Spring 2014

Risk-averse optimization in networks

Maciej Wladyslaw Rysz
University of Iowa

RISK-AVERSE OPTIMIZATION IN NETWORKS

by

Maciej Wladyslaw Rysz

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Industrial Engineering
in the Graduate College of
The University of Iowa

May 2014

Thesis Supervisor: Associate Professor Pavlo Krokhmal

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Maciej Wladyslaw Rysz

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Industrial Engineering at the May 2014 graduation.

Thesis Committee: _____
Pavlo Krokhmal, Thesis Supervisor

_____
Yong Chen

_____
Geb Thomas

_____
Karim Abdel-Malek

_____
Asghar Bhatti

ACKNOWLEDGEMENTS

I would like to express my genuine gratitude to Dr. Pavlo Krokhmal for his guidance and support throughout my studies. I would also like to thank the committee members, Dr. Yong Chen, Dr. Geb Thomas, Dr. Karim Abdel-Malek and Dr. Asghar Bhatti.

ABSTRACT

The primary goal of this research is to model and develop efficient solution techniques for graph theoretical problems with topologically stochastic information that manifests in a various forms. We employ a stochastic programming framework that is based on a formalism of coherent risk measures in order to find minimum-risk graph structures with stochastic vertex weights. Namely, we propose a class of risk-averse maximum weighted subgraph problems that represent a stochastic extension of the so-called maximum weight subgraph problems considered in graph-theoretical literature.

The structural nature of these model poses a twofold challenge in developing efficient solution algorithms. First, accurate quantification of uncertainties in mathematical programming via risk measures in the form of convex constraints typically requires very large representative scenario sets, thus incurring lengthy solution times. In this regard, we first introduce an efficient algorithms for solving large-scale stochastic optimization problems involving measures of risk that are based on certainty equivalents. The second major challenge relates to the fact that problems of finding a maximum vertex subset of a defined property within a network are generally not solvable in polynomial time. Nevertheless, much emphasis has been placed on developing efficient combinatorial solution methodologies that exploit the structural nature of the sought subgraphs. In pursuance of analogous frameworks, we propose a graph-based branch-and-bound algorithm for solving models in the risk-averse maxi-

mum weighted subgraph problem class that is generally applicable to problems where a subgraph's weight is given by a super-additive function whose evaluation requires solving an optimization problem. As an illustrative example of the developed concepts, we consider risk-averse maximum weighted clique and k-club problems.

The mentioned network studies address problems with topologically exogenous information in the form uncertainties induced by stochastic factors associated with vertices. This assumption clearly relies on the premise that the network is structurally unvarying. For many application, however, it may also be of interest to examine decision making under conditions that admit topological changes of the network itself. To this end, we consider a two-stage stochastic recourse maximum graph problem that seeks to maximize the expected size of a subset of vertices over the decision time horizon. Namely, a subset conforming to a predefined structural property is select in the first stage, after which realizations of uncertainty in the form of edge failures and creations arise. Then, a second stage recourse is taken to "repair" the subset selected in the first stage by adding or removing vertices in order to ascertain its the defined property. An exact graph-based branch-and-bound solution criteria is proposed for instances where the sought subsets represent complete graphs.

Numerical experiments for the above studies demonstrating the underlying problem properties and improvements in computational time achieved by the developed algorithms are conducted.

TABLE OF CONTENTS

LIST OF TABLES vii

LIST OF FIGURES ix

LIST OF ALGORITHMS x

CHAPTER

1 RISK AVERSE DECISION MAKING UNDER UNCERTAINTY . . . 1

1.1 Introduction 1
1.2 Coherent and convex measures of risk 2
1.2.1 Implementation in stochastic programming 8

2 A SCENARIO DECOMPOSITION APPROACH FOR STOCHASTIC OPTIMIZATION PROBLEMS WITH A CLASS OF DOWNSIDE RISK MEASURES 12

2.1 Introduction 12
2.2 Scenario decompositon algorithm 15
2.2.1 Efficient solution method for problem (2.4) 22
2.3 Case study: Portfolio optimization with HMCR and LogExpCR measures 28
2.3.1 Portfolio optimization using HMCR measures 29
2.3.2 Portfolio optimization using LogExpCR measures 36
2.4 Computational experiments, scenario data, and results 38
2.5 Conclusions 41

3 ON RISK-AVERSE MAXIMUM WEIGHTED SUBGRAPH PROBLEMS 48

3.1 Introduction 48
3.2 Risk-averse maximum vertex problem 50
3.3 Solution approaches for risk-averse maximum weighted subgraph problems 58
3.3.1 A mathematical programming formulation 58
3.3.2 A graph-based branch-and-bound algorithm 60
3.4 Case studies: Risk-averse maximum weighted clique problem with HMCR measures 65
3.4.1 Case Study I: Isolated risk exposure 66

3.4.2 Case Study II: Neighbor-dependent risk exposures 69
3.4.3 Numerical experiments for Case Study I 73
3.4.4 Numerical experiments for Case Study II 75
3.5 Conclusions 81

4 ON RISK-AVERSE WEIGHTED K-CLUB PROBLEMS 83

4.1 Introduction 83
4.2 Risk-averse stochastic maximum k-club problem 84
4.3 Solution approaches for risk-averse maximum weighted 2-club problems 86
4.3.1 A mathematical programming formulation 87
4.3.2 A graph-based branch-and-bound algorithm 87
4.4 Case study: Risk-averse maximum weighted 2-club problem with HMCR measures 91
4.4.1 Numerical experiments and results 92
4.5 Conclusions 93

5 TWO-STAGE STOCHASTIC RECOURSE MAXIMUM Π PROBLEM 95

5.1 Introduction 95
5.2 Two-stage stochastic recourse maximum Π problem 96
5.3 A combinatorial branch-and-bound technique for solving the two-stage stochastic recourse maximum clique problem 100
5.3.1 First stage branch-and-bound 101
5.3.2 Second stage branch-and-bound 103
5.4 Numerical experiments and results 105
5.5 Conclusion 109

6 SUMMARY AND FUTURE WORK 111

REFERENCES 114

LIST OF TABLES

Table

2.1 Average computation times (in seconds) obtained by solving problems [SOCP] and [SOCP-SD] for $p = 2$. 43

2.2 Average computation times (in seconds) obtained by solving problems [SpCOP] and [SpCOP-SD] for $p = 3$. 44

2.3 Average computation times (in seconds) obtained by solving problems [LpOCP] and [LpOCP-SD] for $p = 3$. 45

2.4 Average computation times (in seconds) obtained by solving a specified number of instances for problems [LECR] and [LECR-SD]. 46

2.5 Average number of partitioned scenarios from solving the scenario decomposition-based problems listed in Sections 2.3.1 – 2.3.2. 47

3.1 Average computation times (in seconds) obtained by solving problem (3.12) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and scenarios $N = 100$. 75

3.2 Average computation times (in seconds) obtained by solving problem (3.12) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and edge density $d = 0.5$. 76

3.3 Average optimal clique sizes obtained by solving problems (3.12) and (3.20) with risk measures (1.6). 78

3.4 Average optimal clique sizes obtained by solving problems (3.19) and (3.20) with risk measures (1.6). 80

4.1 Average computation times (in seconds) obtained by solving problem (4.6) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and scenarios $N = 100$. 94

5.1 Average solution times (in seconds) for problem (5.6) on random graphs with an edge density of 0.5, 20 scenarios, and $M = 0.15$. 108

5.2 Average solution times (in seconds) for problem (5.6) on random graphs with 40 vertices, an edge density of 0.5 and 20 scenarios. 108

5.3 Average solution times (in seconds) for problem (5.6) on random graphs with 40 vertices, 20 scenarios and $M = 0.15$. 109

5.4 Average solution times (in seconds) on random graphs with 40 vertices an edge density of 0.5, and $M = 0.15$. 109

LIST OF FIGURES

Figure

3.1 Average optimal clique sizes obtained by solving problem (3.12) at varying risk tolerances α using graphs with 150 vertices and 50 scenarios ($d = 0.50$). 79

3.2 Comparative optimal clique sizes at ranging risk tolerances α for problems (3.12) and (3.19) using a single graph with 50 vertices and 50 scenarios. . 81

LIST OF ALGORITHMS

Algorithm

2.1 Scenario decomposition algorithm 27

2.2 Solution method for sub-problem (2.4) 28

3.1 Graph-based branch-and-bound method for R-MWSproblems 64

4.1 Graph-based branch-and-bound method for R-MWK 90

5.1 First stage graph-based branch-and-bound method 104

5.2 Second stage graph-based branch-and-bound method 106

CHAPTER 1
RISK AVERSE DECISION MAKING UNDER UNCERTAINTY

1.1 Introduction

Classical methods for measuring "risk" in decision making problems under uncertainty were mostly application-driven, or ad-hoc. While such an approach admits quantification of risk preferences as necessitated by application specific requirements, it may lead to situations where the constructed risk measure lacks properties that are generally necessary for adequate risk management. A notorious example of this kind is served by a metric that is known in financial literature as the Value-at-Risk (VaR) measure, which is widely considered as a *de-facto* standard for measuring risk in the banking industry. Mathematically, VaR is defined as the α-quantile of the loss distribution at a confidence level $\alpha \in (0, 1)$:

$$\mathrm{VaR}_\alpha(X) = \inf\{\eta \mid \mathsf{P}[X \leq \eta] \geq \alpha\}, \tag{1.1}$$

and is generally *non-convex* in random outcome X, thereby not admitting proper reduction of risk via diversification which constitutes a fundamental principle in risk management practice.

Owing to a large degree to the failings of VaR, recent advances in risk theory pioneered by Artzner et al. [4] and Delbaen [18] spawned an axiomatic approach to the construction of risk measures by postulating desirable properties that a "good" risk measures should possess. Below we discuss the definitions of *coherent* and *convex* risk measures and describe their representation in context of mathematical programming

models.

1.2 Coherent and convex measures of risk

In this section we present a class of risk measures that encompass many popular instances in risk management literature. A risk measure $\rho(X)$ over a random outcome X from probability space $(\Omega, \mathcal{F}, \mathsf{P})$ is generally defined as a mapping $\rho : \mathcal{X} \mapsto \mathbb{R}$, where $\mathcal{X}$ is the space of bounded $\mathcal{F}$ - measurable functions $X : \Omega \mapsto \mathbb{R}$. In what follows, it is assumed that X represents a cost or a loss, whereby larger realizations are considered to induce higher risk levels. To be of practical use, however, the current definition of risk measure ρ is normally augmented using additional properties that makes its utilization meaningful in a specific application.

Artzner et al. [4] and Delbaen [18] proposed the following four axioms as the desirable characteristics that a *coherent risk measure* should possess:

(A0) *regularity:* ρ is proper and lower semicontinuous (l.s.c.) on $\mathcal{X}$

(A1) *monotonicity:* $\rho(X) \leq \rho(Y)$ for all $X, Y \in \mathcal{X}$ such that $X \leq Y$

(A2) *subadditivity:* $\rho(X + Y) \leq \rho(X) + \rho(Y)$ for all $X, Y \in \mathcal{X}$

(A3) *positive homogeneity:* $\rho(\lambda X) = \lambda \rho(X)$ for all $X \in \mathcal{X}$ and $\lambda > 0$

(A4) *translation invariance:* $\rho(X + a) = \rho(X) + a$ for all $X \in \mathcal{X}$ and $a \in \mathbb{R}$

An intuitive interpretation of the above axioms is as follows. Axiom (A1) guarantees that lower losses yield lower risk. The sub-additivity axiom (A2) is important in the context of *risk reduction via diversification*. It is also of fundamental significance from the optimization viewpoint, since together with the positive homogeneity axiom

(A3), it yields the all-important convexity property:

$$\rho(\lambda X + (1-\lambda)Y) \leq \lambda\rho(X) + (1-\lambda)\rho(Y) \quad \text{for all} \quad X, Y \in \mathcal{X}, \;\; \lambda \in [0,1]$$

The positive homogeneity property (A3) postulates that losses and risk scale correspondingly. Axiom (A4) allows for eliminating risk of an uncertain loss profile X by adding a deterministic hedge, $\rho(X - \rho(X)) = 0$.

Since being proposed in [4, 18], the axiomatic approach to defining risk measures has been widely adopted in recent literature, and a number of risk functionals tailored to specific preferences emerged thereafter (see, e.g., [28, 54]). In particular, it was argued that the positive homogeneity property (A3) may be omitted in many situations; the corresponding risk measures that satisfy (A1), (A2), and (A4) are called *convex measures of risk* [46].

Observe that the above axiomatic framework of risk measures is not constructive in the sense that it does not provide a functional form of coherent/convex risk measures. Moreover, the ability to employ them in an optimization context heavily depends on the availability of a functional representation, typically in the formalism of convex analysis, that is conducive to implementation in a mathematical programming models. Our interest in these two classes of risk measures stems from the following *infimal convolution* representation that facilitates their use in mathematical programming problems.

Theorem 1.1. [26, 59] *Function $\rho(X)$ is a proper coherent (resp., convex) measure of risk if and only if it can be represented by the following infimal convolution of a*

lower semicontinuous function $\phi : \mathcal{X} \mapsto \mathbb{R}$ *such that* $\phi(\eta) > \eta$ *for all real* $\eta \neq 0$*, and which satisfies (A1)–(A3) (resp., (A1)–(A2)):*

$$\rho(X) = \inf_{\eta} \ \eta + \phi(X - \eta). \tag{1.2}$$

Moreover, the infimum in (1.2) *is attained for all* X*, so* $\inf_\eta$ *may be replaced by* $\min_{\eta \in \mathbb{R}}$.

Representation (1.2) can be used for construction of coherent (convex) risk measures through an appropriate choice of function ϕ. The present work pertains to a special form of $\phi(X)$ in (1.1) that can be derived from considerations involving utility functions and certainty equivalents. In fact, much emphasis has been placed on developing risk-based decision making models that conform to the notions defined by the utility theory of von Neumann and Morgenstern [60]. Namely, if a decision makers preferences "$\succeq$" among outcomes (e.g. if outcome X is preferred to Y, then $X \succeq Y$) are satisfied by a system of axioms (completeness, transitivity, continuity, independence), then there exists a utility function $u : \mathbb{R} \to \mathbb{R}$ such that $\mathbb{E}[u(X)] \geq \mathbb{E}[u(Y)]$ for $X \succeq Y$. A decision-maker's preferences are said to be risk averse if function u is non-decreasing and concave, which can be extended to postulate the concept of second-order stochastic dominance (SSD) if the same a relationship hold true for all such functions u. Namely, a random outcome X is said to dominate outcome Y by the second order stochastic dominance if

$$\int_{-\infty}^{t} F_X(t)dt \leq \int_{-\infty}^{t} F_Y(t)dt \quad \text{for all } t \in \mathbb{R}.$$

Due to the fact that coherent risk measures generally do not comply with SSD, a framework that is consistent with this property is of interest. By means of imposing SSD consistency to risk measure $\rho(X)$, Krokhmal [26] showed that an analog of (1.2) can be constructed by replacing axiom (A1) with

(A1') *SSD isotonicity:* $\rho(X) \leq \rho(Y)$ for all $X, Y \in \mathcal{X}$ such that $(-X) \succeq_{SSD} (-Y)$,

where the resulting risk measure obtained from solving the optimization problem in the following theorem is both coherent and SSD compliant.

Theorem 1.2. **[26]** *Let function $\rho(X)$ satisfy axioms (A1'),(A2), and (A3), and be a l.s.c. function such that $\phi(\eta) > \eta$ for all real $\eta \neq 0$. Then the optimal solution value of stochastic programming problem*

$$\rho(X) = \inf_{\eta} \ \eta + \phi(X - \eta) \tag{1.3}$$

exists and is a proper function that satisfies (A1'),(A2), (A3), and (A4).

Let $v(\cdot)$ be the *deutility* function that quantifies dissatisfaction with loss X (if $-X$ can be regarded as payoff or reward and $u(\cdot)$ is Bernoulli utility function, then $v(t) = -u(-t)$). Expression $\mathrm{CE}(X) = v^{-1}\big(\mathsf{E}v(X)\big)$ represents the *certainty equivalent* (CE) of loss X, i.e., such a deterministic loss that a rational decision maker with deutility function v (or, equivalently, utility function $u(t) = -v(-t)$) would be indifferent between stochastic loss profile X and $\mathrm{CE}(X)$. Then, by taking $\phi(X) = (1-\alpha)^{-1}\mathrm{CE}(X)$, we obtain the following expression for (1.2):

$$\rho(X) = \min_{\eta} \ \eta + \frac{1}{1-\alpha} v^{-1}\big(\mathsf{E}v(X - \eta)\big). \tag{1.4}$$

Expression (1.4) admits a two-stage decision process interpretation. Faced with a future uncertain loss X, assume that the decision-maker can allocate amount η of resources to cover this loss (i.e., take a deterministic loss η) now, and then deal with an uncertain loss $X-\eta$. In this sense, (1.4) is a problem of minimization of resources required to cover loss X, where $1/(1-\alpha) > 1$ is a penalty factor (see a detailed discussion of representation (1.4) and related aspects in [58]).

The conditions on v that guarantee convexity of $\mathrm{CE}(X) = v^{-1}\big(\mathsf{E}v(X)\big)$, and correspondingly of $\phi(X)$, can be found in [11]. Namely, v should be thrice continuously differentiable and $v'(t)/v''(t)$ be convex. In what follows, however, we implicitly assume convexity of the certainty equivalent. Particularly, function $v(t)$ is continuously differentiable, increasing, convex, and, moreover, the certainty equivalent $v^{-1}\big(\mathsf{E}v(X)\big)$ is convex in X.

Several practically interesting risk measure families can be obtained from (1.4) if one requires that $v(t) = 0$ for $t \leq 0$ (i.e., dissatisfaction vanishes for negative losses). To highlight this fact, we rewrite (1.4) as

$$\rho(X) = \min_{\eta} \ \eta + \frac{1}{1-\alpha} v^{-1}\mathsf{E}v\left(X-\eta\right)_+, \tag{1.5}$$

where $X_+ = \max\{0, X\}$ and we adopt an operator-like notation for v and v^{-1}, e.g., $v(X-\eta)_+ \equiv v\big((X-\eta)_+\big)$. Note that in conjunction with the properties of v^{-1}, it signifies that $\phi(X) = (1-\alpha)^{-1}v^{-1}\mathsf{E}v(X)$ in (1.5) satisfies the conditions of Theorem 1.1. Then, the following risk measures can be obtained from representation (1.5):

(i) If $v(t) = t$, then (1.5) defines the well-known Conditional Value-at-Risk measure (CVaR) [44, 45]:

$$\mathsf{CVaR}_{\alpha}(X) = \min_{\eta} \ \eta + (1-\alpha)^{-1}\mathsf{E}(X-\eta)_{+} \tag{1.6}$$

(ii) If $v(t) = t^p$ for $t \geq 0$ and $p > 1$, then representation (1.5) yields a two-parametric family of *higher-moment coherent risk measures* (HMCR) [26]:

$$\mathsf{HMCR}_{p,\alpha}(X) = \min_{\eta} \ \eta + (1-\alpha)^{-1}\big\|(X-\eta)_{+}\big\|_{p} \tag{1.7}$$

(iii) If $v(t) = \lambda^t - 1$, $\lambda > 1$, then one obtains the family of *log-exponential convex measures of risk* (LogExpCR) [57]:

$$\mathsf{LogExpCR}_{\lambda,\alpha}(X) = \min_{\eta} \ \eta + (1-\alpha)^{-1}\log_{\lambda}\mathsf{E}\lambda^{(X-\eta)_{+}} \tag{1.8}$$

Unlike the CVaR and HMCR measures that are coherent, the LogExpCR measure is convex but not coherent as it does not satisfy the positive homogeneity axiom (A3).

A common property of measures (i)–(iii) is that they are "tail" risk measures in the sense that they quantify losses in the tail $\{X : X \geq \eta_{\alpha}^{*}(X)\}$ of the loss distribution, where the location of the "tail cutoff" point $\eta_{\alpha}^{*}(X)$, which is a minimizer in (1.5), is governed by the value of the parameter α (see [26, 59]):

$$\lim_{\alpha \to 1} \eta_{\alpha}^{*}(X) = \operatorname{ess\,sup} X.$$

Perhaps one of the most widely used coherent measures of risk is defined by (i), which represents, roughly speaking, the conditional expectation of losses that may occur in the $(1-\alpha)\cdot 100\%$ of worst realizations of X. Notice that the CVaR measure is

a special case of representation (ii) when $p = 1$, $\mathsf{HMCR}_{1,\alpha}(X) = \mathsf{CVaR}_{\alpha}(X)$. When $p > 1$, HMCR measures quantify losses via higher tail moments $\|(X-\eta)_+\|_p$. These have been shown to be better suited for applications under the presence of "heavy tailed" loss distributions [26]. Likewise, the log-exponential family of convex measures of risk is designed for dealing with heavy-tailed loss distributions, and particularly extreme and catastrophic losses. Another interesting common property of risk measures (i)–(iii) is represented by the second order stochastic dominance isotonicity axiom (A1'). Recall that payoff profile X dominates Y with respect to SSD if the relation $\mathsf{E}[u(X)] \geq \mathsf{E}[u(Y)]$ holds for all non-decreasing concave utility functions u, or, in other words, if every rational risk-averse decision maker prefers X over Y. It is important to note that coherent (convex) measures of risk are generally not SSD-isotonic [28].

Below we discuss the implementation of the described risk measures in mathematical programming models.

1.2.1 Implementation in stochastic programming

Assume that loss X is a function of the decision variable $\mathbf{x}$, $X = X(\mathbf{x}, \omega)$, where event $\omega \in \Omega$. Then, for a compact and convex feasible set $C \subset \mathbb{R}^n$, consider a stochastic programming problem with a risk constraint in the form

$$\min \big\{ g(\mathbf{x}) : \rho\big(X(\mathbf{x}, \omega)\big) \leq h(\mathbf{x}),\ \mathbf{x} \in C \big\}. \tag{1.9}$$

Theorem 1.3. *Consider problem* (1.9) *where set* $C \subset \mathbb{R}^n$ *is compact and convex, and functions* $g(\mathbf{x})$ *and* $h(\mathbf{x})$ *are convex and concave on* $\mathcal{C}$*, respectively. If, further, the cost or loss function* $X(\mathbf{x}, \omega)$ *is convex in* $\mathbf{x}$*, and* ρ *is a coherent or convex measure*

of risk possessing representation (1.2)*, then problem* (1.9) *is equivalent to*

$$\min\big\{g(\mathbf{x}) : \eta + \phi\big(X(\mathbf{x},\omega) - \eta\big) \le h(\mathbf{x}),\ (\mathbf{x},\eta) \in C \times \mathbb{R}\big\}, \tag{1.10}$$

in the sense that (1.9) *and* (1.10) *achieve minima at the same values of the decision variable* $\mathbf{x}$ *and their optimal objective values coincide. Further, if the risk constraint in* (1.9) *is binding at optimality,* $(\mathbf{x}^*, \eta^*)$ *achieves the minimum of* (1.10) *if and only if* $\mathbf{x}^*$ *is an optimal solution of* (1.9) *and*

$$\eta^* \in \arg\min_\eta\ \ \eta + \phi\big(X(\mathbf{x}^*,\omega) - \eta\big).$$

Proof. See [26].

Remark. Note that risk minimization problem

$$\min\big\{\rho\big(X(\mathbf{x},\omega)\big) : \mathbf{x} \in C\big\} \tag{1.11}$$

is obtained from (1.9) by introduction of dummy variable x_{n+1} and letting $g(\mathbf{x}) = h(\mathbf{x}) = x_{n+1}$.

Let function ϕ in (1.10) have the form

$$\phi(X) = (1-\alpha)^{-1} v^{-1} \mathsf{E} v(X_+).$$

Given a discrete set of scenarios $\{\omega_1, \ldots, \omega_N\} = \Omega$ that induce outcomes $X(\mathbf{x},\omega_1), \ldots, X(\mathbf{x},\omega_N)$ in decision vector $\mathbf{x}$, it is easy to see that the risk constraint in (1.10) can

be represented via the following set of inequalities:

$$\eta + (1-\alpha)^{-1} w_0 \leq h(\mathbf{x}) \tag{1.12a}$$

$$w_0 \geq v^{-1}\left(\sum_{j\in\mathcal{N}} \pi_j v(w_j)\right) \tag{1.12b}$$

$$w_j \geq X(\mathbf{x}, \omega_j) - \eta, \quad j \in \mathcal{N} \tag{1.12c}$$

$$w_j \geq 0, \quad j \in \mathcal{N}, \tag{1.12d}$$

where $\mathcal{N}$ denotes the set of scenario indices, $\mathcal{N} = \{1, \ldots, N\}$, and corresponding scenario probabilities are represented as $\mathsf{P}(\omega_j) = \pi_j > 0$, such that $\pi_1 + \cdots + \pi_N = 1$.

In the above discussion it was shown that several risk measure types emerge from different choices of the deutility function v. Here we note that the corresponding representations of constraint (1.12b) in context of HMCR and LogExpCR measures lead to sufficiently "nice", i.e., convex, mathematical programming models. For HMCR measures inequality (1.12b) becomes

$$w_0 \geq \left(\sum_{j\in\mathcal{N}} \pi_j w_j^p\right)^{1/p}, \tag{1.13}$$

which is equivalent to a standard p-order cone under affine scaling. Noteworthy instances of (1.13) for which readily available mathematical programming solution methods exist include $p = 1, 2$. In the particular case when $p = 1$ (which corresponds to CVaR), the problem reduces to a linear programming (LP) model. For instances when $p = 2$, a second-order cone programming (SOCP) model transpires that is efficiently solvable using long-step self-dual interior point methods. However, no similarly efficient solution methods exist for solving p-order conic constrained problems when $p \in (1, 2) \cup (2, \infty)$ due to the fact that the p-cone is not self-dual in this

case. Additional mathematical programming considerations for such instances will be discussed in Chapter 2. Lastly, the following exponential inequality corresponds to constraint (1.12b) when ρ is a LogExpCR measure:

$$w_0 \geq \ln \sum_{j \in \mathcal{N}} \pi_j e^{w_j}, \tag{1.14}$$

which is also convex and allows for the resulting optimization problem be solved using appropriate (e.g., interior point) methods.

CHAPTER 2
A SCENARIO DECOMPOSITION APPROACH FOR STOCHASTIC OPTIMIZATION PROBLEMS WITH A CLASS OF DOWNSIDE RISK MEASURES

2.1 Introduction

Quantification of uncertainties and risk via axiomatically defined statistical functionals, such as the coherent measures of risk, has become a widely accepted practice in stochastic optimization and decision making under uncertainty [28, 54]. Many of such risk measures admit effective utilization in "scenario-based" formulations of stochastic programming models, i.e., the stochastic optimization problems where the random parameters are assumed to have known distribution over a finite support that is commonly called the *scenario set*. A typical instance of such a problem can be written as

$$\min_{\mathbf{x}\in C} \ \rho(X(\mathbf{x},\omega)), \tag{2.1}$$

where risk measure ρ, loss $X(\mathbf{x},\omega)$ and decision vector $\mathbf{x} \in C \subset \mathbb{R}^n$ are defined as previously. In many practical applications accurate approximations of uncertainties may, however, require very large scenario sets ($N \gg 1$), thus potentially leading to substantial computational difficulties.

In this study, we propose an efficient algorithm for solving large-scale stochastic optimization problems involving the risk measures described in Chapter 1. The forthcoming *scenario decomposition algorithm* exploits the special structure of the feasible set induced by the respective risk measures as well as the properties common

to the considered class of risk functionals. As an illustrative example of the general approach, we consider stochastic optimization problems with HMCRs and also apply the proposed method to solve problems with LogExpCR measures.

Perhaps, the most frequently implemented risk measure in problems of type (2.1) is the well known CVaR [44, 45] defined by (1.6). When X is piece-wise linear in $\mathbf{x}$ and set C is polyhedral, problem (2.1) with CVaR objective or constraints reduces to a LP problem. Several recent studies addressed solution efficiency of LPs with CVaR objectives or constraints in the case when the number of scenarios is large. Lim et al. [34] noted that (2.1) in this case may be viewed as a nondifferentiable optimization problem and implemented a two-phase solution approach to solve large-scale instances. In the first phase, they exploit descent-based optimization techniques by circumventing nondifferentiable points by perturbing the solution to differentiable solutions in the "relative" neighborhood. A second phase employs a deflecting subgradient search direction with a step size established by an adequate target value. They further extend this approach by implementing a third phase, resorting to the simplex algorithm after achieving convergence by employing an advanced crash-basis dependent on solutions obtained from the first two phases.

Künzi and Máyer [33] developed a solution technique for problem (2.1) with measure ρ chosen as CVaR that utilized a specialized L-shaped method by reformulating it as a two-stage recourse stochastic optimization problem. However, Subramanian and Huang noted in [49] that the problem structure does not naturally conform to the characteristics of a two-stage stochastic program and introduced a polyhe-

dral reformulation of the CVaR constraint with a statistics based CVaR estimator to solve a closely related version. In a followup study [50], they retained Value-at-Risk (VaR) and CVaR as unknown variables in the CVaR constraints, enabling a more efficient decomposition algorithm, as opposed to [24], where the problem was solved as a canonical integrated chance constraint problem with preceding estimates of VaR. Espinoza and Moreno [20] proposed a different solution method for problems (2.1) with the CVaR measure, where iterative generation of aggregated scenario constraints were used to form smaller relaxation problems, whose optimal outcomes were then used to directly evaluate the respective upper bound on the objective of the original problem.

In this chapter, we develop a generalized scenario decomposition solution framework for solving stochastic optimization problems with centrality equivalent-based risk measures (1.4) by utilizing related principals to the ones described by Espinoza and Moreno [20]. The subsequent sections are organized as follows. In Section 2.2 we propose the scenario decomposition algorithm. Section 2.3 furnishes experimental portfolio optimization benchmark models using HMCR and LogExpCR measures that were used for algorithmic testing. Lastly, experimental results that demonstrate the effectiveness of the developed technique when solving problems with large-scale data sets are presented in Section 2.4.

2.2 Scenario decompositon algorithm

In this section we propose an efficient scenario decomposition algorithm for solving large-scale mathematical programming models using certainty equivalent-based measures of risk. As previously discussed, significant emphasis has been placed on solution techniques pertaining to optimization models using CVaR measures, while little has been done to address computational challenges posed by large-scale scenario sets for problems using the risk functionals in Section 1.2 in a general sense. Consequently, we resort to describing the algorithmic procedures using the general definition of certainty equivalent measures of risk (1.5), and demonstrate its applicability for problems using HMCR measures and LogExpCR measures in order to emphasize its conformity to any appropriate selection of function $v(\cdot)$ consistent with the description in Section 1.2.

The ensuing scenario decomposition algorithm relies on solving a series of relaxation problems composing of linear combinations of scenarios that are systematically decomposed until convergence with the "true" optimal solution is obtained. Naturally, the core assumption behind such a scheme is that sequential solutions of smaller relaxation problems can be achieved within shorter computation times. By virtue of Section 1.2, when the distribution of loss function $X(\mathbf{x}, \omega)$ has a finite support (scenario set) $\Omega = \{\omega_1, \ldots, \omega_N\}$ with probabilities $\mathsf{P}(\omega_j) = \pi_j > 0$, the stochastic

programming problem with risk constraint (1.10) admits the form

$$\min \quad g(\mathbf{x}) \tag{2.2a}$$

$$\text{s. t.} \quad \mathbf{x} \in C \tag{2.2b}$$

$$\eta + (1-\alpha)^{-1} w_0 \leq h(\mathbf{x}) \tag{2.2c}$$

$$w_0 \geq v^{-1}\left(\sum_{j \in \mathcal{N}} \pi_j v(w_j)\right) \tag{2.2d}$$

$$w_j \geq X(\mathbf{x}, \omega_j) - \eta, \quad j \in \mathcal{N} \tag{2.2e}$$

$$w_j \geq 0, \quad j \in \mathcal{N}, \tag{2.2f}$$

where we denoted $\mathcal{N} = \{1, \ldots, N\}$. If we assume that function $g(\mathbf{x})$ and feasible set C are "nice" in the sense that problem $\min\{g(\mathbf{x}) : \mathbf{x} \in C\}$ admits efficient solution algorithms, then formulation (2.2) may present challenges that are two-fold. First, constraint (2.2d) may need a specialized solution approach on account of the chosen function v, especially in the case of large N. Also, when N is large, numerical difficulties may be associated with handling a corresponding number of constraints (2.2e)–(2.2f). In this work, we present an iterative technique for dealing with a large number of scenario-based constraints (2.2e)–(2.2f).

Since the original problem (2.2) with many constraints of the form (2.2e)–(2.2f) may be hard solve, a relaxation of (2.2) can be constructed by aggregating some of the scenario constraints. Let $\{\mathcal{S}_k : k \in \mathcal{K}\}$ denote a *partition* of the set $\mathcal{N}$ of scenario indices (which we will simply call scenario set), i.e.,

$$\bigcup_{k \in \mathcal{K}} \mathcal{S}_k = \mathcal{N}, \quad \mathcal{S}_i \cap \mathcal{S}_j = \emptyset \quad \text{for all} \quad i, j \in \mathcal{K},\ i \neq j.$$

Aggregation of scenario constraints by adding inequalities (2.2e) within sets $\mathcal{S}_k$ produces the following *master problem*:

$$\min \quad g(\mathbf{x}) \tag{2.3a}$$

$$\text{s. t.} \quad \mathbf{x} \in C \tag{2.3b}$$

$$\eta + (1-\alpha)^{-1} w_0 \leq h(\mathbf{x}) \tag{2.3c}$$

$$w_0 \geq v^{-1}\left(\sum_{j \in \mathcal{N}} \pi_j v(w_j)\right) \tag{2.3d}$$

$$\sum_{j \in \mathcal{S}_k} w_j \geq \sum_{j \in \mathcal{S}_k} X(\mathbf{x}, \omega_j) - |\mathcal{S}_k|\eta, \quad k \in \mathcal{K} \tag{2.3e}$$

$$w_j \geq 0, \quad j \in \mathcal{N}. \tag{2.3f}$$

Clearly, any feasible solution of (2.2) is also feasible for (2.3), and the optimal value of (2.3) represents a lower bound on that of (2.2). Since the relaxed problem contains fewer scenario-based constraints (2.3e), it is potentially easier to solve. It would be of interest then to determine the conditions under which an optimal solution of (2.3) is also optimal for the original problem (2.2). Assuming that $\mathbf{x}^*$ is an optimal solution of (2.3), consider *subproblem* problem

$$\min \quad \eta + (1-\alpha)^{-1} w_0 \tag{2.4a}$$

$$\text{s. t.} \quad w_0 \geq v^{-1}\left(\sum_{j \in \mathcal{N}} \pi_j v(w_j)\right) \tag{2.4b}$$

$$w_j \geq X(\mathbf{x}^*, \omega_j) - \eta, \quad j \in \mathcal{N} \tag{2.4c}$$

$$w_j \geq 0, \quad j \in \mathcal{N}. \tag{2.4d}$$

Proposition 2.1. *Consider problem* (2.2) *and its relaxation* (2.3) *obtained by aggregating scenario constraints* (2.2e) *over sets* $\mathcal{S}_k$, $k \in \mathcal{K}$, *that form a partition of*

$\mathcal{N} = \{1, \ldots, N\}$. *Assuming that* (2.2) *is feasible, consider problem* (2.4) *where* $\mathbf{x}^*$ *is an optimal solution of relaxation* (2.3). *Let* $(\eta^{**}, \mathbf{w}^{**})$ *be an optimal solution of* (2.4). *If the optimal value of* (2.4) *satisfies condition*

$$\eta^{**} + (1-\alpha)^{-1} w_0^{**} \leq h(\mathbf{x}^*), \tag{2.5}$$

then $(\mathbf{x}^*, \eta^{**}, \mathbf{w}^{**})$ *is an optimal solution of the original problem* (2.2).

Proof. Let $\mathbf{x}^\circ$ be an optimal solution of (2.2). Obviously, one has $g(\mathbf{x}^*) \leq g(\mathbf{x}^\circ)$. The statement of the proposition then follows immediately by observing that inequality (2.5) guarantees the triple $(\mathbf{x}^*, \eta^{**}, \mathbf{w}^{**})$ to be feasible for problem (2.2). □

The statement of Proposition 2.1 allows one to solve the original problem (2.2) by constructing an appropriate partition of $\mathcal{N}$ and solving the corresponding master problem (2.3). Below we outline an iterative procedure that accomplishes this goal.

Step (0): The algorithm is initialized by including all scenarios in a single partition, $\mathcal{K} = \{0\}$, $\mathcal{S}_0 = \{1, \ldots, N\}$.

Step (1): For a current partition $\{\mathcal{S}_k : k \in \mathcal{K}\}$, solve the master problem (2.3). If (2.3) is infeasible, then the original problem (2.2) is infeasible as well, and the algorithm terminates. Otherwise, let $\mathbf{x}^*$ be an optimal solution of the master (2.3).

Step (2): Given a solution $\mathbf{x}^*$ of the master, solve problem (2.4), and let $(\eta^{**}, \mathbf{w}^{**})$ denote the corresponding optimal solution. If condition (2.5) is satisfied, algorithm terminates with $(\mathbf{x}^*, \eta^{**}, \mathbf{w}^{**})$ being an optimal solution of (2.2) due to Proposition 2.1. If however, condition (2.5) is violated,

$$\eta^{**} + (1-\alpha)^{-1} w_0^{**} > h(\mathbf{x}^*),$$

the algorithm proceeds to Step 3 to update the partition.

Step (3): Determine the set of scenario-based constraints in (2.4) that, for a given solution of the master $\mathbf{x}^*$, are binding at optimality:

$$\mathcal{J} = \{j \in \mathcal{N} : w_j^{**} = X(\mathbf{x}^*, \omega_j) - \eta^{**} > 0\}. \tag{2.6}$$

Then, the elements of $\mathcal{J}$ are removed from the existing sets $\mathcal{S}_k$:

$$\mathcal{S}_k = \mathcal{S}_k \setminus \mathcal{J}, \quad k \in \mathcal{K},$$

and added to the partition as single-element sets:

$$\{\mathcal{S}_0, \ldots, \mathcal{S}_K\} \cup \{S_{K+1}, \ldots, \mathcal{S}_{K+|\mathcal{J}|}\},$$

$$\text{where } \mathcal{S}_{K+i} = \{j_i\} \text{ for each } j_i \in \mathcal{J}, \ i = 1, \ldots, |\mathcal{J}|,$$

and the algorithm proceeds to Step 1.

Theorem 2.2. *Observe that at any given step of the described methodology the partitions are structured such that* $|\mathcal{S}_0| \geq 1$, *while* $|\mathcal{S}_k| = 1$ *for* $k = 1, \ldots, K$. *Then, the scenario decomposition algorithm for problem* (2.2) *terminates after at most* N *iterations.*

Proof. Let us show that during an iteration of the algorithm the size of the partition of the set $\mathcal{N}$ of scenarios increases by at least one.

Let $\{\mathcal{S}_k : k \in \mathcal{K}\}$ be the current partition of $\mathcal{N}$, $(\mathbf{x}^*, \eta^*, \mathbf{w}^*)$ be the corresponding optimal solution of (2.3), and $(\eta^{**}, \mathbf{w}^{**})$ be an optimal solution of (2.4) for the given $\mathbf{x}^*$, such that the stopping condition (2.5) is not satisfied,

$$\eta^{**} + (1 - \alpha)^{-1} w_0^{**} > h(\mathbf{x}^*). \tag{2.7}$$

Let $\bar{\mathcal{S}}^*$ denote the set of constraints (2.4c) that are binding at optimality,

$$\bar{\mathcal{S}}^* = \big\{ j : w_j^{**} = X(\mathbf{x}^*, \omega_j) - \eta^{**} > 0,\ j \in \mathcal{N} \big\}.$$

Next, consider a problem obtained from (2.4) with a given $\mathbf{x}^*$ by aggregating the constraints (2.4c) that are non-binding at optimality:

$$\min \quad \eta + (1-\alpha)^{-1} w_0 \tag{2.8a}$$

$$\text{s. t.} \quad w_0 \geq v^{-1}\Big(\sum_{j \in S_0} \pi_j v(w_j) \Big) \tag{2.8b}$$

$$w_j \geq X(\mathbf{x}^*, \omega_j) - \eta, \quad j \in \bar{\mathcal{S}}^* \tag{2.8c}$$

$$\sum_{j \in \mathcal{S}^*} w_j \geq \sum_{j \in \mathcal{S}^*} X(\mathbf{x}^*, \omega_j) - |\mathcal{S}^*|\eta \tag{2.8d}$$

$$w_j \geq 0, \quad j \in \mathcal{N}, \tag{2.8e}$$

where $\mathcal{S}^* = \mathcal{N} \setminus \bar{\mathcal{S}}^*$. Obviously, an optimal solution $(\eta^{**}, \mathbf{w}^{**})$ of (2.4) will also be optimal for (2.8).

Next, observe that at any stage of the algorithm partition $\{\mathcal{S}_k : k \in \mathcal{K}\}$ is such that there exists at most one set with $|\mathcal{S}_k| > 1$, namely set $\mathcal{S}_0$, and the rest of the sets in the partition satisfy $|\mathcal{S}_k| = 1$, $k \neq 0$. Let us denote

$$\bar{\mathcal{S}}_0 = \mathcal{N} \setminus \mathcal{S}_0 = \bigcup_{k \in \mathcal{K} \setminus \{0\}} \mathcal{S}_k$$

Assume that $\bar{\mathcal{S}}^* \subseteq \bar{\mathcal{S}}_0$. By rewriting the master problem (2.3) as

$$\min \quad g(\mathbf{x}) \tag{2.9a}$$

$$\text{s. t.} \quad \mathbf{x} \in C, \tag{2.9b}$$

$$\eta + (1-\alpha)^{-1} w_0 \leq h(\mathbf{x}) \tag{2.9c}$$

$$w_0 \geq v^{-1}\left(\sum_{j \in \mathcal{N}} \pi_j v(w_j)\right) \tag{2.9d}$$

$$w_j \geq X(\mathbf{x}, \omega_j) - \eta, \quad j \in \bar{\mathcal{S}}_0 \tag{2.9e}$$

$$\sum_{j \in \mathcal{S}_0} w_j \geq \sum_{j \in \mathcal{S}_0} X(\mathbf{x}, \omega_j) - |\mathcal{S}_0|\eta \tag{2.9f}$$

$$w_j \geq 0, \quad j \in \mathcal{N}, \tag{2.9g}$$

we observe that the components $\eta^*, \mathbf{w}^*$ of its optimal solution are feasible for (2.8). Indeed, from (2.9e) one has that

$$w_j^* \geq X(\mathbf{x}^*, \omega_j) - \eta^*, \quad j \in \bar{\mathcal{S}}^*,$$

which satisfies (2.8c), and also

$$w_j^* \geq X(\mathbf{x}^*, \omega_j) - \eta^*, \quad j \in \bar{\mathcal{S}}_0 \setminus \bar{\mathcal{S}}^* = \mathcal{S}^* \setminus \mathcal{S}_0.$$

Adding the last inequalities yields

$$\sum_{j \in \mathcal{S}^* \setminus \mathcal{S}_0} w_j^* \geq \sum_{j \in \mathcal{S}^* \setminus \mathcal{S}_0} X(\mathbf{x}^*, \omega_j) - |\mathcal{S}^* \setminus \mathcal{S}_0|\eta^*,$$

which can then be aggregated with (2.9f) to produce

$$\sum_{j \in \mathcal{S}^*} w_j^* \geq \sum_{j \in \mathcal{S}^*} X(\mathbf{x}^*, \omega_j) - |\mathcal{S}^*|\eta^*,$$

verifying the feasibility of $(\eta^*, \mathbf{w}^*)$ for (2.8). Since (2.9c) has to hold for $(\mathbf{x}^*, \eta^*, \mathbf{w}^*)$, we obtain that

$$\eta^{**} + (1-\alpha)^{-1} w^{**} \leq \eta^* + (1-\alpha)^{-1} w^* \leq h(\mathbf{x}^*),$$

which furnishes a contradiction with (2.7). Therefore, one has to have $\bar{\mathcal{S}}_0 \subset \bar{\mathcal{S}}^*$ for (2.7) to hold, meaning that at least one additional scenario from $\bar{\mathcal{S}}^*$ will be added to the partition during Step 3 of the algorithm. It is easy to see that the number of iterations cannot exceed the number N of scenarios. □

2.2.1 Efficient solution method for problem (2.4)

Although formulation (2.4) may be solved using appropriate mathematical programming techniques, an efficient alternative solution method can be employed by noting that (2.4) is equivalent to

$$\min \ \eta + \frac{1}{1-\alpha} v^{-1} \left(\sum_{j \in \mathcal{N}} \pi_j v\big(X(\mathbf{x}^*, \omega_j) - \eta\big)_+ \right), \tag{2.10}$$

which is a mathematical programming implementation of representation (1.5) under a finite scenario model where realizations $X(\mathbf{x}^*, \omega_j)$ represent scenario losses corresponding to an optimal decision $\mathbf{x}^*$ in the master problem (2.3). An optimal value of η in (2.4) and (2.10) can be computed directly using its properties dictated by representation (1.5).

Namely, let $X_j = X(\mathbf{x}^*, \omega_j)$ represent the optimal loss in scenario j for problem (2.3), and let $X_{(m)}$ be the m-th smallest outcome among $X_1, \ldots, X_N$, such that

$$X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(N)}.$$

The following proposition enables a framework for evaluating η^{**} by establishing its limiting factors among the outcomes.

More precisely, an optimal η^{**} in problem (2.4) corresponding to the "cutoff" point withing the tail of the loss distribution at a given tolerance level $\alpha \in (0,1)$ must satisfy the following proposition:

Proposition 2.3. *Given an increasing convex function $v(\cdot)$ such that $v(t) = \mathcal{O}(t^{1+a})$ for $t \to 0+$ and some $a > 0$, an optimal η^{**} in problems (2.10) and (2.4), where $\alpha \in (0,1)$, can be obtained as a solution of equation*

$$\frac{\sum\limits_{j:X_j>\eta} \pi_j v'(X_j - \eta)}{v'\Big(v^{-1}\big(\sum_{j\in\mathcal{N}} \pi_j v(X_j-\eta)_+\big)\Big)} + \alpha - 1 = 0, \tag{2.11}$$

where v' denotes the derivative of v.

Proof. Given the underlying assumption that v is such that $\phi(X) = (1-\alpha)^{-1} v^{-1} \mathsf{E} v(X)$ is convex, the objective function of (2.10)

$$\Phi_X(\eta) = \eta + \phi(X-\eta) = \eta + \frac{1}{1-\alpha} v^{-1}\Big(\sum_{j\in\mathcal{N}} \pi_j v(X_j-\eta)_+\Big) \tag{2.12}$$

is convex on $\mathbb{R}$. Moreover, the condition $\phi(\eta) > \eta$ for $\eta \neq 0$ of Theorem 1.1 guarantees that the set of minimizers of $\Phi_X(\eta)$ is closed and convex in $\mathbb{R}$. Indeed, it is easy to see that $\Phi_X(\eta)$ is non-monotonic on $\mathbb{R}$: $\Phi_X(\eta) = \eta$ for $\eta \geq X_{(N)}$ and $\Phi_X(\eta) \sim -\frac{\alpha\eta}{1-\alpha}$ for $\eta \ll -1$.

Next, observe that while the assumed behavior of $v(t)$ at $t = 0+$ necessarily requires that $v(0) = 0$, this implied condition is not restrictive as one can always take $v(t) = \tilde{v}(t) - \tilde{v}(0)$ in (1.4) and (1.5) if $\tilde{v}(0) \neq 0$.

Given that $v(t)$ is increasing and continuously differentiable on $[0, \infty)$, its inverse $v^{-1}(t)$ exists for $t \geq 0$ and is also continuously differentiable. For (2.11) to hold, it suffices to show that function $\Phi_X(\eta)$ is continuously differentiable. Observe that under the assumptions on the properties of $v(\cdot)$, it is sufficient to show the continuous differentiability of $\sum_{j \in \mathcal{N}} \pi_j v(X_j - \eta)_+$ with respect to η. To this end, consider the right and left derivatives of function $v(X_j - \eta)_+$ at $\eta = X_j$, $j \in \mathcal{N}$:

$$\frac{d^-}{d\eta} v(X_j - \eta)_+ \bigg|_{\eta = X_j} = \lim_{\epsilon \to 0-} \frac{v(X_j - \eta + \epsilon)_+ - v(X_j - \eta)_+}{-\epsilon} = \lim_{\epsilon \to 0-} \frac{v(\epsilon) - v(0)}{-\epsilon} = 0,$$

where the last equality holds due to the assumption that $v(t) = \mathcal{O}(t^{1+a})$ at $t \to 0+$. In the case of the right-side derivative at $\eta = X_j$, one trivially has

$$\frac{d^+}{d\eta} v(X_j - \eta)_+ \bigg|_{\eta = X_j} = \lim_{\epsilon \to 0+} \frac{v(X_j - \eta - \epsilon)_+ - v(X_j - \eta)_+}{\epsilon} = \lim_{\epsilon \to 0+} \frac{v(0) - v(0)}{\epsilon} = 0.$$

Using the established continuous differentiability of $\Phi_X(\eta)$, expression (2.11) is obtained directly from the first order conditions. □

Remark. Note that conditions of Proposition 2.3 imply that the optimal η^{**} in (2.4) and (2.10) is unique. This is in stark contrast with the corresponding well-known result for the Conditional Value-at-Risk measure $\mathsf{CVaR}_\alpha(X)$, where the set of optimal solutions of (1.6) in a finite scenario case generally represents a non-empty segment of real line, the left endpoint of which is equal to $\mathsf{VaR}_\alpha(X)$. Note, however, that the case of CVaR, where $v(t) = t$, does not satisfy the condition of Proposition 2.3 on the behavior of $v(t)$ at $t = 0+$.

Remark. Expression (2.11) allows for obtaining optimality conditions for η in the special cases corresponding to the families of HMCR and LogExpCR measures considered in this work. Particularly, in the HMCR case one can take $v(t) = t^p$ for $t \geq 0$, whereby optimality condition (2.11) reduces to

$$\frac{\left(\sum_{j\in\mathcal{N}} \pi_j (X_j - \eta)_+^{p-1}\right)^{1/(p-1)}}{\left(\sum_{j\in\mathcal{N}} \pi_j (X_j - \eta)_+^{p}\right)^{1/p}} - (1-\alpha)^{1/(p-1)} = 0, \quad p > 1. \tag{2.13}$$

Similarly, when $v(t) = e^t - 1$, the corresponding condition for LogExpCR measure is given by

$$\frac{\sum_{j:X_j>\eta} \pi_j e^{X_j - \eta}}{\sum_{j\in\mathcal{N}} \pi_j e^{(X_j-\eta)_+}} + \alpha - 1 = 0. \tag{2.14}$$

Recall that the above scenario decomposition algorithm uses subproblem (2.4) for determining the optimal value of η^{**}, as well as for identifying (during Step 3) the set $\mathcal{J}$ of scenarios that are binding at optimality, i.e., for which $X(\mathbf{x}^*, \omega_j) - \eta^{**} > 0$. This can be accomplished with the help of the derived optimality conditions (2.11) as follows.

Step (i): Compute values $X_j = X(\mathbf{x}^*, \omega_j)$, where $\mathbf{x}^*$ is an optimal solution of (2.3), and sort them in ascending order: $X_{(1)} \leq \ldots \leq X_{(N)}$.

Step (ii): For $m = N, N-1, \ldots, 1$, compute values T_m as

$$\begin{aligned} &T_N = 1, \\ &T_m = \frac{\sum_{j=m+1}^{N} \pi_j v'\big(X_{(j)} - X_{(m)}\big)}{v'\left(v^{-1}\left(\sum_{j=m+1}^{N} \pi_j v\big(X_{(j)} - X_{(m)}\big)\right)\right)} - \alpha + 1, \quad m = N-1, \ldots, 1, \end{aligned} \tag{2.15}$$

until m^* is found such that

$$T_{m^*} \leq 0, \; T_{m^*+1} > 0.$$

Step (iii): If $T_{m^*} = 0$, then the solution η^{**} of (2.11) is equal to $X_{(m^*)}$. Otherwise, η^{**} satisfies

$$\eta^{**} \in \big(X_{(m^*)}, X_{(m^*+1)}\big),$$

and its value can be found by solving (2.11) using an appropriate numerical procedure, such as Newton's method. The set $\mathcal{J}$ in (2.6) is then obtained as

$$\mathcal{J} = \{j : X_j = X_{(k)},\ k = m^* + 1, \ldots, N\}.$$

Proposition 2.4. *Given an optimal solution* $\mathbf{x}^*$ *of the master problem* (2.3)*, the algorithm described in steps (i)–(iii) yields the value* η^{**} *and the set* $\mathcal{J}$ *to be used during steps 2 and 3 of the scenario decomposition algorithm.*

Proof. First, observe that the optimal solution η^{**} of (2.4) and (2.11) satisfies

$$\eta^{**} \leq X_{(N)}.$$

Indeed, assume to the contrary that $\eta^{**} = X_{(N)} + \epsilon$ for some $\epsilon > 0$. The optimal value of (2.4) and (1.5) is then equal to $X_{(N)} + \epsilon$, and can be improved by selecting, e.g., $\epsilon = \epsilon/2$.

Next, observe that the quantities T_m (2.15) are equal to the values of the left-hand side in the optimality condition (2.11) computed at $\eta = X_{(m)}$, or, in other words, the values of the derivative of function $\Phi_X(\eta)$ in (2.12) at $\eta = X_{(m)}$. The value of $T_N = 1$ follows directly from the fact that $\Phi_X(\eta) = \eta$ for $\eta \geq X_{(N)}$. Hence, step (ii) consists in determining an interval on which the derivative $\Phi'_X(\eta)$ changes sign. Then, the value of η^{**} such that $\Phi'_X(\eta^{**}) = 0$ is obtained during step (iii),

and the set $\mathcal{J}$ in (2.6) is constructed as the set of scenario indices corresponding to $X_{(m^*+1)}, X_{(m^*+2)}, \ldots, X_{(N)}$.

We conclude by noting that it is not necessary to prove that there always exists $m^* \in \{1, \ldots, N-1\}$ such that $T_{m^*} \leq 0$ and $T_{m^*+1} > 0$. If indeed it were to happen that $T_m > 0$ for all $m = 1, \ldots, N$, this would imply that set $\mathcal{J}$ must contain all scenarios, $\mathcal{J} = \mathcal{N}$, making the exact value of η^{**} irrelevant in this case, since the original problem (2.2) would have to be solved at the next iteration of the scenario decomposition algorithm. □

A formal description of the described scenario decomposition process is presented in Algorithms 2.1–2.2.

Algorithm 2.1 Scenario decomposition algorithm

Initialize: $\mathcal{K} = \{0\}$, $S_0 = \{1, \ldots, N\}$, $\eta^{**}, w_0^{**} = \infty$, $h(\mathbf{x}^*) = -\infty$;
While $\eta^{**} + (1-\alpha)^{-1} w_0^{**} > h(\mathbf{x}^*)$ **do**
  $\mathbf{x}^*$:= solution of master problem (2.3) for a current partition $\{\mathcal{S}_k : k \in \mathcal{K}\}$;
  if (2.3) is infeasible **then**
    STOP
  else
    $(\eta^{**}, \mathbf{w}^{**})$:= solution of sub-problem (2.4) for the given $\mathbf{x}^*$;
  if $\eta^{**} + (1-\alpha)^{-1} w_0^{**} \leq h(\mathbf{x}^*)$ **then**
    STOP;
  else
    Identify binding scenarios: $\mathcal{J} := \{j \in \mathcal{N} : w_j^{**} = X(\mathbf{x}^*, \omega_j) - \eta^{**} > 0\}$;
  for $k = 0, \ldots, |\mathcal{K}|$ **do**
    $\mathcal{S}_k = \mathcal{S}_k \setminus \mathcal{J}$;
  for $i \in \mathcal{J}$ **do**
    $\mathcal{K} := \mathcal{K} \cup \{|\mathcal{K}| + 1\}$;
    $\mathcal{S}_{|\mathcal{K}|} := \{i\}$;
return $\mathbf{x}^*$

Algorithm 2.2 Solution method for sub-problem (2.4)

1. Compute $X_j := X(\mathbf{x}^*, \omega_j)$ for $j \in \mathcal{N}$;
2. Sort X_j in ascending order: $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(N)}$;
3. $m := N$; $T_m := 1$;
4. **while** $T_m > 0$ **do**
5. $\quad m := m - 1$;
6. $\quad T_m := \dfrac{\sum_{j=m+1}^{N} \pi_j v'\big(X_{(j)} - X_{(m)}\big)}{v'\left(v^{-1}\left(\sum_{j=m+1}^{N} \pi_j v\big(X_{(j)} - X_{(m)}\big)\right)\right)} - \alpha + 1$;
7. $m^* := m$;
8. **if** $T_{m^*} = 0$ **then**
9. $\quad \eta^{**} := X_{(m^*)}$;
10. **else**
11. $\quad$ Compute η^{**} by solving (2.11) on interval (T_{m^*+1}, T_{m^*});

2.3 Case study: Portfolio optimization with HMCR and LogExpCR measures

Portfolio optimization problems are commonly used as an experimental platform in risk management and stochastic optimization. Common implementations involve risk-reward analysis via mathematical programming models whose objective or constraints contain risk and reward measures. In this section we illustrate the computational performance of the proposed scenario decomposition algorithm on a portfolio optimization problem, where the investment risk is quantified using HMCR or LogExpCR measures.

A standard formulation of portfolio optimization problem entails determining the vector of portfolio weights $\mathbf{x} = (x_1, \ldots, x_n)^\top$ of n assets so as to minimize the risk while maintaining a prescribed level of expected return. We adopt the traditional

definition of portfolio losses X as negative portfolio returns, $X(\mathbf{x},\omega) = -\mathbf{r}(\omega)^\top \mathbf{x}$, where $\mathbf{r}(\omega) = (r_1(\omega),\ldots,r_n(\omega))^\top$ are random returns of assets. Then, the portfolio selection model takes the general form

$$\min \quad \mathscr{R}\big(-\mathbf{r}(\omega)^\top \mathbf{x}\big) \tag{2.16a}$$

$$\text{s. t.} \quad \mathbf{1}^\top \mathbf{x} = 1 \tag{2.16b}$$

$$\mathsf{E}\left[\mathbf{r}(\omega)^\top \mathbf{x}\right] \geq \bar{r} \tag{2.16c}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{2.16d}$$

where $\mathbf{1} = (1,\ldots,1)^\top$, equality (2.16b) represents the budget constraint, and (2.16b) ensures a minimum expected portfolio return level, $\bar{r}$, and (2.16d) corresponds to no-short-selling constraints. The distribution of the random vector $\mathbf{r}(\omega)$ of assets' returns is given by a finite set of N equiprobable scenarios $\mathbf{r}_j = \mathbf{r}(\omega_j) = (r_{1j},\ldots,r_{nj})^\top$,

$$\pi_j = \mathsf{P}\big\{\mathbf{r} = (r_{1j},\ldots,r_{nj})^\top\big\} = 1/N, \quad j \in \mathcal{N} \equiv \{1,\ldots,N\}. \tag{2.17}$$

2.3.1 Portfolio optimization using HMCR measures

In the case when risk measure ρ in (2.16) is selected as a higher moment coherent risk measure, $\rho(X) = \mathsf{HMCR}_{p,\alpha}(X)$, the portfolio optimization problem (2.16) can be written in a stochastic programming form that is consistent with the general

formulation (2.2) as

$$\min \quad \eta + (1-\alpha)^{-1} w_0 \tag{2.18a}$$

$$\text{s. t.} \quad w_0 \geq \|(w_1, \ldots, w_N)\|_p \tag{2.18b}$$

$$\pi_j^{-1/p} w_j \geq \ -\mathbf{r}_j^\top \mathbf{x} - \eta, \quad j \in \mathcal{N} \tag{2.18c}$$

$$\mathbf{x} \in C, \ \ \mathbf{w} \geq \mathbf{0}, \tag{2.18d}$$

where C represents a polyhedral set comprising the expected return, budget, and no-short-selling constraints on the vector of portfolio weights $\mathbf{x}$:

$$C = \Big\{ \mathbf{x} \in \mathbb{R}^n : \sum\nolimits_{j \in \mathcal{N}} \pi_j \mathbf{r}_j^\top \mathbf{x} \geq \bar{r}, \ \ \mathbf{1}^\top \mathbf{x} = 1, \ \ \mathbf{x} \geq \mathbf{0} \Big\}. \tag{2.19}$$

Due to the presence of p-order cone constraint (2.18b), formulation (2.18) constitutes a p-order cone programming problem (pOCP).

Solution methods for problem (2.18) are dictated by the given value of parameter p in (2.18b). As has been mentioned, in the case of $p = 1$ formulation (2.18) reduces to a LP problem that corresponds to a choosing the CVaR as the risk measure, a case that has received considerable attention in the literature. In view of this, of particular interest are nonlinear instances of problem (2.18) that correspond to values of the parameter $p \in (1, +\infty)$.

Below we consider instances of (2.18) with $p = 2$ and $p = 3$. In the case of $p = 2$, problem (2.18) can be solved using SOCP self-dual interior point methods. In the case of $p = 3$ and, generally, $p \in (1, 2) \cup (2, \infty)$, the p-cone (2.18b) is not self-dual, and we employ two techniques for solving (2.18) and the corresponding master problem (2.3): (i) a SOCP-based approach that relies on the fact that for a

rational p, a p-order cone can be equivalently represented via a sequence of second order cones, and (ii) an LP-based approach that allows for obtaining exact solutions of pOCP problems via cutting-plane methods.

Detailed discussions of the respective formulations of problems (2.18) are provided below. Throughout this section, we use abbreviations in brackets to denote the different formulations of the "complete" versions of (2.18) (i.e., with complete set of scenario constraints (2.18c)). For each "complete" formulation, we also consider the corresponding scenario decomposition approach, indicated by suffix "SD". Within the scenario decomposition approach, we present formulations of the master problem (denoted by subscript "MP"); the respective subproblems are then constructed accordingly. For example, the SOCP version of the complete problem (2.18) with $p = 2$ is denoted [SOCP], while the same problem solved by scenario decomposition is referred to as [SOCP-SD], with the master problem being denoted as $[\text{SOCP-SD}]_{\text{MP}}$ (see below).

SOCP formulation in $p = 2$ case. In case when $p = 2$, formulation (2.18) constitutes a standard SOCP problem that can be solved using a number of available SOCP solvers. In order to apply it to the scenario decomposition algorithm presented in Section 2.2, the master problem (2.3) is formulated with respect to the original

problem (2.18) with $p = 2$ as follows:

$$\begin{aligned}
[\text{SOCP-SD}]_{\text{MP}} := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\
\text{s. t.} \quad & w_0 \geq \|(w_1, \ldots, w_N)\|_2 \\
& \sum_{j \in \mathcal{S}_k} \frac{\pi_j^{1/2}}{\pi^{(k)}} w_j \geq \left(\sum_{j \in \mathcal{S}_k} \frac{\pi_j}{\pi^{(k)}} \mathbf{r}_j^\top \right) \mathbf{x} - \eta, \quad k \in \mathcal{K} \\
& \mathbf{w} \geq \mathbf{0}, \;\; \mathbf{x} \in C.
\end{aligned} \tag{2.20}$$

Note that in the case of $\mathsf{HMCR}_{2,\alpha}$ measure, the function $v_+(t) = t^2$ is positive homogeneous of degree two, which allows for eliminating the scenario probabilities π_j from constraint (2.3d) and formulating the latter in the form of a second order cone in the full formulation (2.18) and the master problem $[\text{SOCP-SD}]_{\text{MP}}$. This affects constraints (2.3d) which then can be written in the form of the second constraint in $[\text{SOCP-SD}]_{\text{MP}}$. The subproblem (2.4) is reformulated accordingly.

SOCP reformulation of p-order cone program. One of the possible approaches of solving pOCP problem (2.18) with $p = 3$ consists in reformulating the p-cone constraint (2.18b) via a set of quadratic cone constraints. Such an exact reformulation is possible when the parameter p has a rational value, $p = q/s$. Then, a (q/s)-order cone constraint in the positive orthant $\mathbb{R}^{N+1}_+$

$$w_0 \geq (w_1^{q/s} + \ldots + w_N^{q/s})^{s/q}, \quad \mathbf{w} \geq \mathbf{0}, \tag{2.21}$$

may equivalently be represented as the following set in $\mathbb{R}^{N+1}_+ \times \mathbb{R}^N_+$:

$$\begin{aligned}
& w_0 \geq \|\mathbf{u}\|_1 \\
& w_j^q \leq u_j^s w_0^{q-s}, \quad j \in \mathcal{N} \\
& \mathbf{w}, \mathbf{u} \geq \mathbf{0}.
\end{aligned} \tag{2.22}$$

Each of the N nonlinear inequalities in (2.22) can in turn be represented as a sequence of three-dimensional rotated second-order cones of the form $\xi_0^2 \leq \xi_1\xi_2$, resulting in a SOCP reformulation of the rational-order cone (2.21) [2, 27, 38]. Such a representation, however, is not unique and in general may comprise different number of rotated second order cones for a given $p = q/s$. In this case study we use the technique of [37], which allows for representing rational order p-cones with $p = q/s$ in $\mathscr{R}^{N+1}$ via $N\lceil\log_2 q\rceil$ second order cones. Namely, in the case of $p = 3$, when $q = 3$, $s = 1$, the 3-order cone (2.21) can be equivalently replaced with $\lceil\log_2 3\rceil N = 2N$ quadratic cones

$$\begin{aligned} & w_0 \geq \|\mathbf{u}\|_1 \\ & w_j^2 \leq w_0 v_j, \;\; v_j^2 \leq w_j u_j, \quad j \in \mathcal{N} \\ & \mathbf{w}, \mathbf{u}, \mathbf{v} \geq \mathbf{0}. \end{aligned} \tag{2.23}$$

In accordance with the above, a p-order cone inequality may be represented in $\mathbb{R}^{N+1}$ by a set of 3D conic constraints and a linear inequality when p is a positive rational number. Thus, the [SpOCP] problem (2.18) takes the following form:

$$\begin{aligned} [\text{SpOCP}] := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\ \text{s. t.} \quad & w_0 \geq \|\mathbf{u}\|_1 \\ & w_j^2 \leq w_0 v_j, \; v_j^2 \leq w_j u_j, \quad j \in \mathcal{N} \\ & \pi_j^{-1/p} w_j \geq \; -\mathbf{r}_j^\top \mathbf{x} - \eta, \quad j \in \mathcal{N} \\ & \mathbf{x} \in C, \; \mathbf{w}, \mathbf{v}, \mathbf{u} \geq \mathbf{0}. \end{aligned} \tag{2.24}$$

The corresponding master problem $[\text{SpOCP-SD}]_{\text{MP}}$ in a scenario decomposition-based method is constructed by replacing constraints of the form (2.18c) in the last problem

as follows:

$$
\begin{aligned}
[\text{SpOCP-SD}]_{\text{MP}} := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\
\text{s. t.} \quad & w_0 \geq \|\mathbf{u}\|_1 \\
& w_j^2 \leq w_0 v_j,\ v_j^2 \leq w_j u_j, \quad j \in \mathcal{N} \\
& \sum_{j \in \mathcal{S}_k} \frac{\pi_j^{1-1/p}}{\pi^{(k)}} w_j \geq \left(\sum_{j \in \mathcal{S}_k} \frac{\pi_j}{\pi^{(k)}} \mathbf{r}_j^\top \right) \mathbf{x} - \eta, \quad k \in \mathcal{K} \\
& \mathbf{x} \in C,\ \mathbf{w}, \mathbf{v}, \mathbf{u} \geq \mathbf{0}.
\end{aligned}
\tag{2.25}
$$

An exact solution method for pOCP programs based on polyhedral approximations. Computational methods for solving p-order cone programming problems that are based on polyhedral approximations [27, 58] represent an alternative to interior-point approaches, and can be beneficial in situations when a pOCP problem needs to be solved repeatedly, with small variations in problem data or problem structure.

Thus, in addition to the SOCP-based approaches for solving the pOCP problem (2.18) discussed above, we also employ an exact polyhedral-based approach with $O(\varepsilon^{-1})$ iteration complexity that was proposed in [58]. It consists in reformulating the p-order cone $w_0 \geq \|(w_1, \ldots, w_N)\|_p$ via a set of three-dimensional p-cones

$$
w_0 = w_{2N-1}, \quad w_{N+j} \geq \|(w_{2j-1}, w_{2j})\|_p, \quad j = 1, \ldots, N-1,
\tag{2.26}
$$

and then iteratively building outer polyhedral approximations of the 3D p-cones until the solution of desired accuracy $\varepsilon > 0$ is obtained,

$$
\|(w_1, \ldots, w_N)\|_p \leq (1+\varepsilon) w_0.
$$

In the context of the lifted representation (2.26), the above ε-relaxation of p-cone inequality translates into $N-1$ corresponding approximation inequalities for 3D p-cones:

$$\|(w^*_{2j-1}, w^*_{2j})\|_p \le (1+\epsilon) w^*_{N+j}, \quad j = 1, \dots, N-1, \tag{2.27}$$

where $\epsilon = (1+\varepsilon)^{1/\lceil \log_2 N \rceil} - 1$. Then, for a given $\varepsilon > 0$, an ε-approximate solution of pOCP portfolio optimization problem (2.18) is obtained by iteratively solving a linear programming problem

$$\begin{aligned}
[\text{LpOCP}] := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\
\text{s. t.} \quad & w_0 = w_{2N-1} \\
& w_{N+j} \ge \alpha_p(\theta_{k_j}) w_{2j-1} + \beta_p(\theta_{k_j}) w_{2j}, \quad \theta_{k_j} \in \Theta_j, \quad j = 1, \dots, N-1 \\
& \pi_j^{-1/p} w_j \ge \ -\mathbf{r}_j^\top \mathbf{x} - \eta, \quad j \in \mathcal{N} \\
& \mathbf{x} \in C, \ \mathbf{w} \ge \mathbf{0},
\end{aligned} \tag{2.28}$$

where coefficients α_p and β_p are defined as

$$\alpha_p(\theta) = \frac{\cos^{p-1}\theta}{(\cos^p\theta + \sin^p\theta)^{1-\frac{1}{p}}}, \quad \beta_p(\theta) = \frac{\sin^{p-1}\theta}{(\cos^p\theta + \sin^p\theta)^{1-\frac{1}{p}}}.$$

If for a given solution $\mathbf{w}^* = (w^*_0, \dots, w^*_{2N-1})$ of [LpOCP], the approximation condition (2.27) is not satisfied for some $j = 1, \dots, N-1$,

$$\|(w^*_{2j-1}, w^*_{2j})\|_p > (1+\epsilon) w^*_{N+j}, \tag{2.29}$$

then a cut of the form

$$w_{N+j} \ge \alpha_p(\theta^*_j) w_{2j-1} + \beta_p(\theta^*_j) w_{2j}, \quad \theta^*_j = \arctan \frac{w^*_{2j}}{w^*_{2j-1}}, \tag{2.30}$$

is added to [LpOCP]. The process is initialized with $\Theta_j = \{\theta_1\}$, $\theta_1 = \pi/4$, $j = 1, \ldots, N-1$, and continues until no violations of condition (2.29) are found. In [58] it was shown that this cutting-plane procedure generates an ε-approximate solution to pOCP problem (2.18) within $O(\varepsilon^{-1})$ iterations.

The described cutting plane scheme can be employed to solve the master problem corresponding to the pOCP problem (2.18). Namely, the cutting-plane formulation of this master problem is obtained by replacing the p-cone constraint (2.18b) with cutting planes similarly to [LpOCP], and the set of N scenario constraints (2.18c) with the aggregated constraints (compare to $[\text{SpOCP-SD}]_{\text{MP}}$):

$$
\begin{aligned}
[\text{LpOCP-SD}]_{\text{LB}} := \min \quad & \eta + (1-\alpha)^{-1} t \\
\text{s. t.} \quad & w_0 = w_{2N-1} \\
& w_{N+j} \geq \alpha_p(\theta_{k_j}) w_{2j-1} + \beta_p(\theta_{k_j}) w_{2j}, \quad \theta_{k_j} \in \Theta_j, \\
& \qquad\qquad\qquad\qquad\qquad\qquad j = 1, \ldots, N-1 \\
& \sum_{j \in \mathcal{S}_k} \frac{\pi_j^{1-1/p}}{\pi^{(k)}} w_j \geq \left(\sum_{j \in \mathcal{S}_k} \frac{\pi_j}{\pi^{(k)}} \mathbf{r}_j^\top \right) \mathbf{x} - \eta, \quad k \in \mathcal{K} \\
& \mathbf{x} \in C, \ \mathbf{w} \geq \mathbf{0}.
\end{aligned}
\tag{2.31}
$$

2.3.2 Portfolio optimization using LogExpCR measures

In order to demonstrate the applicability of the proposed method when solving problems with measures of risk other than the HMCR class, we examine an analogous experimental framework for instances when $\rho(X) = \mathsf{LogExpCR}_\alpha(X)$. The portfolio

optimization problem (2.16) may then written as

$$\begin{aligned} [\text{LogExpCP}] := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\ \text{s. t.} \quad & w_0 \geq \ln \sum_{j \in \mathcal{N}} \pi_j e^{w_j} \\ & w_j \geq \ -\mathbf{r}_j^\top \mathbf{x} - \eta, \quad j \in \mathcal{N} \\ & \mathbf{x} \in C, \ \mathbf{w} \geq \mathbf{0}. \end{aligned} \tag{2.32}$$

Note that in contrast to pOCP and SOCP problems discussed in the previous section, the above formulation is not a conic program since its first constraint that involves logarithm and exponent functions is not a cone. In view of this, we call this problem a log-exponential convex programming problem (LogExpCP), which can be solved with interior point methods.

The corresponding master problem for the scenario decomposition algorithm is obtained from [LogExpCP] by aggregating the scenario constraints in accordance to (2.3):

$$\begin{aligned} [\text{LogExpCP-SD}]_{\text{MP}} := \min \quad & \eta + (1-\alpha)^{-1} w_0 \\ \text{s. t.} \quad & w_0 \geq \ln \sum_{j \in \mathcal{N}} \pi_j e^{w_j} \\ & \sum_{j \in \mathcal{S}_k} w_j \geq - \sum_{j \in \mathcal{S}_k} \mathbf{r}_j^\top \mathbf{x} - |\mathcal{S}_k| \eta, \quad k \in \mathcal{K} \\ & \mathbf{x} \in C, \ \mathbf{w} \geq \mathbf{0}. \end{aligned} \tag{2.33}$$

In the next section we examine the computational performances within each implementation class of problem (2.18).

2.4 Computational experiments, scenario data, and results

The portfolio optimization problems described in Section 2.3.1 and 2.3.2 were implemented in C++ using callable libraries of three solvers, CPLEX 12.5, GUROBI 5.02, and MOSEK 6. Computations ran on a six-core 2.30GHz PC with 128GB RAM in 64-bit Windows environment. In the context of benchmarking, each adopted formulation was tested against its scenario decomposition-based implementation. Moreover, it was of particular interest to examine the performance of the scenario decomposition algorithm using various risk measure configurations, thus, the following problem settings were solved: problems [SOCP]-[SOCP-SD] with risk measure as defined by (1.7) for $p = 2$; problems [SpOCP]-[SpOCP-SD] and [LpOCP]-[LpOCP-SD] with measure (1.7) for $p = 3$; and problems [LECR]-[LECR-SD] with risk measure (1.8). The value of parameter α in the employed risk measures was fixed at $\alpha = 0.9$.

The scenario data in our numerical experiments was generated as follows. First, a set of n stocks ($n = 50, 100, 200$) was selected at random from the S&P500 index. Then, a covariance matrix of daily returns as well as the expected returns were estimated for the specific set of n stocks using historical prices from January 1, 2006 to January 1, 2012. Finally, the desired number N of scenarios, ranging from 1,000 to 100,000, have been generated as N iid samples from a multivariate normal distribution with the obtained mean and covariance.

On account of precision arithmetic errors associated with the numerical solvers, we introduced a tolerance level $\epsilon > 0$ specifying the permissible gap in the stopping

criterion (2.5):

$$\eta^{**} + (1-\alpha)^{-1} w_0^{**} \leq h(\mathbf{x}^*) + \epsilon. \tag{2.34}$$

Specifically, the value $\epsilon = 10^{-5}$ was was chosen to match the reduced cost of the simplex method in CPLEX and GUROBI. In a similar manner, we adjust (2.15) around m^* for precision errors as

$$T_{m^*+1}(p) - \epsilon < 0 \quad \text{and} \quad T_{m^*}(p) + \epsilon > 0.$$

Empirical observations suggest the accumulation of numerical errors is exacerbated by the use of fractional values of scenarios in assets returns, r_{ij}. To alleviate the numerical accuracy issues, the data in respective problem instances of the scenario decomposition algorithm were appropriately scaled.

The results of our numerical experiments are summarized in Tables 2.1 – 2.5 at the end of Chapter 2. Unless stated otherwise, the reported running time values were averaged over twenty instances and the symbol "—" indicates that the computation time limit of 3600 seconds was exceeded. Table 2.1 presents the computational times observed during solving the full formulation, [SOCP], of problem (2.18) with HMCR measure and $p = 2$, and solving the same problem using the scenario decomposition algorithm, [SOCP-SD], with the three solvers, CPLEX, GUROBI, and MOSEK. Observe that the scenario decomposition method performs better for all instances and solvers, with the exception of the largest three scenario instances when using GUROBI with $n = 50$ assets. However, this trend is tampered as the number of assets increases.

Table 2.2 reports the running times observed during solving of the second-order cone reformulation of the pOCP version of problem (2.18) with $p = 3$, in the full formulation ([SpOCP]) and via the scenario decomposition algorithm ([SpOCP-SD]). The obtained results indicate that, although the scenario decomposition algorithm is slower on smaller problem instances, it outperforms direct solution methods as the numbers of scenarios N and assets n in the problem increase. Due to observed numerical instabilities, the CPLEX solver was not considered for this particular experiment.

Next, the same problem is solved using using the polyhedral approximation cutting-plane method described in Section 2.3.1. Table 2.3 shows the running times achieved by all three solvers for problems [LpOCP] and [LpOCP-SD] with $p = 3$. In this case, the scenario decomposition method resulted in order-of-magnitude improvements, which can be attributed by the "warm-start" capabilities of CPLEX and GUROBI's simplex solvers. Consistent with these conclusions is also the fact that the simplex-based solvers of CPLEX and GUROBI yield improved solution times on the full problem formulation comparing to the SOCP-based reformulation [SpOCP], where barrier solvers were invoked. The discrepancy between [LpOCP] and [LpOCP-SD] solution times is especially prominent for MOSEK, but in this case it appears that MOSEK's interior-point LP solver was much less effective at solving the [LpOCP] formulation using the cutting plane method.

Finally, Table 2.4 displays the running times for the discussed implementation of problems [LogExpCR] and [LogExpCP-SD]. Of the three solvers considered in this case study, only MOSEK was capable of handling problems with constraints that

involve sums of univariate exponential functions. Again, the scenario decomposition-based solution method appear to be preferable in comparison to solving the full formulation. Note, however, that computational times were not averaged over 20 instances in this case due to numerical difficulties associated with the native solver for many instances of [LogExpCP].

It is also of interest to comment on the number of scenarios that had to be generated during the scenario decomposition procedure in order to yield an optimal solution. Table 2.5 lists the corresponding average number of scenarios partitioned for each problem type over all instances. Although these numbers may slightly differ among the three solvers, we only present results for MOSEK as it was the only platform used to solve all the problems in Sections 2.3.1 and 2.3.2. Observe that far fewer scenarios are required relative to the total set size N. In fact, as a percentage of the total, the number of scenarios that were generated during the algorithm in order to achieve optimality was between 0.7% and 11% of the set size.

2.5 Conclusions

In this chapter, we considered the efficiency of solving risk-based stochastic optimization problems that utilize large-scale scenario data sets. We exploit the notion that a significant portion of representative scenarios are not required to obtain an optimal solution, and accordingly develop a scenario decomposition technique contingent on the identification and separation of "non-redundant" scenarios by solving a series of smaller relaxation problems whose solution can be numerically evaluated in

the context of the original problem. Numerical experiments on portfolio optimization problems using simulated return data following the covariance structure of randomly chosen S&P500 stocks demonstrate that significant reductions in solution times may be achieved by employing the proposed algorithm. Particularly, performance improvements were observed for the large-scale instances when using HMCR measures with $p = 2, 3$, and LogExpCR measures.

Table 2.1: Average computation times (in seconds) obtained by solving problems [SOCP] and [SOCP-SD] for $p = 2$.

		CPLEX		GUROBI		MOSEK	
n	N	[SOCP]	[SOCP-SD]	[SOCP]	[SOCP-SD]	[SOCP]	[SOCP-SD]
50	1000	1.00	0.46	0.62	0.45	0.26	0.15
	2500	3.03	0.51	1.88	1.07	0.60	0.36
	5000	6.58	0.55	3.81	2.78	1.24	0.72
	10000	13.72	1.35	9.56	7.89	2.56	1.61
	25000	31.03	3.53	32.40	34.04	7.33	5.18
	50000	60.62	9.05	101.09	117.24	17.64	12.43
	100000	137.14	25.25	327.95	449.78	36.78	33.02
100	1000	2.46	0.86	1.73	0.42	0.61	0.18
	2500	6.14	0.99	4.87	1.17	1.50	0.47
	5000	13.69	1.10	11.13	3.55	3.25	1.15
	10000	27.06	2.21	21.94	9.63	6.69	3.03
	25000	72.95	8.85	71.34	37.48	20.41	6.88
	50000	157.25	20.88	185.56	129.37	44.01	16.61
	100000	319.90	58.29	464.12	467.35	79.75	41.58
200	1000	6.87	2.19	5.60	0.58	6.68	0.29
	2500	17.48	2.10	15.36	1.37	4.49	0.73
	5000	34.93	2.98	33.96	4.15	9.36	1.92
	10000	76.13	5.03	63.67	16.50	19.54	5.51
	25000	206.29	24.16	196.45	54.00	53.89	29.15
	50000	447.85	55.93	438.40	152.76	112.47	28.85
	100000	950.17	112.60	998.86	539.46	234.68	61.98

Table 2.2: Average computation times (in seconds) obtained by solving problems [SpCOP] and [SpCOP-SD] for $p = 3$.

		GUROBI		MOSEK	
n	N	[SpOCP]	[SpCOP-SD]	[SpOCP]	[SpCOP-SD]
50	1000	2.58	2.73	0.18	0.63
	2500	10.63	6.61	0.49	0.96
	5000	32.01	19.27	1.06	1.70
	10000	87.27	41.34	2.31	3.49
	25000	198.56	92.39	7.14	6.70
	50000	455.63	540.09	16.36	13.70
	100000	1217.96	2080.34	35.33	30.29
100	1000	7.16	3.14	0.30	0.75
	2500	29.47	8.44	0.85	1.37
	5000	90.25	19.74	1.88	2.32
	10000	277.72	44.31	4.52	3.91
	25000	642.63	92.11	12.66	8.66
	50000	1365.37	1716.37	28.64	15.10
	100000	—	—	65.48	28.29
200	1000	17.86	3.87	0.69	1.01
	2500	78.28	8.65	1.90	1.56
	5000	276.89	22.40	4.41	2.47
	10000	799.65	49.02	9.88	4.84
	25000	2118.11	107.14	29.99	9.60
	50000	—	—	64.52	17.41
	100000	—	—	139.87	34.99

Table 2.3: Average computation times (in seconds) obtained by solving problems [LpOCP] and [LpOCP-SD] for $p = 3$.

		CPLEX		GUROBI		MOSEK	
n	N	[LpOCP]	[LpCOP-SD]	[LpOCP]	[LpCOP-SD]	[LpOCP]	[LpCOP-SD]
50	1000	0.27	0.12	0.22	0.59	0.82	0.46
	2500	1.65	0.24	0.74	0.83	4.26	0.66
	5000	6.81	0.46	2.31	1.54	15.08	1.46
	10000	19.20	1.42	7.73	3.86	60.66	3.75
	25000	31.93	3.93	56.52	13.74	381.67	11.34
	50000	179.49	16.07	117.72	36.51	1412.81	25.47
	100000	903.36	62.79	474.68	112.72	—	54.45
100	1000	0.37	0.13	0.23	0.61	2.94	0.65
	2500	2.22	0.28	0.86	0.98	7.11	1.06
	5000	8.58	0.79	2.82	1.76	32.20	1.95
	10000	28.71	2.18	9.28	4.13	122.75	4.99
	25000	45.37	4.99	35.11	13.13	1138.99	15.34
	50000	200.12	18.80	122.21	39.78	2753.54	34.17
	100000	3336.26	82.79	1316.29	138.74	—	80.15
200	1000	0.61	0.20	0.33	0.89	15.68	1.06
	2500	3.13	0.44	1.30	1.17	20.64	1.37
	5000	13.25	1.01	3.72	2.11	70.49	2.97
	10000	47.97	3.31	13.20	4.72	322.36	8.12
	25000	195.28	6.98	94.45	14.77	2418.52	26.91
	50000	936.60	27.20	665.61	45.43	—	53.62
	100000	—	114.08	3301.44	160.92	—	123.89

Table 2.4: Average computation times (in seconds) obtained by solving a specified number of instances for problems [LECR] and [LECR-SD].

		MOSEK		Instances Solved
n	N	[LECR]	[LECR-SD]	
50	1000	0.61	0.27	12
	2500	0.97	0.58	14
	5000	1.89	1.18	12
	10000	4.88	2.57	9
	25000	14.99	7.94	12
	50000	26.65	18.76	15
	100000	65.45	61.48	17
100	1000	0.57	0.25	17
	2500	1.65	0.53	16
	5000	3.69	1.14	10
	10000	9.18	2.53	15
	25000	24.61	13.83	13
	50000	50.66	39.72	19
	100000	148.54	59.02	16
200	1000	5.25	0.37	19
	2500	4.22	0.75	17
	5000	9.53	1.39	18
	10000	21.17	2.63	17
	25000	62.03	7.59	17
	50000	145.89	16.47	18
	100000	333.73	43.56	19

Table 2.5: Average number of partitioned scenarios from solving the scenario decomposition-based problems listed in Sections 2.3.1 – 2.3.2.

		MOSEK			
n	N	[SOCP-SD]	[SpOCP-SD]	[LpOCP-SD]	[LECR-SD]
50	1000	80.3	24.8	21.3	61.8
	2500	180.8	47.8	47.0	77.8
	5000	349.3	80.3	79.0	104.6
	10000	711.6	133.4	128.3	154.3
	25000	1834.9	232.0	318.3	178.2
	50000	3582.1	445.4	675.0	841.7
	100000	6945.1	774.1	1346.4	1447.5
100	1000	87.2	32.0	27.0	81.4
	2500	191.2	73.6	74.1	107.8
	5000	367.6	107.4	102.4	192.2
	10000	711.1	148.9	156.9	229.7
	25000	1808.6	278.1	348.6	1869.1
	50000	3802.9	457.8	729.7	2418.6
	100000	7323.3	831.3	1395.8	923.4
200	1000	108.2	39.5	36.4	100.7
	2500	201.7	72.7	73.0	154.5
	5000	395.6	116.3	119.6	198.1
	10000	744.0	184.9	171.2	304.6
	25000	1805.5	308.3	347.0	464.2
	50000	3607.8	512.2	697.6	788.1
	100000	7198.9	865.0	1384.3	1153.5

CHAPTER 3
ON RISK-AVERSE MAXIMUM WEIGHTED SUBGRAPH PROBLEMS

3.1 Introduction

For decades, network problems with topologically exogenous information have occupied a prominent place in graph theory and network science literature. A popular class of problems of this type involves finding a subset of minimum or maximum weight conforming to a prescribed structural property in a graph whose vertices are characterized by deterministic weights [6, 7, 21, 32, 39]. Several influential studies have established a foundation for exact combinatorial solution algorithms for such problems [8, 17, 40]. Most notably, Carraghan and Pardalos [17] developed a backtracking branch-and-bound method for efficiently solving the maximum clique problem by exploiting the hereditary property [53] of complete subgraphs. Many extensions of their work improved upon the process of reducing the search space by using vertex coloring schemes for branching and for obtaining upper bounds on the maximum achievable subgraph order (see, e.g., [16, 25, 52]). Analogous weight-based procedures have also been used when seeking a maximum weight subgraph in the presence of deterministic vertex weights [6, 31, 39].

Significant emphasis has also been placed on network problems with uncertain exogenous information evidenced in various forms that influences the overall topology, flow distribution and costs, etc. Particularly common are considerations of stochastic factors in context of network flow and vehicle routing problems where uncertainties

are attributed to arc capacities or node demands [5, 15, 22, 23]. Also, a number of studies examined the effects of probabilistic arc failures in networks [3, 56] and introduced risk-based approaches to minimize the corresponding flow losses [13, 48]. The problem of finding a subset of vertices of maximum cardinality that form a clique with a specified probability, given that edges in the graph can fail with some probabilities, is studied in [36]; a similar approach in application to certain clique relaxations is pursued in [62]. Although uncertainties in most of the aforementioned cases influence decisions related to directed network flows, far less emphasis has been placed on examining decision making regarding optimal subgraph topologies and resource allocation in settings where uncertainties are induced by stochastic factors associated with network vertices.

In this work [47], we employ a stochastic programming framework that is based on formalism of risk measures [29], and in particular, coherent risk measures, in order to find minimum-risk structures in graphs with stochastic vertex weights. Namely, we consider a class of *risk-averse maximum weighted*[1] *subgraph problems* (R-MWSP) that represent a stochastic extension of the so-called maximum weight subgraph problems considered in literature in the context of hereditary graph-theoretical properties. We propose a graph-based branch-and-bound algorithm for solving problems in the R-MWSP class, which is generally applicable to maximum weight subgraph problems

[1]The word "weighted" is reserved to address problems that seek maximal subgraphs of minimum risk subject to stochastic vertex weights. The term "weight" defines a more traditional problem setting where vertex weights are deterministic and a subgraph with the largest cumulative weight is sought.

where a subgraph's weight is given by a super-additive function whose evaluation requires solving an optimization problem. As an illustrative example of the proposed concepts, we consider a risk-averse maximum weighted clique problem.

The remainder of this chapter is organized as follows. In Section 3.2 we introduce the general formulation of R-MWS problems and discuss their properties. Section 3.3 presents solution methods for R-MWSP, including a mathematical programming formulation and a graph-based (combinatorial) branch-and-bound method. Section 3.4 considers a numerical case study on solving risk-averse maximum weighted clique problems using HMCR measures for randomly generated graphs with various densities. Furhter, an extension of the risk-averse maximum weighted clique in context of neighbor dependent risk exposure is also provided in Section 3.4. Solution properties relative to the optimal subgraph sizes obtained from solving the mentioned configurations using CVaR measures are discussed.

3.2 Risk-averse maximum vertex problem

Let $G = (V, E)$ be an undirected graph where each vertex $i \in V$ has a positive weight $w_i > 0$. For any subset S of its vertices, let $G[S]$ denote the subgraph of G *induced by* S, i.e., a graph such that any of its vertices i, j are connected by an edge if and only if (i, j) is an edge in G.

Property Π is said to be *hereditary with respect to induced subgraphs* (*hereditary* for short) if for any graph satisfying Π the removal of a vertex preserves Π in the resulting induced subgraph. Examples of hereditary properties include *"complete"*;

"independent", or *"stable"*; *"degree constrained"*; *"planar"*, etc. Given a hereditary property Π, it may be of interest to find a subgraph of G that satisfies Π and has the largest additive weight, which is known as the *maximum weight subgraph problem*, or the *maximum weight* Π *problem*:

$$\max_{S \subseteq V} \Big\{ \sum_{i \in S} w_i : G[S] \text{ satisfies } \Pi \Big\}. \tag{3.1}$$

A subgraph of G that satisfies Π and whose order cannot be further increased without violating Π is known as a *maximal* Π*-subgraph*; the largest such subgraph represents the *maximum* Π*-subgraph*. Obviously, an optimal solution of the maximum weight Π problem (3.1) is necessarily a maximal Π-subgraph, but may not be its maximum Π-subgraph.

Finding subgraphs of maximum weight with hereditary properties represents a large and important class of graph theoretical problems. A seminal result regarding maximum subgraph problems with hereditary properties was established by Yannakakis [61]. Particularly, property Π is called *nontrivial* if it is satisfied by a single-vertex graph and not satisfied by every graph, and is called *interesting* if the order of graphs satisfying Π is unbounded. Then, the following holds:

Theorem 3.1 (Yannakakis [61]). *If property* Π *is hereditary with respect to induced subgraphs, nontrivial, and interesting, then the maximum* Π *problem*

$$\max_{S \subseteq V} \big\{ |S| : G[S] \textit{ satisfies } \Pi \big\}$$

is NP-complete.

It is straightforward that the statement of this theorem extends to the version of the maximum weight Π problem (3.1). Some of the most well known instances of (3.1) include the maximum weight clique problem and maximum weight independent set problem.

Now we pose the question that served as motivation for the present endeavor: *What if the vertex weights w_i are uncertain?* In this case, extending the deterministic formulation (3.1) into the stochastic domain is not straightforward and requires additional considerations. Indeed, minimization of the random quantity that is represented by the sum of random weights in (3.1) is ill-posed in the context of decision making under uncertainty that requires a deterministic optimal solution. Therefore, the sum of stochastic weights in the objective has to be replaced with a statistical functional that utilizes the distributional information about the weights' uncertainties. The traditional stochastic optimization approach, for example, involves seeking the best "expected outcome", which in this setting would translate into maximizing the expected weight of an induced subgraph $G[S]$. It is easy to see, however, that maximization of the expected subgraph weight trivially reduces to the deterministic maximum weight Π formulation with expected vertex weights: $\mathsf{E}\big(\sum_{i\in S} w_i\big) = \sum_{i\in S} \mathsf{E}w_i$.

In this work, we pursue a *risk-averse* approach and consider the problem of finding the subgraph of G that satisfies property Π and has the lowest risk. Namely, let X_i denote a stochastic variable that is associated with vertex $i \in V$ and assume that the joint distribution of vector $\mathbf{X}_G = (X_1, \ldots, X_{|V|})$ is known. Assuming that the random quantities X_i, $i \in V$, represent costs or losses, consider the problem of

finding the *minimum-risk* subgraph in G with property Π, or the *risk-averse maximum weighted* Π *problem*:

$$\min_{S\subseteq V} \big\{\mathscr{R}(S;\mathbf{X}_G) : G[S] \text{ satisfies } \Pi\big\}. \tag{3.2}$$

In formulation (3.2), the functional $\mathscr{R}(S;\mathbf{X}_G)$ quantifies the risk of the induced subgraph $G[S]$ given the distributional information $\mathbf{X}_G$, and is undefined as yet.

By virtue of Section 1.2 we define the risk $\mathscr{R}(S;\mathbf{X}_G)$ of a subgraph $G[S]$ in(3.2) in the same context as risk measure ρ. This basic definition is also augmented the properties (A1)-(A4) that are dictated by application. Then, given a risk measure ρ that we additionally assume to be lower semi-continuous (l.s.c.), the risk $\mathscr{R}(S;\mathbf{X}_G)$ of a subgraph of G induced on a set of vertices $S \subseteq V(G)$ with uncertain vertex weights X_i can be defined as an optimal value of the following stochastic programming problem:

$$\mathscr{R}(S;\mathbf{X}_G) = \min\left\{\rho\Big(\sum_{i\in S} u_i X_i\Big) : \sum_{i\in S} u_i = 1,\ u_i \geq 0,\ i \in S\right\}. \tag{3.3}$$

Recall that function $f : \mathcal{X} \mapsto \overline{\mathbb{R}}$ is l.s.c. if and only if the sets $\{X \in \mathcal{X} : f(X) \leq a\}$ are closed for all $a \in \mathbb{R}$. Obviously, lower semi-continuity of risk measure ρ is necessary for the minimization problem in (3.3) to be well-posed. In the sequel, it will be implicitly assumed that the risk measure ρ in (3.3) is l.s.c.

The rationale behind definition (3.3) of subgraph risk function $\mathscr{R}(\cdot)$ is that, similarly to many "nice" risk measures, such as those discussed in Chapter 1, it allows for risk reduction through diversification:

Proposition 3.2. *Given a graph $G = (V, E)$ with stochastic weights X_i, $i \in V$, and a l.s.c. risk measure ρ, the subgraph risk function $\mathscr{R}$ defined by* (3.3) *satisfies*

$$\mathscr{R}(S_2; \mathbf{X}_G) \leq \mathscr{R}(S_1; \mathbf{X}_G) \quad \textit{for all} \quad S_1 \subseteq S_2. \tag{3.4}$$

Proof. For $S_1 \subseteq S_2$, denote

$$\mathbf{u}^{(k)} \in \arg\min \left\{ \rho\Big(\sum_{i \in S_k} u_i X_i \Big) : \sum_{i \in S_k} u_i = 1; \;\; u_i \geq 0, \;\; i \in S_k \right\}, \quad k = 1, 2.$$

Then, one immediately has

$$\mathscr{R}(S_2; \mathbf{X}_G) = \rho\Big(\sum_{i \in S_2} u_i^{(2)} X_i \Big) \leq \rho\Big(\sum_{i \in S_1} u_i^{(1)} X_i + \sum_{j \in S_2 \setminus S_1} 0 \cdot X_j \Big) = \mathscr{R}(S_1; \mathbf{X}_G).$$

due to lower semicontinuity of risk measure ρ. □

Note that the power of definition (3.3) via solution of a stochastic programming problem is evidenced in the fact that the property (3.4) of risk reduction via diversification holds for any l.s.c. risk measure $\rho : \mathcal{X} \mapsto \mathbb{R}$. Secondly, property (3.4) implies the following important observation regarding the optimal solution of the risk-averse maximum weighted Π problem (3.2):

Corollary 3.3. *The optimal solution of the risk-averse maximum weighted* Π *problem* (3.2) *with $\mathscr{R}(S; \mathbf{X}_G)$ defined by* (3.3) *is a maximal* Π*-subgraph in G.*

Recall that the same characterization of optimal solutions holds for the deterministic maximum weight Π problem (3.1). This provides justification for calling the risk-minimization problem (3.2) a "risk-averse maximum *weighted* subgraph problem".

In this respect, it is worth mentioning that the presented framework differs from other recent studies that also utilized formally defined risk measures for quantifying the risk in graphs, but relied on explicit maximization of the subgraph's cardinality or weight while requiring that its risk be bounded (see, e.g., [36, 62]):

$$\max_{S\subseteq V} \big\{ |S| : \mathrm{Risk}(S) \leq c_0,\ G[S] \text{ satisfies } \Pi \big\}.$$

Indeed, the proposed definition (3.3) of risk function $\mathscr{R}$ in the R-MWS problem (3.2) implies that maximization of a solution's cardinality is a consequence of risk minimization via diversification.

Further properties of $\mathscr{R}(S; \mathbf{X}_G)$ depend on those of the risk measure ρ in (3.3). The next proposition states that when the risk measure ρ is coherent, or at least possesses the properties (A1), (A3), (A4) defined in Chapter 1, then the corresponding subgraph risk function $\mathscr{R}(S; \mathbf{X}_G)$ satisfies properties analogous to (A1), (A3), (A4) with respect to the stochastic weights vector $\mathbf{X}_G$.

Proposition 3.4. *Let $G = (V, E)$ be an undirected graph, and $\mathbf{X}_G = (X_1, \ldots, X_{|V|})$, and $\mathbf{Y}_G = (Y_1, \ldots, Y_{|V|})$ be vectors of stochastic weights whose components are defined on the same linear space $\mathcal{X}$. If the risk measure ρ in* (3.3) *is l.s.c. and satisfies axioms (A1), (A3), and (A4) of coherency, then for any induced subgraph $G[S]$ the subgraph risk function $\mathscr{R}$ defined in* (3.3) *satisfies the following properties:*

(G1) $\mathscr{R}(S; \mathbf{X}_G) \leq \mathscr{R}(S; \mathbf{Y}_G)$ *for all* $\mathbf{X}_G \leq \mathbf{Y}_G$

(G2) $\mathscr{R}(S; \lambda \mathbf{X}_G) = \lambda \mathscr{R}(S; \mathbf{X}_G)$ *for all* $\mathbf{X}_G$ *and* $\lambda > 0$

(G3) $\mathscr{R}(S; \mathbf{X}_G + a\mathbf{1}) = \mathscr{R}(S; \mathbf{X}_G) + a$ *for all* $a \in \mathbb{R}$

where $\mathbf{1}$ *is the vector of ones, and the vector inequality* $\mathbf{X}_G \leq \mathbf{Y}_G$ *is interpreted component-wise.*

Proof. Consider, for example, property (G1). Denoting, as before,

$$\mathbf{u}^Z \in \arg\min\left\{\rho\Big(\sum_{i\in S} u_i Z_i\Big) : \sum_{i\in S} u_i = 1;\ \ u_i \geq 0,\ \ i \in S\right\},$$

we have

$$\mathscr{R}(S;\mathbf{X}_G) = \rho\Big(\sum_{i\in S} u_i^X X_i\Big) \leq \rho\Big(\sum_{i\in S} u_i^Y X_i\Big).$$

On the other hand, from $X_i \leq Y_i$ it follows that

$$\sum_{i\in S} u_i^Y X_i \leq \sum_{i\in S} u_i^Y Y_i,$$

whence

$$\rho\Big(\sum_{i\in S} u_i^Y X_i\Big) \leq \rho\Big(\sum_{i\in S} u_i^Y Y_i\Big) = \mathscr{R}(S;\mathbf{Y}_G).$$

Properties (G2) and (G3) are verified similarly. □

Observe that $\mathscr{R}(S;\mathbf{X}_G)$ does not obey the sub-additivity with respect to the stochastic weights, i.e., in general

$$\mathscr{R}(S;\mathbf{X}_G + \mathbf{Y}_G) \not\leq \mathscr{R}(S;\mathbf{X}_G) + \mathscr{R}(S;\mathbf{Y}_G).$$

With respect to the traditional risk measures $\rho : \mathcal{X} \mapsto \mathbb{R}$, the failure to satisfy the sub-additivity requirement (or, if positive homogeneity also does not hold, the convexity requirement) implies that such a risk measure is ill fitting for risk reduction via diversification. In other words, it is possible that diversification can result in an increased

risk exposure, as measured by a non-subadditive (correspondingly, nonconvex) risk measure ρ.

In the context of proposed risk function $\mathscr{R}$ for subgraphs, risk reduction via diversification is already ascertained by (3.4), which, with respect to the problem of finding a Π-subgraph with the smallest risk, ensures that adding new vertices to the existing feasible solution that satisfies a hereditary property Π is always beneficial, provided that Π is not violated by the addition of new vertices. Yet, under an additional assumption that the stochastic vertex weights have non-negative support, i.e., $\mathbf{X}_G \geq \mathbf{0}$, the subgraph risk function $\mathscr{R}(S; \mathbf{X}_G)$ can be shown to be "set-subadditive". Namely, one has

Proposition 3.5. *Let the stochastic vertex weights X_i, $i \in V$, of graph $G = (V, E)$ satisfy $X_i \geq 0$, $i \in V$. Then, for any $S_1, S_2 \subseteq V$ the subgraph risk function $\mathscr{R}(S; \mathbf{X}_G)$ defined by* (3.3) *satisfies*

$$\mathscr{R}(S_1 \cup S_2; \mathbf{X}_G) \leq \mathscr{R}(S_1; \mathbf{X}_G) + \mathscr{R}(S_2; \mathbf{X}_G), \tag{3.5}$$

provided that the risk measure ρ in (3.3) *is l.s.c. and satisfies (A1) and (A2).*

Proof. If ρ satisfies axioms (A1) and (A2), then $\rho(X) \geq 0$ for any $X \geq 0$. Immediately, one has $\mathscr{R}(S_1 \cup S_2; \mathbf{X}_G) \leq \mathscr{R}(S_1; \mathbf{X}_G) \leq \mathscr{R}(S_1; \mathbf{X}_G) + \mathscr{R}(S_2; \mathbf{X}_G)$. □

Naturally, in the context of risk-averse maximum weighted Π problems where Π is hereditary, one should also require that S_1, S_2, and $S_1 \cup S_2$ satisfy Π.

Note that the assumption of nonnegative support for vertex weights X_i is analogous to the standard assumption of positive vertex weights in hereditary maximum

weight subgraph problems such as the maximum clique and independent set problems [7, 42].

3.3 Solution approaches for risk-averse maximum weighted subgraph problems

In this section we consider a mathematical programming formulation for the R-MWS Π problem (3.2), where the risk $\mathscr{R}(S)$ of induced subgraph $G[S]$ is defined as in (3.3), and propose a graph-based, or combinatorial branch-and-bound algorithm that represents an extension of the well-known branch-and-bound schemes for the maximum clique problem [17, 39, 40].

3.3.1 A mathematical programming formulation

Given a graph-theoretic property Π, let binary decision variables x_i indicate whether node $i \in V$ belongs to a subset S, such that the induced subgraph $G[S]$ satisfies Π:

$$x_i = \begin{cases} 1, & i \in S \text{ such that } G[S] \text{ satisfies } \Pi \\ 0, & \text{otherwise.} \end{cases}$$

Further, let $\mathbf{\Pi}_G(\mathbf{x}) \leq \mathbf{0}$ denote the *structural constraints* such that for any $\tilde{\mathbf{x}} \in \{0,1\}^{|V|}$, $\mathbf{\Pi}_G(\tilde{\mathbf{x}}) \leq \mathbf{0}$ if and only if $G[\tilde{S}]$ satisfies Π, where $\tilde{S} = \{i \in V : \tilde{x}_i = 1\}$. Then, the following proposition, which we give without proof, formalizes a mathematical programming representation for the risk-averse maximum weighted Π problem (3.2) with risk $\mathscr{R}(S; \mathbf{X}_G)$ defined by (3.3) if the property Π is hereditary on induced subgraphs:

Proposition 3.6. *Let $G = (V, E)$ be an undirected graph with stochastic vertex weights X_i, $i \in V$, and Π be a property hereditary on induced subgraphs. Then, the R-MWS Π problem* (3.2) *with risk defined by* (3.3) *can equivalently be represented as a mixed 0–1 programming problem*

$$\begin{aligned} \min \quad & \rho\big(\mathbf{u}^\top \mathbf{X}_G\big) \\ \text{s. t.} \quad & \mathbf{u}^\top \mathbf{1} = 1 \\ & \mathbf{u} \leq \mathbf{x} \\ & \mathbf{\Pi}_G(\mathbf{x}) \leq \mathbf{0} \\ & \mathbf{x} \in \{0,1\}^{|V|}, \;\; \mathbf{u} \in \mathbb{R}_+^{|V|}. \end{aligned} \tag{3.6}$$

When the property Π in (3.6) denotes graph completeness, one can choose, for example, the well-known *edge formulation* of the maximum clique problem (see, e.g., [42]) to represent the structural constraints (3.6d) as

$$\big\{\mathbf{x} \in \{0,1\}^{|V|} : \mathbf{\Pi}_G(\mathbf{x}) \leq \mathbf{0}\big\} = \big\{\mathbf{x} \in \{0,1\}^{|V|} : x_i + x_j \leq 1 \;\text{ for all }\; (i,j) \in \overline{E}\big\},$$

where $\overline{E}$ represents the complement edges of graph G, whereby the mathematical programming formulation of the R-MWS clique problem (3.2)–(3.3) takes the form

$$\begin{aligned} \min \quad & \rho\Big(\sum_{i \in V} u_i \, X_i\Big) \\ \text{s. t.} \quad & \sum_{i \in V} u_i = 1 \\ & u_i \leq x_i, \quad i \in V \\ & x_i + x_j \leq 1, \quad (i,j) \in \overline{E} \\ & x_i \in \{0,1\}, \;\; u_i \geq 0, \quad i \in V. \end{aligned} \tag{3.7}$$

Formulations (3.6)-(3.7) allow for handling risk measures ρ whose representations come in the form of mathematical programming problems, and can be solved with appropriate (nonlinear) mixed integer programming solvers.

A combinatorial branch-and-bound algorithm that allows for exploiting the structure of problems (3.6)-(3.7) imposed by the underlying graph G is described next.

3.3.2 A graph-based branch-and-bound algorithm

The combinatorial branch-and-bound (BnB) algorithm works by navigating between "levels" of the BnB tree until a subgraph of G that satisfies property Π and is guaranteed to be of lowest risk as measured by (3.3) is found. The algorithm starts at level $\ell = 0$ with a partial solution $Q := \emptyset$, incumbent solution $Q^* := \emptyset$, and a global upper bound $L^* := +\infty$ on risk of Q^*. Throughout the algorithm, the partial solution Q contains the vertices in V such that $G[Q]$ has property Π, and set Q^* induces, per Corollary 3.3, a maximal Π-subgraph whose risk equals L^* in G hitherto.

Within the current branch of the BnB tree, "level" ℓ is associated with the *candidate set* C_ℓ of vertices such that any single vertex of C_ℓ can be added to the current partial solution Q without violating property Π. Branching is performed by removing a *branching* vertex q from C_ℓ and adding it to the partial solution Q. The algorithm is initialized with $C_0 := V$, and, as soon as the partial solution Q is updated after branching at level ℓ, the corresponding candidate set at level $\ell+1$ is constructed by removing all vertices from C_ℓ whose inclusion in Q would break the property Π,

i.e.,

$$C_{\ell+1} := \{i \in C_\ell : G[i \cup Q] \text{ satisfies } \Pi\}. \tag{3.8}$$

As a result, immediately after branching at level ℓ the cardinality of partial solution set Q is equal to $|Q| = \ell + 1$.

The bounding step of the BnB algorithm involves evaluating the quality of the solution that can be obtained by exploring further the subgraph induced by vertices in $Q \cup C_{\ell+1}$. Observe that an exact approach of directly finding the Π-subgraph with the lowest possible risk that is contained in $G[Q \cup C_{\ell+1}]$ entails solving the following restriction of problem (3.6):

$$\begin{aligned} \mathscr{R}(Q \cup C_{\ell+1}; \mathbf{X}_G) = \min \quad & \rho\big(\mathbf{u}^\top \mathbf{X}_G\big) \\ \text{s. t.} \quad & \mathbf{u}^\top \mathbf{1} = 1 \\ & \mathbf{u} \leq \mathbf{x}, \\ & \mathbf{\Pi}_G(\mathbf{x}) \leq \mathbf{0}, \\ & \mathbf{x} \in \{0,1\}^{|V|}, \;\; \mathbf{u} \in \mathbb{R}_+^{|V|}, \\ & x_i = 0, \quad i \in V \setminus (Q \cup C_{\ell+1}). \end{aligned} \tag{3.9}$$

As (3.9) is a (nonlinear) mixed 0–1 problem, solving it at every node of the BnB tree is impractical. Instead, a lower bound on the value of $\mathscr{R}(Q \cup C_{\ell+1}; \mathbf{X}_G)$ given by (3.9) can be computed. However, in contrast to the traditional mixed integer programming approach of constructing a lower bound by relaxing the integrality constraints, we formulate a lower bound problem by completely eliminating the 0-1

variables x_i along with the structural constraints:

$$\begin{aligned} \mathscr{R}(Q \cup C_{\ell+1}; \mathbf{X}_G) \geq \mathcal{L}(Q \cup C_{\ell+1}) := \min \quad & \rho\Big(\sum_{i \in V} u_i \, X_i\Big) \\ \text{s. t.} \quad & \sum_{i \in V} u_i = 1 \\ & u_i = 0, \quad i \in V \setminus (Q \cup C_{\ell+1}) \\ & u_i \geq 0, \quad i \in Q \cup C_{\ell+1}. \end{aligned} \tag{3.10}$$

Observe that the structural constraints $\mathbf{\Pi}_G(\mathbf{x}) \leq \mathbf{0}$ in problem (3.9) are satisfied by variables $\{x_i : i \in Q\}$ (since $G[Q]$ satisfies Π), as well as by variables $\{x_i : i \in Q \cup j_0\}$ for each $j_0 \in C_{\ell+1}$ (since $G[Q \cup j_0]$ for each vertex j_0 in $C_{\ell+1}$ also satisfies Π, per definition (3.8) of the candidate set $C_{\ell+1}$). Hence, the corresponding structural constraints are redundant in (3.9). On the other hand, the structural constraints are not necessarily satisfied by variables $\{x_i : i \in C_{\ell+1}\}$ and $\{x_i : i \in Q \cup C_{\ell+1}\}$, since $G[C_{\ell+1}]$ and $G[Q \cup C_{\ell+1}]$ do not necessarily satisfy Π. Thus, (3.10) is a relaxation of (3.9), and, by virtue of Proposition 3.2, the solution to (3.10) provides a lower bound on the minimum risk achievable in any Π-subgraph induced on the union of Q with any subset of $C_{\ell+1}$, i.e.,

$$\mathcal{L}(Q \cup C_{\ell+1}) \leq \mathscr{R}(Q \cup C_{\ell+1}; \mathbf{X}_G) \leq \mathscr{R}(Q \cup S; \mathbf{X}_G) \;\text{ for any }\; S \subseteq C_{\ell+1}.$$

Observe that if $\ell' = \ell + 1$ represents the next level in the BnB tree, and Q' is the corresponding partial solution, then due to the definition (3.8) of candidate set one has

$$(Q' \cup C_{\ell'+1}) \subseteq (Q \cup C_{\ell+1}),$$

whence the risk $\mathscr{R}(Q \cup C_{\ell+1}; \mathbf{X}_G)$ does not decrease as ℓ increases (or, in other words, as new vertices are added to the partial solution Q and the algorithm proceeds to deeper levels ℓ of the BnB tree). We next show that this observation is an effective bounding criterion to obtain a Π-subgraph of lowest risk in G.

Depending on the computed value of $\mathcal{L}(Q \cup C_{\ell+1})$, the algorithm branches further or prunes/backtracks as follows. If $\mathcal{L}(Q \cup C_{\ell+1}) < L^*$ and $C_{\ell+1} \neq \emptyset$, the algorithm proceeds to select a branching vertex at the next level $\ell+1$; for backtracking purposes, the current branching vertex q at level ℓ is stored as q_ℓ. If $\mathcal{L}(Q \cup C_{\ell+1}) < L^*$ and $C_{\ell+1} = \emptyset$, the subgraph induced by the partial solution Q represents a maximal Π-subgraph in G and is declared as the new incumbent solution, $Q^* := Q$, the global upper bound on risk is updated $L^* := \mathcal{L}(Q \cup C_{\ell+1})$, and algorithm backtracks by removing q from Q.

In the case of $\mathcal{L}(Q \cup C_{\ell+1}) \geq L^*$, the vertex q is removed from Q and the corresponding branch of the BnB tree is fathomed due to the fact that there exists no possibility of achieving a reduction in risk by sequential branching/refinement. Further, if $C_\ell \neq \emptyset$, another branching vertex is selected and removed from C_ℓ and added to Q. Otherwise, if $C_\ell = \emptyset$, the algorithm backtracks to level $\ell - 1$ by removing from Q the most recent branching vertex that was used at level $\ell - 1$, namely vertex $q_{\ell-1}$ (see Algorithm 3.1).

With regard to the branching rule, the observed computational performance suggests that branching on a vertex q with the smallest value of $\rho(X_q)$ or $\mathsf{E}X_q$ is most effective. To this end, vertices in the set $C_0 = V$ are pre-sorted during the

initialization phase of the algorithm in descending order with respect to their risks $\rho(X_i)$ or expected values $\mathsf{E}X_i$, and then the last vertex in C_ℓ is selected for branching.

The outlined branch-and-bound procedure for R-MWS problems is formalized as Algorithm 3.1.

Algorithm 3.1 Graph-based branch-and-bound method for R-MWSproblems

1. **Initialize:** $\ell := 0$; $C_0 := V$; $Q := \emptyset$; $Q^* := \emptyset$; $L^* := \infty$
2. **While** $\ell \geq 0$ **do**
3. **if** $C_\ell \neq \emptyset$ **then**
4. select a vertex $q \in C_\ell$;
5. $C_\ell := C_\ell \setminus q$;
6. $Q := Q \cup q$;
7. $C_{\ell+1} := \{i \in C_\ell : i \cup Q \text{ satisfies } \Pi\}$;
8. solve $\mathcal{L}(Q \cup C_{\ell+1})$;
9. **if** $\mathcal{L}(Q \cup C_{\ell+1}) < L^*$ **then**
10. **if** $C_{\ell+1} \neq \emptyset$ **then**
11. $q_l := q$;
12. $\ell := \ell + 1$;
13. **else**
14. $Q^* := Q$;
15. $L^* := \mathcal{L}(Q \cup C_{\ell+1})$;
16. $Q := Q \setminus q$;
17. **else**
18. $Q := Q \setminus q$;
19. **else**
20. $Q := Q \setminus q_{l-1}$;
21. $\ell := \ell - 1$;
22. **return** Q^*

Depending on the particular form of risk measure ρ, evaluation of the lower bound by solving the relaxed problem (3.10) can be relatively expensive and be a ma-

jor contributor to the overall computational cost of the proposed algorithm. However, certain efficiencies in computing the lower bound value via (3.10) can be implemented by taking into account the properties of the subgraph risk function $\mathscr{R}$. Namely, if at any point $(Q \cup C_{\ell+1}) \subseteq (Q' \cup C')$, where Q' and C' are a partial solution and a candidate set for which the lower bound value $\mathcal{L}(Q' \cup C')$ is known to exceed the current global upper bound, $\mathcal{L}(Q' \cup C') \geq L^*$, then $\mathcal{L}(Q \cup C_{\ell+1}) \geq \mathcal{L}(Q' \cup C') \geq L^*$ due to Proposition 3.2. The vertex q under consideration is then removed from Q and the corresponding subproblem is fathomed. In practice, however, retaining the list of sets $(Q' \cup C')$ with $\mathcal{L}(Q' \cup C') \geq L^*$ and checking whether the current $Q \cup C_{\ell+1}$ is a subset of some $Q' \cup C'$ has proven computationally expensive for even moderately sized problems, and is most notably exacerbated in graph topologies that contain a large number of maximal Π-subgraphs (for example, when the graph density increases in the context of risk averse maximum weighted clique problem). Therefore, a more modest approach is considered where only the vertices from incumbent solutions Q^* are retained and tested against unfathomed sets $(Q \cup C_{\ell+1})$.

3.4 Case studies: Risk-averse maximum weighted clique problem with HMCR measures

In this section, we consider two cases of risk-averse maximum weighted clique problems where loss function $\mathbf{X}_G$ describes "propagation" of uncertainties within the network. In the first case, risk exposures of the network vertices are "isolated" in the sense that the loss (risk) profile at vertex i is determined only by the random factor

X_i at that vertex. In the second case, it is assumed that risk exposure of vertex i depends on its own loss profile X_i as well as losses of the adjacent vertices, thus the overall risk of a selected subset S depends not only on the stochastic factors X_i at individual vertices, but also on their exposure to neighboring vertices within S. This assumption reflects risk interdependence observed in many applications like finance and banking, where inter-agency lending heavily exposes counterparties.

In Sections 3.4.1 – 3.4.2 we present mixed integer programming formulations of risk averse maximum clique problem with "isolated" and "neighbor-dependent" stochastic effects, respectively. In Section 3.4.3 numerical simulations demonstrating the solution performances of the proposed BnB method on problems with isolated stochastic effects are conducted. In Section 3.4.4 the solution properties of problems involving isolated and neighbor-dependent risk exposures are examined. We assume that the losses X_i associated with vertices $i \in V$ have a discrete joint distribution that can be represented by scenario set $\Omega = \{\omega_1, \ldots, \omega_N\}$, such that X_{ik} is the realization of a stochastic factor X_i under scenario $k \in N$ with the corresponding probabilities $\mathsf{P}(\omega_k) = \pi_k > 0$, where $\pi_1 + \cdots + \pi_N = 1$.

3.4.1 Case Study I: Isolated risk exposure

In this section we discuss the computational framework and conduct numerical experiments demonstrating the computational performance of the proposed BnB algorithm when solving the risk-averse maximum weighted clique problem (3.7), where

the loss function was given by

$$\mathbf{X}_{G[S]} = \sum_{i \in S} u_i X_i. \tag{3.11}$$

Mathematical programming models containing HMCR measures in the objective or constraints can be formulated using p-order cone constraints (see Section 2.3). Then, the corresponding formulation (3.7) with risk measure $\rho(X)$ selected as $\mathrm{HMCR}_{p,\alpha}(X)$ takes the form of a mixed integer p-order cone programming (MIpOCP) problem:

$$\begin{aligned}
\min \quad & \eta + (1-\alpha)^{-1} t \\
\text{s. t.} \quad & t \geq \|(y_1, \ldots, y_N)\|_p \\
& \pi_k^{-1/p} y_k \geq \sum_{i \in V} u_i X_{ik} - \eta, \quad k = 1, \ldots, N \\
& \sum_{i \in V} u_i = 1 \\
& u_i \leq x_i, \;\; i \in V \\
& x_i + x_j \leq 1, \;\; (i,j) \in \overline{E} \\
& x_i \in \{0,1\}, \; u_i \geq 0, \; i \in V; \;\; y_k \geq 0, \; k = 1, \ldots, N,
\end{aligned} \tag{3.12}$$

where X_{ik} is the realization of the stochastic weight of vertex $i \in V$ under scenario k, $k = 1, \ldots, N$, and the scenario probabilities $\mathsf{P}(X_1 = X_{1k}, \ldots, X_N = X_{Nk}) = \pi_k$. Similarly, the lower bound problem (3.10) for the combinatorial branch-and-bound

algorithm described in the previous section takes the form

$$\begin{aligned}
\mathcal{L}(Q \cup C_{\ell+1}) = \min \quad & \eta + (1-\alpha)^{-1} t \\
\text{s. t.} \quad & t \geq \|y_1, \ldots, y_N\|_p \\
& \pi_k^{-1/p} y_k \geq \sum_{i \in V} u_i X_{ik} - \eta, \quad k = 1, \ldots, N \\
& \sum_{i \in V} u_i = 1 \\
& u_i \geq 0, \;\; i \in Q \cup C_{\ell+1} \\
& u_i = 0, \;\; i \in V \setminus (Q \cup C_{\ell+1}) \\
& y_k \geq 0, \; k = 1, \ldots, N.
\end{aligned} \tag{3.13}$$

Both the described polyhedral approximation approach and SOCP reformulation approach presented in Section 2.3 have been employed in our implementation of the combinatorial BnB algorithm of Section 3.3.2 in the cases when the lower bound problem (3.13) is nonlinear, i.e., when $p > 1$.

Specifically, a polyhedral approximation of the lower bound problem (3.13) was solved at each node of the BnB tree instead of the exact the nonlinear problem (3.13) itself. This allows for a significant reduction in the computational cost of the BnB method, since the warm-start capabilities of LP simplex solvers can be utilized during repeated solving of the approximating LP problem.

The exact solution method that is based on the SOCP reformulation is employed for solving (3.13) once an incumbent solution is found, and the corresponding optimal value is used to update the global upper bound L^*. Due to the fact that the

described polyhedral approximation is an outer approximation, one has

$$\mathcal{L}_{\mathrm{LP}}(Q \cup C_{\ell+1}) \leq \mathcal{L}(Q \cup C_{\ell+1}), \tag{3.14}$$

where $\mathcal{L}_{\mathrm{LP}}(Q \cup C_{\ell+1})$ is the optimal value given by the polyhedral (LP) approximation of the lower bound problem. This implies that for any $Q \cup C_{\ell'+1}$ containing an incumbent solution Q^*, the following holds

$$\mathcal{L}_{\mathrm{LP}}(Q \cup C_{\ell'+1}) \leq \mathcal{L}_{\mathrm{LP}}(Q^*) \leq \mathcal{L}(Q^*) = L^*,$$

which guarantees the correctness of the BnB algorithm relying on polyhedral approximations. Note, however, that inequality (3.14) also implies that the use of polyhedral approximations instead of the exact nonlinear formulation of the lower bound problem (3.13) allows for delayed pruning of "non-promising" branches of the BnB tree in situations when

$$\mathcal{L}_{\mathrm{LP}}(Q \cup C_{\ell+1}) < L^* \leq \mathcal{L}(Q \cup C_{\ell+1}).$$

Still, in our experience, the computational savings due to the use of polyhedral approximations during the BnB procedure greatly outweigh the costs of possible delayed pruning.

Note also that in the special case of $p = 1$, when $\rho(X) = \mathrm{CVaR}_\alpha(X)$, the lower bound problem (3.13) is an LP problem and thus requires no polyhedral approximation or SOCP reformulation.

3.4.2 Case Study II: Neighbor-dependent risk exposures

Loss functions (3.11) only considered isolated stochastic effects such that the uncertainties affecting one vertex did not impact neighboring vertices. However, as

discussed above, many physical network structures frequently do exhibit shared risk exposure. Thus, it is of interest to consider a form of loss function $\mathbf{X}_G$ that reflects this observation by allowing the risk exposure of a selected vertex i to depend on its own loss profile X_i in addition to the loss profiles of the adjacent vertices included in the selected subset S:

$$\mathbf{X}_{G[S]} = \sum_{i \in S} \left(u_i X_i + \sum_{j \in S \setminus i} \theta_{ij} u_j X_j \right), \tag{3.15}$$

where the parameters θ_{ij} denote the degree of exposure of vertex i to vertex j. It is natural to assume that exposure θ_{ij} is non-zero only if an edge exists between i and j: $(i, j) \in E$. Although the meaning of θ_{ij} ultimately depends on the model application, for simplicity we assume that each vertex in V has uniform exposure to its neighbors:

$$\theta_{ij} = \begin{cases} 1/|V|, & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases}$$

Observe that (3.15) can equivalently be expressed in the form

$$\mathbf{X}_{G[S]} = \sum_{i \in S} u_i X_i \left(1 + \sum_{j \in S \setminus i} \theta_{ji} \right), \tag{3.16}$$

that is similar to the form of loss function (3.11) with isolated exposures if one considers the stochastic factor at vertex i to be defined as $\tilde{X}_i = X_i \big(1 + \sum_{j \in S \setminus i} \theta_{ji}\big)$. Note that in the case when S represents a complete graph, risk profile $\tilde{X}_i$ is dependent on the selected subset of vertices S, and, consequently, on the risk profiles of all neighbors of i.

Introducing binary variables x_i as before, the the *risk-averse maximum clique*

problem with neighbor-dependent risk exposures can be formulated as

$$\min \quad \rho\left(\sum_{i\in V}\left(u_i\, x_i\, X_i + \sum_{j:(i,j)\in E}\theta_{ij}\, x_i\, x_j\, u_j\, X_j\right)\right) \tag{3.17a}$$

$$\text{s. t.} \quad \sum_{i\in V} u_i = 1 \tag{3.17b}$$

$$u_i \le x_i, \quad \forall\ i \in V \tag{3.17c}$$

$$x_i + x_j \le 1, \quad \forall\ (i,j) \in \overline{E} \tag{3.17d}$$

$$x_i \in \{0,1\},\ \ u_i \ge 0,\ \ \forall\ i \in V. \tag{3.17e}$$

where constraint (3.17c) allows for replacing products $u_i x_i$ in the objective with just u_i. Also, selecting function ρ in the objective as the HMCR measure, problem (3.17) reduces to a nonlinear 0–1 mixed integer stochastic optimization problem of the form

$$\min \quad \eta + (1-\alpha)^{-1} t \tag{3.18a}$$

$$\text{s. t.} \quad t \ge \|w_1, \ldots, w_N\|_p \tag{3.18b}$$

$$\pi_k^{-1/p} w_k \ge \sum_{i\in V}\left(X_{ik}\, u_i + \sum_{j:(i,j)\in E}\theta_{ij} X_{jk}\, u_j\, x_i\right) - \eta, \quad \forall\ k = 1, \ldots, N \tag{3.18c}$$

$$\sum_{i\in V} u_i = 1 \tag{3.18d}$$

$$u_i \le x_i, \quad \forall\ i \in V \tag{3.18e}$$

$$x_i + x_j \le 1, \quad \forall\ (i,j) \in \overline{E} \tag{3.18f}$$

$$x_i \in \{0,1\},\ \ u_i \ge 0,\ \ \forall\ i \in V;\ \ w_k \ge 0, \quad k = 1, \ldots, N. \tag{3.18g}$$

The remaining non-linear terms $u_j\, x_i$ in the constraint (3.18c) can be linearized by

introduction of auxiliary variables γ_{ij} as follows:

$$\gamma_{ij} \leq u_j, \quad \forall\, i,j \in V$$

$$\gamma_{ij} \leq u_i, \quad \forall\, i,j \in V$$

$$\gamma_{ij} \geq u_j + x_i - 1, \quad \forall\, i,j \in V$$

$$\gamma_{ij} \geq 0, \quad \forall\, i,j \in V.$$

The following mixed-integer linear formulation for problem (3.17) is then obtained:

$$\begin{aligned}
\min \quad & \eta + (1-\alpha)^{-1} t \\
\text{s. t.} \quad & t \geq \|w_1, \ldots, w_N\|_p \\
& w_k \geq \sum_{i \in V} \left(X_{ik}\, u_i + \sum_{j:(i,j)\in E} \theta_{ij} X_{jk}\, \gamma_{ij} \right) - \eta, \quad k = 1, \ldots, N \\
& \sum_{i \in V} u_i = 1 \\
& u_i \leq x_i, \quad \forall\, i \in V \qquad (3.19) \\
& x_i + x_j \leq 1, \quad \forall\, (i,j) \in \overline{E} \\
& \gamma_{ij} \leq u_j, \quad \forall\, i,j \in V \\
& \gamma_{ij} \leq x_i, \quad \forall\, i,j \in V \\
& \gamma_{ij} \geq u_j + x_i - 1, \quad \forall\, i,j \in V \\
& x_i \in \{0,1\},\ u_i \geq 0,\ \gamma_{ij} \geq 0,\ \forall\, i,j \in V;\ w_k \geq 0, \quad k = 1, \ldots, N.
\end{aligned}$$

Observe that the additional complexity of formulation (3.19) in comparison to (3.12) attributes to the fact that the risk exposure of a vertex in solution set S depends not only on its own risk profile, but also on risk profiles of adjacent vertices included in the solution set.

3.4.3 Numerical experiments for Case Study I

Numerical studies of the risk-averse maximum weighted clique problem with "isolated" risk exposure were conducted on randomly generated Erdös-Rényi graphs [19] of orders $|V| = 50, 100, 150, 200$ and average densities $d = 0.2, 0.5,$ and 0.8. The stochastic weights of graphs' vertices were generated as i.i.d. samples from the uniform $U[0, 1]$ distribution. In particular, we generated scenario sets with $N = 50, 100, 200, 500, 1000$ scenarios for each combination of graph order and density. The risk measure ρ has been selected as an HMCR measure (1.7) with $p = 1, 2, 3$ and $\alpha = 0.9$.

The combinatorial branch-and-bound algorithm of Section 3.3.2 with the additional specializations described above has been coded in C++, and we used the CPLEX Simplex and Barrier solvers for solving the polyhedral approximations and SOCP reformulations of the p-order cone programming lower bound problem (3.13), respectively. In the case of $p = 1$, the CPLEX Simplex solver was used to solve the lower bound problem directly.

The performance of the developed BnB method was compared with that of the mathematical programming formulation (3.12) of the risk-averse maximum weighted clique problem. The MIpOCP problem (3.12) was solved with CPLEX MIP solver in the case of $p = 1$, and CPLEX MIP Barrier solver was applied to the SOCP version of (3.12) in the case of $p = 2$ or SOCP reformulation of (3.12) in the case of $p = 3$.

The computations were ran on an Intel Xeon 3.30GHz PC with 128GB RAM, and version 12.5 of the CPLEX solver in Windows 7 64-bit environment was used.

The numerical experiments are summarized in Tables 3.1 – 3.2. All reported running times were averaged over twenty instances and symbol "—" indicates that the time limit of 7200 seconds was exceeded. Table 3.1 summarizes the computational times corresponding to the aforementioned problem configurations with a fixed number of scenarios of $N = 100$. Observe that the BnB algorithm provides one to two orders of magnitude advantage in running time over the CPLEX MIP solver for all configurations, except that of $p = 1$ and $d = 0.8$. For the consecutive set of experiments, Table 3.2 demonstrates the effect of variations in the scenario size N for different graph orders and values of p while maintaining a constant average graph density of $d = 0.5$. The specified edge probability was chosen due to the fact that the size of the mathematical programming (3.12) formulation is density dependent. Mainly, the number of structural constraints $x_i + x_j \leq 1, (i, j) \in \overline{E}$ in (3.12) increases as d decreases. The opposite relationship holds true for the BnB algorithm, as the search space expands with the number of edges. Thus, a "fair" comparison between the two solution methods can be made on graphs with density $d = 0.5$.

It follows from Tables 3.1 and 3.2 that the computational advantages of the combinatorial BnB algorithm over the direct solution approach become more pronounced (up to two orders of magnitude) with increase in p, i.e., as full formulation (3.12) and the lower bound problem (3.13) become more difficult. Also of interest is the fact that the BnB method often yields better solution times for problems with $p = 3$ than $p = 2$. This is a consequence of a known property of the employed cutting-plane algorithm for solving polyhedral approximations of p-order cone programming

problems, which becomes more effective as p increases [30].

Table 3.1: Average computation times (in seconds) obtained by solving problem (3.12) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and scenarios $N = 100$.

		$d = 0.2$		$d = 0.5$		$d = 0.8$	
p	$\|V\|$	BnB	CPLEX	BnB	CPLEX	BnB	CPLEX
1	50	0.08	1.10	0.37	1.31	3.04	1.90
	100	0.24	6.43	4.02	28.06	206.46	121.27
	150	0.74	38.37	26.86	220.17	4065.16	2434.66
	200	1.67	118.13	73.73	1074.93	—	—
2	50	0.40	18.54	1.66	45.67	14.50	156.26
	100	1.38	110.67	19.37	412.90	956.93	2555.77
	150	3.37	629.38	124.99	2293.96	6154.76	—
	200	3.68	2822.38	166.44	—	—	—
3	50	1.35	54.58	2.38	91.98	14.15	273.10
	100	2.43	215.97	17.66	625.52	716.22	4644.90
	150	4.41	927.03	102.28	3560.27	—	—
	200	7.24	3031.77	412.74	—	—	—

3.4.4 Numerical experiments for Case Study II

In this section we conduct numerical experiments demonstrating graph structural solution properties of problems (3.12) and (3.19). In particular, of specific interest in the presented case study were the sizes of the optimal risk-averse cliques produced by both formulations in comparison to the maximum clique size in the respective graphs obtained without considering stochastic effects, i.e., as a solution of

Table 3.2: Average computation times (in seconds) obtained by solving problem (3.12) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and edge density $d = 0.5$.

		$\|V\| = 50$		$\|V\| = 100$		$\|V\| = 150$		$\|V\| = 200$	
p	N	BnB	CPLEX	BnB	CPLEX	BnB	CPLEX	BnB	CPLEX
1	50	0.19	1.15	1.40	11.88	4.43	43.49	13.09	130.45
	100	0.37	1.31	4.02	28.06	26.86	220.17	73.73	1074.93
	200	0.87	3.01	14.64	71.93	84.83	443.74	329.76	2550.12
	500	4.70	10.40	72.90	219.40	429.80	1794.60	2118.60	—
	1000	14.87	28.82	259.48	702.97	1909.48	6094.66	—	—
2	50	0.80	22.96	4.10	167.30	12.89	961.32	37.67	3668.54
	100	1.66	45.67	19.37	412.90	124.99	2293.96	166.44	—
	200	6.57	109.72	131.44	907.95	797.04	5961.69	900.50	—
	500	61.10	552.80	970.10	—	3221.70	—	—	—
	1000	194.59	965.69	3669.37	—	—	—	—	—
3	50	1.22	34.85	3.96	245.79	11.99	1040.01	34.30	3847.40
	100	2.38	91.98	17.66	625.52	102.28	3560.27	412.74	—
	200	5.21	261.83	60.59	2388.44	333.61	—	1424.27	—
	500	20.10	1299.60	248.70	—	1751.90	—	—	—
	1000	58.00	3277.93	768.53	—	5634.04	—	—	—

problem

$$\begin{aligned} \max \quad & \sum_{i \in V} x_i \\ \text{s. t.} \quad & x_i + x_j \leq 1, \quad \forall\ (i,j) \in \overline{E} \\ & x_i \in \{0,1\}, \quad \forall\ i \in V, \end{aligned} \tag{3.20}$$

with variables x_i defined as before.

For the purpose of examining the optimal solution properties resulting from problems (3.12) and (3.19), we only consider risk measure ρ to be chosen as CVaR (1.6), $\rho(X) = \mathsf{CVaR}_\alpha(X)$. Randomly generated Erdös-Rényi graphs and random scenario data corresponding to each vertex $i \in V$ were generated as previously. In what follows, a total of twenty instances of problems (3.12), (3.19) and (3.20) have been solved for each combination of graph size/scenario set, and all reported optimal clique sizes were averaged accordingly. To demonstrate the effect of the degree of risk aversion on the size of risk-averse maximum clique as given by the confidence level α of the CVaR measure, we also solve (3.12) and report the average clique size for various levels of α. We compared the average sizes of risk-averse maximum cliques as given by (3.12) or (3.19) with the average size of deterministic maximum clique as given by (3.20) over the same randomly generated graph instances. Finally, optimal clique sizes in problems (3.12) and (3.19) for a fixed graph instance size with varying levels of parameter α are compared.

Table 3.3 demonstrates averaged optimal clique sizes with respective computation times for the discussed implementations of problems (3.12) and (3.20) for randomly generated graphs of sizes $|V| = 50, 75, 100, 150, 200$ and average densities

$d = 0.25, 0.5, 0.75$. In all instances, the number of scenarios (i.e., realizations of the vector $(X_1, \ldots, X_{|V|})$) was fixed at 50, and the confidence level of the CVaR measure was chosen as $\alpha = 0.9$. One of the observations that can be drawn from Table 3.3 is that the average sizes of risk-averse maximum cliques are smaller than the corresponding sizes of risk-neutral (deterministic) maximum cliques.

Table 3.3: Average optimal clique sizes obtained by solving problems (3.12) and (3.20) with risk measures (1.6).

d	$\|V\|$	N	Clique size	Risk-averse clique size
0.25	25	50	3.6	3.5
	50	50	4.9	3.9
	75	50	5.2	4.2
	100	50	5.4	4.5
	150	50	6.1	4.6
	200	50	6.6	4.7
0.5	25	50	5.7	5.1
	50	50	7.8	6.3
	75	50	8.4	6.4
	100	50	9.2	6.6
	150	50	10.1	7.2
	200	50	11	7.8
0.75	25	50	9.6	8
	50	50	12.7	9.7
	75	50	15.6	11.1
	100	50	16.7	12.5
	150	50	19.1	12.8
	200	50	20.9	14.4

We next examine optimal subgraph sizes obtained by solving problem (3.12) in relation to varying levels of parameter α (note that larger values of α correspond to more risk-averse preferences) in the CVaR measure (1.6). We let α take values

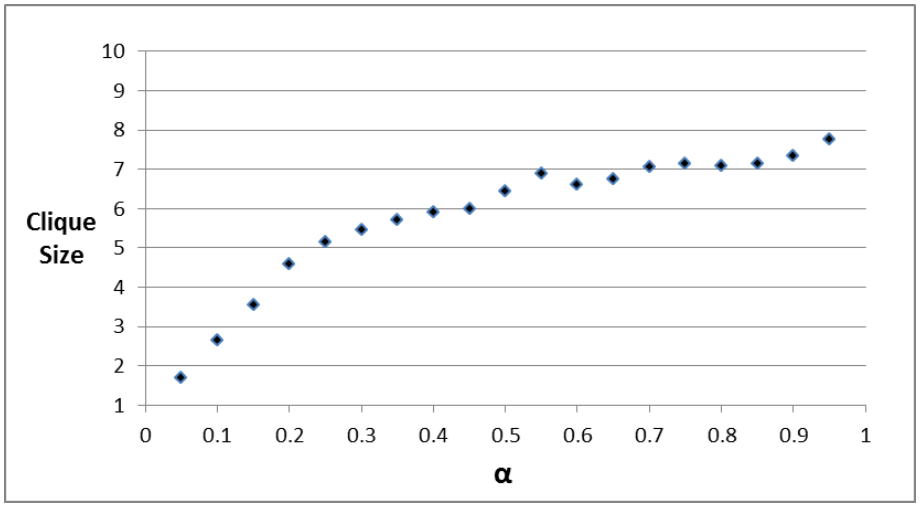

Figure 3.1: Average optimal clique sizes obtained by solving problem (3.12) at varying risk tolerances α using graphs with 150 vertices and 50 scenarios ($d = 0.50$).

within the range $[0.05, 0.95]$ at increments of 0.05 in randomly generated graphs with 150 vertices, an average density of $d = 0.50$ and 50 scenarios. Figure 3.1 establishes a strong relation between α and average optimal subgraphs. Noteworthy are low α levels (e.g. $\alpha = 0.05$) which occasionally reduce the optimal subgraph to a single vertex, confirming that lax risk requirements prevent appropriate, if any, diversification among multiple vertices. Furthermore, a transition from low risk tolerance inducing no diversification ($\alpha \approx 0.05$) towards effectual levels ($\alpha \geq 0.1$) expands optimal clique sizes at high rates over interval $\alpha \in [0.05, 0.25]$, while consecutive restrictions have a more moderate impact. This outcome is consistent with properties of coherent risk measures, such as CVaR, which allow for efficient diversification due to the subadditivity/convexity property.

The results shown in Figure 3.1 can be illustrated through financial settings,

where lax risk constraints are commonly associated with little or no asset diversification, whereas stricter requirements correspond to increased risk aversion leading to improved diversification. In a network setting, specifically problem (3.12), we can analogously express lacking diversification over vertices for insufficiently large α-levels corresponding to low degree of risk aversion. Initial incremental increases in α are reflected in steep clique size growth rates, with a dissipating effect as $\alpha \to 1$.

In regard to neighbor-dependent exposure risk considerations, for construction of problem (3.18) we impose $\theta_{ij} = 1/|V|$ over all vertices $i \in V$ and conduct computational simulations for graphs of sizes $|V| = 25, 50, 75$ and average densities of $d = 0.25, 0.50$. In each problem instance the distribution of uncertainties was modeled using 50 scenarios and the confidence level α of the CVaR measure was set at $\alpha = 0.9$. Table 3.4 reports the resulting average clique sizes and corresponding computation times when risk exposures of the vertices depend on the risk profiles of their neighbors. Observe that optimal risk-averse clique sizes are significantly smaller on average in comparison to the same instances in Table 3.3. Furthermore, in Figure 3.2 we demonstrate that problem (3.19) consistently requires higher levels of α to attain similar optimal subgraph sizes.

Table 3.4: Average optimal clique sizes obtained by solving problems (3.19) and (3.20) with risk measures (1.6).

d	$\|V\|$	N	Clique size	Risk-averse clique size
0.25	25	50	3.6	3.1
	50	50	4.9	3.3
	75	50	5.2	3.9
0.5	25	50	5.7	4.2
	50	50	7.8	4.8
	75	50	8.4	6.1

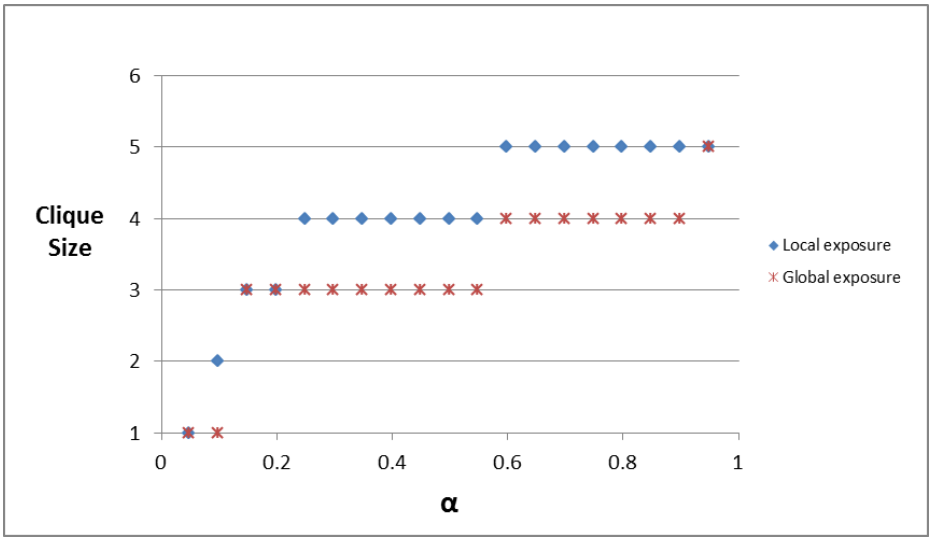

Figure 3.2: Comparative optimal clique sizes at ranging risk tolerances α for problems (3.12) and (3.19) using a single graph with 50 vertices and 50 scenarios.

3.5 Conclusions

We have considered a class R-MWS problems which entail finding a network subgraph of minimum risk satisfying hereditary structural properties. A distinguishing feature of the problem setting considered in this study is the assumption that stochastic factors in the underlying networks are associated with vertices, as opposed to the prevalent literature settings where uncertainties are attributed to network arcs. We employ the HMCR measures as a rigorous framework for quantifying the distributional information of the stochastic vertex weights. By means of diversification properties of the introduced optimization-based risk function for measuring risk of subgraphs, it was shown that the inclusion of additional vertices in a partial solution promotes the minimization of risk; hence, optimal solutions to R-MWS problems are maximal subgraphs. A combinatorial branch-and-bound algorithm utilizing the risk- and graph-related aspects of the problem structure was developed and tested on a special case of the risk-averse maximum clique problem.

Numerical experiments on randomly generated Erdös-Rényi graphs demonstrate that the proposed algorithm may significantly reduce solution times relative to an equivalent mathematical programming counterpart. Notably, when the desired subgraph property defined a clique, improvements were observed for all the tested graph configurations when using the HMCR measures with $p = 2, 3$, and for graphs with edge probabilities of less than 0.8 when using an HMCR measure with $p = 1$. Properties of optimal risk-averse stochastic weighted maximum cliques with "isolated" and "neighbor-dependent" risk exposures were examined.

CHAPTER 4
ON RISK-AVERSE WEIGHTED K-CLUB PROBLEMS

4.1 Introduction

In the previous chapter we addressed problems seeking subgraphs of minimum risk that posses hereditary properties, and focused on particular instances that defined property Π as a clique (complete-graph). In many practical applications, the requirement that the desired subgraph must be complete may, however, impose excessive restrictions, and warrant some structural relaxation in terms of member connectivity. For instance, rather than identifying a subset with all vertices pairwise adjacent, an adjacency requirement based on a distance or degree threshold may suffice. As a consequence, several clique relaxation models have been proposed in graph theory literature. A comprehensive review on clique relaxation models is provided in [53].

In this work we focus on a specific model, the *k-club* [1], where subgraph members may also be indirectly connected via at most k intermediary members. We adopt this setting and extend the techniques introduced in Chapter 3 to address problems seeking subgraphs of minimum risk that represent a k-club. An analogous framework utilizing the information of stochastic vertex weights by means of coherent risk measures is employed to define a *risk-averse maximum weighted k-club* (R-MWK) problem as finding the lowest risk k-club in a network. As an illustrative example, we focus on instances when $k = 2$ and utilize a mathematical programming formulation for the maximum 2-club problem introduced in [10]. A branch-and-bound

method for finding maximum k-clubs [41] is modified to accommodate the conditions of R-MWK problems by bounding solutions in a coherent risk measure context. We compare the solution performance of the proposed algorithm relative to an equivalent mathematical programming counterpart problem for R-MWK problems when $k = 2$. The remainder of the paper is organized as follows. In Section 4.2 we introduce several clique relaxation models and examine the general representation of R-MWK problems . Section 4.3 presents a mathematical programming formulation and a combinatorial branch-and-bound method for R-MWK problems with $k = 2$. Finally, Section 4.4 furnishes numerical studies demonstrating the computational performance of the developed branch-and-bound method on problems where risk is quantified using higher-moment coherent risk measures.

4.2 Risk-averse stochastic maximum k-club problem

As previously, let $G = (V, E)$ be a graph where $G[S]$ represents the subgraph of G induced by S such that any pair of vertices (i, j) share an edge in S only if (i, j) is an edge in G. To ease notation, define $\mathcal{Q}$ as a desired property which the induced graph $G[S]$ must satisfy. The present case considers the instances when $\mathcal{Q}$ represents a certain relaxation of the *completeness* property, such that a subgraph with property $\mathcal{Q}$ represents a *clique relaxation*.

Depending on the characteristic of a complete graph that is relaxed, the clique relaxations can be categorized into *density-based*, *degree-based* and *diameter-based* relaxations. The density of a graph $G = (V, E)$ is defined as a ratio $D(G) = |E|/\binom{|V|}{2}$,

where the denominator represents the number of edges in a complete graph with $|V|$ vertices. Evidently, a complete graph has a density of 1. Then, for a fixed $\gamma \in (0, 1)$, graph G is called a γ-*quasi-clique* [55], if its density is at least γ:

$$D(G) \geq \gamma, \quad \text{or, equivalently,} \quad |E| \geq \gamma \binom{|V|}{2}.$$

The γ-quasi-clique is, therefore, a density-based relaxation of the clique concept, and as such is different from the k-*clique*, which is one of the diameter-based clique relaxations. Namely, let $d_G(i, j)$ be the distance between nodes $i, j \in V$, measured as the number of edges in the shortest path between i and j in G. Then, the subgraph $G[S]$ induced by a subset of nodes $S \subset V$ of the graph G is called a k-clique if

$$\max_{i,j \in S} \; d_G(i, j) = k.$$

Note that the definition of the k-clique does not require that the shortest path between $i, j \in S$ belong to $G[S]$. If one requires that the shortest path between any two vertices i, j in S belong to the induced subgraph $G[S]$, then the subset S such that

$$\max_{i,j \in S} \; d_{G[S]}(i, j) = k, \tag{4.1}$$

is called a k-*club*. Note that a k-club is also a k-clique, while the inverse is not true in general. The shortest path connecting two vertices in a clique is 1, thus 1-clique and 1-club are cliques. For a vertex $i \in V$, its *degree* $\deg_G(i)$ is defined as the number of adjacent vertices: $\deg_G(i) = |\{j \in V : (i, j) \in E\}|$. A degree-based clique relaxation, known as k-*plex,* is defined as a subset S of V such that the degree of each vertex in the induced subgraph $G[S]$ is at least $|S| - k$ [9]:

$$\deg_{G[S]}(i) \geq |S| - k \quad \text{for all} \quad i \in S,$$

(observe that the degree of each vertex in a clique of size n is equal to $n-1$).

In what follows, we consider the case when $\mathcal{Q}$ represents a distance-based relaxation of the clique model in the sense of the k-club definition (4.1) when $k \geq 2$. Throughout the remainder of this study we let property $\mathcal{Q}_{G[S]}$ define a k-club as

$$\mathcal{Q}_{G[S]} = \{S \subseteq V \mid \forall i, j \in S : d_{G[S]}(i,j) \leq k\}. \tag{4.2}$$

When seeking a subgraph S with the maximum additive vertex weights, $w_i > 0$, that satisfies property $\mathcal{Q}_{G[S]}$, a *maximum weight k-club problem* can take the form

$$\max_{S \subseteq V} \Big\{ \sum_{i \in S} w_i : G[S] \text{ satisfies } \mathcal{Q}_{G[S]} \Big\}. \tag{4.3}$$

Clearly, the optimal subgraph $G[S]$ in problem (4.3) will be maximal, but not necessarily the maximum subgraph with property $\mathcal{Q}_{G[S]}$. We next extend the techniques introduced in Chapter 3 to propose problems seeking subgraphs of minimum-risk that conform to property $\mathcal{Q}$,

$$\min_{S \subseteq V} \big\{ \mathscr{R}(S; \mathbf{X}_G) : G[S] \text{ satisfies } \mathcal{Q} \big\}, \tag{4.4}$$

and, particularly, where $\mathcal{Q}$ represents a k-club.

4.3 Solution approaches for risk-averse maximum weighted 2-club problems

In this section we consider a mathematical programming formulation for the R-MWK problem when $k = 2$, and where the risk $\mathscr{R}(S)$ of induced subgraph $G[S]$ is defined by (3.3). Also, we propose a combinatorial branch-and-bound algorithm

utilizing the solution space processing principals for finding maximum k-clubs introduced by Pajouh and Balasundaram [41].

4.3.1 A mathematical programming formulation

When the property $\mathcal{Q}$ denotes a 2-club, one can choose the *edge formulation* of the maximum 2-club problem proposed by Balasundaram et al. [10], whereby the mathematical programming formulation of the R-MWS problem with $k = 2$ takes the form

$$\begin{aligned} \min \quad & \rho\Big(\sum_{i\in V} u_i\, X_i\Big) \\ \text{s. t.} \quad & \sum_{i\in V} u_i = 1 \\ & u_i \le x_i, \quad i \in V \\ & x_i + x_j - \sum_{l\in\mathcal{N}^{\cap}(i,j)} x_l \le 1, \quad (i,j) \in \overline{E} \\ & x_i \in \{0,1\}, \;\; u_i \ge 0, \quad i \in V, \end{aligned} \tag{4.5}$$

where $\overline{E}$ represents the complement edges of graph G, and $\mathcal{N}^{\cap}(i,j)$ denotes the vertices that are both adjacent to vertex i and vertex j. A combinatorial branch-and-bound algorithm for solving R-MWK problems is described next.

4.3.2 A graph-based branch-and-bound algorithm

The following BnB algorithm for solving R-MWK problems entails efficient processing of solution space by traversing "levels" of the BnB tree until a subgraph $G[S]$ that represents a maximal 2-club of minimum risk in G as measured by (3.3) is found. The algorithm begins at level $\ell = 0$ with a partial solution $Q := \emptyset$, incumbent

solution $Q^* := \emptyset$, and an upper bound on risk $L^* := +\infty$, where Q consists of the vertices of the induced subgraph with property $\mathcal{Q}$, and Q^* contains vertices corresponding to a maximal $\mathcal{Q}$-subgraph whose risk equals L^* in G. A set of candidate vertices C_ℓ is maintained at each level ℓ, from which a certain branching vertex q is selected and added to the partial solution Q, or simply deleted from set C_ℓ without being added to Q. In order to ensure that the proper vertices are removed from Q when the algorithm backtracks between levels of the BnB tree, we introduce set $F := \emptyset$ to account for the levels at which nodes were created to delete a vertex q from C_ℓ.

Due to the distance-based properties of k-clubs, considerations are warranted upon transferring or deleting a vertex q from candidate set C_ℓ, as the structural integrity of corresponding to the graph induced by Q and the candidate set at the subsequent level $C_{\ell+1}$ may be affected. Thus, the removal of q from C_ℓ to add to Q, and the deletion of q from C_ℓ without adding it to Q are considered independently via the construction of two BnB tree nodes for any given current node at level ℓ. The first node is created to include q in Q, while the other to delete q from C_ℓ. The necessary structural properties of Q and $C_{\ell+1}$ at each node are described next.

Consider a k-clique in graph G as a subset S that satisfies

$$\{S \subseteq V \mid \forall i, j \in S : d_G(i, j) \leq k\},$$

and observe that any k-club in G also satisfies the properties of a k-clique, while a k-clique is not necessarily a k-club for $k \geq 2$. Further, both reduces to a complete graph in the case of $k = 1$. By this notion, an incumbent solution Q^* defines a k-club

if the following conditions are maintained for all graphs $G[Q \cup C_{\ell+1}]$:

(C1) Q is a k-clique in $G[Q \cup C_{\ell+1}]$

(C2) $d_{G[Q \cup C_{\ell+1}]}(i,j) \leq k, \ \ \forall i \in Q, \ \ \forall j \in C_{\ell+1}$

The algorithm is then initialized with $C_0 := V$. Whenever a vertex q is selected from C_ℓ and added to Q, the candidate set at level $\ell+1$ must be accordingly constructed by removing all vertices from C_ℓ whose distances to vertex q are larger than k,

$$C_{\ell+1} := \{j \in C_\ell : d_{G[Q \cup C_\ell]}(q,j) \leq k\}.$$

In situations when the deleted vertices serve as intermediaries, their removal from C_ℓ may, however, impose pairwise distance violations among the vertices in $Q \cup q$ with respect to condition (C2). In other words, after removing vertex q from C_ℓ, the distance between a pair of vertices $(i,j) \in Q$ follows $d_{G[Q \cup C_{\ell+1}]}(i,j) > k$. In such cases, the corresponding vertex of the BnB tree is fathomed and the algorithm backtracks to level ℓ. If a BnB tree node is created to delete vertex q, the candidate set $C_{\ell+1}$ is likewise constructed by eliminating vertices that violate (C2). If the removal of vertices from the candidate sets in either of the above cases results in a violation of (C1), then the corresponding BnB node is fathomed.

Subsequent steps of evaluating the quality of the solution via problem (3.10) and the processing of search space remain unchanged with regard to the algorithm described in Section 3.3.2. The outlined BnB for R-MWK problems is formalized in Algorithm 4.1.

Algorithm 4.1 Graph-based branch-and-bound method for R-MWK

1. **Initialize:** $\ell := 0$; $C_0 := V; Q := \emptyset$; $Q^* := \emptyset$; $L^* := \infty; F := \emptyset$;
2. **While** (**not** STOP) **do**
3. **if** $C_\ell \neq \emptyset$ **then**
4. select a vertex $q \in C_\ell$;
5. $C_\ell := C_\ell \setminus q$;
6. $Q := Q \cup q$;
7. $C_{\ell+1} := \{i \in C_\ell : d_{G[Q \cup C_\ell]}(q, i) \leq k \; \forall i \in C_\ell\}$;
8. **if** Q is a k-clique in $G[Q \cup C_{\ell+1}]$ **then**
9. solve $\mathcal{L}(Q \cup C_{\ell+1})$;
10. **if** $\mathcal{L}(Q \cup C_{\ell+1}) < L^*$ **then**
11. **if** $C_{\ell+1} \neq \emptyset$ **then**
12. $\ell := \ell + 1$;
13. **else**
14. $Q^* := Q$;
15. $L^* := \mathcal{L}(Q \cup C_{\ell+1})$;
16. $Q := Q \setminus q$;
17. **if** $\ell \notin F$ **then**
18. $Q := Q \setminus q$
19. $C_{\ell+1} := \{j \in C_\ell : d_{G[Q \cup C_\ell]}(i, j) \leq k, \; \forall i \in Q, \}$;
20. **if** $C_{\ell+1} \neq \emptyset$ **then**
21. **if** Q is a k-clique in $G[Q \cup C_{\ell+1}]$ **then**
22. $F := F \cup \ell$;
23. go to step 9;
24. **else**
25. go to step 3;
26. **else**
27. $F := F \setminus \ell$;
28. **else**
29. **if** $\ell \notin F$ **then**
30. $Q := Q \setminus q$;
31. **else**
32. $F := F \setminus \ell$;
33. **else**
34. $Q := Q \setminus q$;
35. $C_{\ell+1} := \{j \in C_\ell : d_{G[Q \cup C_\ell]}(i, j) \leq k, \; \forall i \in Q, \}$;
36. **if** Q is a k-clique in $G[Q \cup C_{\ell+1}]$ **then**
37. $F := F \cup l$;
38. go to step 9;
39. **else**
40. go to step 3;
41. **else**
42. $\ell := \ell - 1$;
43. **if** $\ell = -1$ **then**
44. STOP
45. **if** $\ell \notin F$ **then**
46. $Q := Q \setminus q$;
47. **else**
48. $F := F \setminus l$;
49. **return** Q^*

4.4 Case study: Risk-averse maximum weighted 2-club problem with HMCR measures

We consider a R-MWK problem with $k = 2$ where random factors X_i at vertex $i \in V$ are isolated in the sense that its loss profile is independent of stochastic factors at other vertices. Assuming that the loss function $\mathbf{X}_G$ is defined by expression (3.11) and is quantified using the HMCR measure (1.7), the mathematical programming formulation (4.5) with risk measure $\rho(X)$ as $\text{HMCR}_{p,\alpha}(X)$ takes the form of a mixed integer p-order cone programming problem:

$$
\begin{aligned}
\min \quad & \eta + (1-\alpha)^{-1} t \\
\text{s. t.} \quad & t \geq \|(w_1, \ldots, w_N)\|_p \\
& \pi_k^{-1/p} w_k \geq \sum_{i \in V} u_i X_{ik} - \eta, \quad k = 1, \ldots, N \\
& \sum_{i \in V} u_i = 1 \\
& u_i \leq x_i, \quad i \in V \\
& x_i + x_j - \sum_{l \in \mathcal{N}^{\cap}(i,j)} x_l \leq 1, \quad (i,j) \in \overline{E} \\
& x_i \in \{0,1\},\ u_i \geq 0,\ i \in V;\ \ w_k \geq 0,\ k = 1, \ldots, N.
\end{aligned}
\tag{4.6}
$$

Observe that upon exclusion of the graph structural inequalities and variables, an equivalent lower bounding form to problem (3.13) is obtained for implementation in the combinatorial BnB algorithm.

4.4.1 Numerical experiments and results

Numerical studies of the risk-averse maximum weighted 2-club problem were conducted on randomly generated Erdös-Rényi graphs of orders $|V| = 25, 50, 100$ with average densities $d = 0.0125, 0.025, 0.05, 0.1, 0.15$. The specified edge probabilities were chosen due to empirical observations indicating that graphs of order $|V| \geq 50$ commonly reduces to a 2-club when the density is in the range [0.15,0.25]. The stochastic weights of graphs' vertices were generated as i.i.d. samples from the uniform $U[-0.5, 0.5]$ distribution. Scenario sets with $N = 100$ were generated for each combination of graph order and density. The HMCR risk measure (1.7) with $p = 1, 2, 3$ and $\alpha = 0.9$ was used.

The BnB algorithm of Section 4.3 has been coded in C++, and we used the CPLEX Simplex and Barrier solvers for solving the polyhedral approximations and SOCP reformulations of the p-order cone programming lower bound problem (3.13), respectively. In the case of $p = 1$, the CPLEX Simplex solver was used to solve the lower bound problem directly.

The computational performance of the mathematical programming model (4.6) was compared with that of developed BnB algorithm. In the case of $p = 1$, problem (4.6) was solved with CPLEX MIP solver. The CPLEX MIP Barrier solver was used for the SOCP version in the case of $p = 2$, and using the SOCP reformulation in the case of $p = 3$.

The experimental results are furnished at the end of this chapter in Table 4.1, where computation times were averaged over five instances and the symbol "—"

indicates that the time limit of 7200 seconds was exceeded. Observe that the BnB algorithm outperforms the CPLEX MIP solver over all the listed graph configurations, and one to two orders of magnitude in performance improvements were witnesses for the majority of instances. Similarly to the results in the previous chapter, the relative performances also becomes more pronounces with an increase in p.

4.5 Conclusions

We have considered a R-MWK problems which entail finding a k-club of minimum risk in a graph. HMCR risk measures were utilized for quantifying the distributional information of the stochastic factors associated with vertex weights. It was shown that the optimal solutions to R-MWK problems are maximal k-clubs. A combinatorial BnB solution algorithm was developed and tested on a special case of the R-MWK problem when $k = 2$. Numerical experiments on randomly generated graphs of various configurations suggest that the proposed BnB algorithm significantly reduces solution times in comparison with the mathematical programming model solved using the CPLEX MIP solver.

Table 4.1: Average computation times (in seconds) obtained by solving problem (4.6) using the proposed BnB algorithm and CPLEX with risk measure (1.7) and scenarios $N = 100$.

		$d = 0.0125$		$d = 0.025$		$d = 0.05$		$d = 0.1$		$d = 0.15$	
p	$\|V\|$	CPLEX	BnB	CPLEX	BnB	CPLEX	BnB	CPLEX	BnB	CPLEX	BnB
1	25	0.47	0.06	0.54	0.04	0.46	0.04	0.31	0.04	0.32	0.08
	50	1.32	0.13	0.74	0.14	0.79	0.18	1.29	0.33	2.47	1.91
	100	1.99	0.07	3.25	0.38	6.00	2.19	57.62	40.90	—	—
2	25	11.00	0.56	9.63	0.72	6.24	0.33	6.38	0.37	10.57	0.43
	50	16.20	0.69	14.89	0.52	19.01	0.46	46.19	1.10	167.51	4.91
	100	38.25	0.61	119.15	1.15	253.27	2.91	973.18	70.45	—	—
3	25	40.48	0.90	25.65	0.81	15.53	0.42	15.26	0.66	27.25	0.86
	50	35.89	1.11	31.80	1.21	42.39	1.09	90.74	1.55	232.49	5.36
	100	70.47	1.08	188.71	1.54	316.38	3.13	1455.73	62.73	—	—

CHAPTER 5
TWO-STAGE STOCHASTIC RECOURSE MAXIMUM Π PROBLEM

5.1 Introduction

In Chapters 3 – 4 we addressed network problems with topologically exogenous information in the form of uncertainties induced by stochastic factors associated with network vertices that were structurally unvarying. In many application, however, it may also be of interest to examine conditions that admit topological changes to the network itself. For example, disruptions in transportation and telecommunications systems are common events that influence structural integrity, thereby merit consideration in network design. The corresponding uncertainty in such cases could, for example, be defined as the failure (or construction) of links between vertices within a certain time period.

A popular class of problems used for modeling temporal uncertainty in mathematical programming are *multi-stage stochastic recourse problems* [12, 43]. They involve making anticipatory decisions without prior knowledge about the realizations of future uncertainties, and taking recourse actions during latter stages. A *two-stage* stochastic recourse problem entails a first stage decision followed by second stage "corrective" actions after the realizations have materialized. The principal logic underlying such problems is to make decisions while considering future events, and making adjustments after they become known with the intention of minimizing the total cost associated with the decisions made during both stages.

In this regard, we consider a class of *two-stage stochastic graph recourse problems* that admit uncertainties induced by stochastic factors affecting network connectivity between decision stages (e.g. see [14, 35]). Particularly, we develop a stochastic recourse framework that seeks to maximize the expected size of a subgraph with some predefined heredity property Π under the assumption of instability among the edges between decision stages. Namely, a subset of vertices conforming to the desired graph property is selected in the first stage, after which scenario realizations in the form of edge connectivity disruptions arise. Then, a second stage recourse action is taken to "repair" the subset selected in the first stage by adding or removing vertices in order to ascertain its structural property Π in each scenario.

In the next sections we define and formalize a mathematical programming model of the described two-stage stochastic graph recourse problem and introduce a corresponding exact branch-and-bound solution criteria that leverages on the algorithmic efficiencies relative to hereditary graphs. In continuation of the descriptive formulations furnished in previous chapters, we let the structural graph property Π represent a complete graph and tailor the solution technique accordingly.

5.2 Two-stage stochastic recourse maximum Π problem

In order to extend the above description onto the stochastic programming framework, we next present a generalization of a classical two-stage linear stochastic

recourse problem of form:

$$\min \quad \mathbf{c}^\top \mathbf{x} + \mathsf{E}[P(\mathbf{x}, \xi(\omega))] \tag{5.1a}$$

$$\text{s. t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \tag{5.1b}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{5.1c}$$

where vector $\mathbf{c}$ represents the first stage costs and vector $\mathbf{x}$ corresponds to decisions that are made before the observing stochastic parameters ξ. The function $P(\mathbf{x}, \xi(\omega))$ measures the value of the second stage recourse decision $\mathbf{y}(\omega)$ in event $\omega \in \Omega$,

$$P(\mathbf{x}, \xi(\omega)) = \min \quad \mathbf{q}^\top(\omega)\mathbf{y}(\omega) \tag{5.2a}$$

$$\text{s. t.} \quad \mathbf{W}(\omega)\mathbf{y}(\omega) = \mathbf{h}(\omega) - \mathbf{T}(\omega)\mathbf{x} \tag{5.2b}$$

$$\mathbf{y}(\omega) \geq \mathbf{0}. \tag{5.2c}$$

The parameter matrices $\mathbf{W}(\omega)$, $\mathbf{T}(\omega)$ and vectors $\mathbf{q}(\omega)$, $\mathbf{h}(\omega)$ become apparent once ω is witnessed, and decisions $\mathbf{y}(\omega)$ are taken on account of the first stage decision $\mathbf{x}$. Hence, the recourse function $\mathsf{E}[P(\mathbf{x}, \xi(\omega))]$ produces the average value (cost) of the actions taken in events $\{\omega_1, \ldots, \omega_N\} \in \Omega$ for a given $\mathbf{x}$. For purposes of the forthcoming extension of problems (5.1)–(5.2) a 0-1 integer restriction is imposed on the decision variables in $\mathbf{x}$ and $\mathbf{y}(\omega)$.

As defined above, $G = (V, E)$ represents an undirected graph where $G[S]$ denotes the subgraph of G induced by S. The present work emphasizes a two-stage stochastic recourse framework for finding graphs conforming to a hereditary property Π during both decision stages. Finding subgraphs of maximum cardinality with hereditary properties composes an important class of graph theoretical problems,

also known as *maximum* Π *problems*:

$$\max_{S\subseteq V}\Big\{\sum_{i\in S} x_i : G[S] \text{ satisfies } \Pi\Big\}, \tag{5.3}$$

where binary decision variables x_i indicate whether a vertex $i \in V$ belongs to a subset $S \subseteq V$, such that the induced subgraph $G[S]$ satisfies Π.

A particular drawback of problem (5.3) is the absence of stochastic considerations regrading topological changes of the network, which, among others, is a common concern in many real-life applications. To this end, we consider uncertainty in the form of variation in vertex connectivity between decision stages of problem (5.1)–(5.2), and assume that a subset $S \subseteq V$ is selected from graph G_0 in the first stage, such that the induced subgraph $G_0[S]$ satisfies property Π. Then, stochastic factors are observed in the form of scenarios $\mathcal{G} = [G_1(V, E_1), \ldots, G_N(V, E_N)]$, where $\mathcal{G}_k = G_k(V, E_k)$ are realizations of graphs with a corresponding sets of edges E_k in scenarios $k = 1, \ldots, N$. Assuming that $\Delta_k^+ \subseteq V \setminus S$ and $\Delta_k^- \subseteq S$ represent the subsets of vertices that are added and removed from S in scenario k, respectively, the second stage recourse establishes that the induced graphs $G_k[S_k]$ of set $S_k := (S \setminus \Delta_k^-) \cup \Delta_k^+$ satisfy Π for $k \in N$. The corresponding second stage binary decision variables for scenario k are given by

$$y_{ik} = \begin{cases} 1, & i \in S_k \text{ such that } G_k[S_k] \text{ satisfies } \Pi \\ 0, & \text{otherwise.} \end{cases}$$

Let $\Pi_{G_k}(\mathbf{z}) \leq \mathbf{0}$ represents the *structural constraints* such that for any $\tilde{\mathbf{z}} \in \{0,1\}^{|V|}$, $\Pi_{G_k}(\tilde{\mathbf{z}}) \leq \mathbf{0}$ if and only if $G[\tilde{S}]$ satisfies Π, where $\tilde{S} = \{i \in V : \tilde{z}_i = 1\}$.

Then, by virtue of (5.1) and (5.2), the *two-stage stochastic recourse maximum* Π *problem* takes the form

$$\max \quad \mathbf{1}^\top \mathbf{x} + \mathsf{E}[P(\mathbf{x}, \mathcal{G})] \tag{5.4a}$$

$$\text{s. t.} \quad \Pi_{G_0}(\mathbf{x}) \leq \mathbf{0} \tag{5.4b}$$

$$\mathbf{x} \in \{0,1\}^{|V|}, \tag{5.4c}$$

and

$$P(\mathbf{x}, \mathcal{G}_k) = \max \quad \mathbf{1}^\top \mathbf{y}_k \tag{5.5a}$$

$$\text{s. t.} \quad \Pi_{G_k}(\mathbf{y}_k) \leq \mathbf{0} \tag{5.5b}$$

$$|\Delta_k^+| + |\Delta_k^-| \leq M \tag{5.5c}$$

$$\mathbf{y}_k \in \{0,1\}^{|V|}, \tag{5.5d}$$

where the *budget* parameter M restricts the number of vertices that can be added and removed within scenario k.

If the property Π in (5.4) and (5.5) defines graph completeness, the edge formulation is once again selected to represent the structural constraints (5.4b) and (5.5b) during both decision stages,

$$\big\{\mathbf{z} \in \{0,1\}^{|V|} : \Pi_G(\mathbf{z}) \leq \mathbf{0}\big\} = \big\{\mathbf{z} \in \{0,1\}^{|V|} : z_i + z_j \leq 1 \ \text{ for all } \ (i,j) \in \overline{E}\big\},$$

Then, given the probability of each scenario π_k such that $\sum_{k \in N} \pi_k = 1$, the *two-stage stochastic recourse maximum clique problem* admits the following 0-1 integer

programming form:

$$\max \quad \sum_{i \in V} x_i + \sum_{k \in N} \pi_k \left(\sum_{i \in V} y_{ik} \right) \tag{5.6a}$$

$$\text{s. t.} \quad x_i + x_j \leq 1, \quad (i,j) \in \overline{E} \tag{5.6b}$$

$$y_{ik} + y_{jk} \leq 1, \quad (i,j) \in \overline{E}, \;\; k = 1, \ldots, N \tag{5.6c}$$

$$\sum_{i \in V} |x_i - y_{ik}| \leq M, \quad k = 1, \ldots, N \tag{5.6d}$$

$$x_i \in \{0,1\}, \;\; y_{ik} \in \{0,1\} \quad i \in V, \;\; k = 1, \ldots, N. \tag{5.6e}$$

In what follows we demonstrate that this model may be solved using only graph-based BnB techniques and independent of optimization software packages. Also, observe that in constraint (5.6d) the costs of maintaining a vertex $i \in V$ cancels if it is selected in both stages, and that the budget M is constant among all the scenarios $k = 1, \ldots, N$. It would, however, be straightforward to enhance this simplistic assumption by imposing nonuniform cost structures and budgetary restrictions while maintaining validity of the forthcoming algorithmic methodology via minor corresponding adjustments.

5.3 A combinatorial branch-and-bound technique for solving the two-stage stochastic recourse maximum clique problem

In this section we consider an exact graph-based BnB algorithm for solving problems (5.4) and (5.5) that extends the algorithmic framework proposed for deterministic maximum clique problems [39, 40, 42]. Particularly, significant emphasis was placed on developing a two-stage stochastic recourse criteria utilizing the notions

of the well-known BnB algorithm introduced by Tomita et al. [51, 52], where partial graph solutions are augmented by selecting vertices from independent candidate set partitions, also know as *numbering* or *coloring* classes. Namely, the current implementation considers two separate algorithms working in tandem to solve problems (5.4) and (5.5), where a numbering- and budgetary-based bounding procedure for evaluating the solution qualities of the former is imposed, while the solution space of the latter is efficiently processed with regard to the structural and budgetary restrictions (5.5b)–(5.5c), respectively. Although the forthcoming technique was specifically tailored for solving problem (5.6), we emphasize that generalizations for any hereditary property Π are admitted, provided there exists a corresponding property-specific bounding criteria (described below).

5.3.1 First stage branch-and-bound

The first stage BnB algorithm begins at level $\ell = 0$ with partial solution $Q := \emptyset$, current objective value in (5.4) $\mathcal{Z} := 0$, incumbent solution $Q^* := \emptyset$ and its objective value $\mathcal{Z}^* := 0$. Throughout the algorithm the partial solution Q contains the vertices in V such that $G_0[Q]$ has property Π. At each node of the BnB tree, a candidate set C_ℓ of vertices is maintained from which any single vertex is added to augment the partial solution Q without violating property Π. The algorithm is initialized with $C_0 := V$ and once a branching vertex $q \in C_\ell$ is selected the corresponding candidate set at level $\ell+1$ is constructed by eliminating all the vertices from C_ℓ whose inclusion in Q would violate the property Π. Thus, the cardinality of partial solution set Q is

equal to $|Q| = \ell + 1$.

Bounding of the current solution Q involves determining the best-case solution that can be obtained by exploring further the subgraph induced by vertices in $Q \cup C_\ell$. To this end, we utilize a *numbering* algorithm [51] to estimate the maximum achievable Π-subgraph from consecutive branching. Namely, the vertices are first sorted in degree descending order and a minimum positive integer $No[p]$ is assigned to each vertex $p \in C_{\ell+1}$ such that $No[p] \neq No[s]$ if vertices p and s are connected by and edge, i.e., $(p, s) \in E$. Thus, vertices in a single number class $No[\cdot]$ (i.e., vertices with the same assigned number $N[\cdot]$) form an independent set where no two vertices within that class share an edge. The size of the maximum possible clique stemming from the vertices in the candidate set, $\upsilon(C_\ell)$, is then given by $\upsilon(C_\ell) \leq max\{No[p] : p \in C_\ell\}$. An upper bound on a feasible first stage solution to problem (5.6) may then be expressed as

$$|Q| + max\{No[p] : p \in C_\ell\} + |Q| + M \leq Q^*, \tag{5.7}$$

which, if satisfied, eliminates the necessity of further exploring the corresponding search space, and vertex q is removed from the current solution Q. If it is not satisfied, then the algorithm branches further by selecting the vertex $q \in C_\ell$ such that $No[q] = max\{No[p] : p \in C_\ell\}$. Notice that the term $|Q| + M$ in (5.7) accounts for the potential contribution of the recourse action in the optimal value of (5.6) by assuming that if no vertices are removed from the first stage subgraph during the second stage decisions, then M vertices can be added within the budget.

Given a first stage solution Q where $\mathbf{x} := \{i \in V[G_0] : x_i = 1\}$, the second

stage problems $P_k(\mathbf{x}, \mathcal{G}_k)$, $k = 1, \ldots, N$ are then solved using the method described in Algorithm 5.2 (discussed below). Two situations may arise when solving $P(\mathbf{x}, \mathcal{G}_k)$, $k \in \mathcal{N}$. First, if any problem $P(\mathbf{x}, \mathcal{G}_k)$ is infeasible for solution Q, the recourse function is likewise infeasible, in which case vertex q is removed from Q and the algorithm backtracks. Otherwise, the recourse function is feasible and the current objective value in Problem (5.4) is updated as $\mathcal{Z} = |Q| + \mathsf{E}[P(\mathbf{x}, \mathcal{G})]$. Next, the incumbent objective value $\mathcal{Z}^*$ is replaced by $\mathcal{Z}$ if $\mathcal{Z}^* < \mathcal{Z}$. Finally, if the candidate set $C_{\ell+1} \neq \emptyset$, then the algorithm branches and the vertex numbers $No[p]$ are updated accordingly. The described first stage procedure is formalized in Algorithm 5.1.

5.3.2 Second stage branch-and-bound

The procedure for solving the second stage problems $P(\mathbf{x}, \mathcal{G}_k)$, $k = 1, \ldots, N$ is identical to the first stage technique in terms of augmenting the subgraph solution with vertices from candidate sets that individually would satisfy the property Π. A key difference, however, pertains to efficient search space processing via bounding, which incorporates budgetary restrictions associated with constraint (5.6c) as opposed to a vertex numbering criteria.

For notation clarity let the current second stage and incumbent solution be denoted by Q_k and Q_k^*, respectively, and $r \in C_\ell^k$ represent the branching vertex. As noted previously, once a vertex r is selected and added to the current solution Q_k, the candidate set at the next level $C_{\ell+1}^k$ is created. Given the first stage solution Q and the current second stage solution Q_k, the left-hand-side of constraint (5.6c) is

Algorithm 5.1 First stage graph-based branch-and-bound method

1. **Initialize:** $\ell := 0$; $C_0 := V$; $Q := \emptyset$; $Q^* := \emptyset$; $\mathcal{Z} = \mathcal{Z}^* = 0$; M
2. **While** (**not** STOP) **do**
3. **if** $C_\ell \neq \emptyset$ **then**
4. select a vertex $q \in C_\ell$;
5. $C_\ell := C_\ell \setminus q$;
6. **if** $|Q| + max\{No[p] : p \in C_\ell\} + |Q| + M > Q^*$ **then**
7. $Q := Q \cup q$;
8. $\mathcal{Z} = |Q|$;
9. $C_{\ell+1} := \{i \in C_\ell : i \cup Q$ satisfies $\Pi\}$;
10. **for** $k \in \mathcal{N}$ **do**
11. $\mathbf{x} := \{i \in Q\}$;
12. solve $P_k(\mathbf{x}, \mathcal{G}_k)$;
13. **if** $P_k(\mathbf{x}, \mathcal{G}_k)$ infeasible **then**
14. $Q := Q \setminus q$;
15. goto Step 3;
16. **else**
17. $\mathcal{Z} = \mathcal{Z} + \mathsf{E}[P_k(\mathbf{x}, \mathcal{G})]$;
18. **if** $\mathcal{Z} > \mathcal{Z}^*$ **then**
19. $\mathcal{Z}^* := \mathcal{Z}$;
20. **if** $C_{\ell+1} \neq \emptyset$ **then**
21. $\ell := \ell + 1$;
22. Find $\{No[p] : p \in C_\ell\}$;
23. **else**
24. $Q := Q \setminus q$;
25. **else**
26. **if** $C_{\ell+1} \neq \emptyset$ **then**
27. $\ell := \ell + 1$;
28. Find $\{No[p] : p \in C_\ell\}$;
29. **else**
30. $Q := Q \setminus q$;
31. **else**
32. $\ell := \ell - 1$;
33. **if** $\ell = -1$ **then**
34. STOP
35. $Q := Q \setminus q$;
36. **return** $\mathcal{Z}^*$

computed as $A = \sum_{i \in V} |x_i - y_{ik}|$, where $y_{ik} = 1$ if $i \in Q_k$. Observe that the number of vertices in C^k_{l+1} which can reduce A is given by $B = |\mathbf{x} \cap C^k_{\ell+1}|$. Thus, if $A - B > M$, then a violation in restriction (5.6c) exists and the current branch corresponding to vertex r is fathomed. On the other hand, several considerations are warranted when $A - B \leq M$. First, if $A > M$, then the algorithm branches as condition (5.6c) may potentially be satisfied via the vertices in C^k_{l+1}. Second, if $A = M$ and $B = 0$, then adding more vertices to Q_k will violate constraint (5.6c). Hence, Q_k is compared against Q^*_k and if $|Q_k| > |Q^*_k|$, then $Q^*_k := Q_k$, and the algorithm backtracks. Third, if $A \leq M$, then (5.6c) is satisfied via vertices in Q_k, and, consequently, Q_k replaces Q^*_k if $|Q_k| > |Q^*_k|$. In this case, if the candidate set $C^k_{\ell+1} = \emptyset$, then the algorithm backtracks, whereas, if $C^k_{\ell+1} \neq \emptyset$, then a new branching vertex $r \in C^k_{\ell+1}$ is selected and it proceeds to level $l = l + 1$. Algorithm (5.2) outlines the described second stage solution technique.

5.4 Numerical experiments and results

Numerical studies on the two-stage stochastic recourse maximum clique problem (5.6) were conducted on randomly generated Erdös-Rényi graphs of orders $|V| = 20, 25, 30, 35, 40$; average densities of $d = 0.1, 0.3, 0.5, 0.7$; and scenario sets $N = 5, 10, 15, 20, 25, 30, 35, 40$. All tested graph instance-specific configurations regarding $|V|$, d and N were maintained constant during both decision stages in problem (5.6). Budgetary restrictions $M = \lceil \epsilon |V| \rceil$ were also examined for each graph instance, where $\epsilon = 0.1, 0.15$ and 0.2.

Algorithm 5.2 Second stage graph-based branch-and-bound method

1. **Initialize:** $\mathbf{x} := \{i \in V[G_0] : x_i = 1\}; \ell := 0; C_0^k := \{V \setminus \mathbf{x}, \mathbf{x}\}; Q_k := \emptyset;\ Q_k^* := \emptyset$
2. **While** (**not** STOP) **do**
3. **if** $C_\ell^k \neq \emptyset$ **then**
4. select a vertex $r \in C_\ell^k$;
5. $C_\ell^k := C_\ell^k \setminus r$;
6. $Q_k := Q_k \cup r$;
7. $C_{\ell+1}^k := \{i \in C_\ell^k : i \cup Q_k \text{ satisfies } \Pi\}$;
8. $A = \sum_{i \in V} |x_i - y_{ik}|$ where $y_{ik} = 1$ if $i \in V[G_k]$;
9. $B = |\mathbf{x} \cap C_{\ell+1}^k|$;
10. **if** $A - B > M$ **then**
11. $Q_k := Q_k \setminus r$;
12. **else**
13. **if** $A - B = M$ **and** $B = 0$ **then**
14. **if** $|Q_k| > |Q_k^*|$ **then**
15. $Q_k^* := Q_k$;
16. $Q_k := Q_k \setminus r$
17. **else**
18. $\ell := \ell + 1$;
19. **if** $A \leq M$ **and** $|Q_k| > |Q_k^*|$ **then**
20. $Q_k^* := Q_k$
21. **else**
22. $\ell := \ell - 1$;
23. **if** $\ell = -1$ **then**
24. STOP
25. $Q_k := Q_k \setminus r$;
26. **return** $P(\mathbf{x}, \mathcal{G}_k) := Q_k^*$

The combinatorial first and second stage BnB algorithms described above were coded using C++, and the CPLEX MIP solver was used for solving the problem 5.6 in its given form. The computations were ran the same platform as the numerical experiments in previous chapters.

The computational performance of both solution techniques over all configurations of vertices, densities, scenarios and budgetary restrictions are reported in Tables 5.1 – 5.4. Five instances of each configuration were generated and the corresponding solution times were averaged. A maximum solution time limit of 7200 seconds was imposed and the symbol "—" is used to indicate that the time limit was exceeded. Further, if only a portion of the instances for a given graph configuration solved within the time limit, then the number of instances that achieved a solution are presented in parenthesis along with the corresponding average run time.

Table 5.1 summarizes the computational times with an assigned budget of $M = \lceil 0.15|V| \rceil$ and an edge density of $d = 0.5$. Observe that the BnB algorithm provides one to two orders of magnitude in running time reductions for all configurations over the CPLEX MIP solver. A similar trend is presented for budgetary considerations in Table 5.2, where the same density is maintained, but the number of vertices is fixed at $|V| = 40$. For the consecutive experiment, Table 5.3 demonstrates the effect of variations in the average edge densities d for different graphs. Note that in this case the BnB algorithm also provides an advantage for all the tested instance, except that of $d = 0.7$. This aberration is due to the relative improvement in efficiency of the CPLEX solver resulting from the decrease in the number of graph structural

constraints (5.6b) (5.6c) associated with an increase in edge density. Lastly, the effects of incremental increases in the number of scenarios N is presented in Table 5.4. Clearly, the relative performance of the BnB algorithm improves as the number of scenarios becomes larger.

Table 5.1: Average solution times (in seconds) for problem (5.6) on random graphs with an edge density of 0.5, 20 scenarios, and $M = 0.15$.

$\|V\|$	BnB	CPLEX
20	0.05	2.44
25	0.15	6.08
30	0.57	33.01
35	2.39	135.34
40	5.66	374.21

Table 5.2: Average solution times (in seconds) for problem (5.6) on random graphs with 40 vertices, an edge density of 0.5 and 20 scenarios.

M	BnB	CPLEX
0.10	1.80	93.73
0.15	5.66	374.21
0.20	46.39	—

Table 5.3: Average solution times (in seconds) for problem (5.6) on random graphs with 40 vertices, 20 scenarios and $M = 0.15$.

d	BnB	CPLEX
0.1	0.02	3.09
0.3	0.42	714.15 (3)
0.5	5.66	374.21
0.7	245.67	82.49

Table 5.4: Average solution times (in seconds) on random graphs with 40 vertices an edge density of 0.5, and $M = 0.15$.

N	BnB	CPLEX
5	1.61	7.09
10	2.97	50.78
15	4.08	105.65
20	5.66	374.21
25	6.57	449.39
30	8.28	628.51 (4)
35	8.29	867.04 (3)
40	9.77	—

5.5 Conclusion

We have introduced a new class of two-stage stochastic recourse maximum Π problems for finding the maximum expected size of a graph among both decision stages that satisfies a defined structural property Π. We address instances where topological network uncertainties manifest in the form of connectivity disturbances between decision time periods. Although the proposed model can admit a broad range

of graph properties, we considered the case when property Π represents a hereditary subgraph.

A combinatorial BnB algorithm exploiting the structure of two-stage recourse problems was developed. We propose two decision-stage-specific algorithms and demonstrate that ensuing reductions in computational times are achievable when the property Π represents graph completeness. Numerical simulations on randomly generated graphs indicate that several orders of magnitude in solution time savings are possible via the proposed BnB algorithm in comparison to an equivalent mathematical programming solver. Namely, for all the tested graph configurations other than ones with an edge density of $d = 0.7$, one or more orders of magnitude in performance improvements were witnessed. Future work will consider efficiency enhancements relative to search space bounding of the second stage problem solutions.

CHAPTER 6
SUMMARY AND FUTURE WORK

The principal objective of this work is to model and develop efficient solution techniques for graph theoretical problems with stochastic information that manifests in a various forms. A stochastic programming framework that is based on the formalism of coherent measures of risk was adopted in order to find minimum-risk structures in graphs with stochastic vertex weights. The structural nature of such problems poses several computational challenges. Namely, in many practical applications accurate approximations of risk commonly demand very large representative scenario data sets. To this end, we propose an efficient algorithm utilizing the notion that a significant portion of representative scenarios that are used for quantifying risk are, in fact, not required to obtain an optimal solution. A scenario scenario decomposition technique contingent on the identification and separation of "non-redundant" scenarios by solving a series of smaller relaxation problems whose solution can be numerically evaluated in the context of an original problem is developed. Particular emphasis is placed on solving large-scale stochastic optimization problems involving HMCR and LogExpCR measures of risk. Numerical experiments on portfolio optimization problems using simulated return data demonstrate that significant reductions in solution times may be achieved by employing the introduced algorithm.

Another major computational challenge relates to the fact that problems of finding a maximum (minimum) subset within a graph are generally not solvable in polynomial time. Thus, the proposed risk-averse maximum weighted subgraph prob-

lem class exhibits similar characteristics and merits graph-based solution methods specifically tailored to exploiting the sought subgraph property. In this regard, we introduce several branch-and-bound algorithms for solving R-MWS problems when the subgraphs conform to hereditary and non-hereditary properties. As an illustrative example of the two concepts a risk-averse maximum weighted clique problem and a risk-averse maximum weighted 2-club problem were considered. In both cases, the branch-and-bound algorithms provided significant reductions in running time over the conventional optimization software solvers for the vast majority of graph configurations.

Conditions that admit topological modifications of networks in the form changes in the structural integrity via the failure (or construction) of links between vertices were also examined. One potential framework conforming to this notion entails a two-stage stochastic recourse maximum Π problem that seeks to maximize the expected cardinality of a subgraph satisfyting the heredity property Π under the assumption of variations in network edges between decision stages. Namely, a subset of vertices composing a Π-subgraph is select in the first stage, after which realizations of uncertainty in the form of edge failures and creations arise. Then, the second stage recourse is taken to "repair" the subset selected in the first stage by adding or removing vertices in order to ascertain property Π in each graph structural scenario. A mathematical programming framework the two-stage stochastic recourse maximum Π problem has been suggested and a branch-and-bound algorithm utilizing the problem structure was developed for each decision stage. Numerical simulations demonstrate that the

technique can reduce solution times by several orders of magnitude in comparison to an equivalent mathematical programming solver package.

It is our intention to further investigate the effectiveness of supplementing the branch-and-bound algorithms described in Chapters 3–4 with the scenario decomposition technique in Chapter 2 in order to evaluate the solution quality associated with each node of the branch-and-bound tree more efficiently. Further, in the case of non-hereditary risk-averse maximum weighted subgraph problems, it is also of interest to investigate the impact of using different solution space elimination criterion to address the additional computational expense associated with navigating between levels of the branch-and-bound tree. In the case of the two-stage stochastic graph recourse problems, future work will address efficiency considerations relative to search space bounding of the second stage recourse problems via a combination of vertex numbering methods and budgetary restrictions.

REFERENCES

[1] Richard D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:3–113, 1973.

[2] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1):3–51, 2003.

[3] Y. P. Aneja, R. Chandrasekaran, and K. P. K. Nair. Maximizing residual flow under an arc destruction. *Networks*, 38(4):194–198, 2001.

[4] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

[5] Alper Atamtrk and Muhong Zhang. Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4):662–673, 2007.

[6] L. Babel. A fast algorithm for the maximum weight clique problem. *Computing*, 52(1):31–38, 1994.

[7] Egon Balas and Jue Xue. Minimum weighted coloring of triangulated graphs, with application to maximum weight vertex packing and clique finding in arbitrary graphs. *SIAM J. Comput.*, 20(2):209–221, March 1991.

[8] Egon Balas and Chang Sung Yu. Finding a maximum clique in an arbitrary graph. *SIAM J. Comput.*, 15(4):1054–1068, November 1986.

[9] B. Balasundaram, S. Butenko, and I. Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59:133–142, 2011.

[10] Balabhaskar Balasundaram, Sergiy Butenko, and Svyatoslav Trukhanov. Novel approaches for analyzing biological networks. *Journal of Combinatorial Optimization*, 10:23–39, 2005.

[11] A. Ben-Tal and M. Teboulle. An old-new concept of convex risk measures: An optimized certainty equivalent. *Mathematical Finance*, 17(3):449–476, 2007.

[12] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer, New York, 1997.

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

[14] Immanuel Bomze, Markus Chimani, Michael Jnger, Ivana Ljubi, Petra Mutzel, and Bernd Zey. Solving two-stage stochastic steiner tree problems by two-stage branch-and-cut. In Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park, editors, *Algorithms and Computation*, volume 6506 of *Lecture Notes in Computer Science*, pages 427–439. Springer Berlin Heidelberg, 2010.

[15] Ann M. Campbell and Barrett W. Thomas. Probabilistic traveling salesman problem with deadlines. *Transportation Science*, 42(1):1–21, 2008.

[16] Renato Carmo and Alexandre Zge. Branch and bound algorithms for the maximum clique problem under a unified framework. *Journal of the Brazilian Computer Society*, 18(2):137–151, 2012.

[17] Randy Carraghan and Panos M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, 9(6):375 – 382, 1990.

[18] F. Delbaen. Coherent risk measures on general probability spaces. In K. Sandmann and P. J. Schönbucher, editors, *Advances in Finance and Stochastics: Essays in Honour of Dieter Sondermann*, pages 1–37. Springer, 2002.

[19] P. Erdös and A. Rényi. On the evolution of random graphs. 5:17–61, 1960.

[20] Daniel Espinoza and Eduardo Moreno. Fast sample average approximation for minimizing conditional-value-at-risk. *Preprint Paper*, 2012.

[21] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Trans. Program. Lang. Syst.*, 5(1):66–77, January 1983.

[22] Gregory D. Glockner and George L. Nemhauser. A dynamic network flow problem with uncertain arc capacities: Formulation and problem structure. *Operations Research*, 48(2):233–242, 2000.

[23] Anupam Gupta, Viswanath Nagarajan, and R. Ravi. Technical noteapproximation algorithms for vrp with stochastic demands. *Operations Research*, 60(1):123–127, 2012.

[24] Willem K. Klein Haneveld and Maarten H. van der Vlerk. Integrated chance constraints: reduced forms and an algorithm. *Computational Management Science*, 3:245–269, 2006.

[25] Janez Konc and Dušanka Janezic. An improved branch and bound algorithm for the maximum clique problem. *proteins*, 4:5, 2007.

[26] P. Krokhmal. Higher moment coherent risk measures. *Quantitative Finance*, 7(4):373–387, 2007.

[27] P. Krokhmal and P. Soberanis. Risk optimization with p-order conic constraints: A linear programming approach. *European Journal of Operational Research*, 301(3):653–671, 2010.

[28] P. Krokhmal, M. Zabarankin, and S. Uryasev. Modeling and optimization of risk. *Surveys in Operations Research and Management Science*, 16(2):49–66, 2011.

[29] P. Krokhmal, M. Zabarankin, and S. Uryasev. Modeling and optimization of risk. *Surveys in Operations Researh and Management Science*, 16(2):49–66, 2011.

[30] Pavlo A. Krokhmal and Policarpio Soberanis. Risk optimization with p-order conic constraints: A linear programming approach. *European Journal of Operational Research*, 201(3):653–671, 2010.

[31] Deniss Kumlander. A new exact algorithm for the maximum-weight clique problem based on a heuristic vertex-coloring and a backtrack search.

[32] Deniss Kumlander. On importance of a special sorting in the maximum-weight clique algorithm based on colour classes. In HoaiAn Le Thi, Pascal Bouvry, and Tao Pham Dinh, editors, *Modelling, Computation and Optimization in Information Systems and Management Sciences*, volume 14 of *Communications in Computer and Information Science*, pages 165–174. Springer Berlin Heidelberg, 2008.

[33] Alexandra Künzi-Bay and János Mayer. Computational aspects of minimizing conditional value-at-risk. *Computational Management Science*, 3(1):3–27, 2006.

[34] Churlzu Lim, Hanif D. Sherali, and Stan Uryasev. Portfolio optimization by minimizing conditional value-at-risk via nondifferentiable optimization. *Computational Optimization and Applications*, 46(3):391–415, 2010.

[35] Ivana Ljubi, Petra Mutzel, and Bernd Zey. Stochastic survivable network design problems. *Electronic Notes in Discrete Mathematics*, 41(0):245 – 252, 2013.

[36] Zhuqi Miao, Balabhaskar Balasundaram, and Eduardo Pasiliao. An exact algorithm for the maximum probabilistic clique problem. *Working paper*.

[37] Y. Morenko, A. Vinel, Z. Yu, and P. Krokhmal. On p-cone linear discrimination. *European Journal of Operational Research*, 231:784–789, 2013.

[38] Y. E. Nesterov and A. Nemirovski. *Interior Point Polynomial Algorithms in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[39] Patric R. J. Östergrard. A new algorithm for the maximum-weight clique problem. *Nordic J. of Computing*, 8(4):424–436, December 2001.

[40] Patric R. J. Östergrard. A fast algorithm for the maximum clique problem. *Discrete Applied Mathematics*, 120(1–3):197–207, 2002. Special Issue devoted to the 6th Twente Workshop on Graphs and Combinatorial Optimization.

[41] F. Mahdavi Pajouh and B. Balasundaram. On inclusionwise maximal and maximum cardinality -clubs in graphs. *Discrete Optimization*, 9(2):84 – 97, 2012.

[42] Panos M. Pardalos and Jue Xue. The maximum clique problem. *Journal of Global Optimization*, 4:301–328, 1994.

[43] A. Prékopa. *Stochastic Programming*. Kluwer Academic Publishers, 1995.

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

[45] R. T. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, 26(7):1443–1471, 2002.

[46] A. Ruszczyński and A. Shapiro. Optimization of convex risk functions. *Mathematics of Operations Research*, 31(3):433–452, 2006.

[47] Maciej Rysz, Mohammad Mirghorbani, Pavlo Krokhmal, and Eduardo Pasiliao. On risk-averse maximum weighted subgraph problems. *Journal of Combinatorial Optimization*.

[48] Alexey Sorokin, Vladimir Boginski, Artyom Nahapetyan, and PanosM. Pardalos. Computational risk management techniques for fixed charge network flow problems with uncertain arc failures. *Journal of Combinatorial Optimization*, 25(1):99–122, 2013.

[49] D. Subramanian and P. Huang. A novel algorithm for stochastic linear programs with conditional-value-at-risk (cvar) constraints. *IBM Research Report, RC24752*, 2008.

[50] D. Subramanian and P. Huang. An efficient decomposition algorithm for static, stochastic, linear and mixed-integer linear programs with conditional-value-at-risk constraints. *IBM Research Report, RC24752*, 2009.

[51] Etsuji Tomita and Tomokazu Seki. An efficient branch-and-bound algorithm for finding a maximum clique. In CristianS. Calude, MichaelJ. Dinneen, and Vincent Vajnovszki, editors, *Discrete Mathematics and Theoretical Computer Science*, volume 2731 of *Lecture Notes in Computer Science*, pages 278–289. Springer Berlin Heidelberg, 2003.

[52] Etsuji Tomita, Yoichi Sutani, Takanori Higashi, Shinya Takahashi, and Mitsuo Wakatsuki. A simple and faster branch-and-bound algorithm for finding a maximum clique. In Md.Saidur Rahman and Satoshi Fujita, editors, *WALCOM: Algorithms and Computation*, volume 5942 of *Lecture Notes in Computer Science*, pages 191–203. Springer Berlin Heidelberg, 2010.

[53] Svyatoslav Trukhanov, Chitra Balasubramaniam, Balabhaskar Balasundaram, and Sergiy Butenko. Algorithms for detecting optimal hereditary structures in graphs, with application to clique relaxations. *Computational Optimization and Applications*, 56(1):113–130, 2013.

[54] S. Uryasev and R. T. Rockafellar. The fundamental risk quadrangle in risk management, optimization and statistical estimation. *Surveys in Operations Research and Management Science*, 18:33–53, 2013.

[55] A. Veremyev, V. Boginski, P. Krokhmal, and D. Jeffcoat. Dense percolation in large-scale mean-field random networks is provably "explosive". *PLOS ONE*, 2012.

[56] Bram Verweij, Shabbir Ahmed, AntonJ. Kleywegt, George Nemhauser, and Alexander Shapiro. The sample average approximation method applied to stochastic routing problems: A computational study. *Computational Optimization and Applications*, 24(2-3):289–333, 2003.

[57] A. Vinel and P. Krokhmal. On log-exponential convex measures of risk. *Working paper*, 2013.

[58] A. Vinel and P. Krokhmal. On polyhedral approximations in p-order cone programming. *Working paper*, 2013.

[59] A. Vinel and P. Krokhmal. On valid inequalities for mixed integer p-order cone programming. *Journal of Optimization Theory and Applications*, page in press, 2013.

[60] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, 1953 edition, 1944.

[61] M. Yannakakis. Node-and edge-deletion np-complete problems. In *STOC'78: Proceedings of the 10th Annual ACM Symposium on Theory of Computing*, pages 253–264, New York, 1978. ACM Press.

[62] O. Yezerska, S. Butenko, and V. Boginski. Detecting robust cliques in the graphs subject to uncertain edge failures. *Working paper*.