume that the layer slot was not filled at the start of time horizon and not all edges are drawn in the figure. Since the layer slot is empty, the

options we have are to add in o_1, add in o_2, or do nothing. Suppose that the edge that flows from S to o_1 is chosen and the action in o_1 is to add a cleaned layer. Then, since the layer has been added already, meaning the slot is no longer empty, the remaining options are to either do a change in o_2 or go to sink (T), and in this case we choose the edge that goes from o_1 to T. In summary, this particular layer slot started out as empty, and then we added a cleaned layer to that slot in o_1.

In the formulation, recall that there is a corresponding reactor potential RP_{ij}^{la} and a cost C_{ij}^{la} for each edge. Since the objective is to minimize total cost, and we currently have no restriction on the NO_x reduction, the optimal solution is to do nothing across all edges and plants, which is certainly infeasible in reality since the minimum NO_x reduction should still be met. As mentioned earlier, since RP is directly proportional to NO_x, we would like to ensure that before each outage o_i across all layers for each plant, a certain RP value is met to ensure that necessary outages are not skipped. Consequently, at the start of each outage, we would like to add a constraint where the instantaneous RP value meets at least a pre-specified minimum value. Furthermore, at the end of the time horizon, power plants are not shut down. They still must run for a certain period until the next outage in the next time horizon. Therefore, we would like to impose a constraint that specifies how many months we would like the plant to run without an outage before the next outage in the next time horizon. The number of months is then converted to a minimum RP value needed. Figure 4.4 illustrates an example.

From Figure 4.4, we expanded the example by adding an additional layer slot to illustrate the RP constraints. Notice that the figure still represents a single plant with two outages. As mentioned earlier, to ensure that no necessary outages are skipped by the optimization, we apply minimum RP constraints at the start of each outage, o_1 and o_2. These minimum RP constraints are derived from the pre-specified minimum

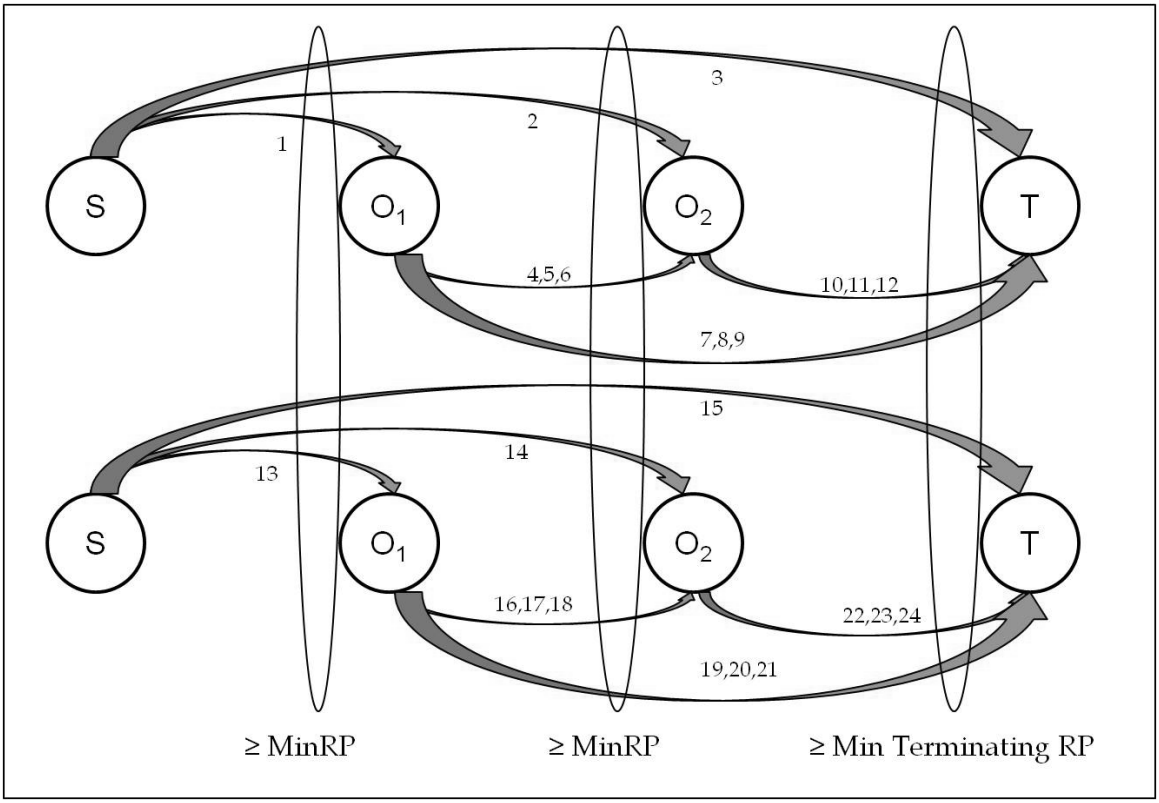


Figure 4.4. Example of the relaxed MCFP with instantaneous RP constraints.

NO_x requirements for the plant. This minimum RP value implies that at the start of each outage, the minimum NO_x requirement must be met. Furthermore, observe that at the sink node, which is at the end of time horizon, a minimum terminating RP constraint is added. This constraint is added to ensure that at the end of time horizon, the NO_x reduction for each plant can still be maintained until the next outage after the time horizon.

For each outage $i \in O$ and each layer $l \in L$, we let $E_l(i)$ be the set of edges from the node representing outage i in the sub-network representing layer l. The relaxed 0-1 integer program to solve the fleet SCR management problem is given by the following.

$$\min \sum_{l\in L}\sum_{a\in A}\sum_{(i,j)\in E} C_{ij}^{la}x_{ij}^{la} \tag{4.2}$$

$$\text{s.t.} \sum_{l\in L}\sum_{a\in A}\sum_{(j,k)\in E_l(i)} RP_{jk}^{la}x_{jk}^{la} \geq \min \text{RP} \qquad \forall i \in O \tag{4.3}$$

$$\sum_{l\in L}\sum_{a\in A}\sum_{j|\exists(i,j)\in E} x_{ij}^{la} \leq 1 \qquad \forall i \in O \tag{4.4}$$

$$\sum_{a\in A}\sum_{j|\exists(i,j)\in E} x_{ij}^{la} = \sum_{a\in A}\sum_{j|\exists(j,i)\in E} x_{ij}^{la} \qquad \forall i \in O, l \in L \tag{4.5}$$

$$\sum_{a\in A}\sum_{j|\exists(s,j)\in E} x_{sj}^{la} = 1 \qquad \forall l \in L \tag{4.6}$$

$$\sum_{a\in A}\sum_{j|\exists(j,t)\in E} x_{jt}^{la} = 1 \qquad \forall l \in L \tag{4.7}$$

$$x_{ij}^{la} \in \{0,1\}^{|E|} \tag{4.8}$$

The problem is to minimize the total costs across all edges in the fleet subject to flow constraints. Essentially the problem is to find paths where edges flow from sources to sinks in the layer sub-networks that incur the least total costs while satisfying the constraints. The optimal solution is an SCR management schedule for all plants in the fleet that is comparable to Chapter 3. After the MCFP algorithm has been formulated, the next section, we discuss computational experiments.

4.3 Computational Experiments

In this section, we present the computational experiments of the relaxed MCFP. First, we start with a single plant example. Then we expand the problem further by introducing a six-plant example with the same input information as Section 3.6.4. The solver used was CPLEX version 12.1 callable library [50].

4.3.1 Single Plant Computational Experiments

In this section, consider a single plant with five scheduled outages during a five-year planning horizon. From the available four catalyst layer slots, two layers were installed prior to the start of the horizon while two slots were empty. Layers are indexed 1, 2, 3, and 4 where layer 4 is the one closest to the inlet. The pre-specified conditions are that NO_x reduction of at least 75% must be met before each outage and that at the end of time horizon, the plant must go without an outage for 8 months. The summary of an input scheduled outage is shown in Table 4.1, and the optimal solution obtained is shown in Figure 4.5.

Table 4.1. Example of a scheduled outages plan

	Start Date	End Date
o_1	03/15/2010	03/29/2010
o_2	11/16/2011	11/24/2011
o_3	10/27/2012	11/04/2012
o_4	04/06/2013	05/04/2013
o_5	10/24/2014	11/01/2014

Figure 4.5 illustrates the optimal SCR management schedule for the plant. To interpret the figure, layer four needs to be changed with a cleaned layer during o_4, layer three requires a change with a cleaned layer during o_1, layer two needs to be added with a cleaned layer during o_2, and the fourth layer slot is kept emptied for the time horizon. The maintenance plan has incurred the total costs of \$13.20 million. Finally, we compare the computational time with the schedule generation algorithm that is described in Table 4.2.

From the Table 4.2, the results have shown a significant reduction in the computational time from the SGO algorithm for a single plant example. While this may

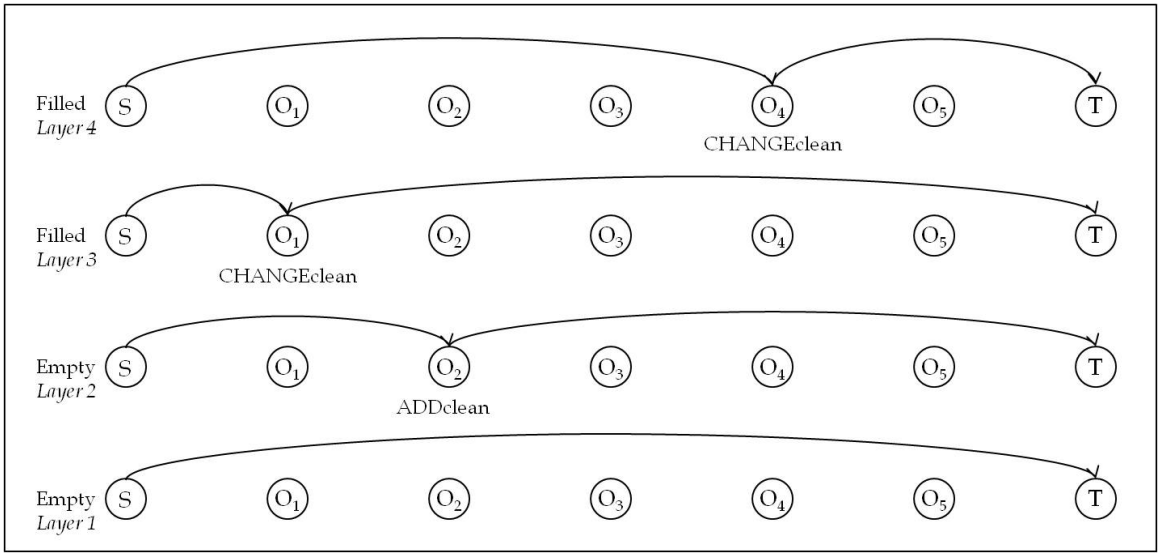


Figure 4.5. Solution of the relaxed MCFP for a single plant.

Table 4.2. Single plant computational time comparison

	SGO		Multi-Commodity Flow Problem
	Fixed NH_3 Slip Policy	Fixed NO_x Policy	(Relaxed MCFP)
CPU Time	5 minutes	22 minutes	<1 second

not seem such a big difference for a single plant, as the problem size gets larger, the difference will also increase exponentially as with the computational time.

4.3.2 Fleet-Wide Computational Experiments

From a single plant example in Section 4.3.1, we expanded the problem further to solve a fleet SCR management problem. As described earlier in Section 3.6.4, we applied a six-plant example with five years planning horizon. The input information were exactly the same as Section 3.6.4 for a consistent computational time and solu-

tion comparison. Table 4.3 displays the optimal SCR management schedule for the relaxed MCFP.

Table 4.3. The relaxed MCFP fleet-wide solution

Start Date	End Date	Plant	Action	Layer
03/15/2010	03/29/2010	1	CHANGEclean	3
11/16/2011	11/24/2011	1	ADDclean	2
04/06/2013	05/04/2013	1	CHANGEclean	4
10/30/2010	11/07/2010	2	CHANGEclean	4
11/08/2011	11/20/2011	2	ADDclean	2
12/02/2013	12/10/2013	2	CHANGEclean	3
11/10/2012	11/20/2012	3	ADDclean	2
03/15/2010	03/29/2010	4	CHANGEclean	3
11/16/2011	11/24/2011	4	ADDclean	2
04/06/2013	05/04/2013	4	CHANGEclean	4
10/30/2010	11/07/2010	5	CHANGEclean	4
11/08/2011	11/20/2011	5	ADDclean	2
12/02/2013	12/10/2013	5	CHANGEclean	3
11/10/2012	11/20/2012	6	ADDclean	2

From Table 4.3, we found that the total cost for the optimal SCR management schedule for the relaxed MCFP during the five-year planning horizon was \$66.73 million. Next, we compare the computational time with the SGO algorithm. Table 4.4 shows the comparison.

Table 4.4. Fleet-wide computational time comparison

	SGO (Fixed NO_x Policy)	Relaxed MCFP
CPU Time	610.67 minutes	2 seconds
Total Costs	\$137.26 million	\$66.73 million

From Table 4.4, we can observe that there was a significant reduction in computational time from 610.67 minutes down to two seconds. However, the total costs were different for the relaxed MCFP as compared to the SGO fixed NO_x policy. As mentioned earlier, the optimal solutions from Table 4.3 were obtained from the exact same input information as the solution shown in Table 3.5. However, for the relaxed MCFP, we ignored the average daily NO_x constraints while the SGO based upon the fixed NO_x policy did not. Recall from Section 4.2, when we defined and formulated the algorithm, we had two types of RP constraints that imply NO_x reduction. There were constraints on the RP value before each outage that must be more than an RP value based upon the minimum NO_x reduction. Furthermore, there were constraints at the end of time horizon for RP that must be greater than an RP value based upon NO_x reduction level that must be maintained over minimum NO_x value until the next outage in the next time horizon. These constraints are based upon instantaneous RP value at a certain point in time. Even though we can obtain the RP value of each edge and then sum them up to get the total RP for the schedule, all of the information here cannot directly derive the average daily NO_x reduction. As described earlier in Chapter 3, for the average daily NO_x reduction, the minimum NO_x reduction required that the average daily NO_x reduction must be satisfied. If we refer to Equations 3.3 and 3.4, NO_x reduction is a function of RP and NH_3 slip, and there is no closed-form function to obtain the average daily NO_x reduction. The only way to obtain the average daily NO_x reduction is with the entire SCR management schedule. An example of a schedule is shown in Table 4.3. In the relaxed MCFP model, edges are generated one by one, therefore we do not have any prior knowledge about the schedule. The only way we can obtain the schedule from this model is after a solution is obtained. To address this limitation of the relaxed MCFP, we introduce

the schedule elimination constraints, where we developed an algorithm to incorporate the average daily NO_x constraint into MCFP.

4.4 Schedule Elimination Constraints

In this section, we introduce MCFP with schedule elimination constraints (MCF-PwSEC). As mentioned earlier, for minimum NO_x constraints, it is the average daily minimum constraint over the time horizon. For the MCFP model, the only way to obtain average daily NO_x reduction is via the schedule after optimization has been done and a solution is obtained. Therefore, after a solution is obtain and the schedule is found, we determine whether or not it violates the average daily minimum NO_x constraint. If it does not, then it is an optimal solution. If it does, then we make that solution infeasible and then re-optimize the problem.

Consider the following set elimination constraints. Let F be the set of all feasible flows in MCFP. Let S_p be the set of all feasible schedules for plant $p \in P$. Let $S_p^c = F \setminus S_p$ be the set of flows in MCFP that are infeasible schedules. For example, schedule $s \in S_p^c$ may violate average daily NO_x constraint. Then, the set of schedule elimination constraints is given by (4.9).

$$\sum_{e \in s} x_e \leq |s| - 1, \qquad \forall s \in S_p^c. \tag{4.9}$$

Figure 4.6 and Algorithm 3 summarizes the MCFPwSEC algorithm.

From Algorithm 3, S_p^c can be potentially huge. In addition, generating all S_p^c at the beginning is probably equivalent to SGO schedule generation. Therefore, we can generate S_p^c dynamically through MCFPwSEC.

From Figure 4.6, we can observe that a cut is added after a schedule is found that violates the minimum average daily NO_x constraint. Cuts are added one by one until the optimal solution is found where the minimum average daily NO_x constraint

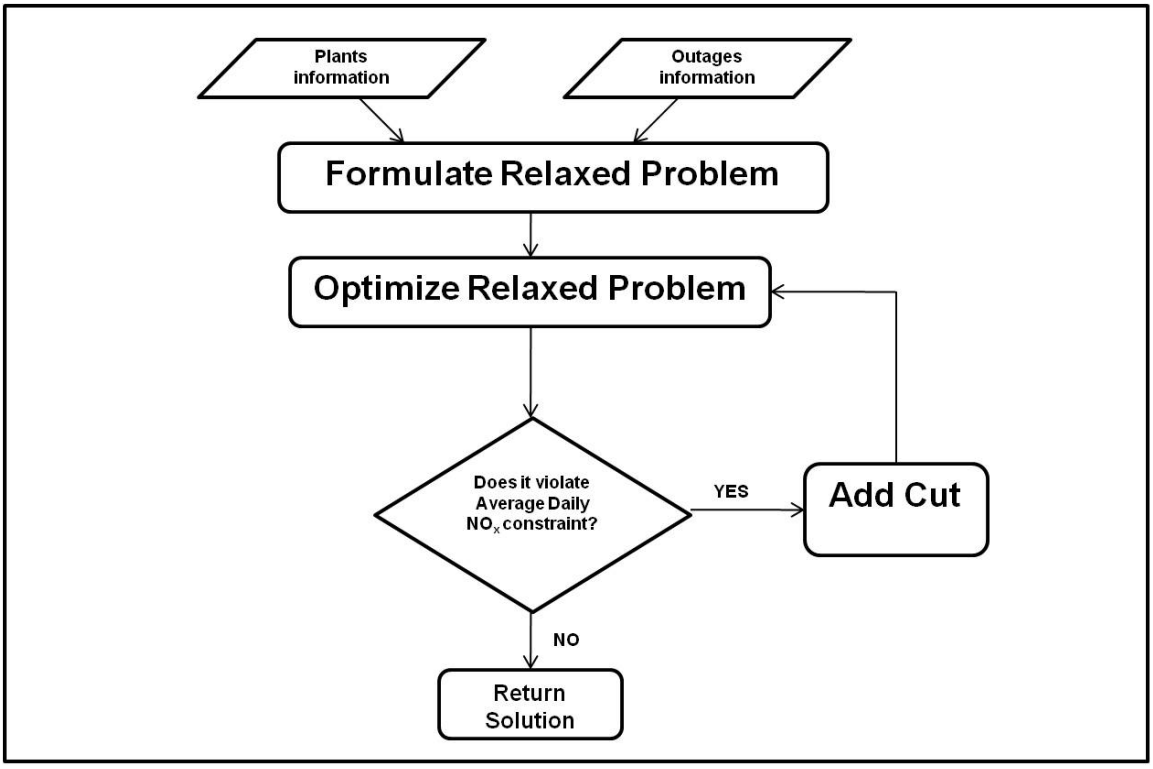


Figure 4.6. Overview of the MCFPwSEC algorithm.

Algorithm 3 Summary of the MCFPwSEC algorithm

Let $\bar{S}_p^c \in S_p^c$

Relaxed problem step: Solve MCFP with schedule elimination constraints from $\bar{S}_p^c$ to obtain s^*.

Feasibility check: If $s^* \in S_p$, then return s^*.

Cut generation step: Set $\bar{S}_p^c \leftarrow \bar{S}_p^c \bigcup \{s^*\}$, and go to Relaxed problem step.

is met. To demonstrate the cut generation, consider a single plant example with the same input information as Section 4.3.1 with the minimum average daily NO_x reduction of 75%. Recall that in Section 4.3.1, a solution is found as shown in Figure 4.5. From the schedule in figure 4.5, we found that the solution consists of $x_{202} = x_{158} = x_{153} = x_{152} = x_{101} = x_{50} = x_0 = 1$. From the solution, we found that the

average daily NO_x reduction of the schedule violates the minimum average daily NO_x constraint, therefore we need to make this solution infeasible. For this example, the first cut would be $x_{202} + x_{158} + x_{153} + x_{152} + x_{101} + x_{50} + x_0 \leq 6$. After 875 cuts, we found the optimal solution that satisfies the minimum average daily NO_x reduction, which is shown in Table 4.5.

Table 4.5. MCFPwSEC optimal solution for a single plant

Start Date	End Date	Plant	Action	Layer
03/15/2010	03/29/2010	1	ADDregen	2
11/16/2011	11/24/2011	1	ADDregen	1
04/06/2013	05/04/2013	1	CHANGEregen	4

Then, we applied the algorithm to the six-plant problem described in Section 4.3.2. We found the optimal schedule after 5250 solution cuts, and it is the exact same solution obtained in Section 3.5. The optimal schedule is displayed in Table 4.6.

Finally we compare the computational time of SGO with the fixed NO_x policy with MCFPwSEC in Table 4.7.

From Table 4.7, we can observe that the optimal schedule is found after 57.11 minutes for the six-plant example. While the computational time is still significantly faster than the SGO algorithm for MCFPwSEC model, we would like to explore ways to further reduce the computational time. By observation of the cuts, we found that a single cut MCFPwSEC is not very efficient in eliminating infeasible solutions. Recall that there are 9 SCR layer actions that are ADDnew, ADDregen, ADDclean, CHANGEnew, CHANGEregen, and CHANGEclean. If we refer to the initial optimal solution for a single plant example from Figure 4.5, the sequence of actions were

Table 4.6. Fleet-wide optimal schedule from MCFPwSEC

Start Date	End Date	Plant	Action	Layer
03/15/2010	03/29/2010	1	ADDregen	2
11/16/2011	11/24/2011	1	ADDregen	1
04/06/2013	05/04/2013	1	CHANGEregen	4
10/30/2010	11/07/2010	2	ADDclean	2
11/08/2011	11/20/2011	2	ADDclean	1
11/02/2012	11/10/2012	2	CHANGEclean	4
10/18/2014	11/15/2014	2	CHANGEclean	3
11/10/2012	11/20/2012	3	ADDclean	2
11/15/2014	11/25/2014	3	ADDclean	1
03/15/2010	03/29/2010	4	ADDregen	2
11/16/2011	11/24/2011	4	ADDregen	1
04/06/2013	05/04/2013	4	CHANGEregen	4
10/30/2010	11/07/2010	5	ADDclean	2
11/08/2011	11/20/2011	5	ADDclean	1
11/02/2012	11/10/2012	5	CHANGEclean	4
10/18/2014	11/15/2014	5	CHANGEclean	3
11/10/2012	11/20/2012	6	ADDclean	2
11/15/2014	11/25/2014	6	ADDclean	1

Table 4.7. Fleet with MCFPwSEC comparison

	SGO Fixed NO_x Policy	MCFPwSEC
CPU Time	610.67 minutes	57.11 minutes

CHANGEclean, ADDclean, and CHANGEclean. As mentioned earlier, cleaning of a catalyst is the least effective in terms of NO_x reduction, where regeneration is the next best alternative, and a new layer is obviously the best option. Since the objective is to minimize cost, the first solution we obtain is usually involves cleaning due to the fact that it is also the least expensive option. Therefore, after we cut of this solution, for the next iteration, the solution found was the sequence CHANGEclean, ADDclean, CHANGEregen. We can observe that each cut only marginally improves

the solution and adding one cut at a time and re-optimizing is not effective in terms of the computational efficiency. To potentially improve the computational efficiency, we introduce heuristic generation of schedule elimination constraints that will be described in Section 4.5.

4.5 Heuristic Generation of Schedule Elimination Constraints

In this section, we introduce additional elimination constraints that can be generated based upon heuristics. Instead of adding a single cut for the infeasible solution one at a time, we introduce multiple cuts to speed up the algorithm performance (multi-cut MCFPwSEC). As we examine the cuts generated in Section 4.4, we found that the cut in each iteration only marginally improves the solution. As mentioned earlier, cleaning a catalyst is least effective at reducing NO_x but is also the least expensive method. Consequently, the optimization would choose this before regeneration and including a new layer. From this observation, we find that after the initial solution is found, we can try the same sequence of outages but replace them with a new layer. For example, if the sequence of outages were CHANGEclean, ADDclean, and CHANGEclean, then we know that, from Section 4.4, after we add the cut, the next solution will be CHANGEclean, ADDclean, and CHANGEregen. In order to eliminate multiple cuts in each iteration, we can try the sequence CHANGEnew, ADDnew, and CHANGEnew, which replaces the layers with new ones while keeping the outage dates the same. If this new sequence of actions still does not satisfy the minimum average daily NO_x constraint, we can eliminate 27 schedules immediately, which would reduce the number of optimization iterations that could potentially reduce the computational time. The following Figure 4.7 describes the algorithm.

Consider the optimal solution obtained for the single plant example shown in Figure 4.5. Tables 4.8 and 4.9 illustrate an example of the algorithm.

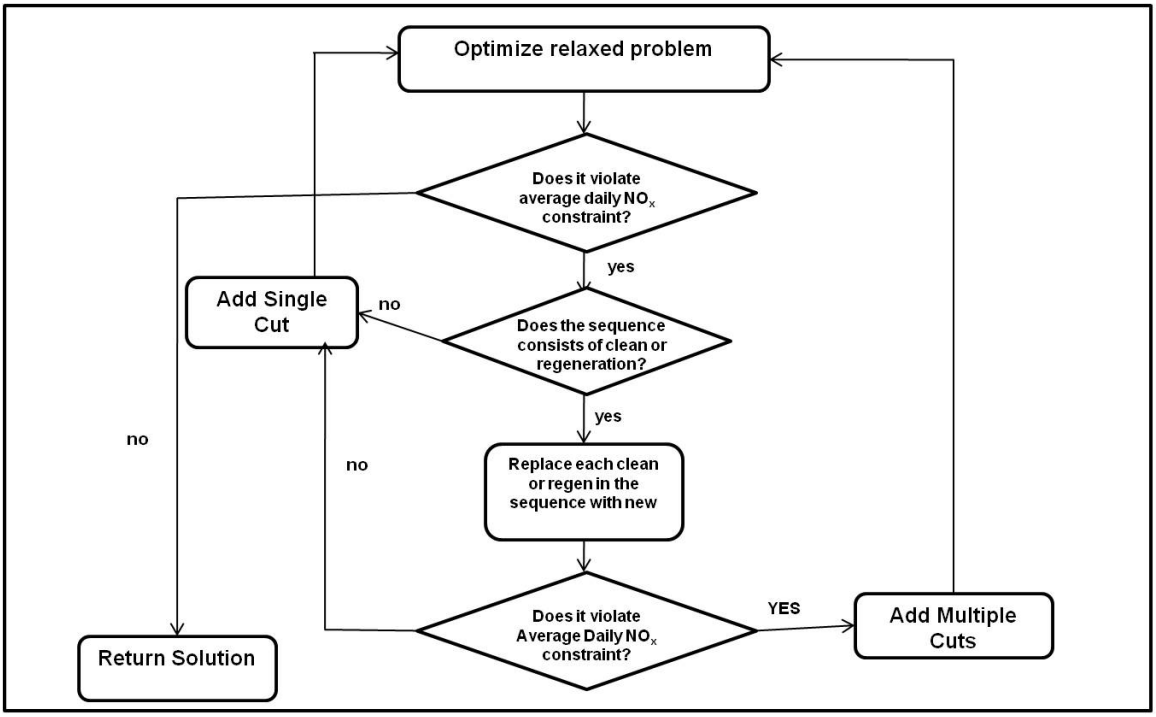


Figure 4.7. Overview of the multi-cut MCFPwSEC algorithm.

Table 4.8. Example of the multi-cut MCFPwSEC algorithm

1	Consider a solution sequence: CHANGEclean, ADDclean, CHANGEclean.
2	Does it violate average daily NO_x constraint? If yes, go to 3. If no, go to 8.
3	Calculate average daily NOx reduction for sequence: CHANGEnew, ADDnew, CHANGEnew.
4	Does it violate average daily NOx constraint? If yes, go to 6. If no, go to 5.
5	Add the schedule elimination constraint for the sequence: CHANGEclean, ADDclean, CHANGEclean, and go to 7.
6	Add heuristic generation of schedule elimination constraint and go to 7.
7	Optimize and go to 1.
8	Return s^*

Table 4.9. Example of step 6 from the multi-cut MCFPwSEC algorithm in Table 4.8

Step 6 from Table 4.8	CHANGEnew, ADDnew, CHANGEnew CHANGEnew, ADDnew, CHANGEregen CHANGEnew, ADDnew, CHANGEclean CHANGEnew, ADDregen, CHANGEnew CHANGEnew, ADDregen, CHANGEregen CHANGEnew, ADDregen, CHANGEclean CHANGEnew, ADDclean, CHANGEnew CHANGEnew, ADDclean, CHANGEregen CHANGEnew, ADDclean, CHANGEclean CHANGEregen, ADDnew, CHANGEnew CHANGEregen, ADDnew, CHANGEregen CHANGEregen, ADDnew, CHANGEclean CHANGEregen, ADDregen, CHANGEnew CHANGEregen, ADDregen, CHANGEregen CHANGEregen, ADDregen, CHANGEclean CHANGEregen, ADDclean, CHANGEnew CHANGEregen, ADDclean, CHANGEregen CHANGEregen, ADDclean, CHANGEclean CHANGEclean, ADDnew, CHANGEnew CHANGEclean, ADDnew, CHANGEregen CHANGEclean, ADDnew, CHANGEclean CHANGEclean, ADDregen, CHANGEnew CHANGEclean, ADDregen, CHANGEregen CHANGEclean, ADDregen, CHANGEclean CHANGEclean, ADDclean, CHANGEnew CHANGEclean, ADDclean, CHANGEregen CHANGEclean, ADDclean, CHANGEclean

From Figure 4.5 and Tables 4.8 and 4.9, we observe that multiple cuts are added per iteration instead of just a single cut in Section 4.4. The number of cuts would depend upon the solution obtained during each iteration. If the sequence consists of clean or regeneration, we replace them with new and compute the average daily NO_x reduction. Then if the sequence with new still does not meet the minimum average daily NO_x requirement, we cut off this solution as well as all the combination of the less effective solutions as shown in Table 4.9, since a new layer is the most effective

action for NO_x reduction. Whereas, if the sequence with a new layer does meet the minimum average daily NO_x requirement, then we just cut off the initial optimal solution and re-optimize since in this case we are nearing the optimal solution. We then apply this algorithm to our initial six-plant example described in Section 4.3.2, and the optimal solution was found to be the same as Section 4.6. We then compare the computational time of multi-cut MCFP with MCFP and SGO with fixed NO_x policy in Table 4.10.

Table 4.10. Fleet with multi-cut MCFPwSEC comparison

	SGO (Fixed NO_x Policy)	MCFPwSEC (Single Cut)	MCFPwSEC (Multi-Cuts)
CPU Time	610.67 minutes	57.11 minutes	20.36 minutes

From the Table 4.10, we have managed to reduce the computational time further from 57.11 minutes down to 20.36 minutes with the multi-cut MCFP. While the computational time of 20.36 minutes is very reasonable for this type of problem, we try to explore potential ways to reduce this even further. As mentioned earlier, in the relaxed MCFP formulation, we cannot directly derive average daily NO_x reduction. In addition, recall that RP is directly proportional to NO_x, so we would like to incorporate a constraint that reduces the generation of schedule elimination constraints in Section 4.4. Consequently, we introduce the average RP constraint that will be described in Section 4.6.

4.6 Average RP Constraints

In this section, we incorporate a constraint that represents the relationship between RP and NO_x reduction. From the MCFP structure, the average daily NO_x reduction cannot be obtained directly prior to a solution. We realize that we would like the RP to be as high as possible. From observations of the solutions and the formulation of edges, we observe the following relationship.

$$DNOX_{\%}(\text{Avg.RP}) \geq \text{Average Daily daily } NO_x \text{ reduction} \tag{4.10}$$

As mentioned earlier in Section 4.2, when edges are formulated, the average daily RP value incurs between two consecutive outages i and j is also calculated. Consequently, if we sum the RP across all edges for each plant over the time horizon, we would obtain the weighted average of RP over the time horizon for the schedule. This weighted average RP can then be used to determine a bound on NO_x reduction. $DNOX_{\%}(\text{Avg.RP})$ represents this calculation. From observation, we have found that this is an upper bound to the average daily NO_x reduction. In addition, we would like these values to be greater than the minimum average daily NO_x value. Therefore, we obtain the following relationship.

$$\begin{aligned} DNOX_{\%}(\text{Avg.RP}) &\geq \text{Average Daily } NO_x \text{ reduction} \\ &\geq \text{Minimum average Daily } NO_x \end{aligned} \tag{4.11}$$

For each edge $i \in E$, let $\overline{RP}_i$ be the average daily RP of edge i. Consequently, we add the following valid inequality as a constraint to the MCFP.

$$\sum_{i \in E} \overline{RP}_i x_i \geq DNOX_{\%}^{-1}(\text{Minimum average daily } NO_x) \tag{4.12}$$

After the constraint 4.12 is added to the optimization model, we ran the experiment again using the MCFPwSEC described in Section 4.4. After the run, we found

the optimal solution with no improvement in the computational time as compared to that of Section 4.4. The computational time found was 60.85 minutes compared with the computational time in Section 4.4 that was 20.35 minutes. The reason for an increase in the computational time is due to the nature of the constraint. Although the added RP constraint resulted in the reduction of 1738 iterations in the generation of schedule elimination constraints, we also introduce large fractional coefficients to the problem. These fractional coefficient reduce the CPU time required to solve the relaxed MCFP. Therefore, the RP constraint has provided no further improvement in terms of the computational efficiency to the MCFP model. The following Table 4.11 summarizes the MCFP model.

Table 4.11. Summary of fleet MCFP comparison

	MCFPwSEC (Single Cut)	MCFPwSEC (Multi-Cuts)	MCFPwSEC (Single Cut) w/Average RP Constraint
CPU Time	57.11 minutes	20.36 minutes	60.85 minutes
Number of Iterations	5250	776	3512

4.7 Conclusion

In this chapter, we proposed the MCFP model to solve the fleet SCR management problem. The motivation for this is purely due to the computational time of the SGO algorithm. While the SGO algorithm discussed in Chapter 3 has produced the optimal solution within a reasonable time of 610.67 minutes. We have found that as the problem size gets larger especially based upon the length of time horizon, the computational time increases exponentially. In this chapter, we first formulated

the relaxed MCFP. We then applied the formulation on a single plant example. We expanded the problem further by applying it to a six-plant example with the same input information as the SGO fixed NO_x policy. Since the average daily NO_x reduction constraint is critical to the model, it is essential to incorporate it into MCFP model. Therefore, we introduced the MCFPwSEC discussed in Section 4.4. From the MCFPwSEC algorithm, we have found significant reduction in computational time from SGO algorithm while achieving the same result. The computational time of 610.67 minutes for SGO has been reduced down to 57.11 minutes. We then explored further to find ways to improve the algorithm performance. From investigation of the cuts generated in each iteration by MCFPwSEC, we introduced multi-cut MCFPwSEC where instead of adding a single cut per iteration, we applied a heuristic method to determine the average daily NO_x reduction for the best solution for the particular sequence and are able to add multiple cuts per iteration. The multi-cut MCFPwSEC algorithm has shown positive results where the computational time has reduced down to 20.36 minutes. Because of the relationship between RP and NO_x reduction, we derived a constraint based upon the relationship. We incorporated an average RP constraint to reduce the generation of heuristic schedule elimination constraints *a priori*. However, since RP values are fractional, we also introduced large fractional coefficients to the problem. Therefore, the RP constraint has provided no further improvement in terms of the computational time to the model. In future research, we can potentially improve the algorithm performance by introducing independent set constraints based upon the RP constraint. Note that the schedule elimination constraints are also a class of independent set constraints. In summary, we have found the MCFP model to be significantly faster than the SGO algorithm and we have managed to reduce the computational time from 610.67 minutes down to 20.36

minutes with multi-cut MCFPwSEC. Finally, Chapter 5 discusses conclusions and future directions.

CHAPTER 5

SUMMARY AND FUTURE DIRECTIONS

In this dissertation, we proposed an SCR management framework to solve the fleet SCR management problem using mathematical optimization techniques. We solved the SCR management problem with two main approaches, which were the SGO algorithm and an MCFP model. From the study on SCR management, we have found little research on optimization of SCR management, and the main focus of most literature is on optimizing the process design. In addition, we have found that most commercial software focuses on SCR management of a single plant. We addressed these limitations by introducing an SCR management framework that maximizes NO_x reduction or minimizes the total operating costs over a fleet-wide set of plants given a scheduled outage plan. In the first part of the dissertation, we proposed the SGO algorithm with recursion to enumerate a set of potential outage schedules for all plants in the fleet. From these generated schedules, a 0-1 large-scale integer programming model, with COIN-OR CBC as the solver, selects an optimal set of schedules. We then demonstrated the effectiveness of the algorithm by providing three computational experiments. The first two computational experiments used the fixed NH_3 slip policy with simulated data. The experiments considered different objectives and Pareto optimal efficient frontiers. The third computational experiment was based upon a modified version of a real-world problem instance using the fixed NO_x policy. From the model, we found that the solution is applicable in a real-world situation. However, one of the main drawbacks of SGO is the computational time due to the recursion method. The problem instance included six plants and a five-

year planning horizon, and the optimal solution was obtained with a computational time of 610.67 minutes. However, as the problem size gets larger, the computational time also increases exponentially. We then addressed this limitation by introducing an MCFP model, where the solver used for the MCFP is the CPLEX 12.1 callable library. Instead of generating schedules, edges are generated to represent the flow of all SCR layers in the fleet. We started out with a relaxed MCFP that solved a single plant example. We then expanded further to that six-plant example used in the SGO computational experiments with the fixed NO_x policy. In order to incorporate the average daily NO_x reduction constraint implemented in SGO to the MCFP model, we introduced schedule elimination constraints that cut off solutions that did not meet the minimum average daily NO_x reduction requirement. Consequently, we developed MCFPwSEC that iteratively solves MCFP and adds a schedule elimination constraint until the minimum average daily NO_x reduction condition is met. From this algorithm, we found that the computational time was 57.11 minutes, which was significantly less than that of the SGO algorithm. We improved this further by introducing multi-cut MCFPwSEC, where instead of cutting infeasible solutions one by one, we could eliminate many schedules at once by observation of the cuts. From there, we reduced the computational time down to 20.36 minutes. To potentially further reduce the computational time, we explored the relationship between average daily NO_x reduction and average daily RP. Consequently, we added an RP constraint to the MCFP in order to eliminate the generation of schedule elimination constraints early in the algorithm execution. We found that although the number of generated cuts decreased, the added constraint provided no improvement in the computational time. The primary reason is due to the nature of the constraints. Since RP values are fractional, when the constraint is added, we also introduced large fractional coefficients into the problem. In future research, we will explore potential ways to

reduce the computational time further, especially on a larger problem instance. From what we have learned from the relationship between average daily NO_x reduction and average daily RP, we could incorporate independent set constraints based upon the RP constraint to the model. Note that the schedule elimination constraints are also a class of independent set constraints. Consequently, we can also explore lifting of either or both independent set constraints based upon the RP constraint and the schedule elimination constraints. Furthermore, for a larger problem instances, branch-and-price may be helpful. Finally, in this dissertation, the scheduled outage plan was assumed to be pre-specified, and swapping outage dates was not allowed. Based upon discussions with users, we have found that this policy is practical in real-world situations. However, we could potentially extend the flexibility of the framework by allowing the swapping of outage dates.

REFERENCES

[1] "Tennessee Valley Authority: Coal-fired power plant," http://www.tva.gov/power/coalart.htm.

[2] J. Staudt and A. Engelmeyer, "SCR catalyst management strategies-modeling and experience," in *Proceedings of the DOE-EPRI-EPA-AWMA Power Plant Air Pollutant Control Mega Symposium: DOE/EPRI/EPA/AWMA*, Washington, DC, May 2003.

[3] E. Rubin, J. Kalagnanam, H. Frey, and M. Berkenpas, "Integrated environmental control concepts for coal-fired power plants," *Journal of the Air and Waste Management Association*, vol. 47, no. 11, pp. 1180–1188, 1997.

[4] "The U.S. Department of Energy and Southern Company Services, Inc.: Topical report number 9, control of nitrogen oxide emissions: Selective catalyst reduction (SCR)," http://www.netl.doe.gov/technologies/coalpower/cctc/topicalreports/pdfs/topical9.pdf, 1997.

[5] H. Frey, "Engineering-economic evaluation of SCR NOx control systems for coal-fired power plants," in *Proceedings of the American Power Conference*, Chicago, Illinois, 1995.

[6] L. Muzio, G. Quartucy, and J. Cichanowicz, "Overview of status of post-combustion NOx control: SNCR, SCR, and hybrid technologies," *International Journal of Environment and Pollution*, vol. 17, no. 1-2, pp. 4–30, 2002.

[7] S. Pritchard, C. DiFrancesco, S. Kaneko, N. Kobayashi, K. Suyrgama, and K. Eda, "Optimizing SCR catalyst design and performance for coal-fired boilers,"

in *Proceedings of the EPRI/EPA 1995 Joint Symposium on Stationary Combustion NO Control*, Kansas City, MO, 1995.

[8] J. Cichanowicz and L. Muzio, "Factors affecting selection of a catalyst management strategy," in *Proceedings of the DOE-EPRI-EPA-AWMA Power Plant Air Pollutant Control Mega Symposium: DOE/EPRI/EPA/AWMA*, Washington, DC, May 2003.

[9] J. Cichanowicz, L. Smith, and L. Muzio, "Twenty-five years of SCR evolution: Implications for U.S. application and operation," in *Proceedings of the EPRI-DOE-EPA Combined Power Plant Air Pollutant Control Symposium: The MEGA Symposium; EPRI/DOE/EPA*, Chicago, IL, August 2001.

[10] J. Cichanowicz, L. Smith, L. Muzio, and J. Marchetti, "100 GW of SCR: Installation status and implications of operating performance on compliance strategies," in *Proceedings of the EPA-EPRI-NETLAWMA Combined Power Plant Air Pollutant Control Mega Symposium: EPA/EPRI/NETL/AWMA*, Washington, DC, May 2003.

[11] E. Rubin, J. Salmento, and H. Frey, "Cost-effective emission controls for coal-fired power plants," *Chemical Engineering Communications*, vol. 74, pp. 155–167, 1988.

[12] S. Pritchard and C. DiFrancesco, "SCR catalyst management: Enhancing operational flexibility," Cormetech, Inc., Tech. Rep., 2006.

[13] J. Mi, "CFD for SCR design," Southern Company Inc., Tech. Rep.

[14] K. Rogers, M. Milobowski, and B. Wooldridge, "Perspectives on ammonia injection and gaseous static mixing in SCR retrofit applications," in *Proceedings of the EPRI-DOE-EPA Combined Utility Air Pollutant Control Symposium*, Atlanta, Georgia, 1999.

[15] J. Chen and H. Frey, "Optimization under variability and uncertainty: A case study for NOx emissions control for a gasification system," *Environmental Science and Technology*, vol. 38, pp. 6741–6747, 2004.

[16] U. Diwekar, E. Rubin, and H. Frey, "Optimal design of advanced power systems under uncertainty," *Energy Conversion and Management*, vol. 38, no. 15, pp. 1725–1735, 1997.

[17] Grabitech, "MultiSimplex software," http://www.grabitech.com/Multisimplex.htm.

[18] FERCo, "CatReact software," http://www.ferco.com/catreact.html.

[19] R. Ahuja, T. Magnanti, and J. Orlin, *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., 1993.

[20] H. Taha, *Operations research: An introduction*, 7th ed. Prentice-Hall, Inc., 2002.

[21] M. Cheng, Y. Li, and D. Du, *Combinatorial optimization in communication networks*, ser. Combinatorial optimization. Springer-Verlag New York, Inc., 2006, vol. 18.

[22] L. Ford and D. Fulkerson, "A suggested computation for maximal multi-commodity network," *Management Science*, vol. 5, no. 1, pp. 97–101, 1958.

[23] A. Sood, *Logical topology design for WDM networks using tabu search*. Library and Archives Canada, 2007.

[24] M. Milano and P. Van Hentenryck, *Hybrid Optimization: The Ten Years of CPAIOR*. Springer, 2010.

[25] T. Hu, "Multi-commodity network flows," *Operations Research*, vol. 11, no. 3, pp. 344–360, 1963.

[26] L. Kantorovich, "Mathematical methods in the organization and planning of production. Publication House of the Leningrad University," *Translated in Management Science*, vol. 6, pp. 366–422, 1960.

[27] F. Hitchcock, "The distribution of a product from several sources to numerous facilities," *Journal of Mathematical physics*, vol. 20, pp. 224–230, 1941.

[28] T. Koopmans, "Optimum utilization of the transportation system," in *Proceedings of the International Statistical Conference*, Washington, DC, 1947.

[29] ——, "Optimum utilization of the transportation system," *Econometrica*, vol. 17, 1949.

[30] G. Dantzig, "Application of the simplex method to a trasportation problem," in *Activity Analysis and Production and Allocation*. New York: Wiley, 1951, pp. 359–373.

[31] L. Ford and D. Fulkerson, "Maximal flow through a network," *Canadian Journal of Mathematics*, vol. 8, pp. 399–404, 1956.

[32] R. Gomory and T. Hu, "Multi-terminal network flows," *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, no. 4, pp. 551–570, 1961.

[33] ——, "An application of generalized linear programming to network flows," *Journal of the Society for Industrial and Applied Mathematics*, vol. 10, no. 2, pp. 260–283, 1962.

[34] D. Tang, "Bi-path networks and multicommodity flows," *IEEE Transactions*, vol. 11, no. 4, pp. 468–473, 1964.

[35] J. Tomlin, "Minimum-cost multicommodity network flows," *Operations Research*, vol. 14, no. 1, pp. 45–51, 1966.

[36] J. Kennington, "A survey of linear cost multicommodity network flows," *Operations Research*, vol. 26, no. 2, pp. 209–236, 1978.

[37] R. Bixby and W. Cunningham, "Converting linear programs to network problems," *Mathematics of Operations Research*, vol. 5, no. 3, pp. 321–357, 1980.

[38] D. Fontes, E. Hadjiconstantinou, and N. Christofides, "A branch-and-bound algorithm for concave network flow problems," *Journal of Global Optimization*, vol. 34, no. 1, pp. 127–155, 2006.

[39] H. Bellmore and H. Ratliff, "Optimal defense of multi-commodity networks," *Management Science*, vol. 18, no. 4, pp. B174–B185, 1971.

[40] B. Baker, "A network-flow algorithm for project selection," *The Journal of the Operational Research Society*, vol. 35, no. 9, pp. 847–852, 1984.

[41] J. Farvolden, W. Powell, and I. Lustig, "A primal partitioning solution for the arc-chain formulation of a multicommodity network flow problem," *Operations Research*, vol. 41, no. 4, pp. 669–693, 1993.

[42] C. Hane, C. Barnhart, E. Johnson, R. Marsten, G. Nemhauser, and G. Sigismondi, "The fleet assignment problem: Solving a large-scale integer program," *Mathematical Programming*, vol. 70, no. 1, pp. 211–232, 1995.

[43] C. Barnhart, N. Boland, L. Clarke, E. Johnson, G. Nemhauser, and R. Shenoi, "Flight string models for aircraft fleeting and routing," *Transportation Science*, vol. 32, no. 3, pp. 208–220, August 1998.

[44] R. McBride, "Advances in solving the multicommodity-flow problem," *Interfaces*, vol. 28, no. 2, pp. 32–41, 1998.

[45] J. Rosenberger, E. Johnson, and G. Nemhauser, "A robust fleet-assignment model with hub isolation and short cycles," *Transportation Science*, vol. 38, no. 3, pp. 357–368, August 2004.

[46] V. Pilla, J. Rosenberger, V. Chen, and B. Smith, "A statistical computer experiments approach to airline fleet assignment," *IIE Transactions*, vol. 40, no. 5, pp. 524–537, 2005.

[47] "COIN-OR Branch-and-Cut MIP Solver," https://projects.coin-or.org/Cbc.

[48] G. Nemhauser and L. Wolsey, *Integer and combinatorial optimization*. Wiley-Interscience, 1999.

[49] M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems," *SIAM Rev.*, vol. 33, no. 1, p. 60100, 1991.

[50] IBM, "CPLEX callable library version 12.1 C API reference manual," ftp://ftp.software.ibm.com/software/websphere/ilog/docs/optimization/cplex/refcallablelibrary.pdf, 2009.

[51] L. Muzio, R. Smith, G. Quartucy, and Q. Qader, "Recent experiences tuning SCR systems," in *Combined Power Plant Air Pollutant Control Mega Symposium: EPA/EPRI/NETL/AWMA*, Washington, DC, May 2003.

[52] "Computational infrastructure for operations research (COIN-OR)," http://www.coin-or.org.

[53] R. Marsten and M. Muller, "A mixed-integer programming approach to air cargo fleet planning," *Management Science*, vol. 26, no. 11, pp. 1096–1107, 1980.

[54] P. Phananiramai, J. Rosenberger, V. Chen, S. Kim, and M. Sattler, "A mathematical optimization technique for managing selective catalytic reduction for coal-fired power plants," COSMOS Technical Report 10-04, The University of Texas at Arlington. Arlington, TX, Tech. Rep., 2010. [Online]. Available: http://www.uta.edu/cosmos/TechReports/COSMOS-10-04.pdf

[55] ——, "A mathematical optimization technique for managing selective catalytic reduction for coal-fired power plants," *Energy Systems*, pp. 1–18, 2011, 10.1007/s12667-011-0030-0. [Online]. Available: http://dx.doi.org/10.1007/s12667-011-0030-0

BIOGRAPHICAL STATEMENT

Passakorn Phananiramai received his Bachelor's degree in Industrial Engineering from Sirindhorn International Institute of Technology (SIIT), Thammasat University, Thailand, in 2003. Before joining The University of Texas at Arlington (UTA), he worked as an Engineer at the Industrial Finance Corporation of Thailand (IFCT). In 2007, he received his Master's degree in Industrial and Manufacturing Systems Engineering at UTA where he has continued pursuing the Ph.D. degree. During his study at UTA, he also worked as a research assistant for Dr. Jay Rosenberger on various Luminant projects. His research interests is in the areas of Operations Research, Mathematical Optimization, Operations Management, and Integer Programming. He is a member of the Center of Stochastic Modeling, Optimization and Statistics (COSMOS) and Institute for Operation Research and Management Science (INFORMS).