NEW APPROACHES TO ROBUST OPTIMIZATION WITH APPLICATIONS

By

MYKYTA I. BOYKO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010

I dedicate my thesis to parents Igor and Olga, wife Lidiya, and son Boris.

ACKNOWLEDGMENTS

I am very thankful to my advisor and mentor Prof. Panos Pardalos for his support during my doctorate studies at the University of Florida. His enthusiastic help and research guidance helped me develop on on both professional and personal levels. I would like to express my gratitude to other members of my doctorate committee, Prof. Vladimir Boginski, Prof. Stan Uryasev, and Prof. William Hager for their valuable contribution to my research. I would also like to express my greatest appreciation to my colleagues from the Center for Applied Optimization. Intensive exchange of ideas and joint research with my fellow graduate students and post docs from the center helped me significantly in my work. Last but not least, I would like to thank my family and friends, who supported and encouraged me in all of my beginnings.

TABLE OF CONTENTS

page

ACKNOWLEDGMENTS 4

LIST OF TABLES 7

LIST OF FIGURES 8

LIST OF SYMBOLS 9

ABSTRACT 10

CHAPTER

1 INTRODUCTION 12

2 STATE OF THE ART REVIEW OF ROBUST OPTIMIZATION 16

2.1 Robust Optimization Formulations 16
2.2 Relation to Stochastic Programming 18
2.3 Scenario-based Optimization Approach 19

3 ROBUST MULTI-SENSOR SCHEDULING 26

3.1 Optimization Techniques for Sensor Networks 26
3.1.1 Positioning Using Angle of Arrival 28
3.1.2 Semidefinite Programming (SDP) for Sensor Network Localization 30
3.1.3 Network Interdiction 32
3.2 Robust Sensors Scheduling 38
3.2.1 Optimization Models for Sensors Scheduling 38
3.2.2 Deterministic setup 43
3.2.3 Problem Setup under Uncertainty 46
3.3 Equivalent Formulations in Cardinality Constraints 47
3.4 Sensor Scheduling in Network-Based Settings 53
3.5 Computational Experiments 57

4 TWO STAGE STOCHASTIC OPTIMIZATION MODEL FOR ROBUST NETWORK FLOW DESIGN 61

4.1 Problem Formulation 61
4.2 Decomposition Method for Network Flow Problem 66
4.3 Computational Experiments 75

5 ROBUST SYSTEM IDENTIFICATION FOR SPACE WEATHER FORECASTING 78

5.1 Modeling Magnetosphere as a Black Box 79
5.2 Robust Model Reconstruction 83
5.3 Nonlinear Structure Reconstruction 84

6 CONCLUSION 86
6.1 Thesis Contribution 86
6.2 Future Work 87
REFERENCES 89
BIOGRAPHICAL SKETCH 94

LIST OF TABLES

Table page

3-1 Performance results for deterministic model (3–57)-(3–64). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$. . . 57

3-2 Performance results for CVaR type deterministic model (3–67). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$. CVaR confidence level $\alpha = 0.9$. 58

3-3 Performance results for CVaR type stochastic model (3–75). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$, number of scenarios $S = 100$. CVaR confidence level $\alpha = 0.9$. 59

3-4 Comparing PSG and CPLEX performance for obtaining approximate solution of CVaR type stochastic problem (3–75)($n = 12$ sites and $T = 10$ time periods). 60

3-5 ILOG CPLEX CPU time (sec) for network deterministic model (3–102)-(3–113). n - number of sites; m - number of sensors. The number of discrete time steps is T = 10. 60

5-1 Leaps-and-bound based variable selection for fixed number of regressors $k = 1, \ldots, 8$ for one step ahead forecasting (linear model). "+" in columns k indicates that the corresponding variable is added to the model. 84

5-2 Leaps-and-bound based variable selection for fixed number of regressors $k = 1, \ldots, 10$ for one step ahead forecasting (bilinear). "+" in columns k indicates that the corresponding variable is added to the model. 85

LIST OF FIGURES

Figure page

2-1 Graphical representation of VaR and CVaR for a random value L specified by probability density function $f_L(x)$. 22

2-2 Graphical representation of VaR and CVaR for a finite set of stochastic scenarios (discrete case). 23

3-1 Example of a possible network. Two nodes are connected by an arc if a sensor can move from one node to another in consequent time periods. 53

3-2 Counterexample ($m = 2$): two sensors cannot perform simultaneous feasible move due to the constraint $x_{1,1} + x_{4,2} \leq 1$ 55

4-1 Effect caused by network failure. In both cases the network lost the same amount of flow. 64

4-2 New loss function in a 6 vertice network. A - initial network, B - minimal cost flow without with CVaR in objective, C deterministic case without CVaR 76

4-3 Flow schedule in the network with and without considering CVaR robust constraint. A - initial network, B - minimal cost flow without without considering reliability issues, C and D - CVaR of loss is bounded by 5 and 10 respectively. 77

LIST OF SYMBOLS, NOMENCLATURE, OR ABBREVIATIONS

AoRDA PSG	American optimal Decision Portfolio Safeguard
$\mathrm{card}(\cdot)$	Cardinality function (number of nonzero components of vector argument)
CVaR	Conditional Value-at-Risk
E	Set of arcs
G	Network graph
$I(\cdot)$	Indicator function
$\inf(\cdot)$	Infimum
LP	Linear programming
$\max(\cdot)$	Maximum
MILP	Mixed integer linear programming
$\min(\cdot)$	Minimum
pdf	Probability density function
Prob$\{\}$	Probability measure of event
R	Set of real numbers
s.t.	Subject to
$\sup(\cdot)$	Supremum
UAV	Unmanned Aerial Vehicle
$\mathcal{U}$	Uncertainty set
VaR	Value-at-Risk
V	Set of vertices

Abstract of dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

NEW APPROACHES TO ROBUST OPTIMIZATION WITH APPLICATIONS

By

Mykyta I. Boyko

August 2010

Chair: Panos M. Pardalos
Major: Industrial and Systems Engineering

The dissertation focuses on a popular percentile-based risk measure known as Conditional Value at Risk (CVaR). CVaR has emerged from financial applications and allows obtaining optimal solutions that are robust with respect to various uncertainties incorporated into models as random variables. In this work I solve three operations research problems using CVaR-based robustness: multi-sensor scheduling problem, network-flow and system identification for space weather prediction.

I develop mathematical programming techniques for solving a class of multi-sensor scheduling problems and formulate robust optimization problems for both deterministic and stochastic cases using linear 0-1 programming techniques. Equivalent formulations are developed in terms of cardinality constraints. I conducted numerical case studies and analyzed the performance of optimization solvers on the problems under consideration.

The next problem I consider is robust network flow problem. I propose a new stochastic formulation of minimum cost flow problem aimed at finding network design and flow assignments subject to uncertain factors, such as network component disruptions/failures. I introduced loss function that is proportional to actual loss of flow in the network. In order to quantify the uncertain loss caused by network failures, I utilized CVaR risk measure. The combination of Lagrangian Relaxation and Benders' decomposition is proposed to solve large problems

Predicting geomagnetic activity applied for Dst-index forecasting has been performed through robust identification of discrete dynamic system. The modeling assumes that the state of magnetosphere plasma is determined by solar wind velocity and the magnitude of southern component of magnetic field. The structure of the system is obtained using statistical techniques. The CVar robust deviation measure has been suggested for obtaining system parameters of a stable prediction model.

CHAPTER 1
INTRODUCTION

The main objective of this dissertation work is to extend existing mathematical programming models to stochastic settings, where some parameters of the models, such as the reliability of connections or penalties, are uncertain. In the real settings typical for the majority of civil and military applications, some components of the system may become temporarily unavailable due to weather conditions and other issues that can affect normal operating procedures. These uncertain factors can make the traditional optimal solution infeasible and therefore unrealistic. That is why, it is crucial to develop mathematical programming models, that will take these factors into account. The objective of the approach suggested in the work is to control and restrict possible losses by utilizing appropriate quantitative risk measures and introducing them into mathematical programming models. Specifically, I propose to use the concept of Conditional Value-at-Risk (CVaR) which has been used in financial engineering applications, and has recently been widely applied to studying scheduling, network flow or forecasting problems.

Rockafellar and Uryasev were the pioneers to develop the general methodology of optimization of CVaR and apply it to portfolio optimization problems [45]. This methodology has been very well received by both academic community and practitioners (as of February 2010, Google Scholar reports more than 1000 citations of this original paper). This dissertation will develop appropriate mathematical programming formulations using CVaR to achieve robust and efficient performance of sensors scheduling, network flow and prediction models.

A general form of a typical optimization problem aims at finding the optimal solution that remains feasible for all possible parameters from the uncertainty set. This traditional approach to finding robust solution has some disadvantages. First, the classical robust problem is more difficult to solve than its deterministic counterpart. Second,

the traditional robust formulation is often criticized for its conservatism. Indeed, the worst case scenario might be a highly unlikely event and, therefore, it would be more reasonable to consider a set of worst case scenarios. In order to avoid the conservatism of traditional robust optimization, we replace worst case scenario with an "averaged" set of worst case scenarios. To accomplish this, let us assume that uncertain parameters are governed by some probability measure. We will also associate some loss with each implementation of the worst case scenarios. In order to quantify the uncertainty, we will use a popular risk measure known as Conditional Value at Risk (CVaR) in the literature. CVaR is closely related to a well-known quantitative risk measure referred to as Value-at-Risk (VaR), which is widely used in financial engineering. The α-CVaR equals to the expectation of the α % of worst cases of the loss caused by uncertainty factors. This work contributes to the area of robust optimization by creating a modeling framework for three application areas using CVaR-type robust constrains. All the problems will be modeled using a large yet final set of scenarios.

One of the applications is the robust sensors scheduling for multiple site surveillance. The task of area surveillance is important in a variety of applications in both military and civilian settings.One of the main challenges that need to be addressed in these problems is the fact that the number of locations (sites) that need to be visited to gather potentially valuable information is often much larger than the number of available surveillance devices (sensors) used for collecting information. Under these conditions, it is necessary to schedule all of the available sensors (that can be installed, for instance, on Unmanned Air Vehicles) in order to maximize the amount of valuable information collected by the sensors. It is possible to formulate this problem in terms of minimizing the information losses associated with the fact that some locations are not under surveillance at certain time moments. In these settings, the information losses can be quantified as both *fixed* and *variable* losses, where fixed losses would occur when a given site is simply not under surveillance at some time moment, while variable

losses would depend on how long a site has not been previously visited by a sensor. Particularly, consideration of variable losses of information is critical for strategically important sites that need to be monitored as closely as possible.

In addition, the parameters that quantify fixed and variable information losses are in many cases *uncertain* by nature. In previous works related to this area, the uncertainties in these parameters were not explicitly taken into account (see, e.g., [63]). However, it is crucially important to develop efficient techniques to minimize or restrict the information losses under uncertainty. This chapter proposes mathematical programming formulations that allow quantifying and restricting the risks of worst-case losses associated with uncertain parameters.

The mathematical programming formulations are first developed for the deterministic case. The natural extensions of these formulations to the stochastic case (with uncertain information loss parameters) are made by utilizing quantitative risk measures that allow to control the conservativeness of the optimal strategy. In particular, the statistical concept referred to as Conditional Value-at-Risk (CVaR) is used in the proposed problem formulations under uncertainty. Using these techniques allows to efficiently incorporate uncertainties into the optimization problems under consideration, as well as to provide the means to balance between the optimality and the robustness of the solutions. Equivalent reformulations and extensions of the given problems are also provided. Several numerical case studies verify the efficiency of the suggested algorithms. Two software packages, IBM CPLEX and AOrDa PSG, are used to solve the case studies.

The second application presents an approach to finding an optimal robust network structure with respect to uncertain factors, such as demands, component failures, etc. A two stage stochastic optimization problem is formulated and Benders decomposition is proposed for solving the problem with a large number of second stage variables. The work extends the model initially suggested by Boginski *et al* in [16]. It is assumed that

any arc of the network can be independently destroyed with the known probability and the loss is determined by the flow alone the failed arcs. The present work suggests a better loss function which is based on the amount of flow undelivered to the consumers due to the realization of a failure scenario. That is the loss can represent penalty fines for failure to deliver a contracted amount of product to consumers. The network is designed to be robust with respect to this penalty. A two stage stochastic optimization problem is formulated and Benders decomposition is proposed for solving the problem with a large number of second stage variables.

The third application will perform system identification using the soft margin approach. The identified linear and bilinear models aim at forecasting solar index activity for predicting geomagnetic storms. Presently, identification and prediction models are widely used for studying linear and non-linear processes in the space. It is possible to predict the Earth magnetosphere by building black box model which related solar wind parameters (such as magnetic field characteristics, solar wind speed, etc) to measurable geophysics indices (for example Kp or Dst indices). Work [19] suggested a black-box technique for modeling complex processes in magnetosphere, by assuming that they are described by an unknown bilinear system. A CVaR-based deviation function is suggested to use in order to provide a better stability of the underlying dynamic system.

CHAPTER 2
STATE OF THE ART REVIEW OF ROBUST OPTIMIZATION

2.1 Robust Optimization Formulations

This chapter describes how data uncertainty can be addressed in optimization models. Any real life optimization problem contains parameters that are not exact in certain sense. The parameters can be measured with some error or caused by changes of condition over time. The parameters can be of a random nature, for example stock quotes or system component failures. The simplest solution is to consider a "most typical" value of a parameter (i.e. average or single available observation) that accounts a single most like averages. However, the deviation from the average can easily make the problem infeasible or result in a solution where perturbed parameters will differ significantly from the initial solution.

Even such natural source of uncertainty caused by approximate measurement or insufficient knowledge of the phenomenon of study can dramatically affect the solution. The book [8] illustrates it with an impressive case study. The authors look into a constraint of the real world linear problem having a thousand of decision variables and 410 constraints. Each of the 1000×410 coefficients of the problem is some real number with several digits. After selecting a constraint, it turns out that a very modest assumption such as the measurement error for the constraint matrix coefficients can deviate within 0.01%, which results in 450% violation of the constraint in the worst case. Moreover, if the actual parameters are assumed to be uniformly distributed on a segment of 0.01% the constraint is violated with probability 0.5. The same constraint is violated by 150% with probability 0.18. Even insufficient precision of data representation significantly affect the quality of the nominal solution.

A general form of a typical optimization problem can be formulated as follows

$$\text{Minimize}_{x \in \mathbb{R}^n} f_0(x, \xi), \tag{2–1}$$

$$\text{s.t.} \quad f_i(x, \xi) \leq 0, \ i = 1, ..., m, \tag{2–2}$$

where ξ are the parameters of the optimization problem.

Traditionally, robust optimization paradigm is based on the following reasonable assumption made for the parameters $\xi \in \mathcal{U} \subseteq R^k$.

- Actual data or parameters of the model are represented with the so-called uncertainty set $\mathcal{U}$.
- By the time we obtain an optimal solution x we do not know the actual values of the uncertain parameters.
- When actual parameters are revealed, the optimal solution x should be "valid" with respect to the uncertain outcome.
- The decision maker cannot tolerate any violation caused by any uncertain outcome, i.e. the margins are "hard". The latter assumption is extremely conservative and is one of the reasons that cause criticism of standard robust techniques.

Thus, a common robust formulation for (2–1)-(2–2) can be written as

$$\text{Minimize}_{x \in \mathbb{R}^n} \max_{\xi \in \mathcal{U}} f_0(x, \xi), \tag{2–3}$$

$$\text{s.t.} \quad f_i(x, \xi) \leq 0, \ i = 1, ..., m, \forall \xi \in \mathcal{U}. \tag{2–4}$$

The latter (2–3)-(2–4) problem is called the *robust counterpart* of the original uncertain problem. This solution is called *robust solution* to the uncertain problem. The uncertainty set $\mathcal{U}$ is often chosen either as a polytop or ellipsoid in k-dimensional space. The simplest version of a polytop can be a parallelepiped produced by interval uncertainty

$$\mathcal{U} = \{[a_1, b_1] \times [a_2, b_2] \times ... \times [a_k, b_k]\}. \tag{2–5}$$

That is each of the component ξ_i belongs to a segment. An ellipsoidal uncertainty assumes that vector of actual parameters lies inside a unit ball with respect to some Mahalanobis metric:

$$\mathcal{U} = \{\xi \in R^k | \xi^T \Sigma \xi \leq 1\} \tag{2–6}$$

for some positive semidefinite $k \times k$ matrix Σ.

2.2 Relation to Stochastic Programming

Robust optimization and stochastic optimization are closely related disciplines. In particular, both of the disciplines address uncertainties. When the uncertain parameters are stochastic by nature, i.e. they are described by probabilistic distribution, a stochastic optimization problem can be formulated for a deterministic counterpart. For example, let ξ be a stochastic parameter. Then the chance constraint can be introduced for the deterministic counterpart (2–1) –(2–2). This constraint practically eliminates the possibility of violations, i.e. the probability of constraint violation $\epsilon \ll 1$ is close to zero. That is if x^* is a robust optimal decision value and f^* is the optimal value,

$$P(f^* < f_0(x, \xi), f_i(x, \xi) \leq 0) \geq 1 - \epsilon. \tag{2–7}$$

In this case, ϵ is the parameter that characterizes wiliness of a decision maker to accept the risk. In particular, the robust solution due to its extreme conservatism will remain feasible for all possible $\epsilon \in [0, 1]$. In many practical cases the chance threshold ϵ can be relatively small. For example, when we do not perform any life-threatening activities, i.e. process sociological data, make aggressive investment on the stock market, a 5% chance that something potentially goes wrong is a rather non restrictive assumption. However, there are applications where the classical robust approach would be more appropriate, such as nuclear plant design or pension fund investments. Nevertheless, even for nuclear plant designs very little chance of failures order of 10^{-9} and less is considered.

At the same time, there are a few challenges related to the stochastic optimization approach. First, the underlying distribution of parameters ξ is not always known. For example, for the stock market, where stochastic optimization is applied intensively, there is no agreement whether stock returns are adequately described with log normal distribution. Second, the straightforward approach to quantify the probability constraint with probability function will result in such a deterministic formulations that will be

computationally intractable. For example, even for the simplest case when deterministic counterpart is LP and parameters $\xi_i,\ i = 1, ..., k$ are uniformly distributed on some intervals, evaluating the probability of constraint violations of (2–7) becomes NP-hard [8]. That is given the input of rational x, rational intervals $[a_i, b_i],\ i = 1, ..., k$ and rational $\delta \in (0, 1)$ there is no algorithm that evaluates the probability with accuracy δ whose running time is bounded with polynom of the bit size of the input (unless P=NP). Moreover, the function $P(.) \leq 1 - \epsilon$ is generally not convex, which makes even good deterministic problems unsolvable under the uncertainty setup.

2.3 Scenario-based Optimization Approach

Finding an exact solution for a general robust problem (2–3)–(2–4) is a nontrivial task that requires special techniques for every particular case. One of the approaches is to allow violation of the robust constraint (2–4) for some very small "fraction" of parameter ξ. Let us assume that ξ has a distribution measured by a probability function Prob. The above assumption results in a chance-constrained optimization problem and is written as follows

$$\text{Minimize}_{x \in \mathbb{R}^n} f_0(x), \tag{2–8}$$

$$\text{s.t.} \qquad \text{Prob}\{f(x, \xi) > 0\} \leq \alpha. \tag{2–9}$$

The parameter $\alpha \in (0, 1)$ is the probability of the violation of constraints (2–4).

Even if the probability distribution for ξ is known, solving such a problem in general setup could be a challenge. A useful approach is to sample uncertain parameter ξ similarly to Monte Carlo simulation and then solve the problem where each scenario provides a set of constraints:

$$\text{Minimize}_{x \in \mathbb{R}^n} f_0(x), \tag{2–10}$$

$$\text{s.t.} \qquad f(x, \xi^s) < 0,\ s = 1, ..., N. \tag{2–11}$$

If all constraints and the objective function are convex, the above problem can be solved efficiently. Let $\hat{x}_N$ be the optimal solution of the problem (2–10)-(2–11). Note that $\hat{x}_N$ depends on N random scenarios and consequently is a random value itself. Now, it is important to estimate how accurately the random approximate solution $\hat{x}_N$ represents the robust optimal solution of (2–10)-(2–11). In other words, how many scenarios need to be sampled to guarantee with certain confidence that the optimal solution violates only a "small" portion (e.g. α%) of the uncertainty set $\mathcal{U}$. Campi and Calafiore have proved the following theorem in [18] which answers the question.

Theorem 2.1 (Campi and Calafiore)**.** *Fix two real numbers $\epsilon \in (0,1)$(level parameter) and $\beta \in (0,1)$ (confidence parameter).If*

$$N \geq N(\alpha,\beta) = \frac{2}{\alpha}\ln\frac{1}{\beta} + 2n + \frac{2n}{\alpha}\ln\frac{2}{\alpha} \tag{2–12}$$

then, with probability no smaller than $1-\beta$, the optimal solution $\hat{x}_N$ of the scenario problem is α-level robustly feasible, that is

$$\mathrm{Prob}\{\xi \in \mathcal{U} : f(x,\xi) > 0\} < \alpha.$$

We can conclude that after relaxing traditional robust constraint we invade into the area of percentile-based optimization. For further discussion let us introduce the notion of Value at Risk (VaR).

Definition 1. *Given some confidence level $\alpha \in (0,1)$ the VaR of the random value L at the confidence level $alpha$ is given by the smallest number l such that the probability that the random value L exceeds threshold l is not larger than $(1-\alpha)$*

$$\mathrm{VaR}_\alpha(L) = \inf\{l \in \boldsymbol{R} : \mathrm{Prob}\{L \geq l\} \leq 1-\alpha\} = \inf\{l \in \boldsymbol{R} : F_L(l) \geq \alpha\}$$

VaR initially emerged in financial engineering as a popular risk measure of the portfolio of random outcome securities. Speaking informally VaR risk measure

answers the question, “What is the lower bound of portfolio loss in $(1-\alpha)$ % worst case scenarios?”

Using the definition of VaR we can write a problem similar to (2–8)-():

$$\text{Minimize}_{x\in\mathbb{R}^n} f_0(x), \tag{2–13}$$

$$\text{s.t.} \quad \text{VaR}_\alpha(f(x,\xi)) \leq 0, \tag{2–14}$$

That is we expect the problem constraint to be robust for a large portion of stochastic outcomes.

Despite wide popularity of VaR there is a criticism of this risk measure. First of all Var does not care about the “tail” of the distribution which potentially can result in underestimation of rare yet high-impact losses, as a result could be a source of false confidence. Second, VaR risk measure is not subadditive (non convex). The relation

$$\text{VaR}(aX+(1-a)Y) \leq a\text{VaR}(X)+(1-a)\text{VaR}(Y),$$

where $a \in [0,1]$ does not necessary hold. The above described process makes incorporating VaR into optimization models to be a challenging process. It is also unnatural from practical perspective: the VaR-based risk of a combined portfolio can be larger than the sum of the VaRs of its components. This contradicts widely recognized opinion that diversification does not increase investment risk.

Let us introduce the notion of Conditional Value-at-Risk (CVaR) [45, 50]. CVaR is closely related to the well-known quantitative risk measure referred to as Value-at-Risk (VaR). By definition, with respect to a specified probability level $(1-\alpha)$ (in many applications the value of $(1-\alpha)$ is set rather high, e.g. 95%), the α-VaR is the lowest amount ζ such that with probability $(1-\alpha)$, the loss will not exceed ζ. Whereas for continuous distributions the α-CVaR is the conditional expectation of losses *above* that amount ζ. As we can see, CVaR is a more conservative risk measure than VaR, which

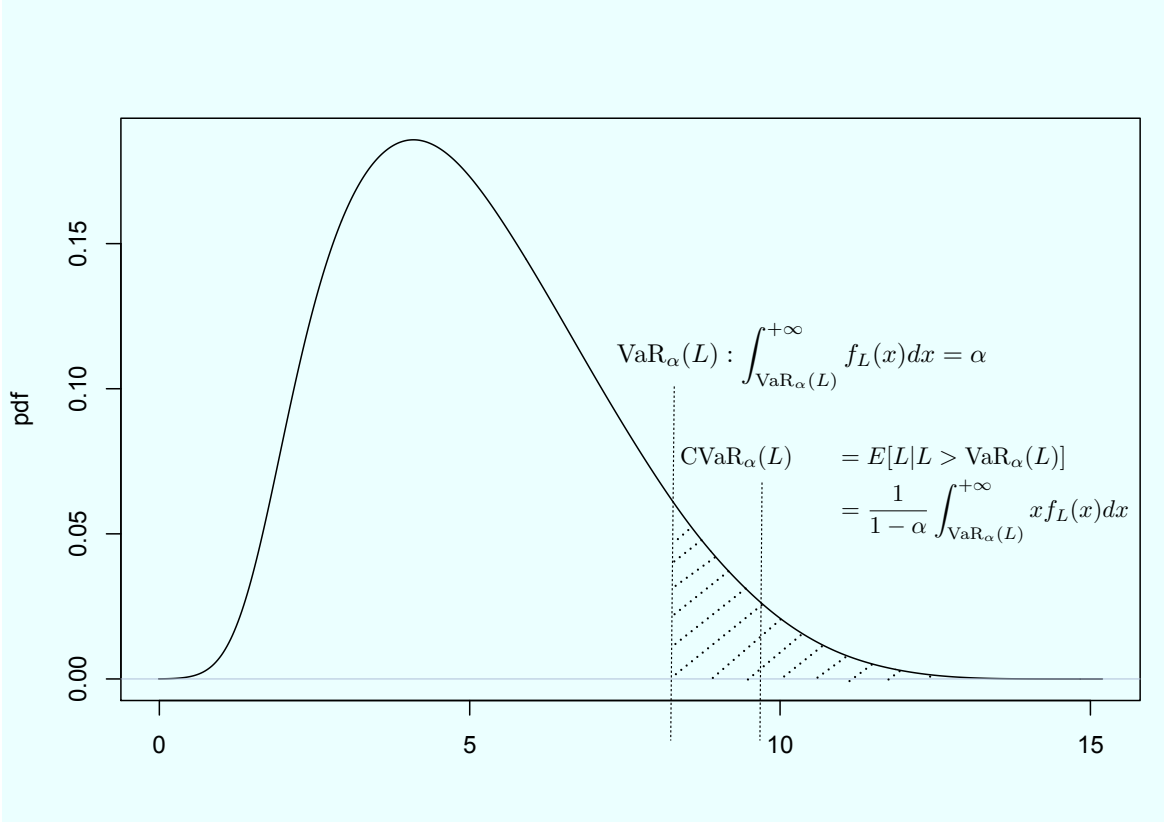


Figure 2-1. Graphical representation of VaR and CVaR for a random value L specified by probability density function $f_L(x)$.

means that minimizing or restricting CVaR in optimization problems provides more robust solutions with respect to the risk of high losses (see figure 2-1).

Formally, α-CVaR of random variable L can be expressed as

$$\mathrm{CVaR}_\alpha(L) \;=\; (1-\alpha)^{-1}\int_{\mathrm{VaR}_\alpha(L)}^{+\infty} x f_L(x)\,dx, \tag{2–15}$$

L is driven by decision vector x and the vector of uncertain parameters ξ. Then if distribution of ξ is known, we can write CVaR of $L(x,\xi)$ as a function of decision parameter x

$$\phi_\alpha(x) = \mathrm{CVaR}_\alpha(L(x,\xi)) = (1-\alpha)^{-1}\int_{y:L(x,y)>\mathrm{VaR}_\alpha(L(x,y)} L(x,y) f_\xi(y)dy, \tag{2–16}$$

Generally, α indicates the *level of conservatism* the decision maker is willing to accept. The closer α approaches to 1, the narrower the range of worst cases becomes in a corresponding optimization problem.

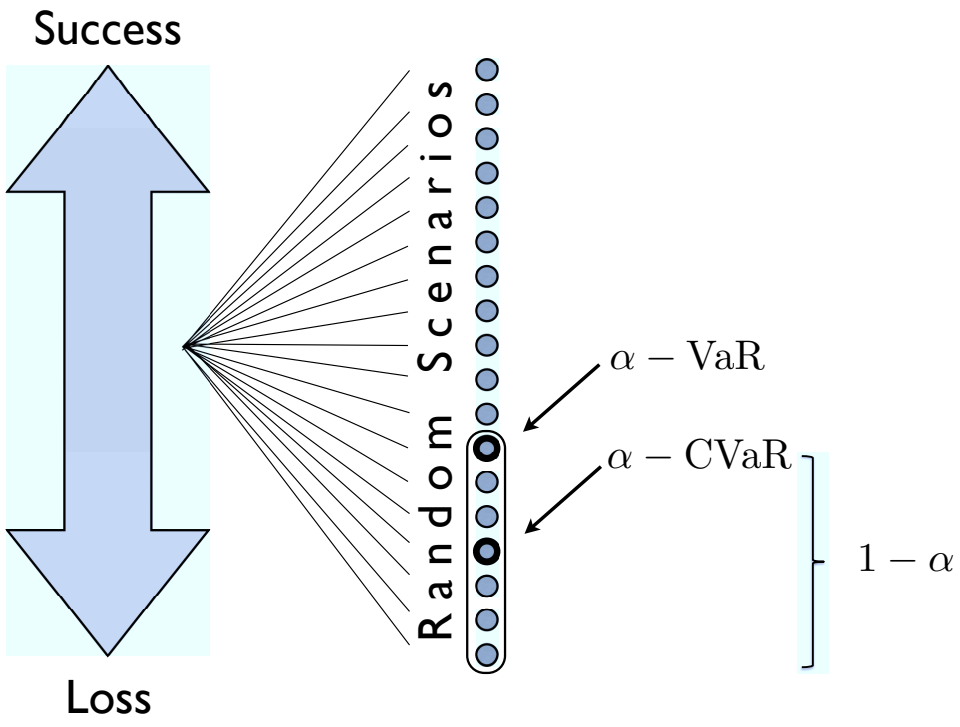


Figure 2-2. Graphical representation of VaR and CVaR for a finite set of stochastic scenarios (discrete case).

CVaR is defined in a similar way for discrete or mixed distributions as it is illustrated in figure 2-2). The reader can find the formal definition of CVaR for general case in [46, 50].

Rockafellar and Uryasev [45, 54] have demonstrated that minimizing $\phi_\alpha(\mathbf{x})$ is equivalent to minimizing the function

$$F_\alpha(\mathbf{x}, \zeta) \;=\; \zeta + (1-\alpha)^{-1} \int_{y \in \mathbb{R}^m} [L(\mathbf{x}, \mathbf{y}) - \zeta]^+ \, p(\mathbf{y}) \, d\mathbf{y}, \tag{2–17}$$

where

$$[t]^+ = \begin{cases} t, & t \geq 0, \\ 0, & t < 0, \end{cases}$$

and the variable ζ corresponds to the VaR value, as it was introduced above.

The significance of this result includes the fact that the function $F_\alpha(x, \zeta)$ can be further simplified by sampling the probability distribution of ξ according to its c.d.f. F_ξ. If the sampling generates a collection of vectors (scenarios) ξ^1, ξ^2,...,xi^S, the corresponding approximation to $F_\alpha(x, \zeta)$ is

$$\tilde{F}_\alpha(x, \zeta) = \zeta + \frac{1}{S(1-\alpha)} \sum_{s=1}^{S} [L(x, y^s) - \zeta]^+. \tag{2–18}$$

The expression $\tilde{F}_\alpha(x, \zeta)$ is convex and piecewise linear with respect to ζ. Moreover, if the function $L(x, y)$ is linear with respect to x, $\tilde{F}_\alpha(x, \zeta)$ can be easily minimized using Linear Programming techniques.

Let us emphasize that the solution $\hat{x}_N$ of the discrete problem is a random value that depends on the randomly sampled scenarios. In order to ensure correctness of such Monte Carlo type approach, it is important to quantify the number of scenarios to provide the solution with certain level of confidences as it was done by Campi and Calafiore [18] for a similar type of problem. The first step in this direction was made in [16] where Boginski *et al* investigated how many scenarios are required to ensure that the true value of the CVaR function $F_\alpha(x, \zeta)$ was close enough to the sample (scenario-based) value that was used in the LP formulation with a high confidence level. In particular, the upper bound required a number of scenarios on the input size was found for a robust network design problem.

Now we are ready to formulate the general framework for obtaining robust solution using CVaR risk measure. If we associate the magnitude of violation of robust constraint with the loss function L, than robust formulation similar to (2–8)-(2.3) can be written as a

nice convex programming model:

$$\text{Minimize}_{x \in \mathbb{R}^n} f_0(x), \tag{2–19}$$

$$\text{s.t.} \quad \text{CVaR}_\alpha f(x, \xi) \leq 0 \tag{2–20}$$

which can be approximated by the linear programming optimization model if all of the objective and constraint functions are linear with respect to decision variables.

CHAPTER 3
ROBUST MULTI-SENSOR SCHEDULING

The chapter develops optimization framework for scheduling several sensors in uncertain environments. I have applied CVaR risk measure to achieve robust scheduling solution in deterministic and stochastics settings. The model is extended for the network set up. The problems, initially formulated as mixed integer linear programs are later reformulated in terms of cardinality functions used in constraints. The theorems are formulated and proved to justify such transition. Cardinality formulations have been solved by AoRDA Portfolio Safeguard software package that used the heuristics optimized for cardinality functions.

This chapter starts with a comprehensive survey into the state of the arts in optimization methods for sensors and sensors networks. These methods are used for such important tasks as scheduling, positioning, building sensors network topologies capable to protect themselves against enemy intrusion, etc.

3.1 Optimization Techniques for Sensor Networks

Organizing sensors into the networks has been recently studied in the literature [3, 6, 27, 29, 35, 36, 43, 60, 64, 65]. Various sensors are used in both civilian and military tasks. Sensing devices can be deployed in either static or dynamic settings, where the positions of each sensor can be permanent or dynamically changing (such as in the case of sensors installed on air vehicles). Multiple sensor systems are commonly represented as networks, since besides collecting important information, sensing devices can transmit and exchange information via wireless communication between sensor nodes. Therefore, network (graph) structures are convenient and informative in terms of efficient representation of the structure and properties thereof. To analyze and optimize the performance of sensor networks, mathematical programming techniques are extensively used.

Before considering a particular problem of robust scheduling for sensors, let us give a brief review of some of the recent developments in mathematical programming as applied to sensor network research. Various types of optimization problems can be formulated and solved in this context [17, 55, 60]. Moreover, the pursued tasks can vary from optimizing the network performance to network interdiction, where the goal is to disrupt enemy networks by interfering with communication network integrity. I will outline the formulations and briefly describe the solution methods used to tackle these problems.

In many cases, the identified optimization problems are challenging from the computational viewpoint, and efficient algorithms need to be developed to ensure that the near-optimal solutions are found quickly. This is essential to guarantee that the decisions regarding efficient operations of sensor networks could be made in a real-time mode, which can be crucial in many applications. Moreover, the uncertain factors that commonly arise in real-world situations also need to be incorporated into the mathematical programming problems, which makes them even more challenging to formulate and solve.

In particular, let us mention several important problems that have recently been addressed in the literature. Let us start the discussion with the description of recent promising developments in the area of sensor network localization, which allow identifying global positions of all the nodes in a network using limited and sometimes noisy information. It turns out that semidefinite programming techniques can be efficiently used to tackle these problems. Next, the problems of single and multiple sensor scheduling for area surveillance, including the setups under uncertainty will be discussed. The issues of wireless communication network connectivity and integrity, as well as network interdiction problems will be reviewed.

3.1.1 Positioning Using Angle of Arrival

An important class of problems is a determining sensors position, known as sensors localization in the literature. In many cases, it is important to be aware of the sensor's physical coordinates. Installing GPS receivers in every sensor is not always optimal from the cost-related and other perspectives. Typically, only a few nodes (seed nodes, landmarks, etc) of the network are equipped with GPS and know their physical location. The rest of the nodes can only communicate with other nodes and determine relative location characteristics such as distance, angles, etc. Various localization techniques are used to obtain location of all the sensors in the network.

A wide range of sensors applications reveals different requirements to the network topology identification [33, 47, 57, 59]. For example, network parameters can be influenced by land surface, transmission characteristics, energy consumption policy, etc. Ad hoc and dynamic networks also require identification of node coordinates. Utilizing mathematical programming techniques often allows to find efficient solutions.

Typically, only a few nodes (seed nodes, landmarks, etc) of the network are equipped with GPS and know their physical location. The rest of the nodes can only communicate with other nodes and determine relative location characteristics such as distance, angles, etc. Various localization techniques are used to obtain location of all the sensors in the network. The following method assumes that the network consists of two types of nodes: usual and more capable nodes which know its position. Niculescu and Nath propose a method according to which nodes in an ad hoc network collaborate in finding their position and orientation, assuming that a small part of the network has a position capability. Also, every node in the network has a capability to determine the angle of the arriving signal (AOA).

Each node in the network has one fixed main axis (which may not be the same for different nodes) and is able to measure all angles against this axis.

Every node in an ad hoc network can communicate only with its immediate neighbors within the radio range, and its neighbors may not always be landmarks, i.e., the nodes that know their position. Niculescu Nath propose in [42] a method to forward orientation in a way that would allow the nodes which are not in direct contact with landmarks to determine its orientation with respect to the landmarks. Orientation means bearing, radial, or both. Bearing is an angle measurement with respect to another object. A radial is the angle under which the object is seen from another point. The authors examine two algorithms: Distance Vector Bearing (DV-Bearing), which allows each node to get a bearing to landmark, and DV-Radial. The propagation in both algorithms works the following way: nodes adjacent to landmarks determine their bearing/radial directly from landmark and send the network the information about their position.

DV-Bearing algorithm works the following way: nodes A, B, and C are neighbors and can communicate with each other. Suppose that the node A needs to find its bearing to node L, which is not within radio range of node A but within radio range of nodes B and C. Since A, B, and C can locate each other than the node A can determine all the angles in triangles ABC and BCL. However, that would allow to calculate the angle LAC and consequently the bearing of A to L, which is equal to $c + LAC$. Once node A knows three bearings to landmarks, which are not at the same line, it can calculate its own location by triangulation.

The DV-Radial algorithm works very similarly. The only difference is that node A needs to know not only bearings of nodes B and C to node L, but also the radials of B and C from L. The knowledge of radials improves accuracy of the algorithm. When all angles are measured against the same direction (for example, when compass is available) then these two methods become identical.

3.1.2 Semidefinite Programming (SDP) for Sensor Network Localization

Let us review localization problem with provided information on distances for anchor nodes and unknown sensors nodes. Suppose we consider localization problem on the plane. We have m known points $a_k \in R^2$, $k = 1, ..., m$ and n unknown nodes $x_j \in R^2$, $j = 1, ..., n$. Let us consider three sets of node pairs N_e, N_l, N_u. For pairs in N_e we know exact distances d_{kj} between a_k and x_j and $\hat{d}_{ij}$ between x_i and x_j. N_l is a set of pairs with known lower bounds $\underline{r}_{kj}$ and $\underline{r}_{ij}$. Finally, N_u is a set of upper bounds $\overline{r}_{kj}$ and $\overline{r}_{ij}$. Naturally our goal is to minimize estimation error, which immediately leads us to the following non-convex optimization problem

$$\begin{aligned}
\text{Minimize} \quad & \sum_{(i,j)\in N_e,\ i<j} |\|x_i - x_j\|^2 - \hat{d}_{ij}^2| \\
& + \sum_{(k,j)\in N_e} |\|a_k - x_j\|^2 - d_{kj}^2| \\
& + \sum_{(i,j)\in N_l,\ i<j} (\|x_i - x_j\|^2 - \underline{r}_{ij}^2)_- \\
& + \sum_{(k,j)\in N_l} (\|a_k - x_j\|^2 - \underline{r}_{kj}^2)_- \\
& + \sum_{(i,j)\in N_u,\ i<j} (\|x_i - x_j\|^2 - \overline{r}_{ij}^2)_+ \\
& + \sum_{(k,j)\in N_u} (\|a_k - x_j\|^2 - \overline{r}_{kj}^2)_+,
\end{aligned} \tag{3–1}$$

where $(u)_-$ and $(u)_+$ are defined as

$$(u)_- = \max\{0, -u\}$$

$$(u)_+ = \max\{0, u\}.$$

The norm of vector x is defined as $\|x\| = \sqrt{x^T x}$. Works [15, 53] study semidefinite relaxation of the problem.

The formulated problem can be rewritten by introducing matrix notation and slack variables as

$$\begin{aligned}
\text{Minimize} \quad & \sum_{(\mathrm{i,j})\in \mathrm{N_e},\ \mathrm{i<j}} (\alpha_{\mathrm{ij}}^{+} + \alpha_{\mathrm{ij}}^{-}) + \sum_{(\mathrm{k,j})\in \mathrm{N_e},} (\alpha_{\mathrm{kj}}^{+} + \alpha_{\mathrm{kj}}^{-}) \\
& + \sum_{(i,j)\in N_l,\ i<j} \beta_{ij}^{-} + \sum_{(k,j)\in N_l} \beta_{kj}^{-} \\
& + \sum_{(i,j)\in N_u,\ i<j} \beta_{ij}^{+} + \sum_{(k,j)\in N_u} \beta_{ij}^{+} \\
\text{s.t.} \quad & \\
& s_{ij}^{T} Y e_{ij} - d_{ij} 2 = \alpha_{ij}^{+} - \alpha_{ij}^{-},\ \forall i,j \in N_e,\ i<j, \\
& (a_k; e_j)^{T} \begin{pmatrix} I & X \\ X^{T} & Y \end{pmatrix} (a_k; e_j) - (\hat{d}_{kj})^2 = \alpha_{kj}^{+} - \alpha_{kj}^{-}, \\
& \forall k,j \in N_e, \\
& e_{ij}^{T} Y e_{ij} - (d_{ij})^2 \geq -\beta_{ij}^{-},\ \forall i,j \in N_l,\ i<j \\
& (a_k; e_j)^{T} \begin{pmatrix} I & X \\ X^{T} & Y \end{pmatrix} (a_k; e_j) - (\underline{r}_{kj})^2 \geq -\beta_{kj}^{-}, \\
& \forall k,j \in N_l, \\
& e_{ij}^{T} Y e_{ij} - (d_{ij})^2 \leq \beta_{ij}^{-},\ \forall i,j \in N_l,\ i<j \\
& (a_k; e_j)^{T} \begin{pmatrix} I & X \\ X^{T} & Y \end{pmatrix} (a_k; e_j) - (\underline{r}_{kj})^2 \leq \beta_{kj}^{-}, \\
& \forall k,j \in N_l, \\
& \alpha_{ij}^{+}, \alpha_{ij}^{-}, \alpha_{kj}^{+}, \alpha_{kj}^{-}, \beta_{ij}^{+}, \beta_{ij}^{-}, \beta_{kj}^{+}, \beta_{kj}^{-} \leq 0, \\
& Y = X^{T} X.
\end{aligned}$$

Here $X = [x_1, x_2, \ldots, x_n]$ is a $2 \times n$ matrix. In order to cope with nonconvexity equality, $Y = X^T X$ is replaced with $Y \succeq X^T X$ or equivalently

$$Z := \begin{pmatrix} I & X \\ X^{T} & I \end{pmatrix} \succeq 0.$$

The last relation leads to a standard SDP formulation:

$$\begin{aligned}\text{Minimize} \quad & \sum_{(i,j)\in N_e,\ i<j} (\alpha_{ij}^{+} + \alpha_{ij}^{-}) + \sum_{(k,j)\in N_e,} (\alpha_{kj}^{+} + \alpha_{kj}^{-}) \\ & + \sum_{(i,j)\in N_l,\ i<j} \beta_{ij}^{-} + \sum_{(k,j)\in N_l} \beta_{kj}^{-} \\ & + \sum_{(i,j)\in N_u,\ i<j} \beta_{ij}^{+} + \sum_{(k,j)\in N_u} \beta_{ij}^{+}\end{aligned}$$

s.t.

$$(1;0;\mathbf{0})^T Z(1;0;\mathbf{0}) = 1,$$

$$(0;1;\mathbf{0})^T Z(0;1;\mathbf{0}) = 1,$$

$$(1;1;\mathbf{0})^T Z(1;1;\mathbf{0}) = 2,$$

$$(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) - \alpha_{ij}^{+} + \alpha_{ij}^{-} = (\hat{d}_{ij})^2, \forall i,j \in N_e, i<j,$$

$$(a_k;e_j)^T Z(a_k;e_j) - \alpha_{kj}^{+} + \alpha_{kj}^{-} = (\hat{d}_{kj})^2, \forall k,j \in N_e,$$

$$(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) + \beta{-}_{ij} \geq (\underline{r}_{ij})^2, \forall i,j \in N_l, i<j,$$

$$(a_k;e_j)^T Z(a_k\ e_j) + \beta_{kj}^{-} \geq (\underline{r}_{kj})^2, \forall k,j \in N_l,$$

$$(\mathbf{0};e_{ij})^T Z(\mathbf{0};e_{ij}) - \beta{+}_{ij} \leq (\overline{r}_{ij})^2, \forall i,j \in N_u, i<j,$$

$$(a_k;e_j)^T Z(a_k\ e_j) - \beta_{kj}^{+} \leq (\overline{r}_{kj})^2, \forall k,j \in N_u,$$

$$\alpha_{ij}^{+}, \alpha_{ij}^{-}, \alpha_{kj}^{+}, \alpha_{kj}^{-}, \beta_{ij}^{+}, \beta_{ij}^{-}, \beta_{kj}^{+}, \beta_{kj}^{-} \leq 0,$$

$$Z \succeq 0.$$

Ye, Biswas, and So [15, 53] provide a criterion of solution existence and uniqueness, as well as statistical interpretation of the formulation in case when distance values are random values with normally distributed measurement errors. The SDP problems are solved using interior point algorithms. The numerical experiments results demonstrate the efficiency of the proposed approach.

3.1.3 Network Interdiction

An important issue in military applications is to neutralize the communication in the sensors network of the enemy. This problem is known as jamming or eavesdropping a

wireless communication network. Let us introduce optimization formulations that allow to place jamming devices delivering maximal harm to the adverse sensors network. We start by considering a deterministic case when node location is known.

The goal of jamming is to find a set of locations for placing jamming devices that suppresses the functionality of the network. Let n jamming devices be used to jam m communicating sensors. The underlying assumption is that the sensors and jammers can be located on a fixed set of locations V. The jamming effectiveness of device j is calculated as

$$d : (V \times V) \mapsto R,$$

where d is a decreasing function of the distance from the jamming device to the node being jammed.

The cumulative level of jamming energy received at node i is defined as

$$Q_i := \sum_{j=1}^{n} d_{ij},$$

where n is the number of jamming devices. As a result, jamming problem can be formulated as the minimization of the number of jamming devices placed, subject to a set of covering constraints:

$$\begin{aligned} &\text{Minimize n} \\ &\text{s.t.} Q_i \geq C_i, \quad i = 1, 2, \dots, m. \end{aligned}$$

Seeking the optimal placement coordinates $(x_j, y_j), j = 1, 2, \dots, n$ for jamming devices given the coordinates $(X_i, Y_i), i = 1, 2, \dots, m$ leads to non-convex formulations for most functions d. Thus, integer programming models for the problem are proposed.

A fixed set $\mathcal{N} = \{1, 2, \dots, n\}$ of possible locations for the jamming devices and a set of communication nodes are introduced by Commander *et al* in [22]. Define the decision

variable x_j as

$$x_j = \begin{cases} 1, \text{ if a jamming device is installed at location } j, \\ 0, \text{ otherwise.} \end{cases} \tag{3–2}$$

Then we have the optimal network covering formulation given as

$$\text{Minimize } \sum_{j=1}^{n} c_j x_j \tag{3–3}$$

$$\text{s.t.} \tag{3–4}$$

$$\sum_{j=1}^{n} d_{ij} x_j \geq C_i, \quad i = 1, 2, \ldots, m, \tag{3–5}$$

$$x_j \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \tag{3–6}$$

In this instance the objective is to minimize the cost of jamming devices used while achieving some minimum level of coverage at each node. If $c_j = 1$, the number of jammers is minimized.

If the goal is to suppress sensors communications we can minimize jamming cost with respect to the required level of connectivity index. Communication between nodes i and j is assumed to be destroyed if at least one of the nodes is jammed. Further, let $y_{ij} := 1$ if there exists a path from node i to node j in the jammed network and let $z_i = 1$

be an indicator that i-th node is jammed. This can be formulated as

$$\text{Minimize} \sum_{j=1}^{n} c_j x_j \tag{3–7}$$

s.t.

$$\sum_{\substack{j=1 \\ j \neq i}}^{m} y_{ij} \leq L, \ \forall \ i \in \mathcal{M}, \tag{3–8}$$

$$M(1 - z_i) > S_i - C_i \geq -M z_i, \ \forall \ i \in \mathcal{M}, \tag{3–9}$$

$$y_{ij} \text{ is consistent with the network and } z_i \tag{3–10}$$

$$x_j \in \{0, 1\}, \ \forall \ j \in \mathcal{N}, \tag{3–11}$$

$$z_i \in \{0, 1\}, \ \forall \ i \in \mathcal{M}, \tag{3–12}$$

$$y_{ij} \in \{0, 1\}, \ \forall \ i, j \in \mathcal{M}, \tag{3–13}$$

where $S_i := \sum_{j=1}^{n} d_{ij} x_j$ denote the cumulative level of jamming at node i, $M \in R$ is some large constant. This problem can be formulated as a mixed integer linear problem. The justification is provided in [22].

Finally, Commander *et al* provides percentile based formulation for deterministic jamming problems. Suppose it is determined that jamming some fraction $\alpha \in (0, 1)$ of the nodes is sufficient for effectively dismantling the network. This can be accomplished by including α-VaR constraints in the original model. Let $y : \mathcal{M} \mapsto \{0, 1\}$ be an indicator that equals to one if node i is jammed ($y_i = 1$).

To find the minimum number of jamming devices that will allow covering $\alpha \cdot 100\%$ of the network nodes with prescribed levels of jamming C_i, we must solve the following integer program

$$\text{Minimize} \sum_{j=1}^{n} c_j x_j \tag{3–14}$$

s.t.

$$\sum_{i=1}^{m} y_i \geq \alpha m, \tag{3–15}$$

$$\sum_{j=1}^{n} d_{ij} x_j \geq C_i y_i, \quad i = 1, 2, \ldots, m, \tag{3–16}$$

$$x_j \in \{0, 1\}, \quad j = 1, 2, \ldots, n, \tag{3–17}$$

$$y_i \in \{0, 1\}, \quad i = 1, 2, \ldots, m. \tag{3–18}$$

The α-CVaR optimization model for network covering can be formulated as a mixed integer linear program using standard linearization framework:

$$\text{Minimize} \sum_{j=1}^{n} c_j x_j \tag{3–19}$$

s.t.

$$\zeta + \frac{1}{(1-\alpha)m} \sum_{i=1}^{m} \max\left\{ C_{\min} - \sum_{j=1}^{n} x_j d_{ij} - \zeta,\ 0 \right\} \leq 0, \tag{3–20}$$

$$\zeta \in \mathbb{R}, \tag{3–21}$$

$$x_j \in \{0, 1\}. \tag{3–22}$$

The VaR and CVaR models can also be written for connectivity suppression models in the similar fashion. We refer the reader to [22] for details.

The deterministic formulations of the wireless network jamming problem are extended in [21] to tackle the stochastic jamming problem formulations using percentile type constraints. These formulations consider the case when the exact topology of the network to be jammed is not known but we know the distribution of network parameters.

Since the exact locations of the network nodes are unknown, it is assumed that a set of intelligence data has been collected and from that a set $\mathcal{S}$ of the most likely scenarios have been compiled. Scenario $s \in \mathcal{S}$ contains both the node locations

$\{(\xi_1^s, \eta_1^s), (\xi_2^s, \eta_2^s), \dots, (\xi_m^s, \eta_m^s)\}$ and the set of jamming thresholds $\{C_1^s, C_2^s, \dots, C_m^s\}$. For each scenario $s \in \mathcal{S}$, the set $\mathcal{M}_s = \{1, 2, \dots, m_s\}$ needs to be jammed. Taking into account all of the scenarios we can write a program for node covering problem:

$$\text{Minimize } \sum_{k=1}^{n} c_k x_k, \tag{3–23}$$

$$\text{s.t.}$$

$$\sum_{k=1}^{n} d_{ik}^s x_k \geq C_i^s, \ i = 1, 2, \dots, m_s, s = 1, 2, \dots, S, \tag{3–24}$$

$$x_k \in \{0, 1\}, \ k = 1, 2, \dots, n, \tag{3–25}$$

It is unlikely to find the solution that can provide effective jamming strategy for all scenarios. Therefore, the notion of percentile-based risk measures can be utilized to develop formulations of the robust jamming problems incorporating these risk constraints. Robust node covering problem with Value-at-Risk constraints can be formulated as

$$\text{Minimize } \sum_{k=1}^{n} c_k x_k, \tag{3–26}$$

$$\text{s.t.}$$

$$\sum_{k=1}^{n} d_{ik}^s x_k \geq C_i^s \rho_i^s, \ \forall \ s \in \mathcal{S}, \forall \ i \in \mathcal{M}_s, \tag{3–27}$$

$$\sum_{i=1}^{m_s} \rho_i^s \geq \alpha m_s, \ \forall \ s \in \mathcal{S}, \tag{3–28}$$

$$x_k \in \{0, 1\}, \ \forall \ k \in \mathcal{N}, \tag{3–29}$$

$$\rho_i^s \in \{0, 1\}, \ \forall \ s \in \mathcal{S}, \forall \ i \in \mathcal{M}_s, \tag{3–30}$$

The loss function can be considered as the difference between the energy required to jam network node i, namely C_i^s, and the cumulative amount of energy received at node i due to x over each scenario. With this the robust node covering problem with

CVaR constraints is formulated as follows.

$$\text{Minimize } \sum_{k=1}^{n} c_k x_k, \tag{3–31}$$

s.t.

$$\zeta^s + \frac{1}{(1-\alpha)m_s} \tag{3–32}$$

$$\sum_{i=1}^{m_s} \max\left\{ C^s_{min} - \sum_{k=1}^{n} d^s_{ik} x_k - \zeta^s, 0 \right\} \leq 0, \ \forall\, s \in \mathcal{S}, \tag{3–33}$$

$$x_k \in \{0,1\}, \ \forall\, k \in \mathcal{N}, \tag{3–34}$$

$$\zeta^s \in \mathbb{R}, \ \forall\, s \in \mathcal{S}. \tag{3–35}$$

The CVaR constraint (3–33) implies that for the $(1-\alpha)\cdot 100\%$ of the worst (least) covered nodes, the average value of $f(x)$ is less than or equal to 0.

3.2 Robust Sensors Scheduling

Surveillance is an important task that can be effectively performed by an intelligent network of sensors. For example, satellites can be equipped with cameras to monitor Earth surface for different events, such as forest fire, border crossing or enemy hostile activity. Another example is traffic monitoring at the roads and intersections. Many scientific publications have recently appeared in the literature due to increasing interest in the problem of finding optimal schedule for sensors [20, 30, 32, 44, 52, 58]. Most common technological and budget constraint is a low number of sensors to monitor all the objects of interest simultaneously. Thus, finding the schedule that reduces potential loss of limited observations is a task of high importance.

3.2.1 Optimization Models for Sensors Scheduling

This section introduces a general mathematical framework for multi-sensor scheduling problems. Before introducing my model, I provide a brief overview of the research conducted by Javuz and Jeffcoat. Initially, we utilize the concepts introduced in [63] that were developed for a deterministic case of a single-sensor scheduling problem. We then generalize and extend these formulations to the more realistic cases

of multi-sensor scheduling problems, including the setups in uncertain environments. Assume that there are m sensors and n sites that need to be observed at every discrete time moment $t = 1, \dots, T$. We assume that a sensor can observe only one site at one point of time and immediately switch to another site at the next time moment. Since m is usually significantly smaller than n, we have breaches in surveillance that can cause a loss of potentially valuable information. Our goal is to build a strategy that optimizes a potential loss associated with not observing certain sites at some time moments.

Further on we introduce the binary decision variables

$$x_{i,t} = \begin{cases} 1, & \text{if } i\text{-th site is observed at time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{3–36}$$

and integer variables $y_{i,t}$ that denote the last time site i was visited as of the end of time t, $i = 1, \dots, n,\ t = 1, \dots, T,\ m < n$.

We associate a fixed penalty a_i with each site i and a variable penalty b_i of information loss. If a sensor is away from site i at time point t, the fixed penalty a_i is incurred. Moreover, the variable penalty b_i is proportional to the time interval when the site is not observed. We assume that the variable penalty rate can be dynamic; therefore, the values of b_i may be different at each time interval. Thus the loss at time t associated with site i is

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}). \tag{3–37}$$

In the considered setup, we want to minimize the *maximum penalty* over all time points t and sites i

$$\max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\}. \tag{3–38}$$

Besides, $x_{i,t}$ and $y_{i,t}$ are related via the following set of constraints. No more than m sensors are used at each time point; therefore

$$\sum_{i=1}^{n} x_{i,t} \leq m,\ \forall t = 1, \dots, T. \tag{3–39}$$

Time $y_{i,t}$ is equal to the time when the site i was last visited by a sensor by time t. This condition is set by the following constraints:

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–40}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–41}$$

It should be noted that the above constraints automatically ensure that the feasible values of $y_{i,t}$ are integer. It is easy to verify by considering possible values of binary variables $x_{i,t}$. Therefore, in the following mathematical programming problems, we can set the variables $y_{i,t} \in R$. The inclusion of these constraints will make these variables integer in any feasible solution. This enables us to decrease the number of integer (binary) variables in the considered problems.

Consequently, using the notation $C = \max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\}$ and standard linearization techniques, we can formulate the multi-sensor scheduling optimization problem in the deterministic setup as the following mixed integer linear program:

$$\text{Minimize C} \tag{3–42}$$

$$\text{s.t.} \quad C \geq a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–43}$$

$$\sum_{i=1}^{n} x_{i,t} \leq m,\ \forall t = 1, \dots, T, \tag{3–44}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–45}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–46}$$

$$y_{i,0} = 0,\ \forall i = 1, \dots, n, \tag{3–47}$$

$$x_{i,t} \in \{0, 1\},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–48}$$

$$y_{i,t} \in \boldsymbol{R}, \forall i = 1, \dots, n,\ \forall t = 0, \dots, T. \tag{3–49}$$

Single Sensor Scheduling problem is NP-hard [62] Therefore, various greedy heuristics are used. The idea behind greedy algorithm suggested by Javuz and Jeffcoat in [62] is simple. At time $t = 1$ we find the site with the smallest penalty. Then, at next

time period we find another site with the smallest penalty. This sequence is repeated for all T time intervals. Thus, the complexity of suggested approach is $O(nT)$. Yavuz and Jeffcoat has also suggested the "look ahead" modification of greedy heuristic which takes more computational time. However, computational experiments demonstrate that the solution is improved compared to the initial simple greedy optimization.

The stochastic nature of scheduling surveillance reduces predictability of sensors behavior and, as a resul,t plays an important role for military tasks. Here Javuz and Jeffcoat assumed that sites were chosen randomly based on probability p_{ij} of transition from i-th site to j-th. Then, sensor scheduling can be considered as a Markov chain stochastic process and characterized by steady state probabilities π_i.

The goal of stochastic approach is to find such steady state probabilities that minimize maximum loss. Let r_i be the visit period of site i. Then, the penalty of information loss at site i is $a_i + (r_i - 1)b_{i,t}$ at time t. Let us consider a sufficiently small planning horizon with time-invariant site dynamics. This allows us to reduce $b_{i,t}$ to b_i and denote this approach as static. Visiting site i for every $r_i > 0$ periods is equal to spending $\pi_i = 1/r_i$ of the sensor's time at the site i. The optimal schedule is achieved when $\sum_i \pi_i = 1$; i.e., all the available time is utilized. Also, the sensor never stays at any site for two consecutive periods of time. Thus $r_i \geq 2$ (or $\pi_i \leq 0.5$) should be satisfied for each site. Then, the non-linear model for obtaining optimal stationary probabilities is formulated as

$$\text{Minimize } \max_i\{a_i + \left(\frac{1}{\pi_i} - 1\right) b_i\} \tag{3–50}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \pi_i = 1, \tag{3–51}$$

$$\pi_i \leq 0.5, \forall i = 1, \dots, n \tag{3–52}$$

$$\pi_i \in R, \forall i = 1, \dots, n \tag{3–53}$$

$$\tag{3–54}$$

A heuristic solves this nonlinear continuous problem. Utilizing constraints (3–52), Javuz and Jeffcoat define a lower bound on the objective function value with

$$C^L = \max_i \{a_i + b_i\}.$$

Then, we set

$$C = \max_i \{a_i + \left(\frac{1}{\pi} - 1\right) b_i\} = C^L$$

and calculate

$$r_i = (C - a_i)/b_i + 1$$

and

$$\pi_i = 1/r_i$$

for all i. Note that $\pi_i = 0.5$ for the sites with $a_i + b_i = C$ and $\pi_i < 0.5$ for the rest constraints. If the determined probabilities add up to one then we have found the optimal solution and can terminate. If sum of probabilities is less than one, the solution is optimal again and we can shift some π-s up to make $\sum_i \pi_i = 1$. In the case when $\sum_i \pi_i > 1$ the found C is infeasible and we can apply iterative procedures, such as bisection, to find such C that $\sum_i \pi_i = 1$.

The static approach minimizes average penalty, determined by steady probabilities, and does not address the cases of long lasting absence at a site. Taking into account the fact that some random outcomes may result in extremely long penalties, it is reasonable to increase the probabilities of visiting the sites that were visited a long time ago. On the other hand, probability of observing the recently visited sites should be decreased. Recall that $y_{i,t}$ represent the last time when the site i was visited by the time t. The probability of visiting a site must depend on the difference $t - y_{i,t}$. Thus it will increase probability of visiting overdue sites.

To create a preference for visiting site i at time t, the following adjustment factors are proposed

$$q_{i,t} = \pi_i \cdot \left(\frac{t - y_{i,t}}{r_i} \right)^k$$

It is bigger than 1 for overdue sites and less than 1 for the sites that have been visited within their expected visiting periods. The parameter k is a user-defined parameter, which determines the weight of the adjustment factor. The probabilities of visiting each site at specific time point t are based on previous history and can be computed as $p_i = q_i/Q$, where $Q = \sum_{i=1}^{n} q_i$.

Finally, a hybrid method is based on a combination of the greedy algorithm and the stochastic method, discussed above. The first step calculates the penalty of not visiting site i: $c_i = a_i + b_{i,t}(t - y_{i,t})$. The next step calculates preference values to visit each site: $q_i = \left(\frac{c_i}{c_{max}}\right)^k$, where $c_{max} = \max_i\{c_i\}$. Finally, the probabilities of visit are equal to: $p_i = q_i/Q$, where $Q = \sum_{i=1}^{n} q_i$.

3.2.2 Deterministic setup

The simplest case is to model one sensor that observes a group of sites at discrete time points. Some physical systems require virtually zero time for changing a site being observed. For example, the time of a camera refocusing installed on a satellite is negligibly small. This assumption leads to the model proposed by Yavuz and Jeffcoat in [63].

Assume that we need to observe n sites during T time periods. During every period a sensor is allowed to watch only at one of n sites. The scheduling decision can be modeled using binary variables $x_{i,t}$

$$x_{i,t} = \begin{cases} 1, & \text{if } i\text{-th site is observed at time } t, \\ 0, & \text{otherwise,} \end{cases} \tag{3–55}$$

t is a discreet variable and $t = 1, 2, ..., T$. If a site i is not observed for some period of time, it leads to the penalty that is proportional to the time of not observing

this site. This penalty can be modeled using another group of decision variables. Let $y_{i,t}$ denote the time of last visiting site i before time moment t. Let us note that variables $x_{i,t}$ completely determine values of $y_{i,t}$. Fixed penalty a_i and variable penalty b_{it} are associated with site i at time moment t. Thus, the penalty at time t associated with site i is

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}). \tag{3–56}$$

[63] suggests minimizing maximum loss over all sites and time intervals. Thus the objective function is defined as $\max_{i,t}\{a_i + b_{i,t}(t - y_{i,t})\}$. This objective function can be linearized and consequently the problem looks as following

$$\text{Minimize } C \tag{3–57}$$

$$\text{s.t.} \quad C \geq a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}),\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–58}$$

$$\sum_{i=1}^{n} x_{i,t} \leq 1,\ \forall t = 1, \dots, T, \tag{3–59}$$

$$0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–60}$$

$$t x_{i,t} \leq y_{i,t} \leq t,\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–61}$$

$$y_{i,0} = 0,\ \forall i = 1, \dots, n, \tag{3–62}$$

$$x_{i,t} \in \{0, 1\},\ \forall i = 1, \dots, n,\ \forall t = 1, \dots, T, \tag{3–63}$$

$$y_{i,t} \in \boldsymbol{R}, \forall i = 1, \dots, n,\ \forall t = 0, \dots, T. \tag{3–64}$$

Constraints (3–59) ensure that the sensor visits only one site at a time. Constraints (3–60) -(3–61) set the dependence $y_{i,t}$ on $x_{i,t}$. That is $y_{i,t}$ is set to t if and only if the sensor is observing site i at time t otherwise $y_{i,t} = y_{i,t-1}$.

For every site i and every time moment t, we can calculate the penalty associated with the last time a sensor visited this site (see formula (3–56)). Let us pick $(1 - \alpha)$ % of *worst cases* among these $n \times T$ penalty values. Then instead of minimizing the *maximum* loss, we can minimize the *average* loss taken over these $(1 - \alpha)$ % percent of worst-case penalty values. Although this formulation is deterministic, we will

demonstrate that it is equivalent to computing $(1-\alpha)$ Conditional Value-at Risk (CVaR) for a set of random outcomes having equal probabilities $p_{i,t} = \frac{1}{nT}$.

Thus, we can generalize our formulation and write the objective function for our problem as

$$\text{Minimize}_{\,x,y} \text{CVaR}_{\alpha}\left[L(x,y,i,t))\right], \tag{3–65}$$

where

$$L(x,y,i,t) = a_i(1-x_{i,t}) + b_{i,t}(t-y_{i,t}) \tag{3–66}$$

The particular extreme case when $\alpha \to 1$ corresponds to minimizing maximum penalty over all t-s and i-s. This case corresponds to the problem (3–57)-(3–64). The other extreme case $\alpha = 0$ gives average taken over all time points and sites (if we assume uniform distribution). In the latter case we care about average loss. Besides, there is a high chance of not paying enough attention to particular bad outcomes.

Using the general approach outlined in chapter 2 in formulas (2–16)-(2–17), our problem is now formulated as follows:

$$\text{Minimize}_{\,x,y,\zeta} \zeta + \frac{1}{1-\alpha}\sum_{i,t} p_{i,t}\left[a_i(1-x_{i,t}) + b_{i,t}(t-y_{i,t}) - \zeta\right]^{+} \tag{3–67}$$

$$\text{s.t. constraints } (3\text{–}59)\text{-}(3\text{–}64),$$

$$\zeta \in \boldsymbol{R}, \tag{3–68}$$

where the values of $p_{i,t}$ can all be set equal to $1/nT$ as indicated in the beginning of this section.

Furthermore, this problem formulation can be easily transformed into a linear mixed integer problem by introducing a set of artificial variables $z_{i,t}$ that will lead to the following problem with a set of $n \times T$ additional constraints.

$$\text{Minimize}_{\,x,y,\zeta}\ \zeta + \frac{1}{nT(1-\alpha)} \sum_{i,t} y_{i,t} \tag{3–69}$$

$$\text{s.t. } a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t}) - \eta \leq y_{i,t} \tag{3–70}$$

$$y_{i,t} \geq 0 \tag{3–71}$$

$$\text{constraints } (3\text{–}59)\text{-}(3\text{–}64),$$

$$\zeta \in R, \tag{3–72}$$

3.2.3 Problem Setup under Uncertainty

To extend the deterministic problem formulations to a more realistic setup, where the values of the penalty parameters are uncertain, I propose a new CVaR-based formulation of multi-sensor scheduling problems.

In this setup, assume that the fixed and variable penalty values $\mathbf{a}_i$ and $\mathbf{b}_{i,t}$ are random variables with given joint distributions. Further, we can consider a set of penalty values $(a_i^s, b_{i,t}^s)$, $s = 1, \ldots, S$ corresponding to S discrete samples (or *scenarios*) as an approximation of the joint distribution. Then for each $s = 1, \ldots, S$ the loss function can be written as:

$$L(x, y, i, t, s) = a_i^s(1 - x_{i,t}) + b_{i,t}^s(t - y_{i,t}). \tag{3–73}$$

It is appropriate to consider $(1 - \alpha)$ % of worst-case penalties over all indices i, t, s. We can then chose a measure of loss as an average over these $(1 - \alpha)$ % worst cases and minimize the average. Namely, we minimize

$$\text{CVaR}_\alpha \left[L(x, y, i, t, s) \right]. \tag{3–74}$$

Using the approach described in the previous section, we obtain the following robust optimization problem that explicitly takes into account the uncertain penalty parameters:

$$\text{Minimize}_{x,y,\zeta}\ \zeta + \frac{1}{nTS(1-\alpha)} \sum_{i,t,s} \left[\left(a_i^s(1-x_{i,t}) + b_{i,t}^s(t-y_{i,t})\right) - \zeta\right]^+ \tag{3–75}$$

s.t.

constraints (3–59)-(3–64), (3–68).

As mentioned above, this formulation can be linearized by introducing extra variables and constraints, and the linear mixed integer formulation is provided below.

$$\text{Minimize}_{x,y,\zeta}\ \zeta + \frac{1}{nTS(1-\alpha)} \sum_{i,t,s} y_{i,t,s} \tag{3–76}$$

s.t.

$$a_i^s(1-x_{i,t}) + b_{i,t}^s(t-y_{i,t}) - \zeta \leq y_{i,t,s} \tag{3–77}$$

$$y_{i,t,s} \geq 0 \tag{3–78}$$

constraints (3–59)-(3–64), (3–68).

3.3 Equivalent Formulations in Cardinality Constraints

In this section, we show that the developed linear mixed integer programming problems can be equivalently reformulated as problems with cardinality constraints. As it will be discussed later, solving these equivalent reformulations can provide better computational speed and performance in finding near-optimal solutions of the considered problems. It should be noted that due to the high dimensionality and complexity of these problems, it is often impossible to find exact optimal solutions in a reasonable time; however, it is often useful in practice to utilize heuristic techniques that can find near-optimal solutions fast.

There exist heuristics [38] as well as software packages [1] which can solve optimization problems formulated in terms of cardinality constraints. Cardinality function

simply equals to the number of non-zero components of its vector argument. More formally, for $x = (x_1, \ldots, x_n)^T \in R^n$

$$\text{card}(x) = \sum_{i=1}^{n} I(x_i),$$

where I is an indicator function defined as:

$$I(z) = \begin{cases} 1, & z \neq 0; \\ 0, & \text{otherwise.} \end{cases} \tag{3–79}$$

This section presents a problem formulated in terms of cardinality function. This new problem is equivalent to the initial formulation (3–57)-(3–64) that can be written as

Problem (I):

$$\begin{aligned}
& \text{Minimize } \max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_{i,t} \leq m, \ \forall t = 1, \ldots, T, && \text{(3–80)} \\
& 0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, && \text{(3–81)} \\
& t x_{i,t} \leq y_{i,t} \leq t, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, && \text{(3–82)} \\
& y_{i,0} = 0, \ \forall i = 1, \ldots, n, && \text{(3–83)} \\
& x_{i,t} \in \{0, 1\}, \ \forall i = 1, \ldots, n, \ \forall t = 1, \ldots, T, && \text{(3–84)} \\
& y_{i,t} \in R, \forall i = 1, \ldots, n, \ \forall t = 0, \ldots, T. && \text{(3–85)}
\end{aligned}$$

The new problem can be formulated as:

Problem (II):

$$\begin{aligned}
&\text{Minimize } \max_{i,t}\{a_i(1-x_{i,t})+b_{i,t}(t-y_{i,t})\} \\
\text{s.t.}\quad & \text{card}(\tilde{x}_t)\le m,\ \forall t=1,\dots,T, && \text{(3–86)}\\
& 0\le y_{i,t}-y_{i,t-1}\le t x_{i,t},\ \forall i=1,\dots,n,\ \forall t=1,\dots,T, && \text{(3–87)}\\
& y_{i,t}\le t,\ \forall i=1,\dots,n,\ \forall t=1,\dots,T, && \text{(3–88)}\\
& y_{i,0}=0,\ \forall i=1,\dots,n, && \text{(3–89)}\\
& 0\le x_{i,t}\le 1,\ \forall i=1,\dots,n,\ \forall t=1,\dots,T, && \text{(3–90)}\\
& y_{i,t}\in \boldsymbol{R}, \forall i=1,\dots,n,\ \forall t=0,\dots,T. && \text{(3–91)}
\end{aligned}$$

where $\tilde{x}_t=(x_{1,t},\dots,x_{N,t})^T$.

The following theorem provides the relation between the two problems

Theorem 3.1. *The set of optimal solutions of problem (I) belongs to the set of optimal solutions of problem (II). Moreover, if a point* (x^{II},y^{II}) *is an optimal solution for (II), the optimal solution of (I)* (x^{I},y^{I}) *can be constructed as*

$$\begin{aligned}
x^I_{i,t}&=\lceil x^{II}_{i,t}\rceil,\\
y^I_{i,t}&=\max_{\tau\le t}\{\tau x^I_{i,\tau}\}.
\end{aligned}$$

In order to prove the theorem we will use an auxiliary formulation.

Problem (III):

$$
\begin{aligned}
& \text{Minimize } \max_{i,t}\{a_i(1-x_{i,t})+b_{i,t}(t-y_{i,t})\} \\
\text{s.t.} \quad & \text{card}(\tilde{x}_t) \leq m,\ \forall t = 1,\dots,T, \\
& 0 \leq y_{i,t} - y_{i,t-1} \leq t x_{i,t},\ \forall i = 1,\dots,n,\ \forall t = 1,\dots,T, \\
& y_{i,t} \leq t,\ \forall i = 1,\dots,n,\ \forall t = 1,\dots,T, \\
& y_{i,0} = 0,\ \forall i = 1,\dots,n, \\
& 0 \leq x_{i,t} \leq 1,\ \forall i = 1,\dots,n,\ \forall t = 1,\dots,T, \\
& y_{i,t} \in \boldsymbol{R}, \forall i = 1,\dots,n,\ \forall t = 0,\dots,T.
\end{aligned}
$$

Denote by $z^{(I)}$, $z^{(II)}$ and $z^{(III)}$ the optimal objective values of problems (I)-(III) consequently.

Lemma 1. *For every optimal solution of (III) there exists a solution that will be both feasible and optimal in (I) and (III) (i.e. formulations (I) and (III) are equivalent in this sense).*

Proof. Equations (3–81)-(3–82) enforce that

$$y_{i,t} = \max_{\tau \leq t}\{\tau x_{i,\tau}\}.$$

If there exists an optimal solution for (III) such that $x_{i,t} = 1$ but $y_{i,t} < t$ for some i, t (i.e. it is not feasible in (I)), you can build a new solution that has the same values of $x^*_{i,t} = x_{i,t}\ \forall i, t$ and $y^*_{i,t} = \max_{\tau \leq t}\{\tau x_{i,\tau}\}$. This solution will be feasible for both formulations (I) and (III).

Moreover, $\forall i, t\ y_{i,t} <= y^*_{i,t}$, i.e. the objective value will not increase and, therefore, this solution will be also optimal for (III). Obviously, this solution will be feasible and optimal for formulation (I) (since $z^{(III)} \leq z^{(I)}$).

Thus, for every optimal solution of (III) there exists a solution that will be both feasible and optimal in (I) and (III) that means that these two formulations are equivalent.

□

Proof of Theorem 3.1. Let us consider some optimal solution of (II) $x^0_{i,t},\ y^0_{i,t}$ and build a new solution $x^*_{i,t} = I\{x^0_{i,t} > 0\};\ y^*_{i,t} = y^0_{i,t}$. This solution will still be feasible and optimal for (II) since it will not increase the objective value.

Obviously, this solution will be feasible and optimal for (III) ($z^{(II)} \leq z^{(III)}$).

According to Lemma 1 for every optimal solution of (III) there exists a solution $x^{**}_{i,t} = x^*_{i,t}, y^{**}_{i,t} = \max_{\tau \leq t}\{\tau x^*_{i,\tau}\}$ that will be both feasible and optimal for (I) and (III).

This solution will be integer and feasible for (II). Since $\forall\ i, t\ y^*_{i,t} \leq y^{**}_{i,t}$, the objective value will not increase and, therefore, this solution will also be optimal for (II).

Thus, there always exists the optimal integer solution for (II) that will be also optimal for (I)

In order to prevent the solution of (II) from being non-integral, we add a penalty to the objective of (II):

Problem (IV):

$$\begin{aligned} & \text{Minimize } \max_{i,t}\{a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})\} + \lambda \cdot (m \cdot T - \sum_{i,t} x_{i,t}) \\ \text{s.t.} \quad & \text{constraints } (3\text{–}86)\text{-}(3\text{–}91), \end{aligned}$$

where $\lambda > 0$.

Corollary 1. *Problems (I) and (IV) have the same set of optimal values of* $\{x_{i,t}\}$.

Similar theorems can be proven for the other formulations, namely percentile deterministic and stochastic setups. For deterministic case problem (Ia) which is equivalent to (3–67) is related to reformulated in terms of cardinality problem (IIa):

Problem (Ia):

$$\begin{aligned} & \text{Minimize } \mathrm{CVaR}_{\alpha}\{a_i(1-x_{i,t})+b_{i,t}(t-y_{i,t})\} \\ \text{s.t.} \quad & \text{constraints } (3\text{–}80)\text{-}(3\text{–}85). \end{aligned}$$

Problem (IIa):

$$\begin{aligned} & \text{Minimize } \mathrm{CVaR}_{\alpha}\{a_i(1-x_{i,t})+b_{i,t}(t-y_{i,t})\} \\ \text{s.t.} \quad & \text{constraints } (3\text{–}86)\text{-}(3\text{–}91). \end{aligned}$$

Theorem 3.2. *The set of optimal solutions of problem (Ia) belongs to the set of optimal solutions of problem (IIa). Moreover if a point* (x^{II}, y^{II}) *is an optimal solution for (IIa), the optimal solution of (Ia)* (x^{I}, y^{I}) *can be constructed as*

$$\begin{aligned} x^{I}_{i,t} &= \lceil x^{II}_{i,t} \rceil, \\ y^{I}_{i,t} &= \max_{\tau \le t}\{\tau x^{I}_{i,\tau}\}. \end{aligned}$$

For the stochastic case problem (Ib), which is equivalent to (3–75) is related to the reformulated in terms of cardinality problem (IIb):

Problem (Ib):

$$\begin{aligned} & \text{Minimize } \mathrm{CVaR}_{\alpha}\{a^{s}_{i}(1-x_{i,t})+b^{s}_{i,t}(t-y_{i,t})\} \\ \text{s.t.} \quad & \text{constraints } (3\text{–}80)\text{-}(3\text{–}85). \end{aligned}$$

Problem (IIb):

$$\begin{aligned} & \text{Minimize } \mathrm{CVaR}_{\alpha}\{a^{s}_{i}(1-x_{i,t})+b^{s}_{i,t}(t-y_{i,t})\} \\ \text{s.t.} \quad & \text{constraints } (3\text{–}86)\text{-}(3\text{–}91). \end{aligned}$$

Theorem 3.3. *The set of optimal solutions of problem (Ib) belongs to the set of optimal solutions of problem (IIb). Moreover if a point* (x^{II}, y^{II}) *is an optimal solution for (IIb), the*

optimal solution of (lb) (x^l, y^l) *can be constructed as*

$$x^l_{i,t} = \lceil x^{ll}_{i,t} \rceil,$$

$$y^l_{i,t} = \max_{\tau \le t} \{\tau x^l_{i,\tau}\}.$$

3.4 Sensor Scheduling in Network-Based Settings

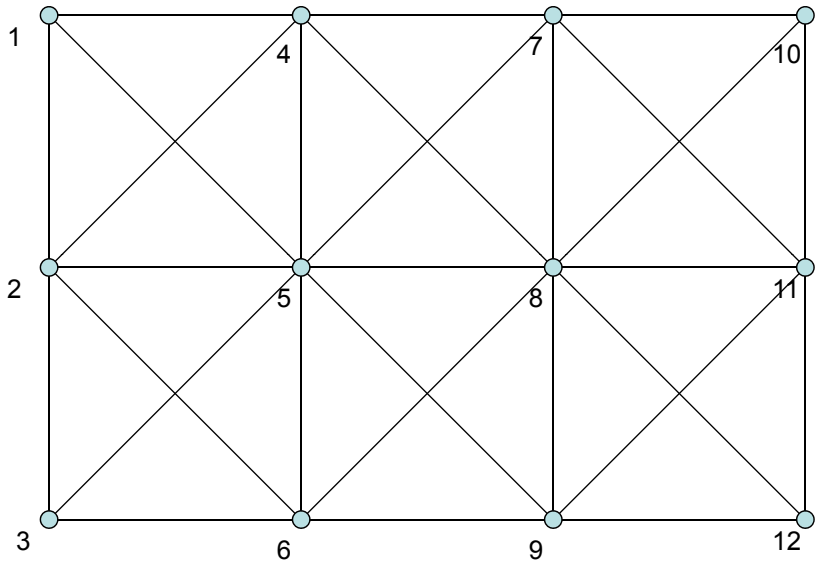


Figure 3-1. Example of a possible network. Two nodes are connected by an arc if a sensor can move from one node to another in consequent time periods.

First, let us discuss a special case of one sensor ($m = 1$) to give an idea of this modeling approach. In the case when surveillance requires sensors to physically move from one site to another, their transition abilities are limited with distance or other constraint (for example, a mountain can be a natural obstacle for UAV to move between sites). In this case, each site can be modeled as a node of a network $G = (V, E)$. Whenever there is no arc between two nodes i and j, we add the inequality

$$x_{i,t} + x_{j,t+1} \le 1, \tag{3–92}$$

that prohibits the infeasible move $i \rightarrow j$ in consequent time periods t and $t+1$. Formulations (3–57)-(3–64), (3–67) and (3–75) can be slightly modified to obtain corresponding formulations for one sensor ($m = 1$). If we demand the sensor to start and come back to a depot located at the certain site, we can optionally set the initial ($i_0 \in \{1, ..., n\}$) and final ($i_T \in \{1, ..., n\}$) locations of the sensor.

Thus, for the special case when $m = 1$ problem (3–57)-(3–64) can be formulated on the network:

$$\text{Minimize } \max_{i,t}\{a_i(1 - z_{i,t}) + b_{i,t}(t - y_{i,t})\} \tag{3–93}$$

$$\text{s.t. constraints } (3–59)\text{-}(3–64),\ (m = 1),$$

$$x_{i,t} + x_{j,t+1} \leq 1 \text{ whenever } (i,j) \notin E, \forall t = 1, \dots T,\ i,j = 1, \dots, n \tag{3–94}$$

$$x_{i_0,1} = 1,\ \text{where } i_0 \in \{1, \dots, n\} \text{ is the initial location of the sensor} \tag{3–95}$$

$$x_{i_T,1} = 1,\ \text{where } i_T \in \{1, \dots, n\} \text{ is the final location of the sensor} \tag{3–96}$$

In this formulation constraints (3–94) prohibit infeasible moves between not connected nodes. (3–95) and (3–96) set initial and final destination for the sensor. The other formulations, namely deterministic (3–67) and stochastic (3–75), can be easily adapted for network case ($m = 1$) in the same way by adding network constraints (3–94)-(3–96) to the existing sets of constraints.

This approach, however, may not be easily extended for the cases of two or more sensors. If we simply add (3–94)-(3–96) to existing non-network formulations, we can arrive at the situation which prevents feasible moves when two or more sensors are involved. Figure 3-2 provides a counterexample. Let two sensors at time moment $t = 1$ be located at nodes 1 and 3. Although they could move to nodes 2 and 4 respectively at the next time point $t = 2$, the constraint $x_{1,1} + x_{2,4} \leq 1$ would prohibit this move. To avoid

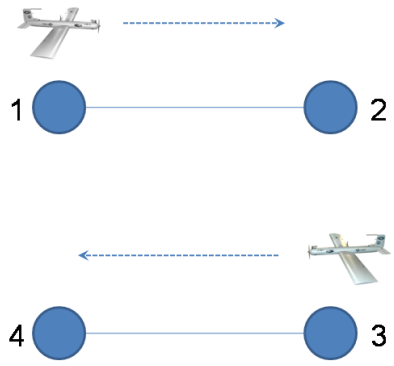


Figure 3-2. Counterexample ($m = 2$): two sensors cannot perform simultaneous feasible move due to the constraint $x_{1,1} + x_{4,2} \leq 1$

such a situation we need to add one more index for decision variable x:

$$x_{i,t,k} = \begin{cases} 1, & \text{sensor } k \text{ is surveilling site } i \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases} \tag{3–97}$$

We can ensure that every sensor is assigned to a site at every time period T with the constraint:

$$\sum_{i=1}^{n} x_{i,t,k} = 1, \ \forall k = 1, \dots, m, \ \forall t = 1, \dots, T. \tag{3–98}$$

Let us introduce $z_{i,t}$ indicating whether site i is observed at time t, namely

$$z_{i,t} = \begin{cases} 1, & \text{if any sensor is surveilling site } i \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases} \tag{3–99}$$

Variables $z_{i,t}$ and $x_{i,t,k}$ can be related with the constraint

$$z_{i,t} \leq \sum_{i=1}^{n} x_{i,t,k} \leq m \cdot z_{i,t}, \tag{3–100}$$

which states that site i is being observed at time t ($z_{i,t} = 1$) if and only if at least one sensor is present at site i ($\sum_{i=1}^{n} x_{i,t,k} > 0$).

The loss function is written as

$$L(x, y, i, t) = a_i^s(1 - z_{i,t}) + b_{i,t}^s(t - y_{i,t}). \tag{3–101}$$

If we want to minimize the maximum loss using the formulated above constraints and similarly to the non-network setup, we can reformulate the deterministic maximum loss minimization problem (3–57)-(3–64) on network as follows:

$$\text{Minimize } \max_{i,t}\{a_i(1 - z_{i,t}) + b_{i,t}(t - y_{i,t})\} \tag{3–102}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{i,t,k} = 1, \ \forall k = 1, \dots, m, \ \forall t = 1, \dots, T, \tag{3–103}$$

$$z_{i,t} \le \sum_{k=1}^{m} x_{i,t,k} \le m \cdot z_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{3–104}$$

$$0 \le y_{i,t} - y_{i,t-1} \le t z_{i,t}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{3–105}$$

$$t z_{i,t} \le y_{i,t} \le t, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \tag{3–106}$$

$$y_{i,0} = 0, \ \forall i = 1, \dots, n, \tag{3–107}$$

$$x_{i,t,k} + x_{j,t+1,k} \le 1,$$
$$\text{whenever } (i, j) \notin E, \forall t = 1, \dots T, \ i, j = 1, \dots, n, \ k = 1, \dots, m \tag{3–108}$$

$$x_{i_{0,k},1,k} = 1,$$
$$\text{where } i_{0,k} \in \{1, \dots, n\} \text{ is the initial location of sensor } k \tag{3–109}$$

$$x_{i_{T,k},T,k} = 1,$$
$$\text{where } i_{T,k} \in \{1, \dots, n\} \text{ is the final location of sensor } k, \tag{3–110}$$

$$x_{i,t,k} \in \{0, 1\}, \ \forall i = 1, \dots, n, \ \forall t = 1, \dots, T, \ \forall k = 1, \dots, m, \tag{3–111}$$

$$y_{i,t} \in \boldsymbol{R}, \forall i = 1, \dots, n, \ \forall t = 0, \dots, T, \tag{3–112}$$

$$z_{i,t} \in \{0, 1\}, \forall i = 1, \dots, n, \ \forall t = 1, \dots, T. \tag{3–113}$$

Formulations for CVaR stochastic and deterministic cases as well as the linearized formulation can be obtained the same way as in non network case.

3.5 Computational Experiments

Table 3-1. Performance results for deterministic model (3–57)-(3–64). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	320	330	330	332	332
	psg value	375	376	376	370	375
	%	14.7%	12.2%	12.2%	10.3%	11.5%
	time: cplex/psg	31.2/2.2	79.1/2.4	134.9/2.5	167.7/2.7	198.9/2.7
m=2	cplex value	240	245	250	260	265
	psg value	305	310	304	310	310
	%	21.3%	21%	17.8%	16.1%	14.5%
	time: cplex/psg	38.4/2.2	142.7/2.3	928/2.5	2042.7/2.6	5898.9/2.8
m=3	cplex value	206	210	215	217	224
	psg value	233	250	265	256	275
	%	11.6%	16%	18.9%	15.2%	18.5%
	time: cplex/psg	30.7/2.3	39.3/2.4	81.1/2.5	1003.6/2.7	9317.9/2.9
m=4	cplex value	190	194	196	200	200
	psg value	215	217	242	237	242
	%	11.6%	10.6%	19%	15.6%	17.4%
	time: cplex/psg	6/2.3	76.5/2.5	64.6/2.6	231.4/2.7	589.3/2.8
m=5	cplex value	183	185	188	190	190
	psg value	196	202	215	217	220
	%	6.6%	8.4%	12.6%	12.4%	13.6%
	time: cplex/psg	1.4/2.3	2.2/2.5	38/2.6	51.9/2.7	68/3
m=6	cplex value	165	170	170	183	185
	psg value	185	185	197	195	200
	%	10.8%	8.1%	13.7%	6.2%	7.5%
	time: cplex/psg	1.1/2.4	1.5/2.5	2.9/2.7	29.1/2.8	123/3
m=7	cplex value	155	160	163	168	170
	psg value	160	171	185	190	188
	%	3.1%	6.4%	11.9%	11.6%	9.6%
	time: cplex/psg	0.3/2.3	0.9/2.5	1.8/2.8	113.2/2.9	25.2/3

The computational experiments were performed on the test problems using two commercial optimization software solvers: ILOG CPLEX [2] and AOrDA PSG [1]. The performance of the solvers is compared in tables 3-1-3-3 (each table corresponds to one of the three problem formulations). It can be observed that CPLEX finds exact solutions, however, it takes too much time for large instances, especially for the problems under uncertainty. PSG allows sacrificing quality for time, i.e. the obtained solutions for

Table 3-2. Performance results for CVaR type deterministic model (3–67). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$. CVaR confidence level $\alpha = 0.9$.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	307.1	318.4	317	318.7	318
	psg value	360.1	357.9	356.8	357.6	353.7
	%	14.7%	11%	11.2%	10.9%	10.1%
	time: cplex/psg	45.5/2.8	99.3/3	74/2.9	62.8/3.1	130.9/3
m=2	cplex value	231.9	241.1	245.2	-	-
	psg value	297	299	297.5	293.5	291.6
	%	21.9%	19.4%	17.6%	-	-
	time: cplex/psg	1072.8/3	10924.6/3.1	16212.1/3.1	-/3.2	-/3.2
m=3	cplex value	198.6	205.3	-	-	-
	psg value	223.6	237.3	245	255.4	260.9
	%	11.2%	13.5%	-	-	-
	time: cplex/psg	1910.7/2.8	45540.4/2.8	-/2.9	-/3	-/3.4
m=4	cplex value	187.9	190.4	-	-	-
	psg value	211.4	207.2	230	218.2	221.8
	%	11.1%	8.1%	-	-	-
	time: cplex/psg	7989.2/3.2	23852.7/3	-/3.3	-/3.4	-/3.2
m=5	cplex value	173.3	179.3	-	-	-
	psg value	193.4	192.6	197.8	207.7	210
	%	10.4%	6.9%	-	-	-
	time: cplex/psg	7672.6/2.9	25593/2.8	-/3.3	-/3.3	-/3.7
m=6	cplex value	161.1	165.3	-	-	-
	psg value	179.5	182.8	185.2	195.2	190.9
	%	10.2%	9.5%	-	-	-
	time: cplex/psg	2250.3/2.8	36618.1/3.3	-/3.4	-/2.9	-/3.4
m=7	cplex value	142.4	-	-	-	-
	psg value	150.9	166.6	173.4	179	183.8
	%	5.6%	-	-	-	-
	time: cplex/psg	6640.9/2.9	65122.8/3.3	-/3	-/3	-/3.7

cardinality formulations are not globally optimal, but the computational time is negligibly small. The numerical experiments show that local solutions differ from global ones in 10-20 % for most cases.

Table 3-4 compares performance of CPLEX and PSG in finding approximate solutions for stochastic case ($n = 12$ sites and $T = 10$ time periods). We stopped CPLEX when it found the objective as small as the PSG objective value (and recorded the computation time). It appears that PSG outperforms CPLEX for problems with a

Table 3-3. Performance results for CVaR type stochastic model (3–75). n - number of sites; m - number of sensors. The number of discrete time steps is fixed: $T = 10$, number of scenarios $S = 100$. CVaR confidence level $\alpha = 0.9$.

		n=8	n=9	n=10	n=11	n=12
m=1	cplex value	-	-	-	-	-
	psg value	394	398.7	390.1	395.6	394.9
	%	-	-	-	-	-
	time: cplex/psg	-/25.9	-/28	-/35.8	-/40.2	-/47.7
m=2	cplex value	-	-	-	-	-
	psg value	297.9	304	321.7	324.4	318.6
	%	-	-	-	-	-
	time: cplex/psg	-/32.7	-/38.6	-/51.4	-/67.8	-/76
m=3	cplex value	-	-	-	-	-
	psg value	260.4	260.1	262.4	264.4	269
	%	-	-	-	-	-
	time: cplex/psg	-/38.4	-/45.4	-/56.3	-/71.4	-/84.6
m=4	cplex value	-	-	-	-	-
	psg value	229.4	231.6	242.9	236.3	250.8
	%	-	-	-	-	-
	time: cplex/psg	-/47.2	-/60.2	-/72.5	-/84.6	-/98.4
m=5	cplex value	-	-	-	-	-
	psg value	209.1	212.4	220.8	224.6	230.1
	%	-	-	-	-	-
	time: cplex/psg	-/56.1	-/72.3	-/83.7	-/96.6	-/119.5
m=6	cplex value	-	-	-	-	-
	psg value	193.2	199.9	209.2	210.3	218.7
	%	-	-	-	-	-
	time: cplex/psg	-/51.8	-/67.3	-/86.1	-/112.9	-/121.4
m=7	cplex value	-	-	-	-	-
	psg value	164.4	186.3	187.6	198.6	204.4
	%	-	-	-	-	-
	time: cplex/psg	-/45	-/66.5	-/89.8	-/111.3	-/133.5

large number of stochastic scenarios while they have similar performance for small size problems. Therefore, based on the size of the problem and user requirements, one can determine the appropriate equivalent problem formulation and the optimization solver that can be used to find an optimal or a near-optimal solution.

We performed experiments on network formulation using the network provided on figure 3-1. Table 3-5 provides CPU times in seconds for obtaining the exact solution

Table 3-4. Comparing PSG and CPLEX performance for obtaining approximate solution of CVaR type stochastic problem (3–75)($n = 12$ sites and $T = 10$ time periods).

	PSG Value	PSG Time (sec)	CPLEX Time (sec)
m=1	389.2	103	32
m=2	318.532	110	185
m=3	276.332	133	230
m=4	246.648	136	247
m=5	229.234	208	210
m=6	218.166	158	330
m=7	204.1	201	260
m=8	193.199	191	220
m=9	180.976	155	250
m=10	164.746	191	280
m=11	146.607	180	200

Table 3-5. ILOG CPLEX CPU time (sec) for network deterministic model (3–102)-(3–113). n - number of sites; m - number of sensors. The number of discrete time steps is T = 10.

	n=6	n=7	n=8	n=9	n=10	n=11	n=12
m=1	0.31	0.77	0.76	2.16	0.90	0.50	1.21
m=2	3.96	3.96	8.00	15.85	238.42	174.93	315.76
m=3	0.93	21.48	22.20	9.17	340.86	350.85	2037.73
m=4	0.20	0.31	0.74	34.26	14.45	926.64	436.80
m=5	0.32	1.79	2.62	31.40	91.38	62.20	3359.69
m=6		0.58	1.79	4.36	69.76	19.05	238.59
m=7			2.12	4.08	22.87	19.57	47.20
m=8				0.79	6.29	7.14	38.69
m=9					1.17	3.19	6.81
m=10						0.79	2.15
m=11							1.94

for the deterministic network case (3–102)-(3–113) in ILOG CPLEX. Computation was performed for a number of sites from 6 to 12 and 10 discrete time steps.

CHAPTER 4
TWO STAGE STOCHASTIC OPTIMIZATION MODEL FOR ROBUST NETWORK FLOW DESIGN

Optimization on networks is probably the oldest type of problems studied in the field of operation research. Hundreds of formulations and applications have been developed over the decades. The interest in optimization can be explained by its numerous applications in transportation, energy, computing, and many other engineering disciplines [4, 10, 24, 25, 51]. In order to model real world problems more efficiently, many of these applications need to address uncertainties, adverse actions of enemies against the network as well as their combination. The latter significantly complicates the network problems due to the increase in the computational complexity of the resulting optimization problems. Some of the problems discussed in the literature include the cases when the length of arc is a random variable [23], arc capacities are random [28], an arc is subject to failure [26], robust flow under assumption that the worst outcome happens [11].

This chapter solves a network flow problem where the conditional expectation of worst collateral flow loss is constrained. I introduce a loss function that characterizes the total loss of flow that happens in the network as a result of arc failure. Then, a combination of Benders' decomposition and Lagrangian relaxation is used to solve the two stage stochastic formulation.

4.1 Problem Formulation

Let a network be represented by a directed graph $G = (V, E)$ and each arc $(i, j) \in E$ have an associated cost $c_{i,j}$ per unit of flow passed alone the arc from node i to j. Let maximum capacity value $u_{i,j}$ denote the maximum amount of flow that can be pushed alone the arc (i, j). Let d_i denote the demand or supply for each node $i \in V$. We assume that positive values of d_i represent the excess of flow or supply and negative values represent demands. The well-known deterministic minimum cost network flow

problem can be formulated as the following linear problem:

$$\text{Minimize} \sum_{(i,j)\in E} c_{i,j}x_{i,j}, \tag{4–1}$$

$$\text{s.t.}$$

$$\sum_{\{j:(i,j)\in E\}} x_{i,j} - \sum_{\{j:(j,i)\in E\}} x_{j,i} = d_i, \forall i \in V, \tag{4–2}$$

$$0 \leq x_{i,j} \leq u_{i,j}, \forall i,j \in V. \tag{4–3}$$

Here, we minimize the total cost of transferring the flow $x_{i,j}$ in the network G provided that all demands are satisfied and no spillage occurs (constraints (4–2)).

Boginski *et al* suggested in [16] a robust stochastic formulation of this problem in the case when every arc (i,j) of the network arcs can be independently destroyed with known probabilities $p_{i,j}$ for each arc (i,j).

Let a vector of Bernoulli random variables y represent the uncertain arc failures in the network:

$$y_{i,j} = \begin{cases} 1, & \text{with probability } p_{i,j}; \\ 0, & \text{with probability } 1 - p_{i,j}. \end{cases}$$

In order to approximate the stochastic outcome, it was suggested to generate a finite set of scenarios:

$$y^s_{i,j} = \begin{cases} 1, & \text{if arc } (i,j) \text{ fails under scenario } s, \\ 0, & \text{otherwise .} \end{cases} \tag{4–4}$$

The function representing the loss of taking decision x under realization of random variable y is called loss function and is defined in [16] as

$$L(x,y) = \sum_{(i,j)\in E} x_{i,j}y_{i,j}. \tag{4–5}$$

Here, the loss is determined by the flow schedule $x_{i,j}$ and stochastic outcome $y_{i,j}$. The loss is equal to the amount of flow being passed through the disrupted arcs.

The CVaR minimization problem can be written as a linear program using standard techniques described in chapter 2:

$$\text{Minimize} \sum_{(i,j)\in E} c_{i,j}x_{i,j}, \tag{4–6}$$

$$\text{s.t.}$$

$$\sum_{\{j:(i,j)\in E\}} x_{i,j} - \sum_{\{j:(j,i)\in E\}} x_{j,i} = d_i, \forall i \in V, \tag{4–7}$$

$$0 \leq x_{i,j} \leq u_{i,j}, \forall i,j \in V. \tag{4–8}$$

$$\zeta + \frac{1}{S(1-\alpha)} \sum_{s=1}^{S} t_s \leq C \tag{4–9}$$

$$t_s \geq \sum_{(i,j)\in E} x_{i,j}y_{i,j}^s - \zeta, \ \forall s = 1,2,...,S, \tag{4–10}$$

$$\zeta \in R. \tag{4–11}$$

The loss function used in this formulation has certain disadvantages. Particularly, for the cases represented in figure 4-1 the losses $L(\cdot) = 4$, if the arc (1,2) fails and $L(\cdot) = 7$, if the arcs (1,2),(2,3),(2,4), and (2,5) fail. It can be seen, however, that both cases are equivalent in a sense that there is no flow in the network. As a result, the loss is actually the same.

Another essential factor should be considered. It is not especially critical which arcs failed. What matters, however, is whether we satisfy the demand of consumers. Moreover, not all consumers are equal in the real networks. For example, if we deliver electricity in a power network grid, it is crucial to ensure uninterruptable supply of hospitals and other strategic objects, while street lighting has a much lower degree of importance.

Taking into account the above arguments, we suggest that loss should be determined by the amount of undelivered flow caused by arc failures. Let us introduce

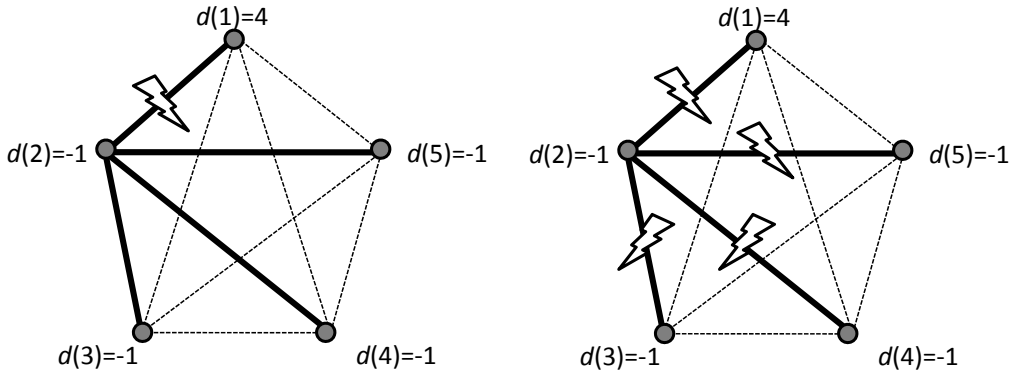


Figure 4-1. Effect caused by network failure. In both cases the network lost the same amount of flow.

variables $x^s_{i,j}$ representing the actual possible flow after some arcs failed in the scenario s and r^s_i is an absolute value of a remaining supply/demand for the node i. Naturally, for transition nodes i the remaining balance is always zero.

Clearly, $x^s_{i,j}$ is determined by the decision vector x and the outcome vector y. The dependence is set by the following set of constraints:

$$0 \leq x^s_{i,j} \leq x_{i,j}(1 - y^s_{i,j}),\ \forall i, j, s; \tag{4–12}$$

$$\sum_{\{j:(i,j)\in E\}} x^s_{i,j} - \sum_{\{j:(j,i)\in E\}} x^s_{j,i} = \mathrm{sign}(\mathrm{d_i}) \cdot \mathrm{r_i^s} \forall \mathrm{s, i}; \tag{4–13}$$

$$0 \leq r^s_i \leq |d^i_s| \tag{4–14}$$

Constraints (4–12) ensure that the remaining flow $x^s_{i,j}$ in the scenario s does not exceed the initially scheduled flow $x_{i,j}$. (4–13) ensures that inflow and outflow are balanced in every node i. (4–14) guarantees that actual inflow and outflow in the network do not exceed the demand and supply. The difference

$$-d_i - r^s_i$$

is computed for each of the consuming nodes, i.e. $d_i < 0$ and characterized by the deficit of flow intended for the recipient at node i. Naturally, our goal is to find such a schedule

that minimizes the loss of a partial or complete failure to satisfy the demand. Let us associate some penalty factor l_i with each consuming node i ($d_i < 0$). Then, the loss function is determined as a solution of the following linear optimization problem:

$$L(x, y) = \min_{\substack{x^s, r^s \\ \text{s.t. } (4\text{–}12)\text{–}(4\text{–}14)}} \sum_{\{i: d_i < 0\}} l_i \cdot (-d_i - r_i^s),$$

Depending on the context, the penalty factor $l_i \geq 0$ can be a monetary forfeit for inability to meet contract obligation or some empirical importance coefficient of the destination node. Clearly, the function $L(x, y)$ is a piecewise linear function with respect to either decision x or stochastic outcome y since all the relations are set by minimizing linear objective function subject to linear constraints which linearly depend on the parameters x and y.

Using the introduced definitions of loss, the robust network flow problem can be formulated as

$$\text{Minimize} \sum_{(i,j) \in E} c_{i,j} x_{i,j}, \tag{4–15}$$

$$\text{s.t.}$$

$$\text{constratints } (4\text{–}2)\text{-}(4\text{–}3),\ (4\text{–}12)\text{-}(4\text{–}14)\ ;$$

$$\text{CVaR}_\alpha L(x, y) \leq C. \tag{4–16}$$

Here, C in the constraint (4–16) is the predefined maximal level of α-CVaR of the loss. The above formulation can be transformed into the following linear problem using the standard technique for linearizing CVaR function:

$$\text{Minimize}_{\,x,x^s,r^s} \sum_{(i,j)\in E} c_{i,j}x_{i,j}, \tag{4–17}$$

s.t.

$$\sum_{\{j:(i,j)\in E\}} x_{i,j} - \sum_{\{j:(j,i)\in E\}} x_{j,i} = d_i, \forall i \in V, \tag{4–18}$$

$$0 \le x_{i,j} \le u_{i,j}, \forall i,j \in V. \tag{4–19}$$

$$0 \le x^s_{i,j} \le (1 - y^s_{i,j})x_{i,j},\ \forall i,j,s; \tag{4–20}$$

$$\sum_{\{j:(i,j)\in E\}} x^s_{i,j} - \sum_{\{j:(j,i)\in E\}} x^s_{j,i} = \text{sign}(\text{d}_\text{i})\text{r}^\text{s}_\text{i},\ \forall \text{s, i, ;} \tag{4–21}$$

$$0 \le r^s_i \le |d_i|; \tag{4–22}$$

$$\zeta + \frac{1}{S(1-\alpha)} \sum_{s=1}^{S} t_s \le C; \tag{4–23}$$

$$t^s \ge \sum_{\{i:d_i<0\}} l_i \cdot \left(-d_i - r^s_i x^s_{j,i}\right) - \zeta,\ t_s \ge 0, \forall s = 1,2,\dots,S; \tag{4–24}$$

$$\zeta, t^s, x_{ij}, x^s_{i,j}, r^s_k \in R,\ \forall (i,j) \in E, \forall k \in V, \forall s = 1,2,\dots,S. \tag{4–25}$$

4.2 Decomposition Method for Network Flow Problem

The formulated problem (4–17)-(4–25) is a partial case of a two stage stochastic problem that can be written as follows:

$$\text{Minimize}_x f(x) + g(x,y),$$

where the deterministic part $f(x)$ can be computed when the decision is taken and the so called second stage part $g(x,y)$ that depends on the initial decision x, stochastic outcome ξ and the response variable y that is determined on the second stage after a random scenario had been implemented. In the case when $g(x,y)$ is linear with respect to x and y and the relations between x and y are set by linear constraint, we can use sampling and finally arrive at the linear program having block-ladder structure:

$$
\begin{array}{lllllllllll}
V = & \text{Minimize} & c^T x & + & f_1^T y_1 & + & f_2^T y_1, & +\ldots+ & f_k^T y_k & & \\
& \text{s.t.} & Ax & & & & & & & = & b, \\
& & B_1 x & + & D_1 y_1 & & & & & = & d_1, \\
& & B_2 x & & & + & D_2 y_2 & & & = & d_2, \\
& & \vdots & & & & & \ddots & & \vdots & \\
& & B_k x & & & & & + & D_1 y_1 & = & d_k.
\end{array}
\tag{4–26}
$$

For example, in our network flow x variables correspond to the initial flow design and second stage respond variables y_i are the remaining flows and balances.

Due to potentially large amount of scenarios the size of the linear program can be huge and, therefore, the problem may not be tractable.

There are several techniques aimed to deal with such a special structure and large amount of variables. They are the internal point methods, column generation algorithms, etc. We will use Benders decomposition method that has been a classical method for solving large scale stochastic programming problems. It was originally designed by Benders [9] to solve mix-integer linear problems, and later extended to nonlinear programs. It has been realized that the algorithm fits the needs of stochastic optimization when the problem is decomposable into deterministic (first stage) and stochastic (second stage) parts.

The problem (4–26) can be rewritten as a following Master Problem (MP):

$$
\begin{array}{lllllll}
MP: V = & \min_x & c^T x & + & \sum_{j=1}^{K} z_i(x), & & \\
& \text{s.t.} & Ax & & & = & b, \\
& & x \geq 0, & & & &
\end{array}
$$

where

$$
\begin{aligned}
P2_i \ : \ z_i(x) = \ & \min_{y_i} \quad f_i^T y_i, \\
& \text{s.t.} \quad D_i y_i \quad = \ d_i - B_i x, \\
& \qquad\quad y_i \geq 0,
\end{aligned}
$$

is a second stage problem that corresponds to scenario i, when the first stage decision variables x are fixed.

By utilizing notion of duality, we can compute $z_i(x)$ via the dual of $P2_i$ where parameter x is moved to objective function. Let us denote the dual as $D2_i$ as

$$
\begin{aligned}
D2_i \ : \ z_i(x) = \ & \max_{p_i} \quad p_i(d_i - D_i x), \\
& \text{s.t.} \quad D_i^T p_i \quad \leq \ f_i.
\end{aligned}
$$

Denote the feasible set of the dual problem as

$$
\mathcal{D}_2 = \{p_i | D_i^T p_i \leq f_i\}. \tag{4–27}
$$

Let

$$
p_i^1, ..., p_i^{I_i}, \tag{4–28}
$$

be the extreme points of the set $\mathcal{D}_2$ and

$$
r_i^1, ..., r_i^{J_i}, \tag{4–29}
$$

be the extreme rays of the set. Let us note that in our specific case, $D2_i$ will always have a finite optional solution because the primal will have a solution for any valid network flow x chosen on a first stage. Therefore, the solution of $D2_i$ will be one of the extreme points $\bar{p}_i$ and the optimal objective function value z_i will satisfy:

$$
z_i(x) = (\bar{p}_i)^T (d_i - B_i x) = \max_k (p_i^k)^T (d_i - B_i x). \tag{4–30}
$$

Thus, we can re-write $D2_i$

$$\begin{aligned} D2_i \ : \ z_i(x) = \ & \min_{z_i} z_i, \\ & \text{s.t.} \quad p_i^j(d_i - D_i x) \quad \leq \ z_i, \ j = 1, ...l_j. \end{aligned}$$

The latter problem assures that optimal value $z_i(x)$ is a dual objective evaluated at the best extreme points of the dual feasible set.

Now we can reformulate the initial Master Problem and write it as

$$\begin{aligned} MP : V = \ & \min_{x, z_1, \ldots, z_k} \quad c^T x \quad + \ \sum_{i=1}^{K} z_i(x), \\ & \text{s.t.} \quad Ax \quad = \ b, \\ & \quad x \geq 0, \\ & \quad (p_i^j)^T(d_i - B_i x) \quad \leq \ z_i, \\ & \quad j = 1, \ldots, J_i, \ i = 1, \ldots, k. \end{aligned}$$

This problem is called Full Master Problem (FMP). In this problem a huge number of second stage variables is replaced with k extra variables z_i. Besides, we have added constraints whose number can be exponentially large in a general case.

Let us note that we can consider a small subset of the full set of constraints. We will obtain a reduced Master Problem which contains only m extreme points of the Full Master Problem's constraints:

$$\begin{aligned} RMP^m : V_m = \ & \min_{x, z_1, \ldots, z_k} \quad c^T x \quad + \ \sum_{i=1}^{K} z_i(x), \\ & \text{s.t.} \quad Ax \quad = \ b, \\ & \quad x \geq 0, \\ & \quad (p_i^j)^T(d_i - B_i x) \quad \leq \ z_i, \\ & \quad \text{for some } i \text{ and } j. \end{aligned}$$

After this problem is solved we obtain an optimal objective value V_m, which is a lower bound of the optimal solution V, i.e.:

$$V^M \leq V.$$

To check if the optimality solution point $\bar{x}, \bar{z}_1, \ldots, \bar{z}_k$ for the reduced problem, we need to verify whether the above point violates any of the constraints that are not included in the reduced problems. Thus, we solve the following K problem:

$$\begin{aligned} Q_i(\bar{x}) = \quad & \max_{p_i} \quad p_i(d_i - B_i\bar{x}), \\ & \text{s.t.} \qquad D_i^T p_i \qquad\qquad \leq \quad f_i, \end{aligned}$$

and check whether $\bar{x}, \bar{z}_1, \ldots, \bar{z}_k$ is optimal for the full master problem. For each of the K problems the algorithm will return an optimal double and primal extreme solutions $\bar{p}_i$ and $\bar{y}_i$ of the optimization problem $Q_i(\bar{x})$. If it appears that the new solution violates the constraint

$$(\bar{p}_i)^T(d_i - B_i\bar{x}) \leq \bar{z}_i,$$

we add this constraint to the reduced master problem.

After $\bar{y}_i$ are computed for all second stage problems we can update an upper bound:

$$UB \leftarrow \min\{c^T\bar{x} + \sum_{i=1}^{K} f_i^T\bar{y}_i\}.$$

If it happens that none of the constraints of K problems is violated, i.e.

$$\max_{l=1,\ldots l_i} (p_i^l)^T(d_i = B_i\bar{x}) \leq \bar{z}_i, \forall i = 1, \ldots K,$$

the solution of reduced master problem is the solution of the full master problem and we can terminate the algorithm. Also, we can stop the algorithm when the difference between the upper and lower bounds is smaller than predefined tolerance level ϵ, i.e.

$$UB - V^k < \epsilon.$$

The formulation presented above (4–17)-(4–25) might not fit the RAM of the computer due to a large number of possible scenarios, and the problem seems to be a perfect candidate for the Benders decomposition algorithm. However, the Benders' decomposition technique cannot be directly applied to (4–17)-(4–25) formulation,

because one of the CVaR related constraints, namely (4–23), violates the block-ladder structure of the constraint matrix.

In order to apply the decomposition algorithm, we need to modify the structure of the problem (4–17)-(4–25) by moving risk constrain to objective using using the mathematical technique of the Lagrangian relaxation.

The Lagrangian relaxation consists in removing one constraint and inserting it inside the objective function with some penalty for its violation. In this case, the constraint is multiplied by a penalty coefficient which represents the Lagrangian multiplier (dual value or simplex multiplier) of the constraint. If this constraint were left in its initial formulation, in the optimal solution, the Lagrangian multiplier would get a specific value inherent to the solution. This solution found for some penalty coefficient λ corresponds to the level of CVaR for the specific quantile that we intend to achieve. Therefore, the magnitude of the penalty coefficient allows us to regulate our risk constraint severity. The greater it is, the more incentive we have to relax our CVaR level and/or percentile and expect smaller costs. In the case of relaxation, we proceed by external iterations on the multiplier values in order to find the penalty coefficient that corresponds to the desired solution. The desired multiplier level can be determined using numerical approximation techniques, for example a bisection method.

Thus, the initial problem

$$\begin{aligned}
&\text{Minimize} \sum_{(i,j)\in E} c_{i,j}x_{i,j}, \\
&\text{s.t.} \\
&\qquad \text{constratints } (4\text{–}2)\text{-}(4\text{–}3),\ (4\text{–}12)\text{-}(4\text{–}14)\ ; \\
&\qquad \text{CVaR}_\alpha L(x,y) \leq C,
\end{aligned}$$

can be transformed into

$$\text{Minimize} \sum_{(i,j)\in E} c_{i,j}x_{i,j} + \lambda \cdot (\text{CVaR}_\alpha L(x,y) - C),$$

s.t.

$$\text{constratints } (4\text{–}2)\text{-}(4\text{–}3),\ (4\text{–}12)\text{-}(4\text{–}14)\ ,$$

where the parameter λ is the mentioned above multiplier. Note that the constant term $-\lambda C$ can be neglected and therefore we omit it later in the text. After performing the standard transformation the LP problem can be written as follows

$$\text{Minimize} \sum_{(i,j)\in E} c_{i,j}x_{i,j} + \lambda \cdot \left(\zeta + \frac{1}{S(1-\alpha)} \sum_{s=1}^{S} t_s\right), \tag{4–31}$$

s.t.

$$\sum_{\{j:(i,j)\in E\}} x_{i,j} - \sum_{\{j:(j,i)\in E\}} x_{j,i} = d_i, \forall i \in V, \tag{4–32}$$

$$0 \le x_{i,j} \le u_{i,j}, \forall i,j \in V. \tag{4–33}$$

$$0 \le x^s_{i,j} \le x_{i,j}(1 - y^s_{i,j}),\ \forall i,j,s; \tag{4–34}$$

$$r^s_i \le |d_i|,\ \forall i \in V, s = 1, \dots S; \tag{4–35}$$

$$\sum_{\{j:(i,j)\in E\}} x^s_{i,j} - \sum_{\{j:(j,i)\in E\}} x^s_{j,i} = \text{sign}(\text{d}_\text{i})\text{r}^\text{s}_\text{i},\ \forall \text{s, i}, \tag{4–36}$$

$$t^s \ge \sum_{\{i:d_i<0\}} l_i \cdot (-d_i - r^s_i) - \zeta,\ t_s \ge 0, \forall s = 1, 2, \dots, S; \tag{4–37}$$

$$\zeta, t^s, x_{ij}, x^s_{i,j}, r^s_i \in R,\ \forall (i,j) \in E,\ \forall i \in V, \forall s = 1, 2, \dots, S. \tag{4–38}$$

Proposition 4.1. *Let (4–17)-(4–25) has optimal solution. Then, there exists such value $\lambda = \lambda^*$ that optimal solution x of problem (4–31)-(4–38) is also optimal for (4–17)-(4–25)*

We provide a formal description of the algorithm applied to the network flow formulation

Input α, λ, d_i, c_{ij}, y_{ij}^s

Initialize Set lower bound

$$LB = 0,$$

Set upper bound to the sum of maximal possible cost and loss:

$$UB = \sum_{(i,j)\in E} c_{i,j}u_{i,j} + \lambda \sum_{i:d_i<0} l_i(-d_i).$$

Write initial reduced master problem:

$$RMP: V_0 = \text{Minimize}_{x,\zeta,z} \quad \lambda\zeta + \sum_{\{j:(i,j)\in E\}} c_{i,j}x_{i,j} + \frac{\lambda}{S(1-\alpha)}\sum_{s=1}^{S} z_s, \quad (4\text{–}39)$$

s.t.

$$\sum_{\{j:(i,j)\in E\}} x_{i,j} - \sum_{\{j:(j,i)\in E\}} x_{j,i} = d_i, \forall i \in V, \quad (4\text{–}40)$$

$$0 \le x_{i,j} \le u_{i,j}, \forall i,j \in V. \quad (4\text{–}41)$$

$$z_s \ge 0 \quad (4\text{–}42)$$

$$x_{i,j}, \zeta, z_s \in R \quad (4\text{–}43)$$

Set $k \leftarrow 0$

Update lower bound Solve reduced master problem. Let V_k be the optimal value and $\bar{x}, z\bar{e}ta$ optimal point at k-th iteration. Then

$$LB \leftarrow \max(LB, V^k).$$

Update upper bound For $\forall s = 1, ..., S$ solve second stage problems for $x = \bar{x}$ and $\zeta = \bar{\zeta}$:

$$z_s(x, \zeta) = \text{Minimize}_{x^s, r^s, t^s} \quad t^s \tag{4–44}$$

s.t.

$$x^s_{ij} \leq (1 - y^s_{ij}) x_{ij}, \tag{4–45}$$

$$\sum_{\{j:(i,j)\in E\}} x^s_{i,j} - \sum_{\{j:(j,i)\in E\}} x^s_{j,i} - \text{sign(d}_\text{i})\text{r}_\text{i} = 0, \forall \text{i} \in \text{V}; \tag{4–46}$$

$$r_i \leq |d_i|, \forall i \in V; \tag{4–47}$$

$$t^s + \sum_{i:d_i<0} l_i r_i \geq - \sum_{i:d_i<0} l_i d_i - \zeta, \tag{4–48}$$

given that

$$x = \bar{x}, \ \zeta = \bar{\zeta}$$

If the solution $z_s(\bar{x}, \bar{\zeta}) > \bar{z}_s$, add a constraint

$$\pi_s^T RHS(x, \zeta) \geq z_s$$

to the reduced master problem (4–39)-(4–39). Here, π_s is the dual solution and $RHS(x, \zeta)$ is the right hand side vector of constraints (4–45)-(4–48) that is a linear function of first stage variables x and ζ.

If the optimal solution of all $z_s(\bar{x}, \bar{\zeta})$ is no larger than $\bar{z}^s$, the reduced master problem solves the full master problem and we terminate. Otherwise, we update the upper bound as follows

$$UB \leftarrow \min \left\{ UB, \lambda\bar{\zeta} + \sum_{\{j:(i,j)\in E\}} c_{i,j}\bar{x}_{i,j} + \frac{\lambda}{S(1-\alpha)} \sum_{s=1}^{S} z_s(\bar{x}, \bar{\zeta}) \right\},$$

Check If

$$UB - LB \leq \epsilon$$

then terminate otherwise increment k:

$$k \leftarrow k + 1$$

and repeat Update lower bound and Update upper bound steps.

Finally The optimal network flow is represented by $\bar{x}$. Optimal cost value is

$$\sum_{(i,j)\in E} c_{ij}\bar{x}_{i,j},$$

and the α-CVaR value at optimality is

$$\lambda\left(\bar{\zeta} + \frac{1}{S(1-\alpha)}\sum_{s=1}^{S}\bar{z}_s\right).$$

The value of corresponding α-VaR is stored in $\bar{\zeta}$.

4.3 Computational Experiments

This section experiments with the old and new loss functions for robust network flow problems. First of all, it is interesting to see how the the network behaves when we tighten the CVaR constraint on the loss.

In order to test the new loss function we considered the example from [16]. For the 6 nodes network shown in Figure 4-2, A, I have generated 100 random scenarios. Deterministic setup was considered and then 70% CVaR was placed into the objective function with the penalty coefficient $\lambda = 10$. We can notice that the flow picture slightly changes as the risk is considered. It is also interesting to notice that unlike the case of the old loss function, there is no fraction flow in the network with the CVaR penalty in the objective.

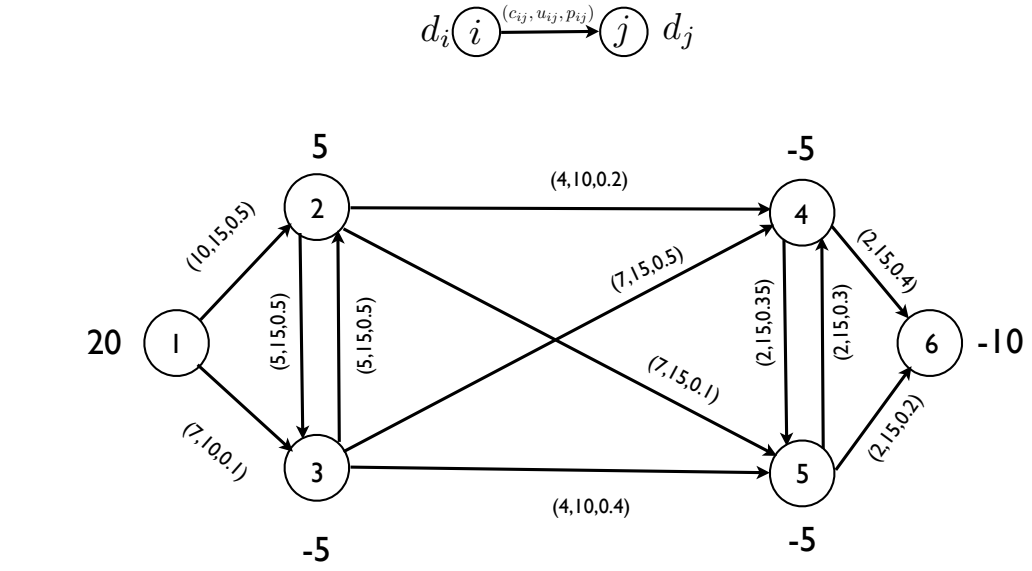
The algorithm converged in three iterations:

Iteration 1 Upper bound =440, Lower Bound=526.667.

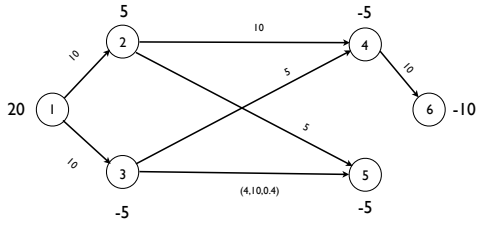
Iteration 2 Upper bound =501.667, Lower Bound=526.667.

Iteration 3 Upper bound =506.667, Lower Bound=506.667.

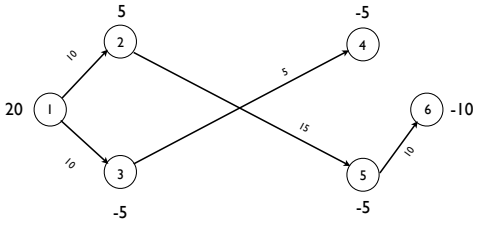
Objective function if the CVaR is included is 425, CVaR value is 8.17. In the deterministic case the objective is 295 and the CVaR of loss in this case is 23.17.



A Initial Network

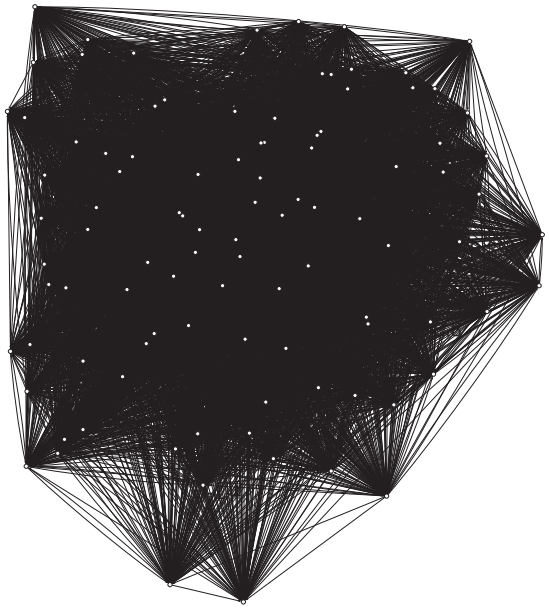


B $10\text{CVaR}_{0.7}(L)$ in objectuve

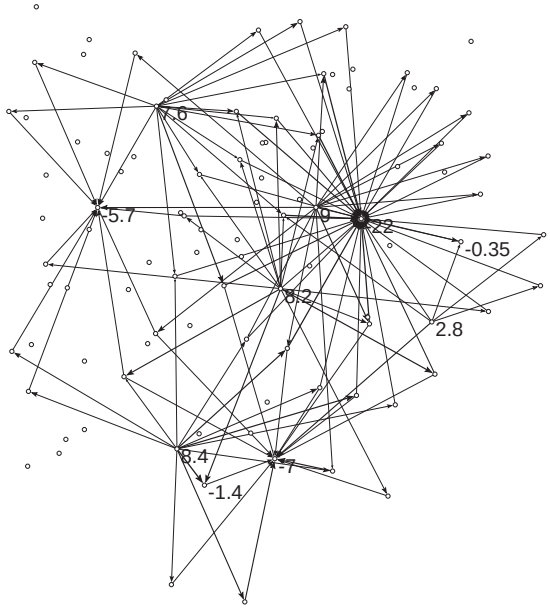


C Deterministic solution

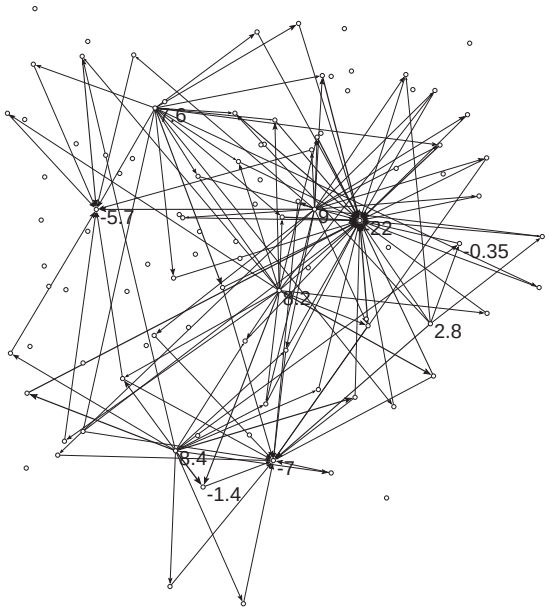
Figure 4-2. New loss function in a 6 vertice network. A - initial network, B - minimal cost flow without with CVaR in objective, C deterministic case without CVaR



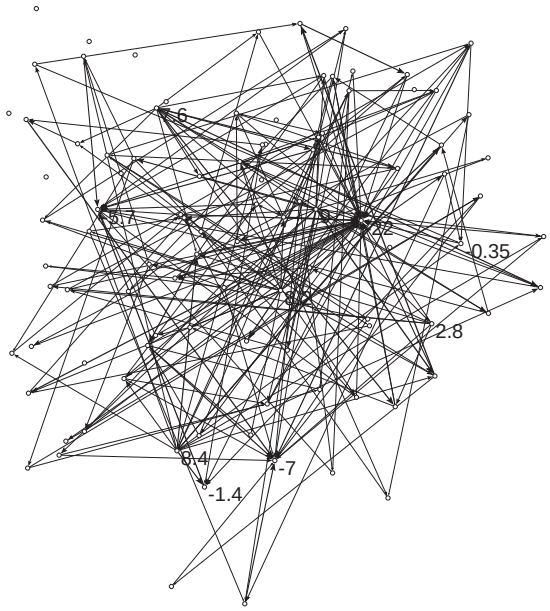
A Initial Network



B Deterministic Solution



C CVaR ≤ 10



D CVaR ≤ 5

Figure 4-3. Flow schedule in the network with and without considering CVaR robust constraint. A - initial network, B - minimal cost flow without without considering reliability issues, C and D - CVaR of loss is bounded by 5 and 10 respectively.

CHAPTER 5
ROBUST SYSTEM IDENTIFICATION FOR SPACE WEATHER FORECASTING

It is known that space, biological and technical systems can be significantly affected by the processes of transferring solar energy to the Earth magnetosphere or ionosphere [5, 37, 39]. Major factors of this influence are studied in the context of space weather problems. Solar influence on near-earth space is characterized with geomagnetic indices. These indices are computed based on measurements and represent only a small part of the complex phenomenon of interaction between the Sun and the Earth. The problem of selecting the proper index and relating it to a certain physical process is a complex task that is necessarily preceded by extensive research [5]. Therefore, it is essential to be able to predict such characteristics of geomagnetic activity as Kp-index, Ap-index, Dst-index, etc. The purpose of the present chapter is to provide an adequate prediction of Dst-index prediction in details. According to the modern views in science, this index represents global behavior of magnetosphere plasm under the influence of solar wind.The Dst is a geomagnetic index is constructed by averaging the horizontal component of the geomagnetic field from mid-latitude and equatorial magnetograms from all over the world. Practically Dst index is determined based on the measurements from satellites on hourly basis. Negative Dst values indicate a magnetic storm is in progress, the more negative Dst is the more intense the magnetic storm.

It is extremely important to select an adequate mathematical model for Dst-index prediction. There are several models described in the literature, such as neural network, regression, principal component based models, and group method of data handling models. A non-linear black-box model is chosen for this problem. The primary feature of nonlinear “black box” is using time series, composed of two solar wind parameters, as an input and Dst-index as an output [5]. Obviously, the complexity of this problem will increase along with the number of model variables, when the multi-dimension time series is used for characterizing solar wind.

5.1 Modeling Magnetosphere as a Black Box

Presently, identification and prediction models are widely used for studying linear and non-linear processes in the space [7, 12–14, 31, 34, 37]. It is possible to predict the Earth magnetosphere by building black box model which related solar wind parameters (such as magnetic field characteristics, solar wind speed, etc) to measurable geophysics indices (for example Kp or Dst indices). To build such a model we should keep in mind that the magnetosphere stays in weak turbulent state. It is known that linear processes in plasma lead to the energy exchange between waves and particles, while nonlinear processes lead to the exchange via three or four wave interactions. Typically, the higher then third order degree of wave interactions are neglected in plasma theory [5, 7, 56] that significantly simplifies the modeling. Solar wind parameters are recorded by satellites on a regular basis while geomagnetic indices are measured on the Earth surface. Therefore, the time series corresponding to solar wind characteristics and geomagnetic indices can be used as input and output for non-linear model. In order to build the model itself, the known methods of dynamic system identification can be applied [34, 37].

The most direct approach to magnetosphere modeling consists in accounting for the entire chain interactions in magnetosphere in the model. Such an approach, however, does not look implementable because of insuperable mathematical and physical complexity of the magnetosphere processes. The impossibility of direct modeling stimulated researchers to look for alternative approaches. One of them is based on using qualitative considerations.

For example, let us consider the reasoning behind the functional interdependence of Dst-index and product of solar wind v and southern component of interplanet magnetic field B_z

$$D_{st} = f(vB_z). \tag{5–1}$$

It is known that the induced by solar wind electric field $\vec{E}$ influences on plasma particles [5]. The electric field $\vec{E}$ is a dominating factor of mass transition in magnetosphere under the unperturbed conditions and determines the kinetic energy of the particles $W = W(\vec{E})$. A charged particle, which has kinetic energy $W = W_{\perp} + W_{\parallel}$ and charge value q, drifts with the average speed of

$$\vec{V}_D = \frac{W_{\perp} + 2W_{\parallel}}{qB^3} \left[\vec{B} \times \nabla B\right]. \tag{5–2}$$

The total drift of the particles induces circle electric current near the Earth. Thus, the field $\vec{E}$ determined the amount of electric current. The electric field that effectively penetrates the magnetosphere is determined by the southern component of electric field

$$|\vec{E}| = \frac{1}{c} v B_z \tag{5–3}$$

and the circle current $\vec{I}$ defines Earth magnetic field depression (i.e. Dst index). Apparently, there must exist non linear interdependence of Dst index and the mentioned electric field.

Basing on the detected functional dependency, we can reconstruct Dst index from the measured values of solar wind and southern component of magnetic field. This approach was utilized in the series of works on identification of discrete Dst index prediction models [7, 12] and explanation on the physical processes that drive the index dynamic. One of the simplest ways of such identification is to construct a nonlinear transfer function using Fourier representation for input and output data. A similar approach is used in this chapter.

The notion of black-box is well known in computer science, electrical engineering and other engineering fields. It is based on the assumption that inner components of a system are not important for describing the system. It is sufficient to know the relation between input and output to characterize the system. The internal state of such a system is set by the external factor $u(t)$, which is usually a multi dimension time series.

This factor is called the input of dynamic system. The inner state of the black box at time t could be set by vector $\psi(t)$ if we knew it. Since we do not actually know this vector, we can use some experimental measurements $y(t)$ instead. We assume that $y(t)$ is somehow functionally dependent on $\psi(t)$. Let us call $y(t)$ an output of the nonlinear black box. Then, the problem of system identification consists in searching analytical description of the model (i.e. $\psi(t)$) based on the input and output.

If the system is linear, the output $y(t)$ can be represented as the convolution of the $u(t)$ and impulse frequency characteristics $h(\tau)$

$$y(t) = \int_0^\infty h(\tau)y(t-\tau)d\tau. \tag{5–4}$$

If we switch from time variable t to frequency f, Fourier transformation $h(\tau)$ function is called linear amplitude-frequency characteristics $H_{\text{lin}}(f)$. In this case it binds spectral components of input and output with the equation

$$y_f = H_{\text{lin}}(f)u(f). \tag{5–5}$$

The absolute value of complex value $H_{\text{lin}}(f)$ characterized the degree of amplifying the system's input spectrum component. The phase corresponds to the delay between the input and output at frequency f. The $f(\tau)$ and $H_{\text{lin}}(f)$ functions are equally valuable for describing the black box. Thus, the output of the linear systems is completely defined by the input and pulse-frequency characteristics. The integral equation (5–4) can be easily generalized for non linear case using Volterra series [48]. In this case, the input signal is defined by equation

$$\begin{aligned} y(t) = \int_0^\infty h_1(\tau)y(t-\tau)d\tau + \int_0^\infty \int_0^\infty h_2(\tau_1,\tau_2)u(t-\tau_1)u(t-\tau_2)d\tau_1 d\tau_2 + \\ + \int_0^\infty \dots \int_0^\infty h_i(\tau_1,\dots,\tau_i)u(t-\tau_1)\cdot\dots\cdot u(t-\tau_i)d\tau_1\dots d\tau_i + \dots, \end{aligned} \tag{5–6}$$

where $h_i(\tau_1,\dots,\tau_i)$ is the i-th Volterra kernel. If all the kernels are known, it is possible to compute the output and to study the properties of the system.

In a case of discrete input ,the integrals in (5–6) are replaced with sums

$$y_k = \sum_{n_1} h_1(n_1)u_{k-n_1} + \sum_{n_1,n_2} h_1(n_1)h_2(n_1,n_2)u_{k-n_1}u_{k-n_2} + \ldots + \sum_{n_1,\ldots,n_i} h_1(n_1)\ldots h_i(n_1,\ldots,n_i)u_{k-n_1}\ldots u_{k-n_i}, \tag{5–7}$$

where k represents discretized time.

Fourier transformation applied to (5–6) leads to the following equation for the output:

$$y_f = H_1(f)U(f) + \sum_{f_1,f_2:f_1+f_2=f} H_2(f_1,f_2)u_{f_1}u_{f_2} + \ldots . \tag{5–8}$$

$H_i(f_1,\cdots,f_i)$ is a generalized frequency transfer function which accounts nonlinear properties of the system. As we previously mentioned, such representation can be limited with three wave interaction for weak turbulent space plasma. Therefore, the nonlinear processes having wave order four and more can be neglected. Apparently, $H_1(f)$ has the physical meaning equivalent to $H_{\text{lin}}(f)$. Function $H_2(f_1,f_2)$ depends on two frequencies and represents nonlinear quadratic interaction between the components f_1 and f_2 (f_i can take negative value. It also represents energy transfer to the system output of frequency $f = f_1 + f_2$. Let us note that (5–8) can be strictly justified using the theory of turbulent plasma [49, 56]. The correspondence between mathematical formulation of the turbulent plasma theory and Volterra series is important for the analysis of an experimental measurement.

In this chapter, we will use discrete black-box model that allows representing the output with Volterra series:

$$y(k) = F[y(k-1),\ldots,y(k-n_y),\ldots,u(k-1),\ldots,u(k-n_u),\xi(k),\ldots,\xi(k-n_\xi)], \tag{5–9}$$

where $F[\cdot]$ is a polynom of variables $u(k)$, $y(k)$, and $\xi(k)$.

The least squares method is traditionally used for determining unknown coefficients of linear black box model from experimental data. However, this approach is not applicable to discrete polynomial model because the number of polynom members

dramatically increases along with polynom degree and time series size. Therefore, we will apply the statistical structure identification techniques known as leaps-and-bounds in the literature [40]. The idea of the procedure is to add only such nonlinear components that significantly improves prediction quality and prevent model overfitting. We add only the component whose combination significantly contributes to the prediction quality measured using certain criteria.

Let us assume the magnetosphere to be a system with one input and one output. Let us note that the model can also be improved by adding extra terms and noise parameters. We did not study this possibility at the present stage of the work. The computational experiments demonstrated reliable prediction efficiency when $B_z v$ is chosen as an input parameter.

5.2 Robust Model Reconstruction

From the phisycal considerations stated in the previous section it follows that the magnetosphere can be modeled as a linear of bilinear system where state vector $x(t)$ represents the magnetosphere index while control vector $u(t)$ corresponds to a solar activity. Since the magnetosphere is driven by solar parameter as well as internal state, the dynamic system can be modeled as bilinear system with positive feedback.

A bilinear system is set by the following set of differential equations:

$$\dot{x}(t) = (\sum_{i=1}^{q} A_i u_i(t))x(t) + Bx(t) + f, \tag{5–10}$$

where A_i, B are $m \times n$ matrices, f is n-dimensional vector. $u(\cdot) : R \mapsto R^q$ is an input or control function and $x(\cdot) : R \mapsto R^n$ is a state of the system. By identification we understand restoration of parameters A_i, B and f if input $u(\cdot)$ and output $x(t)$ are known at some time moments.

To solve this problem usually the system is discretized using Runge-Kutta or other methods

$$x_{k+1} = (\sum_{i=1}^{q} A_i u_k)x_k + Bx_k + f. \tag{5–11}$$

Thus, we can consider the system in discrete time periods, for example every hour, and model the magnetosphere using a discrete bilinear system

Then, based on known x_k and u_k we determine A_i, B and f to minimize the difference between predicted state on the next step x_{k+1} and observable state $\bar{x}_{k+1}$. That is we arrive at regression-type optimization problem:

$$\min_{A_i,B,f} \mathcal{D}(e_1, e_2, ..., e_N), \tag{5–12}$$

$$\text{s.t.} \quad e_k = x_k - \bar{x}_k, \ k = 1, \dots, N, \tag{5–13}$$

$$x_{k+1} = (\sum_{i=1}^{q} A_i u_k)\bar{x}_k + Bx_k + f, k = 0, \dots, N-1. \tag{5–14}$$

[19] considered least squares deviation measure $\mathcal{D}(\cdot) = \|e\|_2^2$ that has analytical solution. Later in [Yatsenko2009] Yatsenko *et al* claimed that if deviation of state is limited it leads to more stable solutions.

The natural deviation measure of such kind is a CVaR based deviation measures (i.e. $\mathcal{D}(e) = \text{CVaR}_\alpha(e) - \text{CVaR}_\beta(-e)$). Using standard linearization techniques we can reduce system reconstruction to linear problem to obtain stable solutions.

5.3 Nonlinear Structure Reconstruction

Table 5-1. Leaps-and-bound based variable selection for fixed number of regressors $k = 1, \dots, 8$ for one step ahead forecasting (linear model). "+" in columns k indicates that the corresponding variable is added to the model.

	Maximal number of regressors							
Factor	1	2	3	4	5	6	7	8
Dst	+	+	+	+	+	+	+	+
$V(-1) \cdot Bz(-1)$		+	+	+	+	+	+	+
$V(-1)$			+	+	+	+	+	+
$V(-1) \cdot By(-1)$					+	+	+	+
$By(-1) \cdot Bz(-1)$						+	+	+
$V(-1) \cdot Bx(-1)$							+	+
$Bz(-1)$							+	+
$Bx(-1) \cdot Bz(-1)$								+

It is important to know which components and their combination of solar activity are the best predictor of the system. It is also essential to determine how many steps in time

Table 5-2. Leaps-and-bound based variable selection for fixed number of regressors $k = 1, \dots, 10$ for one step ahead forecasting (bilinear). "+" in columns k indicates that the corresponding variable is added to the model.

	Maximal number of regressors							
Factor	1	2	3	4	5	6	7	8
$Dst(-1)$	+	+	+	+	+	+	+	+
$V(-1) \cdot Bz(-1)$		+	+	+	+	+	+	+
$V(-1)$			+	+	+	+	+	+
$Dst(-1) \cdot Bz(-1)$					+	+	+	+
$Dst(-1) \cdot V1(-1) \cdot Bz(-1)$					+	+	+	+
$Dst(-1) \cdot V1(-1)$						+	+	+
$By(-1)$							+	+
$V(-1) \cdot By(-1)$				+				
$Dst(-1) \cdot By(-1) \cdot Bz(-1)$								+

are enough for reliable prediction. We have taken real data available on the NASA web site [41] collected for ten years period from January 1, 2000 until December 31, 2010 and applied variable selection techniques.

In order to determine the best predictors the variable selection statistical technique known as leaps and bounds was applied to all possible solver wind components as magnetic field (B_x, B_y, B_z) and solar wind speed (V) as well as their products. The idea behind leaps and bounds technique is to fix the number of regressors and consider all possible subsets of the fixed size. If the number of regressor is hight the enumeration becomes computationally challenging and therefore various heuristics are used that described in [40].The experiment has shown that the best predictors are Dst index, B_z magnetic field component and their products (refer to tables 5-1 and 5-2. This fact agrees with the previous research.

It has also been experimentally established that adding more than two time period into the model has almost no effect on prediction quality. Thus, there is no necessity to consider many steps as it has been done in previous works [19, 61].

CHAPTER 6
CONCLUSION

6.1 Thesis Contribution

Finding a solution which is robust to various uncertainties is a challenging task that has been researched extensively over the last decade. Motivated by the risk managing techniques in financial engineering, the author utilized and extended financial frameworks for non financial applications. The dissertation developed mathematical programming models utilizing a coherent quantitative risk measure Conditional Value-at-Risk (CVaR). These models allow achieving robust and efficient performance of sensors scheduling, network flow and solar wind prediction in the presence of uncertain factors which often need to be considered in practice.

I developed a mathematical framework for solving a class of sensors scheduling problems. Based on the modeling considered by Javuz and Jeffcoat in [62, 63], I introduced and formulated three robust optimization problems: two for deterministic and one for stochastic case. The obtained 0-1 problems are also re formulated in terms of cardinality functions. Numerical experiments are conducted using two commercial solvers ILOG CPLEX and AOrDa PSG. CPLEX gives exact solutions for small problems. Both solvers give an approximate solution in reasonable time for large problems. For large problems with many stochastic scenarios, PSG is faster than CPLEX in finding good quality approximate solutions. Also, PSG has an intuitive user friendly interface. These features of PSG make it an efficient and convenient tool for solving robust sensors scheduling problems in uncertain environments.

LP based model for network flow problems subject to uncertain arc failures has been developed. I have introduced a loss function that characterizes actual loss of flow in the network when certain arcs are destroyed. The robust network design problem under consideration extends approach developed by Boginski *et al* [16]. I proposed a new more realistic loss function that more adequately describes actual harm caused

by network component failures. The proposed loss function makes the optimization problems more difficult to solve compared to the formulation with previous loss function. In particular, the resulting linear problem includes second stage variables that make a straightforward way of solving LP impossible for a large size real world problem. In order to handle the complexity, I have proposed the technique of Lagrangian Relaxation. The CVaR constraint is relaxed and inserted in the objective function with penalty coefficients. Then, I utilized a special structure of the constraint matrix and used Benders' decomposition techniques which allow solving large scale two stage stochastic problems efficiently.

I have also suggested a robust approach to system identification that extends the framework for system identification developed at [19]. The CVaR-based deviation measure in the objective function serves as a regularization component and provides more stable restored dynamic systems used for space weather prediction.

6.2 Future Work

Robust network flow models need to address theoretical aspects of complexity. Namely, theoretical estimation should be established that provides bounds on how many sampling scenarios are required for the accurate approximation of risk measures. Such approximation was developed in [16] where Boginski *et al* investigated how many scenarios were required to ensure that the true value of the CVaR function $F_\alpha(x, \zeta)$ was close enough to the sample (scenario-based) value that is used in the LP formulation with a high confidence level. In particular, a polynomial upper bound on the number of scenarios was found for a particular problem. Due to complexity of the considered formulation, it is not possible to map the previously used techniques directly to the new framework. Therefore, finding good estimations is a matter of future research endeavors.

The approach considered for space weather prediction provides reliable prediction for magnetosphere indices. Further theoretical condition of system stability should be obtained for linear and especially nonlinear cases. The study of stability for nonlinear

system is a very challenging task in itself and could be a great topic of future studies. Further computational experiments are required to calibrate model parameters, such as CVaR percentile level α.

REFERENCES

[1] 2009. American optimal decision, portfolio safeguard. URL http://www.aorda.com.

[2] 2010. Ilog cplex. URL http://www.ilog.com/products/cplex/.

[3] Abbasi, Ameer Ahmed, Mohamed Younis. 2007. A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.* **30**(14-15) 2826–2841.

[4] Ahuja, R. K., T. L. Magnanti, J. B. Orlin. 1993. *Network Flows: Theory, Algorithms and Applications*. Prentice Hall.

[5] Akasofu, S.I., S Chapman. 1972. *Solar-Terrestrial Physics*. Clarendon Press, Oxford, England.

[6] Akyildiz, Ian F., Tommaso Melodia, Kaushik R. Chowdhury. 2007. A survey on wireless multimedia sensor networks. *Comput. Netw.* **51**(4) 921–960.

[7] Balikhin, M. A., I. Bates, S. Walker. 2001. Identification of linear and nonlinear processes in space plasma turbulence data. *Advances in Space Research* **28**(5) 787–800.

[8] Ben-Tal, Aharon, Laurent El Ghaoui, Arkadi Nemirovski. 2009. *Robust Optimization*. Princeton University Press.

[9] Benders, J. F. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* **4**(1) 238–252.

[10] Bertsekas, D.P. 1998. *Network Optimization-Continuous and Discrete Models*. Athena Scientific, Belmont.

[11] Bertsimas, D., M. Sim. 2003. Robust discrete optimization and network flows. *Mathematical Programming* **98**(1–3) 49–71.

[12] Billings, S. A. 1989. Identification of linear and nonlinear processes in space plasma turbulence data. *International Journal of Control* **50**(5) 1897–1923.

[13] Billings, S. A., W. S. F. Voon. 1983. Structure detection and model validity tests in the identification of nonlinear systems. *Control Theory and Applications*. IEEE, 193–199.

[14] Billings, S. A., Q. M. Zhu. 1994. Nonlinear model validation using correlation tests. *International Journal of Control* **60**(5) 1107–1120.

[15] Biswas, Pratik, Yinyu Ye. 2004. Semidefinite programming for ad hoc wireless sensor network localization. *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM, New York, NY, USA, 46–54.

[16] Boginski, Vladimir L., Clayton W. Commander, Timofey Turko. 2009. Polynomial-time identification of robust network flows under uncertain arc failures. *Optimization Letters* **3**(3) 461–473.

[17] Buczak, Anna L., Henry (Hui) Wang, Houshang Darabi, Mohsen A. Jafari. 2001. Genetic algorithm convergence study for sensor network optimization. *Inf. Sci. Inf. Comput. Sci.* **133**(3-4) 267–282.

[18] Campi, M.C., G. Calafiore. 2004. Decision making in an uncertain environment: the scenario-based optimization approach. J. Andrysek, M. Karny, J. Kracik, eds., *Multiple Participant Decision Making*. Andwanced Knowlede International, 99–119.

[19] Cheremnykh, O., V. Yatsenko, O. Semeniv, IU. Shatokhina. 2008. Nonlinear dynamics and prediction for space weather. *Ukrainian Jounal of Physics* **53**(5) 504–507.

[20] Chhetri, Amit S., Darryl Morrell, Antonia Papandreou-Suppappola. uary. Nonmyopic sensor scheduling and its efficient implementation for target tracking applications. *EURASIP J. Appl. Signal Process.* **2006**(1) 9–9.

[21] Commander, C.W., P.M. Pardalos, V. Ryabchenko, S. Sarykalin, T. Turko, S. Uryasev. 2008. Robust wireless network jamming problems. C.W. Commander, M.J. Hirsch, R.A. Murphey, P.M. Pardalos, eds., *Lecture Notes in Control and Information Sciences*. Springer, 399–416.

[22] Commander, C.W., P.M. Pardalos, V. Ryabchenko, S. Uryasev. 2007. The wireless network jamming problem. *Journal of Combinatorial Optimization* **14:4** 481–498.

[23] Corea, G. A., V. G. Kulkarni. 1990. Minimum cost routing on stochastic networks. *Operations Research* **38**(3) 527–536.

[24] Dantzig, G.B. 1963. *Linear Programming and Extensions*. Princeton University Press, New Jersey.

[25] Dantzig, G.B. 1963. *Linear Programming and Extensions*. Princeton University Press, New Jersey.

[26] Doulliez, Pierre J., M. R. Rao. 1971. Maximal Flow in a Multi-Terminal Network with Any One Arc Subject to Failure. *MANAGEMENT SCIENCE* **18**(1) 48–58.

[27] Ferentinos, Konstantinos P., Theodore A. Tsiligiridis. 2007. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Comput. Netw.* **51**(4) 1031–1051.

[28] Glockner, Gregory D., George L. Nemhauser, Craig A. Tovey. 2001. Dynamic network flow with uncertain arc capacities: Decomposition algorithm and computational results. *Comput. Optim. Appl.* **18**(3) 233–250.

[29] Hollick, Matthias, Ivan Martinovic, Tronje Krop, Ivica Rimac. 2004. A survey on dependable routing in sensor networks, ad hoc networks, and cellular networks. *EUROMICRO '04: Proceedings of the 30th EUROMICRO Conference*. IEEE Computer Society, Washington, DC, USA, 495–502.

[30] Jeong, Jaehoon, Sarah Sharafkandi, David H. C. Du. 2006. Energy-aware scheduling with quality of surveillance guarantee in wireless sensor networks. *DIWANS '06: Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*. ACM, New York, NY, USA, 55–64.

[31] Johansen, T. A. 1997. Constrained and regularized system identification. *In: Preprints IFAC Symposium on System Identification, Kitakyushu*. 1467–1472.

[32] Klappenecker, Andreas, Hyunyoung Lee, Jennifer L. Welch. 2008. Scheduling sensors by tilinglattices. *PODC '08: Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*. ACM, New York, NY, USA, 437–437.

[33] Koutsonikolas, Dimitrios, Saumitra M. Das, Y. Charlie Hu. 2007. Path planning of mobile landmarks for localization in wireless sensor networks. *Comput. Commun.* **30**(13) 2577–2592.

[34] Kuntsevich, V. 2006. *Control under uncertainty: Assured results in control and identification problems*. Naukova Dumka, Kiev, Ukraine.

[35] Kuorilehto, Mauri, Marko Hännikäinen, Timo D. Hämäläinen. 2005. A survey of application distribution in wireless sensor networks. *EURASIP J. Wirel. Commun. Netw.* **5**(5) 774–788.

[36] Li, Y., M. T. Thai, , W. Wu (eds). 2007. *Wireless Sensor Networks and Applications*. Springer.

[37] Ljung, L. 1999. *System Identification - Theory For the User*. PTR Prentice Hall, Upper Saddle River, N.J.

[38] Lobo, M. S., M. Fazel, S. Boyd. 2007. Portfolio optimization with linear and fixed transaction costs. *Annals of Operations Research* **152(1)** 376–394.

[39] Lundstedt, Gleisner, H. Gleisner, H. Lundstedt, P. Wintoft. 1996. Predicting geomagnetic storms from solar-wind data using time-delay neural networks.

[40] Miller, A. 2002. *Subset Selection in Regression*. Chapman & Hall/CRC.

[41] NASA. 2010. OMNI satellite database. URL http://nssdc.gsfc.nasa.gov/omniweb.

[42] Niculescu, Dragos, Badri Nath. 2003. Ad hoc positioning system (aps) using aoa. *The 28th Conference on Computer Communications*. IEEE, 1734–1743.

[43] Pardalos, P.M., Y. Ye, C.W. Commander (eds) V. Boginski. to appear in 2009. *Sensors: Theory, Algorithms, and Applications*. Springer.

[44] Pemberton, Joseph C., III Flavius Galiber. 2001. A constraint-based approach to satellite scheduling. *DIMACS workshop on on Constraint programming and large scale discrete optimization*. American Mathematical Society, Boston, MA, USA, 101–114.

[45] Rockafellar, R. T., S. Uryasev. 2000. Optimization of conditional value-at-risk. *Journal of Risk* **2** 21–41.

[46] Rockafellar, R.T., S.P. Uryasev. 2002. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance* **26** 1443–1471.

[47] Rudafshani, Masoomeh, Suprakash Datta. 2007. Localization in wireless sensor networks. *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, New York, NY, USA, 51–60.

[48] S., Boyd, Chua L. O. 1985. Fading memory and the problem of approximating nonlinear operators with volterra series. *IEEE Transactions on Circuits and Systems* **32**(11) 1150–1161.

[49] Sagdeev, R. Z., A. A. Galeev. 1969. *Nonlinear Plasma Theory*. Benjamin, White Plains, New York.

[50] Sarykalin, S, G Serraino, S Uryasev. 2008. VaR vs CVaR in risk management and optimization. *INFORMS Tutorial* .

[51] Schrijver, A. 2002. On the history of the transportation and maximum flow problems. *Mathematical Programming* **91**(3) 437–445.

[52] Singh, Sumeetpal S., Nikolaos Kantas, Ba-Ngu Vo, Arnaud Doucet, Robin J. Evans. 2007. Simulation-based optimal sensor scheduling with application to observer trajectory planning. *Automatica* **43**(5) 817–830.

[53] So, Anthony Man-Cho, Yinyu Ye. 2007. Theory of semidefinite programming for sensor network localization. *Math. Program.* **109**(2) 367–384.

[54] Uryasev, S. 2000. Conditional value-at-risk: Optimization algorithms and applications. *Financial Engineering News* **14** 1–5.

[55] Venkatesh, Swaroop, R. Michael Buehrer. 2006. A linear programming approach to nlos error mitigation in sensor networks. *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*. ACM, New York, NY, USA, 301–308.

[56] V.N., Tsytovich. 1972. *An Introduction to the Theory of Plasma Turbulence*. Pergamon, Oxford, New York.

[57] Wang, Chen, Li Xiao. 2008. Sensor localization in concave environments. *ACM Trans. Sen. Netw.* **4**(1) 1–31.

[58] Wu, Kui, Yong Gao, Fulu Li, Yang Xiao. 2005. Lightweight deployment-aware scheduling for wireless sensor networks. *Mob. Netw. Appl.* **10**(6) 837–852.

[59] Wu, Kui, Chong Liu, Jianping Pan, Dandan Huang. 2007. Robust range-free localization in wireless sensor networks. *Mob. Netw. Appl.* **12**(5) 392–405.

[60] Yan, Ting, Yu Gu, Tian He, John A. Stankovic. 2008. Design and optimization of distributed sensing coverage in wireless sensor networks. *Trans. on Embedded Computing Sys.* **7**(3) 1–40.

[61] Yatsenko, V. A., O.K. Cheremnykh, V.M. Kuncevich, Semeniv O.V. 2009. Geomagnetic activity model identification and space weather forecasting. *Problems of Control and Informatics* (6) 114–124.

[62] Yavuz, M., D.E. Jeffcoat. 2007. An analysis and solution of the sensor scheduling problem. *Advances in Cooperative Control and Optimization*, vol. 369. Springer, 167–177.

[63] Yavuz, M., D.E. Jeffcoat. 2007. Single sensor scheduling for multi-site surveillance. Tech. rep., Air Force Research Laboratory.

[64] Yick, Jennifer, Biswanath Mukherjee, Dipak Ghosal. 2008. Wireless sensor network survey. *Comput. Netw.* **52**(12) 2292–2330.

[65] Yuan, Yong, Zongkai Yang, Min Chen, Jianhua He. 2006. A survey on information processing technologies in wireless sensor networks. *Int. J. Ad Hoc Ubiquitous Comput.* **1**(3) 103–109.

BIOGRAPHICAL SKETCH

Mykyta Boyko was born in Dnipropetrovsk, Ukraine. He received his bachelor's and master's degrees in applied mathematics from Dnipropetrovsk National University in 2000 and 2001 respectively. Mykyta Boyko worked as a software developer from 2000 until 2005 for large international software companies, where he had an opportunity to work on complex enterprise business solutions.

In 2006, Mykyta Boyko joined the graduate program in Industrial and Systems Engineering at the University of Florida. He received his Master of Science degree in industrial and systems engineering from the University of Florida in August 2007. Mykyta Boyko is the author of several scientific papers and surveys published in peer-reviewed journals and books. He also lectures programming classes at the University of Florida.