Stochastic Programming for Global Supply Chain Planning Under Uncertainty: An Outline

Yingjie Fan[1,3], Frank Schwartz[1], Stefan Voß[1], and David L. Woodruff[2]

[1] IWI, University of Hamburg, 20146 Hamburg, Germany,
{fan.yingjie, frank.schwartz, stefan.voss}@uni-hamburg.de,
[2] University of California, Davis, CA 95616, USA,
dlwoodruff@ucdavis.edu,
[3] Xuzhou University of Technology, Xuzhou 221008, Jiangsu, China

Abstract. When supply chain networks become more complex through the application of modern trends such as outsourcing and global marketing, supply chains become more uncertain. Supply chain planning under uncertainty is a challenge for decision makers. Without considering uncertainties in supply chain planning, global supply chains may suffer enormous economic costs. When probability distributions for uncertain parameters can be estimated, stochastic programming can be used for capturing the characteristics of uncertainties and generating flexible production and transportation plans for global supply chains. This paper presents an outline on how to use stochastic programming for decision support under uncertainty. This includes a high level exposition of how to quantify uncertainties, develop stochastic programming models, generate representative scenarios, apply algorithms for model solving, undertake experimental design and present computational results. Through exemplifying supply chain planning and decision making under uncertainty by using stochastic programming, this paper aims to provide a valuable reference for future research in this area.

Keywords: stochastic programming, supply chain planning, decision making, uncertainty

1 Introduction

With larger and more complex networks, global supply chains (SCs) become more uncertain and unpredictable. Without effective risk mitigation strategies, SCs are vulnerable in uncertain environments. In this paper, uncertainty means that some of the problem data can be represented as random variables. When supply chain (SC) uncertainties can be quantified by random variables, stochastic programming can be used for providing flexible SC plans which helps to mitigate negative impacts from uncertainties in environments with stochastic disruption risks. For a SC where decisions are made or plans are revised periodically with updated information, stochastic programming can be used for

SC decision support in a rolling horizon approach. The aim of stochastic programming is to find an optimal decision in problems involving uncertain data. The treatment of uncertainties depends on the moment when the information becomes available (Birge and Louveaux, 1997).

Although research in the area of SC management is increasing, there is a clear research gap in quantitative analyses for global SC uncertainties, especially for SC disruption risks. In order to boost research in this domain, based on the authors' recent research (Fan et al., 2016, 2017a,b), this paper reviews how to use stochastic programming for supply chain planning under uncertain environments. This introduction addresses the issues:

– how to build stochastic programming models for global SCs under uncertainties,
– how to solve a related model,
– how to design computational experiments, and
– how to analyze and present computational results.

SC uncertainties may include, e.g., customer demand fluctuations, disruption risks at SC partner companies as well as transportation delays. In order to include uncertainties in the process of SC plan generation, these uncertainties should be quantified in advance. Based on historical records analytics, customer demand can be expressed with probability distribution functions. For disruption risks and transportation delays, the lasting time of negative impacts and the point in time of occurrence can be characterized according to historical records as well as real time information. With quantified SC risks, a mathematical model can be developed for a global SC.

The rest of this paper is organized as follows: Stochastic programming and other methods for decision support in uncertain environments are reviewed and compared in Section 2. A two-stage stochastic programming model can be developed by incorporating uncertainties in the second stage. The basic model and the principles for setting up stages are introduced in Section 3. *PySP*, an open-source framework for modeling and solving stochastic programs with a Progressive Hedging Algorithm (*PHA*), can be used for solving the model. Both the algorithm as well as user-defined parameters of the algorithm are explained in Section 4. When the number of possible realizations for uncertainties (scenarios) of a model is large, only a limited number of realizations and therefore only a subset of all possible scenarios is taken into consideration in the computational analyses. These scenarios are called *representative scenarios*. In this case, a *representative scenario* generation method is needed. Different scenario generation methods and their applicable scales are presented in Section 5. In computational experiments, benchmark solutions can be calculated for comparison. Different benchmarks are provided in Section 6. After solving a stochastic programming model, a solution can be evaluated through simulating the solution with a large number of scenarios generated with Monte Carlo sampling for simulating possible realizations. A flowchart is presented to illustrate how to design computational experiments. This paper ends with the conclusions in Section 7.

2 Literature Review

Usually most natural and man-made catastrophes cannot be precisely and accurately predicted, especially with a comparatively long prediction lead time. However, by utilizing big data analytics and other advanced prediction techniques, the probability distribution of occurrence and/or the severity can be predicted for an increasing number of catastrophes, e.g., extreme weathers (Fan et al., 2015). A range of approaches is available for making use of imperfect prediction information for decision support, (e.g., stochastic programming, robust optimization, metaheuristics and simulation-optimization approaches. These approaches are briefly sketched in this section.

When uncertainties can be quantified by random variables, stochastic programming can be used for capturing the essence of uncertainties. *PHA* proposed by Rockafellar and Wets (1991) is a scenario-based decomposition technique for solving stochastic programming problems. In a stochastic programming model, random variables can take on numerous values. It may not be possible and reasonable to take all those values into consideration for solving the model. In many cases, characteristics of uncertainties can be captured by specifying a reasonable number of *representative scenarios* (Løkketangen and Woodruff, 1996). *Out-of-sample* simulation is used for evaluating the quality of solutions generated from *representative scenarios.*

When the prediction information is only known in the form of interval values without the probability distribution of random data, a robust optimization can be implemented. A robustness approach aims at finding solutions that hedge against the worst contingency that may arise (Goren and Sabuncuoglu, 2008; Yu, 1997). The *minmax* criterion can be used for quantifying robustness of a decision.

Approximate solutions of optimization problems can be efficiently produced by using metaheuristics. Metaheuristics benefit from different random-search and parallel paradigms, but they frequently assume that the problem inputs, the objective function, and the set of constraints are deterministic (Caserta and Voß, 2010; Juan et al., 2015).

The computing time for solving large-scale models for real-world problems, such as transportation, production, finance and telecommunication problems, is comparatively long. In order to obtain high-quality solutions for large-scale stochastic problems with a short computing time, simulation-optimization approaches have attracted an increasing number of researchers' attentions (Gosavi, 2015; Juan et al., 2015). Although an optimal solution might not be produced in this way, obtaining an approximate solution for an accurate model of a real system is more meaningful than obtaining the optimal solution for an oversimplified model.

In Fan et al. (2016, 2017a,b), stochastic programming is used for global SCs in environments with stochastic disruption risks. Medium-scale stochastic programming models are investigated in the first two papers and a large-scale stochastic programming model is developed in the latter. The large-scale model is solved by running *PySP* on a High-Performance Computing (HPC) cluster.

Stochastic programming is also used by Haugen et al. (2001), Veliz et al. (2015) and Gade et al. (2016) for lot-sizing and forest planning problems.

In the sequel we explain how to use stochastic programming for SC planning in environments with stochastic disruption risks. A *PHA* according to Rockafellar and Wets (1991) is employed for solving stochastic programming models. Monte Carlo simulation is applied for generating *out-of-sample* scenarios for evaluating the quality of solutions.

3 A Basic Two-Stage Stochastic Programming Model

Two-stage stochastic programming models are widely applied for decision support in uncertain environments. The decision maker takes some action in the first stage in the presence of uncertainties about future realizations. *Recourse decisions* can then be made in the second stage after uncertainties are disclosed.

For a SC, plans for the coming time period (the first-stage decision) are made without perfect information for future realizations. The first-stage decision should be ideal for all those possible realizations. When uncertainties are revealed, additional decisions (*recourse decisions*) can be taken. A *recourse decision* may concern SC plans for the subsequent time period with the knowledge of uncertainties for the coming time period or emergency plans when disruptive events arise. A *recourse decision* depends on the realization of the uncertainty. For a two-stage stochastic programming model for a SC, the overall objective is to minimize the cost of the first-stage decision plus the expected costs over the uncertain scenarios.

Let us focus on two-stage stochastic programming models for global SCs. In order to explain the basic model, the following notation is used:

$\mathcal{S}$	:	The set of possible scenarios
s	:	An individual scenario, $s \in \mathcal{S}$
x	:	The first-stage decision variable
y_s	:	The second-stage decision variable in scenario $s \in \mathcal{S}$
c	:	The first-stage cost coefficient
f_s	:	The second-stage cost coefficient in scenario $s \in \mathcal{S}$
R_s	:	The probability of occurrence of scenario $s \in \mathcal{S}$, $\sum_{s\in\mathcal{S}} R_s = 1$
$\mathcal{Q}_s$	:	The set of constraints in scenario $s \in \mathcal{S}$
T_s^{bang}	:	The point in time of occurrence of a disruptive event in scenario $s \in \mathcal{S}$
T_s^{dur}	:	The duration of negative impacts once a disruption arises in scenario $s \in \mathcal{S}$

Note that x, y_s, c and f_s are vectors. The basic model can be mathematically described as follows (Birge and Louveaux, 1997; Kall and Wallace, 1994):

$$\min_{x,y_s} c \cdot x + \sum_{s\in\mathcal{S}} (R_s \cdot f_s \cdot y_s) \tag{1}$$

$$\text{subject to: } (x, y_s) \in \mathcal{Q}_s \quad \forall s \in \mathcal{S}$$

Each scenario represents a possible realization in the future. The first-stage decision variable, x, is unified for all scenarios. The second-stage variable, y_s, is scenario-specific with the associated cost coefficient f_s. Problem (1) is the well-known extensive form of a two-stage stochastic program.

In order to develop a quantitative analysis of catastrophic disruptions as a stochastic programming problem, each scenario s can be characterized by three parameters: the probability of occurrence (R_s), the point in time of occurrence (T_s^{bang}) and the duration of negative impacts (T_s^{dur}). Parameters for each scenario $s \in \mathcal{S}$ of our investigation are included both in the cost coefficient vector and in the constraints (see (2) and (3)). In this paper, a scenario with a disruption is called a disruptive scenario. A scenario without a disruption is called a non-disruptive scenario.

$$f_s \leftarrow \left(T_s^{bang}, T_s^{dur}\right) \tag{2}$$

$$\mathcal{Q}_s \leftarrow \left(T_s^{bang}, T_s^{dur}\right) \tag{3}$$

The overall probability of all disruptive scenarios for a time-span is assumed to be predictable according to the historical records. The point in time of occurrence (T_s^{bang}) can be assumed to be uniformly distributed within a time-span. The duration of negative impacts (T_s^{dur}) depends on the severity of a disruption and the flexibility of a SC and is assumed to be exponentially distributed.

When specifying the stages, according to the statements above, the first-stage decision is identical for all future realizations and the second-stage decisions depend on the particular realizations. For a two-stage stochastic programming model, the principle of setting up stages is that the first stage is scenario-independent and the second stage is scenario-dependent. Based on this principle, two approaches of setting up stages for our stochastic programming models are introduced:

1. **According to the time line**
   This approach fits for predictable disasters. According to the description in Fan et al. (2017b), probability predictions for predictable disasters are available a certain period of time in advance. Updated predictions become available before the occurrence of a disaster. With periodically updated predictions for disruptions, stages for stochastic programming models can be set up according to the time line. In this case, decisions are periodically updated according to a rolling horizon scheme.

2. **According to uncertainty related and unrelated costs**
   This approach fits for disasters which we call half-predictable disasters. In Fan et al. (2017b), half-predictable disasters are those for which a probability of occurrence can be estimated a proper period of time in advance. The point in time of occurrence of a half-predictable disaster cannot be predicted in advance. In this situation, stages should be set up in a way that uncertainty related costs are assigned to the second stage and costs that are not related to uncertainty are assigned to the first stage.

Algorithm 1: Progressive Hedging Algorithm (PHA)

```
1  $k \leftarrow 0$
2  for $s \in \mathcal{S}$ do
3      $x_s^{(k)} \leftarrow \text{argmin}_{x,y_s} (c \cdot x + f_s \cdot y_s) : (x, y_s) \in \mathcal{Q}_s$
4  $\bar{x}^{(k)} \leftarrow \sum_{s \in \mathcal{S}} R_s \cdot x_s^{(k)}$
5  for $s \in \mathcal{S}$ do
6      $w_s^{(k)} \leftarrow \rho \left( x_s^{(k)} - \bar{x}^{(k)} \right)$
7  $k \leftarrow k + 1$
8  for $s \in \mathcal{S}$ do
9      $x_s^{(k)} \leftarrow \text{argmin}_{x,y_s} \left( c \cdot x + w_s^{(k-1)} x + \rho/2 \left\| x - \bar{x}^{(k-1)} \right\|^2 + f_s \cdot y_s \right) : (x, y_s) \in$
       $\mathcal{Q}_s$
10 $\bar{x}^{(k)} \leftarrow \sum_{s \in \mathcal{S}} R_s \cdot x_s^{(k)}$
11 for $s \in \mathcal{S}$ do
12     $w_s^{(k)} \leftarrow w_s^{(k-1)} + \rho \left( x_s^{(k)} - \bar{x}^{(k)} \right)$
13 $g^{(k)} \leftarrow \sum_{x \in \mathcal{S}} R_s \cdot \left\| x_s^{(k)} - \bar{x}^{(k)} \right\|$
14 if $g^{(k)} \leq \epsilon$ then
15     terminate.
16 else
17     if $k = K$ then
18         terminate and implement a local search to find an identical feasible
           solution for $x$.
19     else
20         go to 7
```

4 Progressive Hedging Algorithm and PySP

In this section, the *PHA* proposed by Rockafellar and Wets (1991) as well as *PySP* are introduced. *PHA* is implemented for solving stochastic problems in different areas, i.e., Haugen et al. (2001), Watson and Woodruff (2011), Veliz et al. (2015) and Gade et al. (2016). In these papers, *PHA* is proven to be an effective method for solving stochastic programming models.

For the optimization problem in Section 3, the basic *PHA* can be stated in Algorithm 1, taking a penalty factor $\rho > 0$, a termination threshold ϵ, and a maximum number of iterations K as input parameters.

PHA is embedded in *PySP*, an open-source framework for modeling and solving stochastic programs in Python. In this framework, the *runph* script provides a command-line interface to solve stochastic programming models with *PHA*.

When *PySP* is implemented for solving stochastic programming models, the maximum number of iterations K and the value of ρ are user-defined parameters. Effective methods for determining element-specific $\rho\,(i)$ values based on problem-specific data are developed in Watson and Woodruff (2011). For independent integer variables, we have:

$$\rho\left(i\right) \leftarrow \frac{c\left(i\right)}{x^{max} + x^{min} + 1} \tag{4}$$

For independent continuous variables, $\rho(i)$ is calculated by:

$$\rho\left(i\right) \leftarrow \frac{c\left(i\right)}{max\left(\left(\sum_{s \in \mathcal{S}} R_s \cdot |x_s^{(0)} - \bar{x}^{(0)}|\right), 1\right)} \tag{5}$$

Element-specific $\rho\left(i\right)$ values are implemented for stochastic programming models in Fan et al. (2016, 2017a,b).

The intention of solving a stochastic programming model with *PySP* is to find a good quality feasible solution, rather than obtaining a provably optimal solution. In particular, it may not be possible to find an optimal solution and prove optimality for a stochastic programming model for a global SC. For SC planning problems, feasible and good quality solutions generated within a reasonable time frame are meaningful and valuable in practice.

5 Scenario Generation

For two-stage stochastic programming models with a small number of possible realizations, the list of all possible realizations can be incorporated in the solution process for stochastic programming problems. For problem instances with a large number of possible realizations for the second stage, it is more efficient to include a certain number of representative possible realizations (*in-sample* scenarios) than to incorporate all possible realizations in the solution process. When different *representative scenarios* are used, the solutions are probably different. In this section, methods for generating *representative scenarios* for capturing characteristics of uncertainties are investigated.

For a stochastic programming model with multiple uncertain parameters, *representative scenarios* are composed by uncertain parameters' *representative values*. The probability for each scenario is deduced from probabilities of *representative values*. Three methods for generating *representative values* for uncertain parameters are presented in this section. In order to exemplify these methods, the duration T^{dur} in Fan et al. (2016) is taken as an example. T^{dur} is the duration of negative impacts for a global SC in case of a disruption which is assumed to be exponentially distributed and uncorrelated with other uncertain parameters (see Fig. 1). Uncertainties are assumed to be uncorrelated for low frequency and high impact SC disruptions because the probability for the occurrence of more than one disruption at the same time period is extremely low. It is possible to explore SCs with complicated multiple intercorrelated uncertainties with Monte Carlo sampling which will be introduced in this section.

Different methods are implemented for generating *representative values* for the uncertain parameter T^{dur}. For scenario s, the value of T^{dur} is indicated by T_s^{dur}. In Fig. 2–5, the values of $T_s^{dur}(s \in \mathcal{S})$ generated with different methods

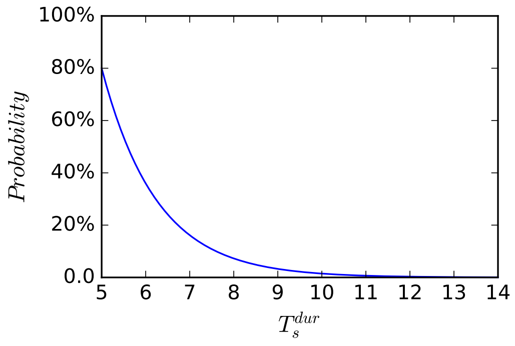


Fig. 1. Probability distribution

are presented. T_s^{bang} $(s \in \mathcal{S})$ in the basic model in Section 3 can be calculated with the same methods.

1. **Selecting equal probability values**
By splitting the probability distribution of an uncertain parameter into a number of equal probability segments, the medians of these segments can be selected as *representative values*. The median is the value that splits the probability distribution into two portions whose areas are identical. Representative values selected in this way have the same probability.
Fig. 2 shows 5 and 10 *representative values*, which are generated with this method for T^{dur}. The probabilities for *representative values*, in case of $T_{Num}^{dur} = 5$ and $T_{Num}^{dur} = 10$, are 20% and 10%, respectively. However, this method need not work for generating discrete *representative values*.

2. **Selecting representative values and calculating probabilities**
For a discrete parameter, *representative values* can be selected at first. In order to calculate the probability of these *representative values*, the probability distribution is split into segments in a way that each *representative value* is the median or close to the median of a segment. The overall probability of a segment is the probability of the *representative value* in this segment.
Fig. 3 gives examples of *representative values* and probabilities generated with this method when T_{Num}^{dur} is 5 and 10. In order to assure that each *representative value* is the median of a segment, in Fig. 3 *representative values* and segments are alternately selected from the left side to the right side one by one. The right frontier of the previous segment is the left frontier of the next segment. This method is implemented in Fan et al. (2016).

3. **Monte Carlo sampling**
An easy way for generating *representative values* for uncertain parameters is to use Monte Carlo sampling. With an uncertain parameter's probability distribution function as the input, *representative values* can be generated by using statistical functions of *SciPy*, which is a collection of mathematical algorithms and functions built in *Python*. Representative values generated with

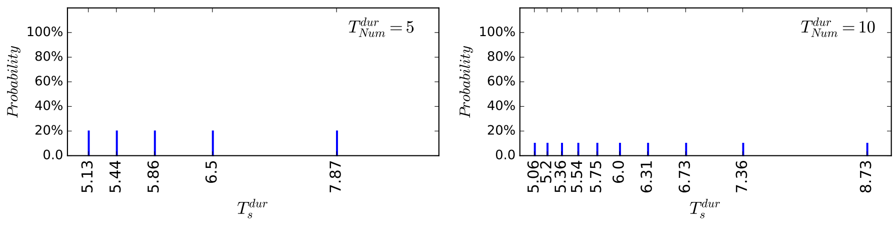


Fig. 2. *Representative values* with equal probability

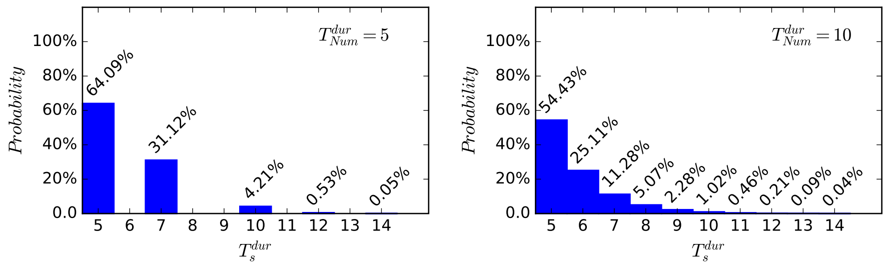


Fig. 3. Discrete *representative values* and their probabilities

Monte Carlo sampling have equal probability. With probability distribution functions, Monte Carlo sampling can also be used for generating *representative values* for multi-dimensional correlated random variables which makes it possible to explore SC planning problems with intercorrelated and real world uncertainties (Joy et al., 1996; Touran and Wiser, 1992).

Representative values and their frequencies generated with Monte Carlo sampling for uncertain parameters are shown in Fig. 4 and Fig. 5. Monte Carlo sampling also fits for uncertain parameters with a large number of dimensions, e.g., the customer demand for each production at each seller in each time period (Fan et al., 2017b).

In order to test the quality of a solution for a stochastic programming model, a number of *out-of-sample* scenarios is required for simulating possible realizations. When Monte Carlo sampling is adopted, it is more accurate to simulate the reality with a larger number of *out-of-sample* scenarios (see Fig. 6). For instance, 500 or 1000 scenarios may be generated with Monte Carlo sampling for simulating possible realizations.

6 Benchmarks

As mentioned in Section 4, the intention of solving stochastic programming models for global SCs is to obtain high quality feasible solutions. Instead of proving optimality, we demonstrate the quality of solutions from stochastic programming

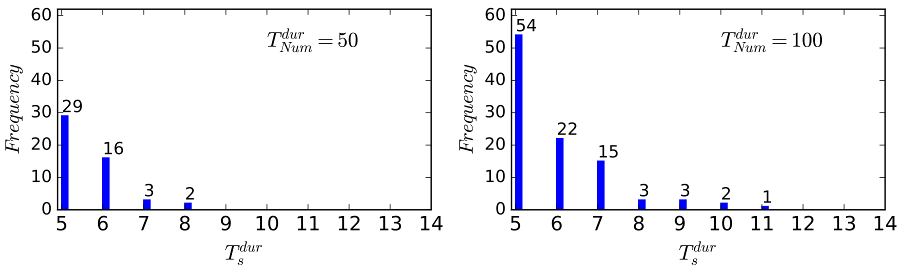


Fig. 4. Monte Carlo methods for generating discrete *representative values*

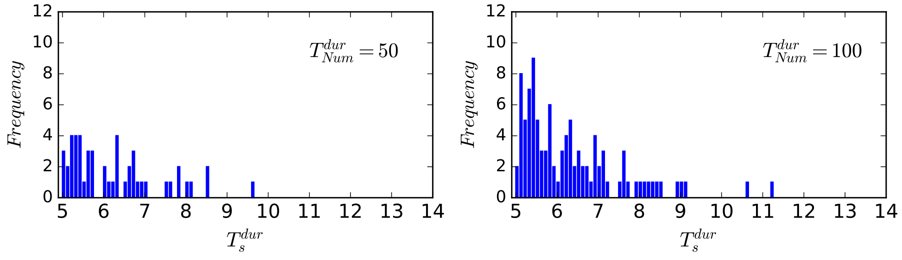


Fig. 5. Monte Carlo methods for generating *representative values*

through comparing with different benchmark solutions. In our previous research, solutions for stochastic programming models generated with *PySP* based on *representative scenarios* are always superior to benchmark solutions. In this section, we describe decision makers with four different attitudes to deal with risk, which are pessimistic, moderate, optimistic and rational attitudes. In the following, the assumptions for these attitudes are introduced (Fan et al., 2017b):

1. **Pessimistic (*pess*)**
   Decision makers with pessimistic attitudes prepare for the worst possible catastrophe, which is characterized by the longest duration and the earli-

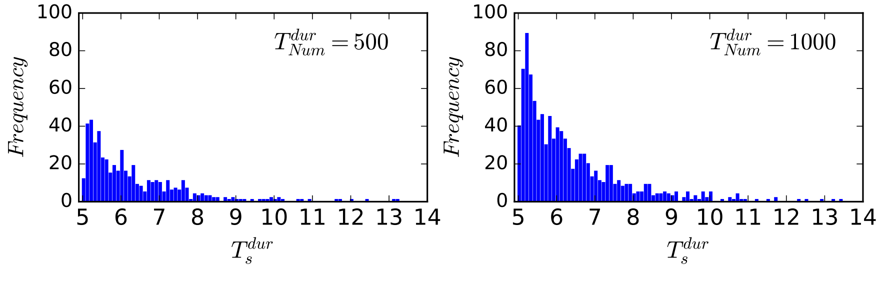


Fig. 6. Monte Carlo methods for generating a large set of *out-of-sample* values

est time of occurrence. f^{pess} and Q^{pess} indicate the parameter set and the constraints set for the worst possible catastrophe.

2. **Moderate (*mod*)**
   Decision makers with moderate attitudes believe that a moderate catastrophe will happen. The moderate catastrophe has the mean duration and the mean occurrence time. f^{mod} and Q^{mod} indicate the parameter set and the constraints set for the moderate catastrophe.

3. **Optimistic (*opt*)**
   Decision makers with optimistic attitudes believe that catastrophes will never happen. They anticipate catastrophes in no way. For this type of decision makers, disruptions appear to be totally unpredictable in the first stage. f^{opt} and Q^{opt} indicate the parameter set and the constraints set for a case without the occurrence of any catastrophe.

4. **Rational (*sp*)**
   Decision makers with rational attitudes are aware of the fact that catastrophes may be of different severities. Probability distributions of their time of occurrence and duration are incorporated in a stochastic programming model for decision support.

In addition, the expected value of *wait-and-see* (*ws*) solutions for each problem instance represents a lower bound. The *ws* solution represents an ideal situation that all uncertainty will be resolved before decisions have to be made. For the sake of a compact presentation, we treat *ws* as an additional element for the set of attitudes. We use a set $\mathcal{U} = \{pess,\ mod,\ opt,\ sp,\ ws\}$ to indicate solutions introduced above.

A large set of *out-of-sample* scenarios is generated with Monte Carlo sampling for simulating possible realizations (see Section 5). Solutions by solving a stochastic programming model (*SP* solutions) and solutions for decision makers with different attitudes are tested with these scenarios. Ω indicates the set of *out-of-sample* scenarios. g_ω indicates the parameter set for each scenario $\omega \in \Omega$. The size of Ω is N. The expected value of *ws* solutions is calculated by (6). The expected values of solutions for decision makers with different attitudes are deduced from (8).

1. **Expected value of *wait-and-see* solutions**
   A *ws* solution is generated until an observation of the uncertainty is made (Madansky, 1960). The expected value of *ws* solutions for *representative scenarios* can be obtained by:

$$EV_{ws} = \frac{1}{N} \sum_{\omega \in \Omega} \left[\min_{x, y_\omega} \left(c \cdot x + g_\omega \cdot y_\omega \right) \right] \tag{6}$$

   As mentioned in Section 5, scenarios generated with Monte Carlo sampling have the same probability. Thus, the probability for each scenario is $\frac{1}{N}$.

2. **Expected value of solutions for decision makers with different attitudes**
The *pess, mod* and *opt* solutions are obtained by solving the model in (1) with a single scenario. f^{pess}, f^{mod} and f^{opt} are the second-stage parameters. Then the stochastic programming model in (1) is transformed into a deterministic model in (7). x^*_u indicates the first-stage optimal solution for a decision maker with an attitude $u \in \{pess, mod, opt\}$.

$$\begin{aligned} x^*_u &\leftarrow \min_{x,y} \left(c \cdot x + f^u \cdot y\right) \\ s.t. \quad (x, y) &\in Q^u, \quad \forall u \in \{pess, mod, opt\} \end{aligned} \tag{7}$$

Note that the *SP* solution, which is indicated by x^*_{sp}, is not included in (7). x^*_{sp} is deduced by solving the stochastic programming model in (1). The expected value of a solution for a decision maker with an attitude $u \in \{pess, mod, opt, sp\}$ is calculated by:

$$\begin{aligned} EV_u &= c \cdot x^*_u + \frac{1}{N} \min_{y_\omega} \sum_{\omega \in \Omega} \left(g_\omega \cdot y_\omega\right) \\ s.t. \quad (x^*_u, y_\omega) &\in \mathcal{Q}_\omega, \quad \forall u \in \{pess, mod, opt, sp\}, \omega \in \Omega. \end{aligned} \tag{8}$$

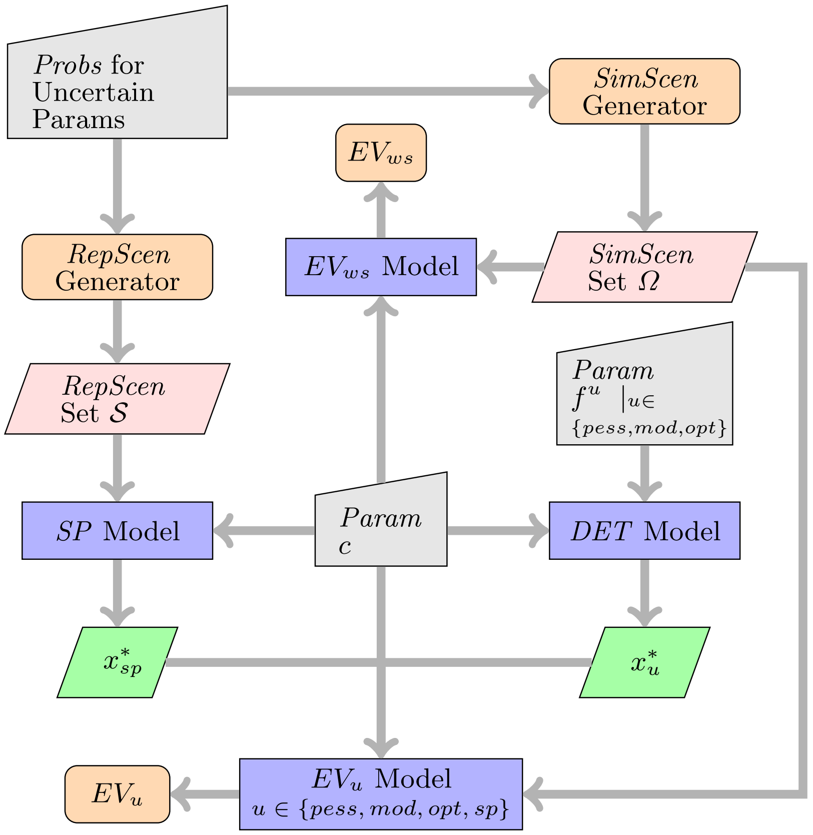
GAP_u indicates the gap between the expected value of a solution x^*_u and the expected value of *ws* solutions. It shows the quality of a solution: The smaller GAP_u, the better the solution. GAP_u is calculated by:

$$GAP_u = \frac{EV_u - EV_{ws}}{EV_{ws}} \tag{9}$$

In Fig. 7, a flowchart is presented for explaining processes of computational experiments. The following abbreviations are used:

Param(s)	:	Parameter(s)
Probs	:	Probability distribution functions
SimScen	:	Scenarios for simulating possible realizations (*out-of-sample*)
RepScen	:	Representative scenarios for the stochastic programming model (*in-sample*)
DET Model	:	Deterministic model, which is the *SP* model with a single scenario

Incorporating probability distribution functions of uncertain parameters as input, *representative scenarios* ($s \in \mathcal{S}$) and scenarios for simulating possible realizations ($\omega \in \Omega$) can be generated with a *RepScen* Generator and a *SimScen* Generator, respectively. Scenario generation methods for *RepScen* Generators depend on the characteristics of uncertain parameters (see Section 5). Monte Carlo sampling can be used for *SimScen* Generators. For problem instances with a medium-scale of possible realizations, the full list of all possible realizations can be included in set Ω (Fan et al., 2014).



Probs for Uncertain Params
SimScen Generator
EV_{ws}
RepScen Generator
EV_{ws} Model
SimScen Set Ω
RepScen Set $\mathcal{S}$
Param $f^u \mid u \in \{pess, mod, opt\}$
SP Model
Param c
DET Model
x^*_{sp}
x^*_u
EV_u
EV_u Model $u \in \{pess, mod, opt, sp\}$

Fig. 7. A flowchart for computational experiments

With *representative scenarios* and the first-stage parameter set c as input, a solution x^*_{sp} for the stochastic programming model can be obtained by using *PySP*. With parameter set f^u and c as input, the first-stage optimal solutions x^*_u for decision makers with attitude $u \in \{pess, mod, opt\}$ are obtained by solving deterministic models (see (7)).

To evaluate the quality of a solution, each solution $x^*_u \mid u \in \{pess, mod, opt, sp\}$ is tested with a large number of scenarios ($\omega \in \Omega$) through an evaluation model. $EV_u \mid u \in \{pess, mod, opt, sp\}$ is calculated according to (8). EV_{ws} is calculated by solving (6). An identical first-stage solution for *ws* solutions is not required. In order to compare the quality of different solutions, final results can be presented with box plots (see Fig. 8). Another way is calculating GAP_u according to (9) and presenting the obtained values in a table.

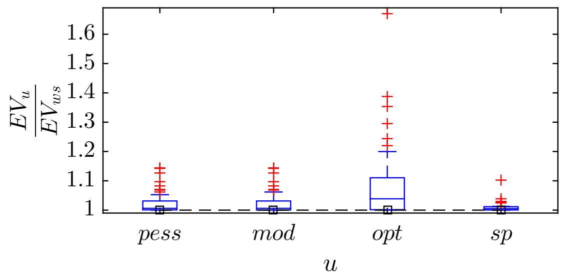


Fig. 8. Presenting results; see Fan et al. (2017a)

7 Conclusion

When we try to realize automated business processes, Industry 4.0, the 5G era or autonomous logistics, autonomous decision making under uncertainty is an essential problem. This paper introduced how to generate and evaluate flexible supply chain plans under uncertainty by using stochastic programming. The framework for supply chain planning problems presented here is a general framework for decision making regarding problems under uncertainty. This paper is meaningful as it provides a valuable reference for the research in the domain of supply chain planning and decision making under uncertainty. For the next step, it is important to develop an autonomous decision making system by combining the framework for supply chain planning with a framework for big data analytics for demand and risk prediction.

Acknowledgements

The author Yingjie Fan acknowledges financial support from the Universtät Hamburg Equal Opportunity Fund.

Bibliography

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

M. Caserta and S. Voß. Metaheuristics: Intelligent problem solving. In V. Maniezzo, T. Stützle, and S. Voß, editors, *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, pages 1–38. Springer US, 2010.

Y. Fan, F. Schwartz, and S. Voß. Flexible supply chain design under stochastic catastrophic risks. In W. Kersten, T. Blecker, and C. Ringle, editors, *Next Generation Supply Chains*, pages 379–406, Berlin, 2014. Epubli.

Y. Fan, L. Heilig, and S. Voß. Supply chain risk management in the era of big data. *Lecture Notes in Computer Science*, 9186:283–294, 2015.

Y. Fan, F. Schwartz, S. Voß, and D. L. Woodruff. Stochastic programming for flexible global supply chain planning. *Flexible Services and Manufacturing Journal*, pages 1–33, 2016. 10.1007/s10696-016-9261-7.

Y. Fan, F. Schwartz, and S. Voß. Flexible supply chain planning based on variable transportation modes. *International Journal of Production Economics*, 183: 654–666, 2017a.

Y. Fan, F. Schwartz, S. Voß, and D. L. Woodruff. Impacts of catastrophe insurance policies on global supply chain operational planning. Institute of Information Systems, University of Hamburg, 2017b.

D. Gade, G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets, and D. L. Woodruff. Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs. *Mathematical Programming*, 157(1): 47–67, 2016.

S. Goren and I. Sabuncuoglu. Robustness and stability measures for scheduling: single-machine environment. *IIE Transactions*, 40(1):66–83, 2008.

A. Gosavi. *Simulation-Based Optimization*. Springer, Berlin, 2nd edition, 2015.

K. K. Haugen, A. Løkketangen, and D. L. Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132(1):116–122, 2001.

C. Joy, P. P. Boyle, and K. S. Tan. Quasi-Monte Carlo methods in numerical finance. *Management Science*, 42(6):926–938, 1996.

A. A. Juan, J. Faulin, S. E. Grasman, M. Rabe, and G. Figueira. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, 2:62–72, 2015.

P. Kall and S. Wallace. *Stochastic Programming*. Wiley, Chichester, 1994.

A. Løkketangen and D. L. Woodruff. Progressive hedging and tabu search applied to mixed integer (0, 1) multistage stochastic programming. *Journal of Heuristics*, 2(2):111–128, 1996.

A. Madansky. Inequalities for stochastic linear programming problems. *Management Science*, 6(2):197–204, 1960.

R. T. Rockafellar and R. J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1): 119–147, 1991.

A. Touran and E. P. Wiser. Monte Carlo technique with correlated random variables. *Journal of Construction Engineering and Management*, 118(2):258–272, 1992.

F. B. Veliz, J.-P. Watson, A. Weintraub, R. J.-B. Wets, and D. L. Woodruff. Stochastic optimization models in forest planning: a progressive hedging solution approach. *Annals of Operations Research*, 232(1):259–274, 2015.

J.-P. Watson and D. L. Woodruff. Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, 8(4):355–370, 2011.

G. Yu. Robust economic order quantity models. *European Journal of Operational Research*, 100(3):482–493, 1997.