INFORMATICA, 2010, Vol. 21, No. 1, 123–138

Multistage K-Means Clustering for Scenario Tree Construction

Kristina ŠUTIENĖ, Dalius MAKACKAS, Henrikas PRANEVIČIUS
Department of Business Informatics, Kaunas University of Technology
Studentų 56-301, LT-51424 Kaunas, Lithuania
e-mail: kristina.sutiene@ktu.lt

Received: January 2008; accepted: October 2009

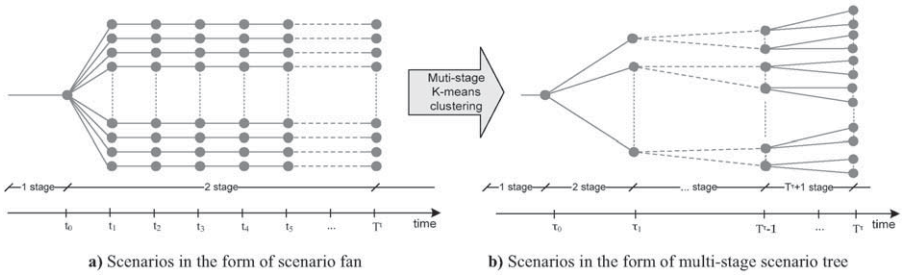
Abstract. In stochastic programming and decision analysis, an important issue consists in the approximate representation of the multidimensional stochastic underlying process in the form of scenario tree. This paper presents the approach to generate the multistage multidimensional scenario tree out of a set of scenario fans. For this purpose, the multistage K-means clustering algorithm is developed. The presented scenario tree generation algorithm is motivated by the stability results for optimal values of a multistage stochastic program. The time complexity of developed multistage K-means clustering algorithm is proved to be linear in regard to the number of scenarios in the fan. The algorithm to determine the branches with nonduplicate information in the multistage scenario tree is also presented as an intermediate result of research.

Keywords: scenario generation, scenario tree, multistage K-means clustering, time complexity.

1. Introduction and Problem Statement

The notion of scenario tree is used in multistage stochastic programs where the main computational approach consists in approximating the underlying multidimensional stochastic process by a process having finitely many scenarios, exhibiting tree structure and starting at a fixed element (Heitsch and Römisch, 2005; Fig. 1b). Each scenario in the tree can be viewed as one realization of a certain multidimensional stochastic data process of the model. The additional features of multistage scenario trees in dynamic decision models are that the process is deterministic at the first time period and that it has to be non-anticipative. The latter feature means that the random data and decision processes at any time do not depend on future realizations of the data process (Gröwe-Kuska *et al.*, 2003).

Approaches of scenario tree generation are based on various principles (Gülpinar *et al.*, 2004; Heitsch, 2003; Heitsch and Römisch, 2005; Mitra, 2006; Dzemyda and Sakalauskas, 2009): (a) bound-based constructions, (b) Monte-Carlo schemes or Quasi Monte-Carlo based methods, (c) EPVI-based sampling and reduction within decomposition schemes, (d) moment-matching principle, (e) probability metric based approximations. There exists a group of scenario generation problems, when it is enough hard to generate a scenario tree directly from historical time series or from statistical models,



a) Scenarios in the form of scenario fan **b)** Scenarios in the form of multi-stage scenario tree

Fig. 1. The usage of multistage K-means clustering method in generating the scenario tree out of scenario fan.

e.g., time series or regression models. Thus, the idea is to start with a good initial approximation of the underlying stochastic input process so that a fan of individual scenarios is created (Fig. 1a). These scenarios can be obtained by sampling or simulation techniques based on stochastic model. Clearly, a good approximation may involve a large number of scenarios. Although such fan of individual scenarios represents a very specific scenario tree (2-stage problem), its structure is not appropriate for the stage-wise decision process and contains a large number of nodes (Heitsch and Römisch, 2005).

To eliminate these disadvantages, the initial scenario fan is modified by bundling similar scenarios to construct the multistage scenario tree. The researches based on such principle have been published, e.g., in Gröwe-Kuska *et al.* (2003) a subset of the initial scenario set is determined and the procedure based on a recursive reduction argument using transportation metrics is presented, in Heitsch and Römisch (2005), Möller *et al.* (2004) the backward and forward scenario tree generation methods based on upper bounds for two relevant ingredients of the stability estimate, namely, the probabilistic and the filtration distance, are described.

We propose the algorithm based on method from cluster analysis to construct the multistage scenario tree by bundling similar scenarios in the scenario fan (Fig. 1). An approach similar to our work is introduced in the article Dupačová *et al.* (2002), but without a detailed clustering algorithm. Due to this, K-means clustering method is modified to treat properly the inter-stage dependencies, and it is implemented while constructing the multistage scenario tree from the fan of individual scenarios. The developed method allows to cluster similar scenarios from d scenario fans. Thus, the multistage d-dimensional scenario tree is obtained, and the method is named as multistage K-means clustering. This paper is a continuous research of paper Pranevičius and Šutienė (2007), where the precision of approximation of scenario fans by the scenario tree is evaluated.

The other question considered in this paper is the redundancy elimination in the multistage scenario tree. The redundancy is identified if the scenario tree is described scenario by scenario, i.e., one could think of the scenarios as paths from the root node to the leaves (Gassmann, 2006). Thus, scenarios may share data for several stages: information up to the branch period is shared between a scenario and its parent scenario. It is useful for system comprehension to remove redundancy and to identify branches in the scenario tree with nonduplicate information (Gassmann and Kristjánsson, 2008). For this purpose,

the algorithm based on idea of labelling on branches is developed and described in this paper.

The third part of the paper addresses to the question of testing the stability of proposed scenario tree generation method, according to the reference of Kaut and Wallace (2003), since the procedure of tree construction involves the randomness. It is demonstrated on a case from asset liability management, which is a problem from stochastic programming.

2. Notation for Scenarios

Scenarios are introduced as atoms of the true discrete probability distribution P or of that discrete probability distribution which approximates the true one. The notation for scenarios is given based on the references Domenica *et al.* (2007), Dupačová *et al.* (2000).

If a stochastic factor evolves in time, we have a stochastic process. Let assume that the stochastic process $\xi = \{\xi_t\}_{t=1}^{T^t}$ is defined on some filtered probability space $(\Omega, S, \mathcal{F}, P)$. The sample space Ω is defined as $\Omega := \Omega_1 \times \Omega_2 \times \cdots \times \Omega_{T^t}$, where $\Omega_t \subset \boldsymbol{R}^d$ are taken as finite dimensional. For instance, these data may correspond to the random return of d financial assets at different time moments t. The σ-algebra S is the set of events with assigned probabilities by measure P, and $\{F_t\}_{t=1}^{T^t}$ is a filtration on S. For scenario based models, one assumes that the probability distribution P is discrete and concentrated on a finite number of points, say, $\xi^s = (\xi_1^s, \ldots, \xi_t^s, \ldots, \xi_{T^t}^s)$, $\xi_t^s = (\xi_t^{s,1}, \ldots, \xi_t^{s,d})'$, $s = 1, \ldots, S$. The probability of ξ_t^s is denoted as $\widehat{\pi}_t^s = \widehat{\pi}_t^s(\xi_t^s)$, $\sum_{s=1}^{S} \widehat{\pi}_t^s = 1$, $t \in \{1, \ldots, T^t\}$. At the current time moment t_0, all scenarios are known with certainty. Thus, the first stage is represented by a single root node ξ_0 (vector in $\boldsymbol{R}^d$). Moving to the second stage, the structure branches into individual scenarios. Such structure of simulated data paths is called as scenario fan (Fig. 1a). It is represented as two-stage problem, as all σ-fields $\mathcal{F}_t$, $t = 1, \ldots, T^t$ coincide. Thus, the probabilities $\widehat{\pi}_t^s = \widehat{\pi}_{t'}^s$, $s = 1, \ldots, S$, $t \neq t'$, $t, t' > 0$. The two-stage stochastic problem has the following properties, as in Dupačová *et al.* (2000): decisions at all time instances are made at once and no further information is expected; except for the first stage no non-anticipativity constraints appear. In general, such properties can be regarded as disadvantages, especially in cases, when decisions are considered to be reformulated during the horizon. By eliminating these disadvantages, the multistage scenario tree is generated.

The multistage scenario tree (Fig. 1b) allows to reflect the inter-stage dependency and decreases the number of nodes while comparing to the scenario fan. The time stage index $\tau \in \{1, \ldots, T^\tau\}$ is associated with time moments when decisions are taken. The structure of multistage scenario tree at initial time moment is described by a sole root node ξ_0 (vector in R^d) and by branching into a finite number of scenarios in every stage. The probability distribution P is concentrated on a finite number of points $\xi^{s_{T^\tau}} = (\xi_1^{s_1}, \ldots, \xi_\tau^{s_\tau}, \ldots, \xi_{T^\tau}^{s_{T^\tau}})$, $\xi_\tau^{s_\tau} = (\xi_\tau^{s_\tau,1}, \ldots, \xi_\tau^{s_\tau,d})'$, $s_\tau = 1, \ldots, S_\tau$, but with varying size of scenarios set S_τ. The probability of $\xi_\tau^{s_\tau}$ is denoted as $\widehat{\widehat{\pi}}_t^{s_t} = \widehat{\widehat{\pi}}_t^{s_t}(\xi_\tau^{s_\tau})$. The stages are connected with possibility to take additional decisions based on newly revealed information. Such information can be obtained periodically (every day, week,

month) or based on some events (expiration of investment portfolio duration). The distinction between stages and time periods of discretization is essential, because in practical application it is important that the number of time periods would be greater than the corresponding nodes. The arcs linking nodes represent various realizations of random variables. The number of branches from each node can vary depending on problem specific requirements and not definitely constant through the tree.

3. Construction of Multistage Multidimensional Scenario Tree out of the Scenario Fans

At this moment, we concentrate on the construction of scenario trees when the underlying stochastic parameters have been determined and the individual scenarios are already generated. Based on scenario's dimension, the scenario trees can be of two types:

- The multistage scenario tree, which is generated from one scenario fan (one-dimensional scenarios).
- The d-dimensional multistage scenario tree, which is generated from d scenario fans (d-dimensional scenarios).

The scenario tree is used to describe the behaviour of one uncertain factor; the d-dimensional scenario tree is used to describe the behaviour of d uncertain factors.

To bundle the individual scenarios into clusters, the clustering procedure is employed. Clustering consists in partitioning of a set of scenarios into subsets, so that the scenarios inside cluster would be more similar than outside the cluster. Since most of clustering methods are developed for data not varying in time, we have to make some modifications in order to cluster the time dependent data, such as are scenarios. Due to this, K-means clustering method (Kaufmann and Rousseeuw, 1990; Teknomo, 2006; Krilavičius and Žilinskas, 2008) is modified to treat properly the inter-stage dependencies, and it is implemented while constructing the multistage scenario tree out of scenario fans. Two factors are used to delineate the structure of scenario tree: the branching scheme and the number of stages. Let assume that branches K_τ are desired from each scenario tree node at stage τ. For example, in the case of $K_\tau = 2$ the pessimistic and optimistic scenarios are considered. In the terminology of cluster analysis, it means that K_τ clusters have to be formed. The following modifications for K-means method have to be done in order to construct the scenario tree with $(\tau + 1)$ stages:

- In the current stage, the new sub-clusters have to be formed from each cluster generated at previous stage.
- Centroids (means) have to be computed only at stage indexed time moments, while the distance measure has to exploit the whole sequence of simulated d-dimensional scenarios.
- The possibility of changing the number of clusters in every stage has to be allowed.
- The probabilities of each node have to be evaluated.

After a discussion of a kind of requirements we are using, the multistage K-means clustering problem and then the algorithm for solving it are described below.

3.1. *The Multistage K-Means Clustering Problem*

Given a set of d-dimensional scenarios $\xi^s = (\xi_t^s)$, where $\xi_t^s = (\xi_t^{s,j})$, $s = 1, \ldots, S$, $t \in \{1, \ldots, T^t\}$, $j = 1, \ldots, d$, and the number K_τ of desired clusters at stage $\tau \in \{1, \ldots, T^\tau\}$, it is needed to solve κ_τ K-means clustering tasks in each stage τ, where

$$\kappa_\tau = \begin{cases} \prod_{j=1}^{\tau-1} K_j, & \tau > 1, \\ 1, & \tau = 1. \end{cases}$$

The input to each of $1, \ldots, \kappa_\tau$ K-means clustering tasks is $\xi^s \in \tilde{C}_\tau^k$, $k = 1, \ldots, \kappa_\tau$, which are formed while performing usual K-means clustering algorithm. The output of stage τ are cluster's centroids $\bar{\xi}^k = (\bar{\xi}_t^k)$, where $t = \tau$, $k = 1, \ldots, \prod_{j=1}^{\tau} K_j$ with assigned probabilities $\widehat{\widehat{\pi}}(\bar{\xi}^k)$.

3.2. *The Multistage K-Means Clustering Algorithm*

Set the stage indexed time moments as $\tau \in \{1, \ldots, T^\tau\}$, then iterate the following steps:

Step 1. Setting initial centroids. Let $\bar{\xi}^k = (\bar{\xi}_t^k)$, $t \in \{1, \ldots, T^t\}$, $k = 1, \ldots, K_\tau$ be the clusters' centroids. Some method can be employed to choose the centroids' positions for initial clusters, sometimes known as "seeds". It might be chosen to be the first K_τ scenarios or scenarios by random since the scenarios are independently generated.

Step 2. Cluster assignment. Assign each scenario $\xi^s = (\xi_t^s)$, $t \in \{1, \ldots, T^t\}$, $s = 1, \ldots, S$ to the cluster $\tilde{C}^k = (\tilde{C}_\tau^k)$, $k = 1, \ldots, K_\tau$, such that centroid $\bar{\xi}^k = (\bar{\xi}_t^k)$ is nearest to $\xi^s = (\xi_t^s)$ by the distance measure, i.e., compute the value of indicator function:

$$\delta\left(\xi^s, \tilde{C}^k\right) = \begin{cases} 1, & D\left(\xi^s, \bar{\xi}^k\right) < D\left(\xi^s, \bar{\xi}^m\right) \quad \text{for all } k, m = 1, \ldots, K_t,\ k \neq m, \\ 0, & \text{otherwise}, \end{cases}$$

where $D(\xi^s, \bar{\xi}^k) = \sum_{t=0}^{T^t} \|\xi_t^s - \bar{\xi}_t^k\|_2 = \sum_{t=0}^{T^t} \sqrt{\sum_{j=1}^{d} (\xi_t^{s,j} - \bar{\xi}_t^{k,j})^2}$, $s = 1, \ldots, S$, $j = 1, \ldots, d$, $k = 1, \ldots, K_\tau$. It is possible to apply other distance metrics, such as Manhattan distance, Maximum norm, Mahalanobis distance, in bundling similar scenarios, only some modifications have to be done in order to exploit the whole simulated data sequence.

Step 3. Centroid's evaluation. Compute $\bar{\xi}^k = (\bar{\xi}_t^k)$ as the mean of all scenarios assigned to cluster $\tilde{C}^k$:

$$\bar{\xi}^k = E\{\xi^s\}_{\xi^s \in \tilde{C}^k} = \frac{1}{|\tilde{C}^k|} \sum_{s=1}^{S} \delta(\xi^s, \tilde{C}^k) \xi^s,$$

where $|\tilde{C}^k| = \sum_{s=1}^{S} \delta(\xi^s, \tilde{C}^k)$ is the number of scenarios in the cluster $\tilde{C}^k$. The calculation of mean can be replaced by other estimate, such as median or mode.

Step 4. Repeat. Go to *Step* 2 until convergence. The termination criteria of convergence may be chosen as follows:

- Termination when convergence criteria is met, e.g., no (or very small) number of scenarios are assigned to different clusters, squared error.
- Termination when a fixed number of iterations has been carried out (this can also ensure stopping without convergence.

Step 5. Calculation of probabilities. Probability of $\bar{\xi}^k$ is equal to the sum of probabilities of individual scenarios ξ^s, belonging to the relevant cluster $\tilde{C}^k$. The probability can be evaluated from:

$$\widehat{\widehat{\pi}}(\bar{\xi}^k) = |\tilde{C}^k|/S,$$

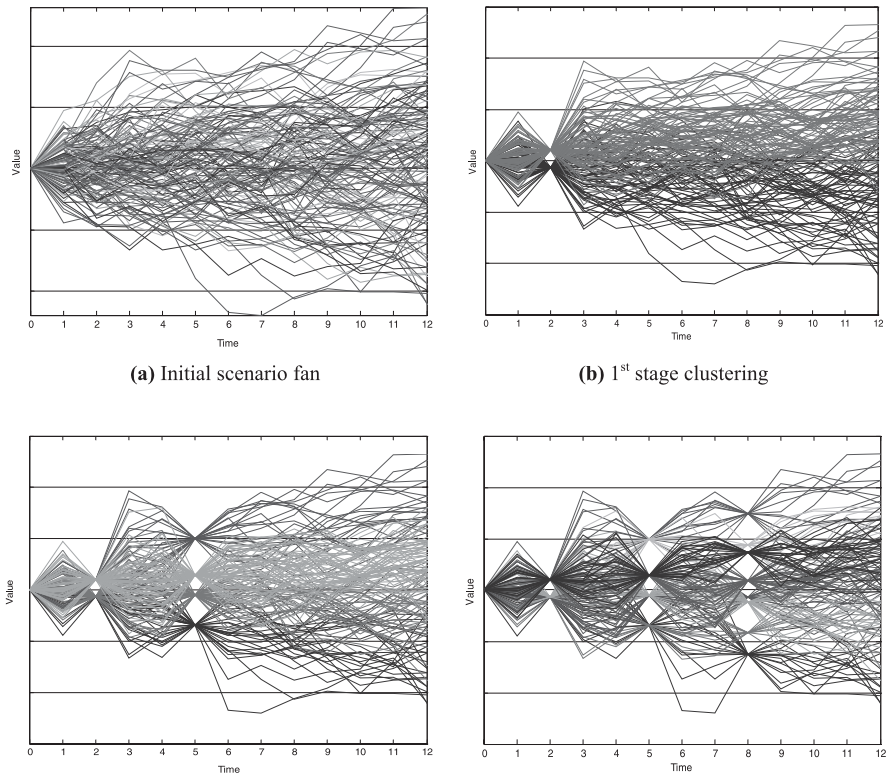
where $|\tilde{C}^k| = \sum_{s=1}^{S} \delta(\xi^s, \tilde{C}^k)$ is the number of scenarios in the cluster $\tilde{C}^k$.

Step 6. Modification. Modify $\xi^s = (\xi_0^s, \ldots, \xi_t^s, \ldots, \xi_{T^t}^s)$ by replacing ξ_t^s with $\bar{\xi}_t^k$ if $\xi_t^s \in \tilde{C}^k$ and $t = \tau$.

Step 7. Repeat. Go to *Step* 1 if the next stage index exists. The clustering procedure starts over for each cluster $\xi_s \in \tilde{C}^j$ formed in the current stage separately, where $j = 1, \ldots, \prod_{j=1}^{\tau} K_j$.

This algorithm produces a separation of scenarios into groups. The given algorithm lets to treat properly the inter-stage dependencies, exploiting the whole sequence of simulated scenario path. New sub-clusters are constructed from previous generated clusters at each defined stage, that's why this approach is named as multistage K-means clustering with varying $K_\tau,\ \tau \in \{1, \ldots, T^\tau\}$ in every stage. The output of multistage K-means clustering algorithm is the multistage scenario tree, which is delineated by nodes containing a cluster of scenarios (vectors in R^d), one of which is designated as centroids $\bar{\xi}^m = (\bar{\xi}_\tau^m),\ \bar{\xi}_\tau^m = (\bar{\xi}_\tau^{m,j}),\ m = 1, \ldots, \prod_{j=1}^{\tau} K_j,\ j = 1, \ldots, d,\ \tau \in \{1, \ldots, T^\tau\}$ with assigned probabilities $\widehat{\widehat{\pi}}(\bar{\xi}_\tau^m)$ and the branching scheme K_τ.

The described idea of bundling scenarios to the clusters is illustrated in Fig. 2, where the scenario fan is chosen to consist of one-dimensional data paths for better illustration of clustering performance. It is assumed that a set of individual scenarios for the entire time horizon (12 time moments) is already generated (Fig. 2a). The scenario fan of 100 scenarios is schematically illustrated. Let assume that we have three decision dates, i.e., we are planning to make decisions at 2, 5 and 8 time moments. Using the given notation, we have $\tau = \overline{1,3}$. The strategy is to construct the scenario tree with two branches $K_\tau = 2$ per each node in every stage τ. With this initial setting, we are ready to construct the 4-stage scenario tree. Thus, at the 1st decision moment ($\tau = 1$, time = 2) two clusters are formed by the first stage of multistage clustering algorithm (Fig. 2b). The centroid of each cluster is computed, which represents the second-stage node. Next, at previous step formed clusters are divided into two sub-clusters. It results that for the 2nd decision moment ($\tau = 2$, time = 5) we have four clusters representing third-stage nodes (Fig. 2c). Finally, the result of the third-stage of K-means clustering algorithm is eight clusters, since two more sub-clusters are formed from previous clusters at the 3rd decision moment ($\tau = 3$, time = 8; Fig. 2d).



(a) Initial scenario fan

(b) 1st stage clustering

(c) 2nd stage clustering

(d) 3rd stage clustering

Fig. 2. Illustration of 4-stage scenario tree's construction.

The computed nodes are denoted by black points in the clustered scenario fan. Joining these points by line, we get the graphical representation of scenario tree (Fig. 3). The constructed scenario tree has 4 stages and 8 scenarios. Such strategy of bundling scenarios to the clusters can continue till the end of time horizon is reached.

The comments about the developed multistage K-means clustering procedure are listed as follows:

- The projection of random variable nearer the time horizon is less critical than those for the near future, because number of scenarios grows smaller down the tree and the centroids that represent the scenario cluster are calculated from a smaller sample size.
- It allows to model extreme events because at every stage the simulated scenarios in all of the clusters are not discarded, and at the next stage all simulated scenarios in all of the clusters are used to determine new sub-clusters.

Two main approaches for determining the appropriate number of clusters are distinguished:

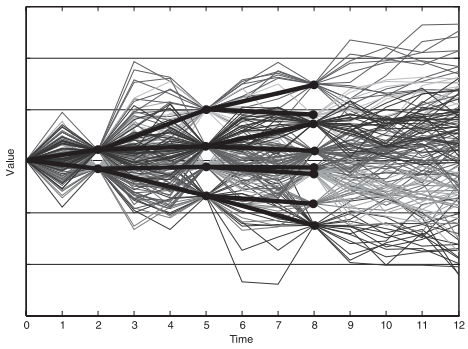


Fig. 3. Graphical representation of 4-stage scenario tree.

- Compatible cluster merging, when the clustering procedure starts with a sufficiently large number of clusters, and then this number is reduced by merging clusters that are similar (compatible) with respect to some predefined criteria (Setnes, 1999).
- Usage of validity measures to assess the goodness of obtained partitions for different values of number of clusters. Different validity measures, such as Dunn's index, Alternative Dunn's index or Silhouette value, have been proposed in the literature, none of them is perfect by oneself (Bruna *et al.*, 2007).

4. The Time Complexity of Multistage K-Means Algorithm

The time complexity refers to a function describing how much time it will take an algorithm to execute, based on the parameters of its input. An exact value of this function is usually ignored in favour of its order, expressed in O-notation.

In the case of one-dimensional scenarios, the time complexity of one-stage K-means clustering algorithm is estimated referring to the results given in references Lin *et al.* (2004), Arthur and Vassilvitskii (2006). We get:

$$T_1(S) = O(ST^tKI),$$

where T^t is the length of scenario, K is the number of clusters specified by the user, and I is the number of iterations until convergence of K-means clustering. The distance measure for one-dimensional scenarios is:

$$D(\xi^s, \bar{\xi}^k) = \|\xi^s - \bar{\xi}^k\|_2 = \sqrt{\sum_{t=0}^{T^t} (\xi_t^s - \bar{\xi}_t^k)^2}, \quad s = 1, \ldots, S,\ k = 1, \ldots, K.$$

Next, let evaluate the time complexity of multistage K-means clustering algorithm.

Theorem. *Assume that the multistage clustering problem consists of T^τ stages. Denote I as the number of iterations until convergence of K-means clustering task. The time complexity of multistage K-means clustering is $O(ST^t dI\sum_{\tau=1}^{T^\tau} K_\tau)$, where K_τ is the number of clusters specified by the user in stage τ.*

Proof. The distance measure of d-dimensional scenarios is given by:

$$D(\xi^s, \bar{\xi}^k) = \sum_{t=0}^{T^t} \|\xi_t^s - \bar{\xi}_t^k\|_2 = \sum_{t=0}^{T^t} \sqrt{\sum_{j=1}^{d} (\xi_t^{s,j} - \bar{\xi}_t^{k,j})^2},$$
$$s = 1, \ldots, S,\ j = 1, \ldots, d,\ k = 1, \ldots, K_\tau.$$

Comparing it with one-dimensional scenarios, the number of operations increases d times. Then, the time complexity of one-stage K-means clustering for d-dimensional scenarios is $T_d(S) = O(dT^t SKI)$. Let introduce κ_τ as the number of clustering tasks to be solved in stage τ, i.e.,:

$$\kappa_\tau = \begin{cases} \prod_{j=1}^{\tau-1} K_j, & \tau > 1, \\ 1, & \tau = 1. \end{cases}$$

In each stage τ, the data set of scenarios is divided into κ_τ non-overlapping sub-sets of size $S_{\tau,j}$, where $\sum_{j=1}^{\kappa_\tau} S_{\tau,j} = S$. Thus, the time complexity of multistage K-means clustering is as follows:

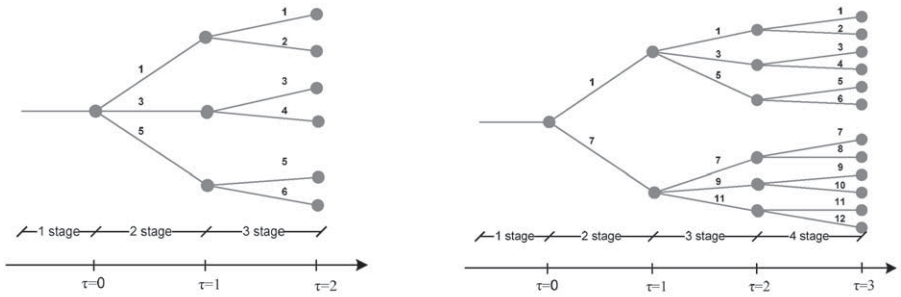
$$T_{\text{multi}}(S) = \sum_{\tau=1}^{T^\tau} \sum_{j=1}^{\kappa_\tau} T_d(S_{\tau,j}).$$

There exist positive constants c and s_0 such that $T_{\text{multi}}(S) \leqslant cdT^t \sum_{\tau=1}^{T^\tau} K_\tau \sum_{j=1}^{\kappa_\tau} S_{\tau,j} I_{\tau,j}$ for all $S_{\tau,j} \geqslant s_0$. Let us assume that $I = \max\limits_{\tau,j} I_{\tau,j}$ is the slowest convergence of K-means clustering task. Thus, we have $T_{\text{multi}}(S) \leqslant cdT^t I \sum_{\tau=1}^{T^\tau} K_\tau \sum_{j=1}^{\kappa_\tau} S_{\tau,j} = cdT^t IS \sum_{\tau=1}^{T^\tau} K_\tau$.

COROLLARY. If $K = K_\tau = \text{const}$ through all stages τ, then the time complexity of multistage K-means clustering is $O(ST^t dIT^\tau K)$.

5. The Exclusion of Duplicate Information from the Multistage Multidimensional Scenario Tree

As was described in the introduction, there exists a need to eliminate the redundant information in the scenario tree. In this section, the developed algorithm, which allows



a) 3-stage scenario tree, $K_1 = 3$, $K_2 = 2$ **b)** 4-stage scenario tree, $K_1 = 2$, $K_2 = 3$, $K_3 = 2$

Fig. 4. The illustration of scenario trees.

to discard the duplicate data from the multidimensional multistage scenario tree, is described.

Let assume that the decision vector $\tau \in \{1, \ldots, T^\tau\}$ is already defined and the number of clusters K_τ is known in every stage τ. We already declared that stages are connected with time moments when decisions have to be taken. Stages also show the branching moment in the scenario tree, where nonduplicate information reveals. Thus, in every stage the new branches are formed, which come up from parent branches; but these new branches are as a parents for the other branches in subsequent stages. That's why in developing the algorithm, we keep the idea that every scenario in the tree has its parent at the same time being a parent itself. Then, the nature of scenario tree leads to the fact that the longest branches appearing from stage τ will be as parents for branches in subsequent stages $\tau + 1, \ldots$. Thus, the proposed algorithm allows to determine the quantity of parent branches starting only from stage τ and their labels in every stage. These branches will capture the nonduplicate information. This purpose is schematically explained in Fig. 4.

In Fig. 4a the quantity of the parent branches starting from Stage 2 is 3 with their labels 1, 3, 5 in Stage 2, and the quantity of the parent branches starting from Stage 3 is 3 with their labels 2, 4, 6 in Stage 3. In Fig. 4b) the quantity of the parent branches starting from Stage 2 is 2 with their labels 1, 7 in Stage 2, the quantity of the parent branches starting from Stage 3 is 4 with their labels 3, 5, 9, 11 in Stage 3, the quantity of the parent branches starting from Stage 4 is 6 with their labels 2, 4, 6, 8, 10, 12 in Stage 4.

The algorithm for determining the parent branches and their labels at each stage $\tau > 0$ is as follows. The procedure has to be continued while the set I is non-empty:

Step 1. Generate the set of possible labels:

$$I = \{1, 2, \ldots, (K_1 K_2 \ldots K_{|\tau|})\},$$

where $|\tau|$ is the length of the decision vector.

Step 2. In the stage indexed by $\tau > 0$, the quantity of all branches is $q_\tau^s = K_1 K_2 \ldots K_\tau$. The quantity q_τ^p of the parent branches starting from the stage indexed by

$\tau > 0$ is $q_\tau^p = K_1 K_2 \dots K_{\tau-1}(K_\tau - 1)$. The label for each parallel branch $i = 1, \dots, q_\tau^s$ is obtained as

$$\vartheta_i^\tau = 1 + (i-1)\Delta_\tau,$$

where Δ_τ is the difference between the label of these branches:

$$\Delta_\tau = \begin{cases} K_{\tau+1} K_{\tau+2} \dots K_{|\tau|}, & \text{if } (|\tau|+1) > (\tau+1), \\ 1, & \text{else.} \end{cases}$$

The new set of labels is formed $I_\tau = \{\vartheta_i^\tau\},\ i = 1, \dots, q_\tau^s$. Then

- If $\tau = 1$, assign the set of labels for parent branches as$L_1^p = I_1$.
- If $\tau > 1$, assign the set of labels for parent branches starting from stage indexed by $(\tau - 1)$ as $L_\tau^a = I_{\tau-1}$ and the set of labels for parent branches starting from stage indexed by τ as $L_\tau^p = I_\tau \backslash I_{\tau-1}$. The initial set I is modified by calculating the complement $I = (I \backslash I_\tau)$.

Then, the new stage is analyzed if I is non-empty.

This algorithm can be applied before generating *stoch* text file of multistage multidimensional scenario tree. *Stoch* text file belongs to SMPS (Stochastic Mathematical Programming System) format (Gassmann and Kristjánsson, 2008), which is widely used in solvers for solving stochastic programming problems.

6. Testing the Scenario Generation Algorithm for Stability

Usually scenario tree generation methods differ in their ability to describe randomness. Since the proposed method involves the randomness, it should be tested for stability (Kaut and Wallace, 2003). Let denote the constructed scenario tree by $\breve{\xi} = \{\breve{\xi}_\tau\}_{\tau=1}^{T^\tau}$. The stability requirement means that if we generate G scenario trees $\breve{\xi}^g = \{\breve{\xi}_\tau^g\}_{\tau=1}^{T^\tau},\ g = \overline{1, G}$ and solve the stochastic programming problem with each tree, we should get approximately the same optimal value of the objective function. This may also be seen as robustness requirement on the scenario generation method. In general, two types of stability tests are performed: in-sample test, and, if feasible, the out-of sample test. The important difference between these two definitions is: we need to solve the scenario-based optimization problem for testing the in-sample stability, but we have to be able to evaluate the "true" objective function for the out-of-sample stability. To do the latter test, we need to have the full knowledge of the underlying distribution, which is not always the case.

6.1. *Asset Liability Management as Multistage Stochastic Linear Problem*

The stability of proposed scenario tree generation algorithm is tested on the optimization problem of Asset Liability Management (ALM; Kouwenberg and Zenios, 2006). In general, there exist J possible asset classes for allocating resources. The solution of optimization model will consist of the initial and recourse decisions for recommended

asset mixes by different combinations applied to the investment portfolio, i.e., weights $(\alpha_1, \ldots, \alpha_J)$ of asset allocation to various investments J. The following formulation is fairly standard in ALM applications of stochastic programming (Hilli *et al.*, 2007).

Inventory constraints are used to describe the dynamics of holdings in each asset class:

$$h^s_{0,j} = h^0_j + p^s_{0,j} - q^s_{0,j},\ h^s_{\tau,j} = R^s_{\tau,j} h^s_{\tau-1,j} + p^s_{\tau,j} - q^s_{\tau,j},$$
$$\tau \in \{1, \ldots, T^\tau\},\ s = 1, \ldots, S_{T^\tau},\ j = 1, \ldots, J,$$

where h^0_j – initial holdings in asset j, $R^s_{\tau,j}$ – return on asset j (random) over stage $[\tau - 1, \tau]$ in scenario s are parameters; $p^s_{\tau,j}$ – non-negative purchases of asset j at time τ in scenario $s, q^s_{\tau,j}$ – non-negative sales of asset j at time τ in scenario $s, h^s_{\tau,j}$ – holdings in asset j in period $[\tau, \tau + 1]$ are decision variables.

Budget constraints are used to guarantee that the total expenses do not exceed revenues:

$$\sum_{j \in J} (1 + k^p_j) p^s_{\tau,j} \leqslant \sum_{j \in J} (1 - k^q_j) q^s_{\tau,j} + V_\tau - L_\tau,$$
$$\tau \in \{0, \ldots, T^\tau\},\ s = 1, \ldots, S_{T^\tau},\ j = 1, \ldots, J,$$

where $k^p_j \geqslant 0$ – transaction costs for buying asset j, $k^q_j \geqslant 0$ – transaction costs for selling asset j, V_τ – cash inflows (random) in period $[\tau - 1, \tau], -L_\tau$ – cash outflows (random) in period $[\tau - 1, \tau]$ are parameters.

Portfolio constraints give limits for the allowed range of portfolio weights:

$$\underline{\mathrm{b}}_j \sum_{j \in J} h^s_{\tau,j} \leqslant h^s_{\tau,j} \leqslant \bar{b}_j \sum_{j \in J} h^s_{\tau,j},$$
$$\tau \in \{0, \ldots, T^\tau\},\ s = 1, \ldots, S_{T^\tau},\ j = 1, \ldots, J,$$

where $\sum_{j \in J} h^s_{\tau,j}$ – total wealth at time τ, $\underline{\mathrm{b}}_j$ – lower bound for the proportion of $\sum_{j \in J} h^s_{\tau,j}$ in asset j, $\bar{b}_j$ – upper bound for the proportion of $\sum_{j \in J} h^s_{\tau,j}$ in asset j are parameters.

Of course, the income should be sufficient to cover the liabilities and to earn the gain. To encourage such outcomes, let Ψ_τ be the target wealth at the horizon $\tau = T^\tau$, $\bar{\mathrm{w}}^s_\tau$ be an excess over target wealth at horizon $\tau = T^\tau$, $\underline{\mathrm{w}}^s_\tau$ be a deficit under target wealth at horizon $\tau = T^\tau$. The objective function will include d_1, the penalty coefficient for the shortfall, and d_2, the reward coefficient for the surplus. Thus, the required wealth constraint is:

$$\sum_{j \in J} R^s_{T^\tau,j} h^s_{T^\tau-1,j} + V_{T^\tau} - L_{T^\tau} - \bar{\mathrm{w}}^s_{T^\tau} + \underline{\mathrm{w}}^s_{T^\tau} = \Psi_{T^\tau}, \quad s = 1, \ldots, S_{T^\tau}.$$

The objective function is given as

$$\min \sum_{s=1}^{S_{T^\tau}} \widehat{\bar{\pi}}_s [d_1 \cdot \underline{\mathrm{w}}^s_{T^\tau} - d_2 \cdot \bar{w}^s_{T^\tau}],$$

where $\widehat{\widehat{\pi}}_s$ – probability of scenario s.

The above presented ALM model is applied for management of insurance company. The revenues from the performance of investment and underwriting business are added to the insurer's asset, while the wealth is depleted both by outflows allocated to various investments and by claims of its clients. The main goal of a company is to earn the profit. The models for asset returns $R^s_{\tau,j}$, for insurance underwriting cash flows V_τ, L_τ are not detailed in this paper and can be found in Kaufmann *et al.* (2001), Hibbert *et al.* (2001). The risk factors of investment activity are described by J-dimensional multistage scenario tree, while the risk factors of insurance underwriting activity are described by scenario fans.

6.2. *The Results of Numerical Experiment*

In the paper Kaut and Wallace (2003), it is stated that in most applications the in-sample test should be sufficient in detecting a possible instability. However, if there is a way to do the out-of-sample test, it is recommended to perform it as well. Since we don't have a representation of the true distribution, we will perform the in-sample stability testing, i.e., we are going to test if holds the following equation:

$$\min_w F(w; \breve{\xi}^g) \approx \min_w F(w; \breve{\xi}^k), \quad \forall g \neq k,\ w = (\underline{w}, \bar{w}),$$

where $F(\cdot)$ is the objective function given in Section 6.1.

The settings for a numerical experiment are as follows. We set $J = 3$, i.e., three asset classes are possible for investment: cash, bonds, stocks. The investments are bounded with lower limit and upper limit as follows: cash $\in [0, 0.2]$, bonds $\in [0.4, 0.7]$, and stocks $\in [0.3, 0.6]$. The investment returns $R^s_{\tau,j}$ (Section 6.1) are described by multistage 3-dimensional scenario tree, which is constructed by multistage K-means clustering algorithm developed in this paper. Stages in scenario tree denote the decision moments: first stage is indexed as $\tau = 0$, and the recourse stages are indexed as $\tau = (1, 3, 6, 10)$ in years. It determines that we have 5 stages during 10 years time horizon. To test the in-sample stability, 100 five-stage 3-dimensional scenarios tree are generated, each of them having branching scheme $K_\tau = 2$ and $K_\tau = 3$. The initial investment consists of initial surplus $\sum_{j=1}^3 h^0_j = 1 \times 10^4$ at time $\tau = 0$. The target wealth is set equal to $\Psi_{T^\tau} = 2.5 \times 10^5$. The sample means and standard deviations of optimal function (Section 6.1) are given in Table 1.

In Table 1, the stability of scenario generation algorithm can be observed if high density clustering scheme is chosen for a small number of simulated data paths; for large number of scenarios, the clustering scheme can be sparser.

Additionally, the statistical t-test method is used to test a null hypothesis whether the difference in the mean value of any two samples is equal to zero. On the whole, fifteen tests (the number of ways that two cases can be chosen from among six cases, i.e., the binomial coefficient $\binom{6}{2}$) were performed. Statistical t-test method showed that sample means are statistically equal for the cases within the same branching scheme. In summary, the obtained results for a value of objective function show in-sample stability of a given scenario tree generation algorithm.

Table 1
In-sample stability test of scenario tree generation algorithm

Case No.	Branching scheme $K_\tau = 2$			Branching scheme $K_\tau = 3$		
	1	2	3	4	5	6
# of simulated data paths (scenario fan)	1000	1500	2000	1000	1500	2000
Mean of objective function, $\cdot 10^5$	-0.6429	-0.7285	-0.8711	-1.5161	-1.3205	-1.1259
Standard deviation of objective function, $\times 10^5$	0.5358	0.6245	0.6330	0.7331	0.7259	0.6113

7. Conclusions

In the present paper, we described the algorithm based on simulation and multistage K-means clustering to generate the multistage d-dimensional stochastic scenario tree from d-dimensional scenario fans. It is proved that the time complexity of the developed multistage K-means clustering algorithm is linear regarding the number of scenarios in the fan. The proposed scenario tree generation algorithm is motivated by the stability of optimal values, obtained from the multistage stochastic optimization problem of asset liability management for insurance company. The algorithm for determining the parent scenarios in the d-dimensional multistage scenario tree is also developed, which allowed to exclude the duplicate information in every stage of scenario tree. This point of view is advantageous because it allows for a reduction of redundancy in the tree.

References

Arthur, D., Vassilvitskii, S. (2006). Worst-case and smoothed analysis of the ICP algorithm with an application to the K-means method. In: *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. CA, pp. 153–164.

Bruna, M. *et al.* (2007). Model-based evaluation of clustering validation measures. *Pattern Recognition*, 40(3), 807–824.

Domenica, N.D. *et al.* (2007). Stochastic programming and scenario generation within a simulation framework: an information systems perspective. *Decision Support Systems*, 42(4), 2197–2218.

Dupačová, J., Consigli, G., Wallace, S.W. (2000). Scenarios for multistage stochastic programs. *Annals of Operations Research*, 100, 25–53.

Dupačová, J., Hurt, J., Štěpán, J. (2002). *Stochastic Modeling in Economics and Finance*. Kluwer Academic, Netherlands.

Dzemyda, G., Sakalauskas, L. (2009). Optimization and knowledge-based technologies. *Informatica*, 20(2), 165–172.

Gassmann, H.I. (2006). *The SMPS Format for Stochastic Linear Programs*. `http://myweb.dal.ca/gassmann/ smps2.htm`

Gassmann, H.I., Kristjánsson, B. (2008). The SMPS format explained. *IMA Journal of Management Mathematics*, 19(4), 347–377.

Gröwe-Kuska, N., Heitsch, H., Römisch, W. (2003). Scenario reduction and scenario tree construction for power management problems. In: *Power Tech Conference Proceedings*. IEEE Bologna, Italy.

Gülpinar, N., Rustem, B., Settergren, R. (2004). Simulation and optimization approaches to scenario tree generation. *Journal of Economic Dynamics & Control*, 28, 1291–1315.

Heitsch, H. (2003). Scenario reduction algorithms in stochastic programming. *Computational Optimization and Applications*, 24, 187–206.

Heitsch, H., Romisch W. (2005). Scenario tree modeling for multistage stochastic programs. *Mathematics for Key Technologies*, DFG Research Center MATHEON, Berlin, Germany, Preprint, 296.

Hibbert, J., Mowbray, P., Turnbull, C. (2001). *A Stochastic Asset Model & Calibration for Long-Term Financial Planning Purposes*. `http://www.actuaries.org.uk/files/pdf/library/proceedings/fin_inv/2001/hibbert.pdf`.

Hilli, P. *et al.* (2007). A stochastic programming model for asset liability management of a Finnish pension company. *Journal Annals of Operations Research*, 125(1), 115–139.

Kaufmann, L., Rousseeuw, P.J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, Canada.

Kaufmann, R., Gadmer, A., Klett, R. (2001). Introduction to dynamic financial analysis. *Astin Bulletin*, 31, 213–250.

Kaut, M., Wallace, S.W. (2003). Evaluation of scenario generation methods for stochastic programming. *Stochastic Programming E-Print Series*, 14. `http://www.speps.org/`.

Kouwenberg, R., Zenios, S.A. (2006). *Handbook of Asset and Liability Management*. Elsevier Science & Technology, 253–304.

Krilavičius,T., Žilinskas, A. (2008). On structural analysis of parliamentarian voting data. *Informatica*, 19(3), 377–390.

Lin, J. *et al.* (2004). Iterative incremental clustering of time series. In: *Proceedings of 9th International Conference on Extending Database Technology*. Greece, pp. 106–122.

Mitra, S. (2006). *A White Paper on Scenario Generation for Stochastic Programming*. `www.optirisk-systems.com/papers/SGwhitepaper.pdf`.

Möller, A., Romisch, W., Weber, K. (2004). A new approach to O&D revenue management based on scenario trees. *Journal of Revenue and Pricing Management*, 3, 265–276.

Pranevičius, H., Šutienė, K. (2007). Scenario tree generation by clustering the simulated data paths. In: *Proceedings of 21st European Conference on Modeling and Simulation*. Prague, pp. 203–208.

Setnes, M. (1999). Supervised fuzzy clustering for rule extraction. In: *Proceedings of FUZZIEEE'99*. Seoul, Korea, pp. 1270–1274.

Teknomo, K. (2006). *K-Means Clustering Tutorial*. `http://people.revoledu.com/kardi/tutorial/kMean/index.html`.

K. Šutienė, doctor of informatics sciences, is a lecturer in the Department of Business Informatics at Kaunas University of Technology. The field of research – statistical learning, stochastic simulation and optimization, operational research applied to business processes.

D. Makackas, doctor of informatics sciences, is an associated professor in the Department of Business Informatics at Kaunas University of Technology. His main research interests include formal methods, real time systems and theory of algorithms.

H. Pranevičius, hab. doctor of mathematical sciences, is a professor in the Department of Business Informatics at Kaunas University of Technology. The field of research – formal specification, validation and simulation of complex systems, knowledge-based simulation, and development of numerical models of systems specified by Markov processes. The results of investigations have been successfully applied in creating the computerized systems for specification, validation and simulation/modeling of computer network protocols, logistics and industrial systems. The theoretical background of investigation is Piece-Linear Aggregate formalism, which permits to use the formal specification for model development and behavior analysis.

Daugiaetapis K-vidurkių klasterizavimo metodas scenarijų medžiui konstruoti

Kristina ŠUTIENĖ, Dalius MAKACKAS, Henrikas PRANEVIČIUS

Sprendžiant stochastinio programavimo ir sprendimų analizės uždavinius, vienas svarbiausių tikslų yra daugiamačio stochastinio proceso apytikslis reprezentavimas scenarijų medžiu. Šiame straipsnyje yra išdėstomas metodas, skirtas daugiamačio daugiaetapio scenarijų medžio generavimui iš scenarijų vėduoklių aibės. Šiam tikslui yra sukurtas daugiaetapis K-vidurkių klasterizavimo algoritmas. Siūlomas scenarijų medžio generavimo metodas yra stabilus daugiaetapės stochastinės programos optimalių reikšmių atžvilgiu. Įrodyta, kad daugiaetapis klasterizavimo algoritmas turi tiesinį laiko sudėtingumą scenarijų skaičiaus vėduoklėje atžvilgiu. Kaip tarpinis rezultatas yra pateiktas algoritmas, kuris leido daugiamačiame scenarijų medyje išskirti nesidubliuojančią informaciją.