Session 5.

Decision Making

THE PERFORMANCE OF PLANNING DECISION MODELS IN UNCERTAIN ENVIRONMENT

Henrikas Pranevicius, Kristina Sutiene

Department of Business Informatics
Kaunas University of Technology
Studentu 56-301, Kaunas, 51424, Lithuania
Tel.: +370 681 52842. Fax: +370 37 451654
E-mail: hepran@if.ktu.lt, kristina.sutiene@stud.ktu.lt

In most cases, the planning has to be carried out under uncertainty. Thus, some or all model parameters are not completely known at the current point of time when decision has to be taken. The purpose of this work is to compare two different approaches of taking decisions for asset liability management problem in insurance: stochastic simulation and stochastic programming (optimisation). Such problems are influenced by a large number of stochastic parameters, and decisions are due to the assets that must be invested over time to achieve sufficient returns to cover liabilities and to achieve goals subject to various uncertainties and various constraints. Two approaches mostly differs how the decisions are made: the first uses efficient frontier concept, while the other uses multistage stochastic optimisation. A common feature of these models is the fact that a stochastic process describing the uncertain environment (asset prices, insurance claims, …) is the most important part of the input data. The pros and cons of these two approaches are discussed, and the obtained results of some considered example are compared.

Keywords: *stochastic simulation, stochastic optimisation, scenario generation, portfolio construction, decision-making*

1. Introduction and Problem Statement

Decision-making under uncertainty is one of the foremost challenges for any financial institution. This is especially true for dynamic decision problems, where the uncertainty is related to the future realizations of certain key variables. Financial institutions, like insurance companies, pension funds, banks, need effective strategic planning for management of their financial resources and liabilities in stochastic environments, with market, economic and actuarial risks all playing an important role. The typical planning horizon is very long. This means that the fund portfolio will have to be rebalanced many times, making "buy&hold" Markowitz's style portfolio optimisation inefficient. The management of financial institutions can also be dictated by a number of solvency requirements which are put in place by the appropriate regulating authorities [1].

The purpose of this paper is to quantitatively compare two different approaches of the same underlying decision problem – investment portfolio management under uncertainty. The first is based on decision-making using stochastic simulation model, while the other is multistage stochastic programming (optimisation). The basic dynamic decision-making problem under uncertainty treated in this paper is: given a fixed planning horizon and a set of portfolio rebalance dates, find the dynamic investment decision (strategy) that maximizes the expected wealth of some enterprise subject to constraints, such as borrowing, portfolio limits, business strategy and other.

The methodology is explained in the context of a specific portfolio problem of insurance company. Insurance business has two basic sides – the collection of premiums for accepting the risk for others and the investment of those premiums. The investment side is similar to other financial institutions like pension funds, banks. The core of the insurance business is to invest the premiums and previous earnings from investments to yield good asset returns over time and to provide resources for insurance claims. These claims have distributions of losses, as with typical liabilities [2]. The responsibility of insurance companies is to hedge the client's risks, while meeting the solvency standards, in such a way that all payments are met. Hence, they are classical asset liability enterprises seeking the methods for the best or optimal management of their resources. The figure below shows the time flow of assets arriving and liability commitments for insurance company.

While comparing the stochastic simulation and stochastic programming (optimisation) approaches, it is important to note that the former allows finding the decision that is the best from the set of tested

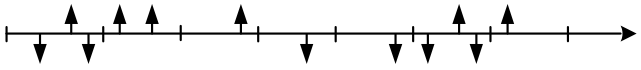

Fig. 1. *Cash flows in insurance*

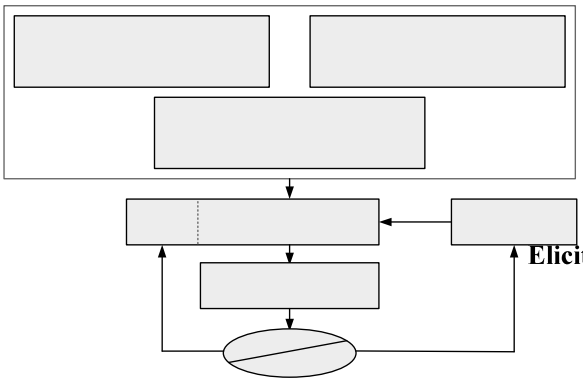

Fig. 2. *Decision-making framework for asset liability management*

alternative decisions, using the efficient frontier concept, while the latter allows to obtain the optimal decision subject to some restrictions and constraints under which the relevant system must operate. Due to the agreement that optimisation problems are not trivial tasks, we tried to improve the stochastic simulation performance, i.e. we involve the possibility for simulation approach to adapt to the new stochastic information available at every stage. This is done by generating the new instance to the model, and then testing the alternative decisions. In other words, simulation analysis is recomputed in each stage with a rolling horizon.

Asset liability modelling and management models usually require knowledge from different fields such as statistics, economics, financial mathematics and optimisation. In a schematic and generic way, the different elements of asset liability models can be organized along three separate poles of interest and a common set of structural considerations [3]. The first task is to declare of future asset and liability cash flows that are to be expected. Then, it is needed to set the objectives of an organization (or an individual). The final task is to choose the set of possible investment vehicles that we want to include in the decision-making process. Such structure creates a framework for decision-making under uncertainty (Fig. 2).

The paper is organized as follows. In Section 2 the current research area in planning decision models under uncertainty is described. The main features of decision simulators are given in Section 3. The role of scenarios and different structures of scenarios are described in Section 4. Finally, two considered approaches are compared through the application of investment portfolio selection.

2. Current State of the Art

Stochastic programming (optimisation) is very intensive research area at the present. It is relatively a new approach applied for decision-making under uncertainty, such as are asset liability management problems. Russell-Yasuda Kasai was the first large scale multi-stage stochastic programming model developed for the Yasuda Fire and Marine Insurance Co., Ltd. in Tokyo in 1991 year [2]. It was a commercial product, and originally it had ten stages and 2048 scenarios. It was too big to solve at that time and became an intellectual challenge for the stochastic programming community. Only in two year duration, the model was solved using parallel processing on several workstations at IBM Research Centre. Yasuda Kasai's previous methodology was to calculate portfolio weights by period using mean-variance model. The multi-stage stochastic programming model was compared with a static mean-variance model that recomputed in each period. The authors demonstrate the superiority of the multistage dynamic portfolio formulation both in terms of return characteristics and general understanding of the company's business and the effect of various policies and constraints [4]. A detailed study of the performance of stochastic dynamic and fixed mix portfolio models was made by Fleten, Hoyland and Wallace [5]. The authors compared two alternatives of a portfolio model for the Norwegian life insurance company Gjensidige NOR. They found that the multi-stage stochastic programming model dominated the fixed-mix approach, but the degree of dominance was much smaller out of sample than in sample, because the stochastic programming model loses its advantage in optimally adapting to the information available in sample case. Also, the performance of the fixed-mix approach improves because the asset mix is updated at each stage. A lot of researches are performed not only in insurance applications, like asset liability enterprises, but also in management of pension funds, cash management in banking. Thus, one of the most important research fields is to compare and evaluate the performance of various decision-making models under uncertainty. The multi-stage stochastic programming approach seems to be very perspective. At the present, the main attention is paid to real applications of stochastic decision models and to development of effective solvers.

The alternative to stochastic programming is dynamic programming. These approaches mainly differ in a way in which they model uncertainty in the environment [6]. Dynamic programming captures randomness by allowing for a continuum of states which can be described at a given point in time by small number of state variables following joint Markov's processes, while stochastic programming in contrast captures uncertainty by

a branching scenario tree where every node represent a joint outcome of all random variables at that decision stage. A major advantage of dynamic programming is that a quite limited amount of situations analytic results can be derived. Drawbacks of this approach are that solutions involve computationally very burdensome numerical solutions procedures and the size of the dynamic programming problem grows exponentially with the number of state variables and becomes quickly unusable for realistic problems.

3. Decision Simulator under Uncertainty

Decision simulator mimics the company's (or decision maker's) decisions over the planning period. An insurance company, for example, makes decisions regarding their asset mix, business strategies, and the firm's capital structure. In most cases, the planning has to be carried out under uncertainty because some or all model parameters are not completely known at the current point of time when decision has to be taken. Despite rich involvement of the future, everything is aimed to make a well hedged decision in the present. The attitude is adopted that a decision will be properly made in the present taking into account the opportunities for modification or correction at later times [7]. Decisions at later times can respond to the information that has become available since the initial decision. Thus, during the time the decisions alternate with observations: initial decision $\rightarrow$ observation $\rightarrow$ recourse decision $\rightarrow$ observation $\rightarrow$ … $\rightarrow$ recourse decision. This sequence doesn't go on indefinitely, but the number of stages can be large enough. Decisions that are taken have no effect on the probability structure. Thus, we have a multi-stage decision-making formulation (Fig. 3).

Thus, the decisions are of two types: the initial decision x_0 taken at the current time moment and the recourse decisions x_t, $t > 0$ taken at $t \in T$ recourse stages. Note that the spaces from which the decisions are to be chosen are taken as finite-dimensional but of possible varying dimensionality. In the finite horizon, we have a record of decisions taken, expressible as a vector $x := (x_0, x_1, \ldots, x_T) \in \mathbf{R}^n$. The constraints on a decision at each stage involve past observations and decisions.

The multi-stage decision-making model must avoid looking into the future in an inappropriate fashion. To prevent this occurrence, the special constraints are added to the model, called non-anticipatory conditions. Thus, the decision process is said to be non-anticipative due to the reference [8], i.e. the decision taken at any time t does not directly depend on future realizations of stochastic parameters or on future decisions. At the time when initial decision must be chosen, nothing about the random elements in our process has been pinpointed. But in making a recourse decision we have the revealed current information until this moment and the residual uncertainty till the end of time horizon. More information on decision-making under uncertainty can be found in references [9], [10], [11].

Due to the purpose of this paper, we emphasis on how decisions are obtained. At one case, these decision-making problems may be quite simple requiring the determination of the values of a small number of manageable variables with only simple conditions to be met; and at the other case they may be large scale and quite complex with thousands of variables and many conditions to be met. Decision-making always involves making a choice between various possible alternatives. Thus, decision problems can be classified into two categories with very distinct features [11]:

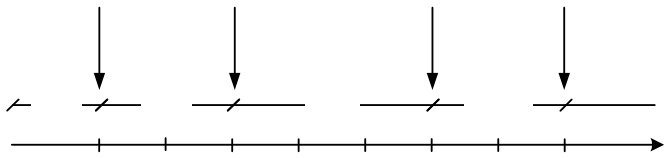

Fig. 3. *The planning period*

Category 1: It includes all decision problems for which the set of possible alternatives for the decision is a finite discrete set typically consisting of a small number of elements, in which each alternative is fully known in complete detail, and any one of them can be selected as the decision.

Category 2: It includes all decision problems for which each possible alternative for the decision is required to satisfy some restrictions and constraints under which the relevant system must operate. Even to identify the set of all possible alternatives for the decision, we need to construct a mathematical model of these restrictions and constraints in this category. Even when there are no constraints to be satisfied in a decision problem, if the number of possible alternatives is either infinite, or finite but very large; it becomes necessary to define the decision variables in the problem, and construct the objective function (the one to be optimised) as a mathematical function of the decision variables in order to find the best alternative to implement.

Decision problems of Category 1 can be solved using the first considered approach – stochastic simulation, where one of the most common techniques used to present the results is the efficient frontier [12]. This is a technique borrowed from finance theory for constructing the investment portfolio framed in terms of risk and return. "Return" is usually defined as arithmetic mean of key variable, and "risk" is defined as corresponding standard deviation. Whatever definition of risk and return we wish to apply, we can define an "efficient" set of decisions (strategies). A decision is called efficient if there is no other one with lower risk at the same level of return, or higher return at the same level of risk. For each level of risk there is a maximal return that cannot be exceeded, giving a rise to an efficient frontier. But we cannot be sure that a decision is really efficient or not. Stochastic simulation is not necessarily a method to come up with an optimal strategy. It is predominantly a tool to compare different decisions. It is important to note that a different measure of risk and return may lead to a different preferred decision. Optimal decisions can be found when decision problems are formulated according to the concept of Category 2, as it is the main concept of the stochastic optimisation approach.

A system for financial decision-making under uncertainty requires a systematic method for projecting the random variables over the planning horizon. In financial and industrial applications, the 'scenario' terminology is used to describe how the future might unfold and can be considered as data streams, whose points are multi-dimensional data.

4. The Role of Scenarios

To construct a multi-stage financial decision support system, the underlying multivariate stochastic data process has to be discrete in time. Mathematically, we have an index for time discretization $\tau = \{0,\ldots,\widetilde{T}\}$, where $\widetilde{T}$ is a time horizon. The stochastic process $\xi = \{\xi_\tau\}_{\tau=1}^{\widetilde{T}}$ is defined on some filtered probability space $(\Omega, S, \mathsf{F}, P)$. At the current time moment $\tau = 0$, value ξ_0 is known with certainty. The sample space Ω is defined as $\Omega := \Omega_1 \times \Omega_2 \times \ldots \times \Omega_{\widetilde{T}}$, where $\Omega_\tau \subset \mathbf{R}^d$. The σ-algebra S is the set of events with assigned probabilities by measure P, and $\{\mathsf{F}_\tau\}_{\tau=1}^{\widetilde{T}}$ is a filtration on S. Let denote the d-dimensional probability distribution function of $\xi_\tau = (\xi_\tau^1,\ldots,\xi_\tau^d)'$ at point $y = (y_1,\ldots,y_d)'$ by $f(y)$, and its d-dimensional cumulative distribution function by $F(y)$. The joint distribution F provides a complete information concerning the behaviour of ξ. The marginal probability distribution function and cumulative distribution function of each element ξ_τ^i at point y_i, $i = 1,\ldots,d$ is denoted by $f_i(y_i)$ and $F_i(y_i)$, respectively. Thus, the underlying probability distribution f is replaced by a discrete distribution P carried by a finite number of atoms $\xi^s = (\xi_1^s,\ldots,\xi_\tau^s,\ldots,\xi_{\widetilde{T}}^s)$, $\xi_\tau^s = (\xi_\tau^{s,1},\ldots,\xi_\tau^{s,d})'$, $s = 1,\ldots,S$ with probabilities $p_s = P(\xi^s)$, $p_s \geq 0$ and $\sum_{s=1}^{S} p_s = 1$. The atoms ξ^s, $s = 1,\ldots,S$ of the distribution P are called as scenarios. Since all scenarios coincide at $\tau = 0$, the initial root node ξ_0 (vector in $\mathbf{R}^d$) is formed, and thus the simulated data paths are called as a scenario fan (Fig. 4), where each scenario can be viewed as one realization of an underlying multivariate stochastic data process. The structure of simulated data paths can be divided into two stages, as all σ-fields F_τ, $\tau = 1,\ldots,\widetilde{T}$ coincide. The first stage is usually represented by a single root node, and the values of random parameters during the first stage are known with certainty. Moving to the second stage, the structure branches into individual scenarios at time $\tau = 1$, as shown in Figure 4. Under requirements of decision-making under uncertainty [8], the decisions are made only at the end of the first stage if structure of scenarios fan is used.

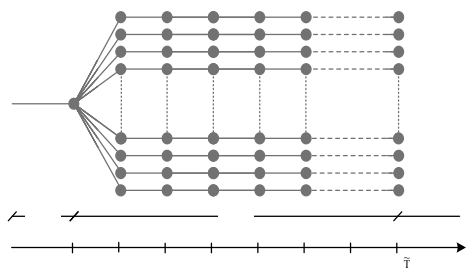

Fig. 4. *Scenario fan*

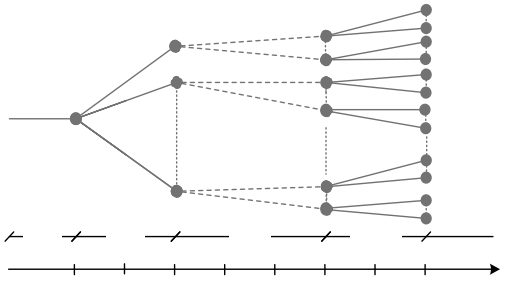

Fig. 5. *Multi-stage stochastic scenario tree*

Let introduce the index for stages $t = \{0,\ldots,T\}$. In general, stages are only those points in time where a decision is made. The time step of discretization has to be smaller than the time step for stages, thus $\{0,\ldots,T\} \subset \{0,\ldots,\tilde{T}\}$. The scenarios are given in the form of multi-stage scenario tree (Fig. 5), which reflects the inter-stage dependency and decreases the number of nodes while comparing to the scenario fan. The structure of multi-stage scenario tree at $t = 0$ is also described by a sole root node and by branching into a finite number of scenarios as it was in previous case. The stages are connected with possibility to take additional decisions based on newly revealed information. Such information can be obtained periodically (every day, week, and month) or based on some events (expiration of investment portfolio duration). The distinction between stages, which correspond to the decision moments, 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. In the scenario tree, each path through the tree from its root to one of its leaves corresponds to one scenario, i.e. to a particular sequence of realizations of random coefficients. If only two stages exist, the scenario tree has the form of scenario fan.

Thus, the factors driving risky events are approximated by a discrete set of scenarios of the required structure. This process is known as scenario generation. Due to the origin of scenarios can be very diverse, to generate them it is natural to use historical data (if available) in conjunction with an assumed background model. In the paper [13], four types of problems are distinguished concerning the level of the available information: (a) full knowledge of underlying probability distribution, (b) known parametric family, (c) sample information, and (d) low information level. Scenarios can be generated using various methods, based on different principles [14]:

- Scenario Generation by Statistical Approaches: Statistical Moment or Property Matching, Principal Components Analysis (PCA), Regression and its variants;
- Scenario Generation by Sampling: Monte Carlo Sampling, Importance Sampling, Bootstrap Sampling, Internal Sampling, Conditional Sampling, Markov Chain Monte Carlo Sampling;
- Scenario Generation by Simulation: Stochastic Process Simulation, Error Correction Model, Vector Auto-regressive;
- Other Scenario Generation Methods: Artificial Neural Networks, Clustering, Scenario Reduction, Hybrid Methods.

A good approximation may involve a very large number of scenarios with probabilities. A better accuracy of uncertainties is described when scenarios are constructed via a simulated data path structure, also known as a scenario fan. But the number of scenarios is limited by the available computing power, together with a complexity of the decision model. To deal with this difficulty, we can reduce the dimension of the initial scenario set by constructing the multistage scenario tree out of it.

5. Comparing the Performance of Stochastic Simulation and Stochastic Optimisation Approaches

We will compare the stochastic simulation and stochastic optimisation approaches in the context of a specific investment portfolio problem of insurance company. The portfolio selection problem is modelled at the strategic level, where resources are allocated among a few aggregated asset classes, such as cash, stock and bonds. The objective is to maximize the expected portfolio value at the end of the horizon net of costs, subject to some constraints.

The randomness is a characteristic of both asset side and liability side of insurance company. Thus, a large number of scenarios has to be created by the means of simulation. More detailed information is given in Section 5.1. In this paper, the investment portfolio selection problem is considered; thus the scenario tree has to be constructed only for asset returns.

The advantage of stochastic programming (optimisation) model is to adapt to the information available in the scenario tree (Fig. 5), allowing to produce the initial and the recourse decisions in every stage. The stochastic simulation approach is performed in the context of scenario fan (Fig. 4); and decisions are made only at the end of the first stage. In such comparison, we anticipate that stochastic optimisation will perform better results. Thus, stochastic simulation is modified using rolling horizon simulations, allowing simulation model to evaluate the system state at recourse decision moments. In stochastic simulation, we have to construct the set of alternatives decisions to be tested, while stochastic optimisation produces the optimal decisions at once due to the given constraints.

5.1. Scenario generation

The asset and liability management is based on scenarios that are used to represent the uncertainty of random parameters. In the case of insurance company, we need a model to generate scenarios that correspond to the different financial variables and asset classes, and a model to generate future paths of liabilities derived from underwriting business. The first type involves stochastic processes or their discrete approximations. The second type is related to well-developed actuarial models, which also have the stochastic dominance. The main challenge of generating scenarios is not to accurately predict rates of return for different asset and liability classes, but to construct scenarios for these classes that are consistent with economic, financial and actuarial

theory. Thus, in this section we will shortly describe the main characteristics of developed scenario generator for asset and liability returns.

Investment portfolio selection problem is to allocate among a few aggregated asset classes, such as cash, bonds, and stocks. We describe shortly the ideas to generate returns for each of these three asset classes.

Let money markets be represented by the 1-month nominal interest rate return. We define $r_{1,t}^{s}$ as generated cash returns in state (t,s). Bonds are represented by the total bond market index. The duration of the bond portfolio changes by time passing. To avoid this, new zero-coupon bonds are chosen, so that the duration of bond portfolio is equal to its desired level. The yield curve for bonds with different maturity, based on references [15], [16], is specified in every decision moment for each scenario. These yields are used to discount future coupon and principal payments in order to obtain market values of the bond portfolio. Let define $r_{2,t}^{s}$ is generated return on the bonds portfolio in state (t,s). Stocks are represented by the stock exchange total index. The regime switching approach is used to model log stock returns in excess of the log return on a risk-less asset: returns distribution is assumed to jump between two possible states, referred to as regimes. Let define $r_{3,t}^{s}$ is generated stock returns in state (t,s). In generating 1-month nominal interest rate returns, we employed the *Gaussian* copula function [16], [17] to model the dependencies between real interest rate and inflation rate. Other copula functions [17] are also available for a study. The initial parameters are set with the reference to the Hibbert's et al. work [15].

The liability side of a company is modelled to the reference [12]. The modification was done on modelling the catastrophe model of losses. The catastrophe event driven co-dependence was included by employing the *Gumbel* copula function; which allows strengthening the dependency between non-catastrophe and catastrophe losses. This copula function enables us to model the dependency in the right side of distribution. The parameters were set to the reference [12].

Both scenarios of asset returns and scenarios of liability flows have the scenario fan structure, as it was shown in Figure 4. But in this paper, the investment portfolio selection problem is considered; thus the scenario tree has to be constructed only for asset returns. We employed the multi-stage K-means clustering method [18], [19] to construct the multistage scenario tree for asset returns.

5.2. Decision support using simulation approach

In order to keep the exposition simple and within reasonable size we will mention only some key relationships of the insurance corporate model. One of the fundamental variables is surplus U_t, defined as the difference between the assets value and the liabilities value [12]. The asset side mainly is driven by the investment activity, while the liability side is driven by the underwriting activity. The amount of surplus reflects the financial strength of an insurance company and serves as a measure for a shareholder value.

Change in the surplus is determined by the following cash flows:

$$\Delta U_t = C_t + \Delta I_t - Z_t - E_t,$$

where C_t – earned premiums, ΔI_t – income from assets investment, Z_t – loss paid in period t, E_t – expenses. The flows for each of this component are generated from scenario generator (Section 5.1.).

The decisions on portfolio composition that are under control of insurance company are chosen and are tested along time horizon. The simulation is performed for each decision separately. Then, the surplus is evaluated at the end of time horizon, and the efficient frontier of decisions is constructed.

5.3. Decision support using stochastic optimisation approach

The following formulation is fairly standard in asset liability management applications of stochastic optimisation.

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

$$h_{0,j}^{s} = h_j^0 + p_{0,j}^{s} - q_{0,j}^{s},$$

$$h_{t,j}^{s} = r_{t,j}^{s} h_{t-1,j}^{s} + p_{t,j}^{s} - q_{t,j}^{s},\ t = 1,\ldots,T-1,\ s = 1,\ldots,S,\ j = 1,\ldots,J,$$

where h_j^0 – initial holdings in asset j, $r_{t,j}^{s}$ – return on asset j (random) over period $[t-1,t]$ in scenario s are parameters; and $p_{t,j}^{s}$ – nonnegative purchases of asset j at time t in scenario s, $q_{t,j}^{s}$ – nonnegative sales of asset j at time t in scenario s, $h_{t,j}^{s}$ – holdings in asset j in period $[t,t+1]$ are decision variables.

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

$$\sum_{j\in J}\left(1+k_j^p\right)p_{t,j}^s \le \sum_{j\in J}\left(1-k_j^q\right)h_{t,j}^s + C_t - L_t,\ t=0,\ldots,T-1,\ s=1,\ldots,S,\ j=1,\ldots,J,$$

where $k_j^p \ge 0$ – transaction cost for buying asset j, $k_j^q \ge 0$ – transaction cost for selling asset j, C_t – cash inflows of underwriting business (random) in period $[t-1,t]$, L_t – cash outflows of underwriting business (random) in period $[t-1,t]$ are parameters.

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

$$\underline{b}_j \sum_{j\in J} h_{t,j}^s \le h_{t,j}^s \le \overline{b}_j \sum_{j\in J} h_{t,j}^s,\ t=0,\ldots,T-1,\ s=1,\ldots,S,\ j=1,\ldots,J,$$

where $\sum_{j\in J} h_{t,j}^s$ – total wealth at time t, $\underline{b}_j$ – lower bound for the proportion of $\sum_{j\in J} h_{t,j}^s$ in asset j, $\overline{b}_j$ – upper bound for the proportion of $\sum_{j\in J} h_{t,j}^s$ in asset j are parameters.

Of course, income should be sufficient to cover the liabilities and to earn the gain. To encourage such outcomes, let Γ_T be the target wealth at the horizon $t=T$, $\overline{w}_T^s$ be an excess over target wealth at horizon $t=T$, $\underline{w}_T^s$ be a deficit under target wealth at horizon $t=T$. 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:

$$\sum_{j\in J} r_{T,j}^s h_{T-1,j}^s + C_T - L_T - \overline{w}_T^s + \underline{w}_T^s = \Gamma_T,$$

and the objective function:

$$\min \sum_{s=1}^{S} \pi_s \left[d_1 \cdot \underline{w}_T^s - d_2 \cdot \overline{w}_T^s\right],$$

where π_s – probability of scenario s.

Given optimisation problem is a multi-stage stochastic program with recourse. The flows for the optimisation model are generated from scenario generator. Random parameters $r_{t,j}^s$ are described by the scenario tree, whose nodes are J-dimensional vectors. Its main advantage is that it fulfils the non-anticipatively requirement, i.e. information up to the branch period is shared between a scenario and its parent scenario, only after the branch occurred will there be the duplicate information. Thus, it allows for a reduction of redundancy in the tree, which compresses the size of stochastic information.

To solve such type of programs, it is preferable that the problem would be written in SMPS format [20]. It is the extension of well-known MPS format for deterministic optimisation programs. This is done using three text files: core file, time file, and stochastic file. To create the time and stochastic files, the scripts are written in Matlab environment using data from scenario generator. The core file is written using the below given matrices. To do this, order the number of possible asset classes for an allocating resources in any way, and let J be the number of assets. Define vectors:

$$H_t = \begin{bmatrix} h_{t,1} \\ \vdots \\ h_{t,J} \end{bmatrix},\ \widetilde{H} = \begin{bmatrix} h_1^0 \\ \vdots \\ h_J^0 \end{bmatrix},\ P_t = \begin{bmatrix} p_{t,1} \\ \vdots \\ p_{t,J} \end{bmatrix},\ Q_t = \begin{bmatrix} q_{t,1} \\ \vdots \\ q_{t,J} \end{bmatrix},\ R_t = \begin{bmatrix} r_{t,1} \\ \vdots \\ r_{t,J} \end{bmatrix},\ K^p = \begin{bmatrix} 1+k_1^p \\ \vdots \\ 1+k_J^p \end{bmatrix},\ K^q = \begin{bmatrix} 1-k_1^q \\ \vdots \\ 1-k_J^q \end{bmatrix},$$

$x_t = \left[H_t \quad P_t \quad Q_t \quad \overline{w}_t \quad \underline{w}_t\right]'$, $\overline{B} = diag\left(\overline{b}_1,\ldots,\overline{b}_J\right)$, $\underline{B} = diag\left(\underline{b}_1,\ldots,\underline{b}_J\right)$, $\widetilde{I} = 1_{J\times J}$ (all elements are equal to one). Then, we may express the given problem in matrix notation. For $t=0,\ldots,T$, define

$$A_t = \begin{bmatrix} I_{J\times J} & -I_{J\times J} & I_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ 0_{1\times J} & (K^p)' & -(K^q)' & 0 & 0 \\ I_{J\times J} - \overline{B}\widetilde{I} & 0_{J\times J} & 0_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ \underline{B}\widetilde{I} - I_{J\times J} & 0_{J\times J} & 0_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ 0_{1\times J} & 0_{1\times J} & 0_{1\times J} & -\delta_{tT} & \delta_{tT} \end{bmatrix},\ T_t = \begin{bmatrix} -I_{J\times J} R_t(1-\kappa_t) & 0_{J\times J} & 0_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ 0_{1\times J} & 0_{1\times J} & 0_{1\times J} & 0 & 0 \\ 0_{J\times J} & 0_{J\times J} & 0_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ 0_{J\times J} & 0_{J\times J} & 0_{J\times J} & 0_{J\times 1} & 0_{J\times 1} \\ \delta_{tT}(R_t)' & 0_{1\times J} & 0_{1\times J} & 0 & 0 \end{bmatrix},$$

$$b_t = \begin{bmatrix} \kappa_t \widetilde{H}_0 \\ C_t - L_t \\ 0_{J\times 1} \\ 0_{J\times 1} \\ \delta_{tT}\left(\Gamma_t - C_t + L_t\right) \end{bmatrix}, \qquad c_t = \begin{bmatrix} 0_{1\times J} & 0_{1\times J} & 0_{1\times J} & -d_2\delta_{tT} & d_1\delta_{tT} \end{bmatrix},$$

$$\kappa_t = \begin{cases} 1, & \text{if } t = 0, \\ 0, & otherwise. \end{cases}, \quad \delta_{tT} = \begin{cases} 1, & \text{if } t = T, \\ 0, & otherwise. \end{cases}$$

In the first stage, the initial decision x_0 has to be chosen from the set $\left\{x_0 \in \mathbf{R}^{3J+2} : A_0 x_0 \propto b_0\right\}$ at a direct cost $c_0 x_0$. The notation $\propto$ denotes the equality or inequality respectively Depending on the decision x_0 taken at present and the realizations $\left\{\xi_\tau\right\}_{\tau=1}^{\widetilde{T}}$ that would be available in the future; there would be indirect costs due to the recourse actions. If the realization ζ_1 is observed, then the recourse decision x_1 is chosen from the set $\left\{x_1 \in \mathbf{R}^{3J+2} : A_1 x_1 \propto b_1 - T_1\left(\xi_1\right)x_0\right\}$ at a direct cost $c_1 x_1$. Such logic of finding decisions is applied to all stages until the end of time horizon is reached.

5.4. Numerical application

Settings for a numerical experiment are as follows. Scenario fan for each of the stochastic element consists of 1000 scenarios, which are generated with 1-month step during 10 years time horizon. The first stage is the initial time moment $t = 0$, and the recourse stages are $t = \left(1, 3, 6, 10\right)$ in years. It determines that we have 5 stages during 10 years time horizon. We will examine the performance of different asset mixes by different combinations of the following weights applied to the remaining portfolio. The lower bound for bonds is usually statutory restrictions; thus stocks and cash investments are chosen so that the total weights in the remaining portfolio sum up to 100 %. For our experiment, we set the lower bound for bonds equal to 40 percent of total assets.

For a stochastic simulation approach, the set of alternative decisions are formed: nine different portfolio compositions on asset structure are given in Table 1. The decision vector represents the weights on cash, bonds, and stocks respectively. We evaluated the surplus at the end of time horizon for two cases: (a) first stage decisions are tested using usual simulation, and (b) first and recourse stages decisions are tested using rolling horizon simulations, allowing the rebalancing of the portfolio at recourse stages. The surplus reward is presented by the mean, and the surplus risk is presented by the standard deviation. The results are given in Table 1. The efficient frontiers of strategies are depicted in Figure 6 – Figure 7.

Table 1. The results of performance of stochastic simulation approach

Case No.	1	2	3	4	5	6	7	8	9
Investment weights	(0 0.4 0.6)	(0 0.6 0.4)	(0 0.8 0.2)	(0.1 0.4 0.5)	(0.1 0.6 0.3)	(0.1 0.8 0.1)	(0.2 0.4 0.4)	(0.2 0.6 0.2)	(0.2 0.8 0)
No rebalancing									
Surplus Reward, 10^5	3.8278	3.8437	3.6606	3.9884	3.9538	3.7171	4.1536	4.0669	3.7752
Surplus Risk, 10^5	7.4851	7.4855	7.0028	7.8474	7.7270	7.1235	8.2097	7.9686	7.2443
With rebalancing									
Surplus Reward, 10^5	4.5143	4.5369	4.5595	4.5042	4.5268	4.5494	4.4941	4.5167	4.5393
Surplus Risk, 10^5	8.4533	8.4989	8.5460	8.4254	8.4716	8.5193	8.3981	8.4448	8.4932

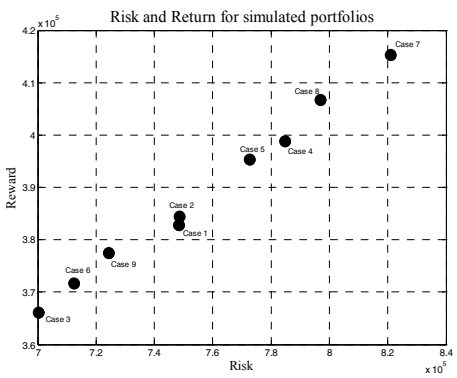

Fig. 6. *Efficient frontier of strategies (no rebalancing)*

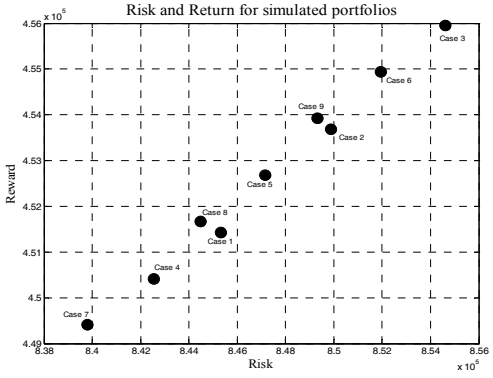

Fig. 7. *Efficient frontier of strategies (with rebalancing)*

The obtained results show that portfolio rebalancing allows achieving more surpluses at the end of time horizon. Efficient frontier helps to choose a decision according to the risk tolerance. If preferable risk is rather low, non-rebalanced portfolio generates higher returns if resources are invested in bonds (Case 3, Case 6, and Case 9). It is because that bonds portfolio duration is equal to 5 years and generates higher yields than stocks at the time horizon. If the investment portfolio is rebalanced at each stage, the investment in stocks generates higher revenue (Case 1, Case 7, and Case 7).

Now let us return to the stochastic optimisation approach (Section 5.3.). The transaction costs for purchasing and selling the assets are ignored. The investments in asset classes are bounded with lower limit and upper limit as follows: cash=(0, 0.2), bonds=(0.4, 0.9), and stocks=(0.3, 0.6). According to the settings for a numerical experiment, the non-zero pattern of constraints is depicted in Figure 8. The scenario tree with five stages and with two scenarios per each node is generated; it is used as an input for a considered problem. The multi-stage stochastic program was solved using SLP_IOR solver, developed by P.Kall and J.Mayer (University of Zurich, Switzerland). The given problem is imported in SMPS standard. The obtained optimal value of objective function is equal to $-4.70642 \cdot 10^5$, i.e. the surplus over target wealth $1 \cdot 10^5$. The distribution of objective function is depicted in Figure 9.

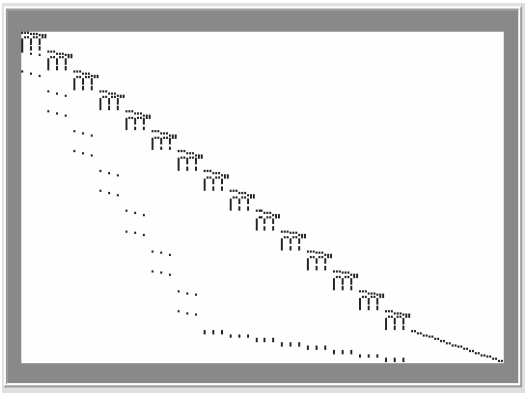

Fig. 8. *Non-zero pattern of constraints*

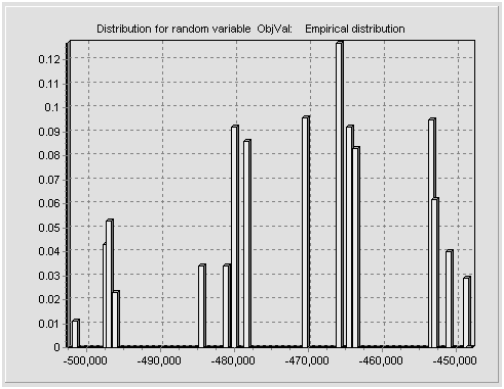

Fig. 9. *Distribution of objective function*

In the first, second, and the third stages the optimal portfolio composition is (0, 0.4, 0.6) in cash, bonds, and stocks respectively. In the fourth stage the optimal portfolio composition is (0, 0.7, 0.3), i.e. it is recommended to invest more in bonds, because of low stock returns.

6. Conclusions

In this paper, the performance of two alternative portfolio models has been compared. The portfolio selection problem considers dynamic decision-making under uncertainty. The results show that multistage stochastic programming (optimisation) dominates the stochastic simulation approach with the efficient frontier concept, but the degree of domination is not very high. The reason for this is that a modified input to the simulation approach is generated in every stage, and the decision model is tested on alternative strategies. It is a drawback of this approach, because it is very time consuming. The multistage stochastic optimisation allows choosing optimal decisions, which means that the objective will be achieved in optimally way. But this field is still missing good solvers, and at this moment it is a very intensive research area.

References

1. Dempster, M.A.H., Germano, M., Medova, E.A., Villaverde, M. Global asset liability management, *British Actuarial Journal,* Vol. 9(1), 2003, pp. 137-195.
2. Ziemba, W.T. *The Stochastic Programming Approach to Asset, Liability, and Wealth Management.* USA: The Research Foundation of the Association for Investment Management and Research, 2003. 192 p.
3. Collomb, A. *Dynamic asset allocation by stochastic programming methods.* Thesis of PhD, 2004, 117 p. – http://www.stanford.edu/group/SOL/dissertations/collombthesis.pdf
4. Carino, D.R., Myers, D.H., Ziemba, W.T. Concepts, technical issues, and uses of the Russell-Yasuda Kasai Financial Planning Model, *Operations Research,* Vol. 46(4), 1998, pp. 450-462.
5. Fleten, S.E., Hoyland, K., Wallace, S.W. The performance of stochastic dynamic and fixed mix portfolio models, *European Journal of Operational Research,* Vol. 140, 2002, pp. 37-49.
6. Halling, M., Popa, C., Randl, O. *Stochastic Optimization, Tree Structures and Portfolio Choice, Working Paper.* 2005, 24 p. – http://www.univie.ac.at/finance/download/research/pap2005_01.pdf

7. Domenica, N.D., Birbilis, G., Mitra, G., Valente, P. Stochastic programming and scenario generation within a simulation framework: an information systems perspective, *Decision Support Systems,* Vol. 42(4), 2007, pp. 2197-2218.
8. Dupacova, J., Consigli, G., Wallace, S.W. Scenarios for multistage stochastic programs, *Annals of Operations Research,* Vol. 100, 2000, pp. 25-53.
9. Yu, L.Y., Ji, X.D., Wang, S.Y. Stochastic programming models in financial optimization: survey, *AMO – Advanced Modeling and Optimization*, Vol. 5(1), 2003, pp. 1-26.
10. Mulvey, J.M. Multi-period Stochastic optimization models for long-term investors, *Quantitative Analysis in Financial markets*, Vol. 3, 2001, pp. 66-85.
11. Murty, K.G. *Optimization models for decision making: Self-Teaching Web-book, Ch. 1*. USA, 2003. – http://ioe.engin.umich.edu/people/fac/books/murty/opti_model/
12. Kaufmann, R., Gadmer, A., Klett, R. Introduction to Dynamic Financial Analysis, *ASTIN Bulletin International Actuarial Association – Brussels*, Vol. 31, 2001, pp. 213-250.
13. Dupačová, J. Stochastic programming: approximation via scenarios, *Aportaciones Mathematicas, Ser. Communicationes*, Vol. 24, 1998, pp. 77-94.
14. Mitra, S. *Scenario generation for stochastic programming. White paper. Opt-risk Systems*. UK, 2006. 34 p.
15. Hibbert, J., Mowbray, P., Turnbull, C. *A stochastic asset model & calibration for long-term financial planning purpose: Technical Report.* UK: Barrie&Hibbert Limited, 2001. 76 p.
16. Pranevicius, H., Sutiene, K. Simulation of dependence between assets returns in insurance. In: *Proceedings of International Conference on Operational Research: Simulation and Optimization in Business and Industry*. Tallinn, 2006, pp. 23-28.
17. Embrechts, P., McNeil, A., Straumann, D. Correlation and Dependency in Risk Management: Properties and Pitfalls. In: *Risk management: value at risk and beyond / M. A. H. Dempster (Ed.).* UK: Cambridge University Press, 2002, pp. 176-224.
18. Pranevicius, H., Sutiene, K. Scenario tree generation by clustering the simulated data paths. In: *Proceedings of the 21st European Conference on Modelling and Simulation.* Prague, 2007, pp. 203-208.
19. Pranevicius, H., Sutiene, K. Copula effect on scenario tree, *IAENG International Journal of Applied Mathematics*, Vol. 37 (2), 2007, pp. 112-126.
20. Gassmann, H.I., Kristjansson, B. The SMPS format explained. In: *IMA Journal of Management Mathematics.* Oxford, 2007, pp.1-31.

AN APPROACH FOR DECISION SUPPORT SYSTEM DESIGN FOR EVALUATION OF WATER CONTAMINATION PROCESSES

Dale Dzemydiene, Salius Maskeliunas

Institute of Mathematics and Informatics
Akademijos str. 4, Vilnius, LT-08303, Lithuania
E-mail: daledz@ktl.mii.lt, mask@ktl.mii.lt

The activities of large enterprises, institutions, and organizations should be based on versatile responsibility of enterprises and stimulation of efficiency, paying ever more attention to the requirements of sustainable development and to the issues of environment protection: strategic and tactical planning and control, estimation of economic-social balance, application of information technologies, and constant check of systems, as well as to legal regulation effect. The information representation methods play an important role in solving decision-making problems for the development of the consultative systems in environment contamination (especially water resources) evaluation processes. The main components of decision support system development in environment evaluation sector are analysed by using E-nets modelling techniques in this article. The models of reviewing the applications and decision-making are designed by three levels of detailing and using imitation modelling. The processes of contamination are monitoring and represented by data warehouse structures and proposed for users.

Keywords: *web services, sustainable development, environment contamination, decision support system, information representation, E-nets*

1. Introduction

The economic growth of a state does not ensure the necessary and sufficient conditions for social welfare and employment; the level of employment is apparently insufficient as well as the quality of working places, and structurally unfavourable unemployment.

The principles of environment protection consist of multiple components and make up a totality of requirements that can be presented as standards of enterprise functioning, requirements of healthy human environment, permission for functioning, taxes for cause pollution, etc. [1], [2], [3], [14]. These principles give rise to very important problem of legal regulation, the significance of which is analysed in this research by relation to the legal system. The proper selection of novel work organization methods, knowledge management systems, modern information-communication technologies, and up-to-date methods of their control as well as the skills of their mastering allow us to realize the sustainable development problems of organizations more efficiently.

The goals of Baltic countries (the states situated near the Baltic Sea) development follows up the goals of Agenda 21 [2], in which the most important features and requirements of sustainable development are summarized: to establish conditions for all inhabitants in order to set up accommodation; to improve administration of populated localities; to support sustainable planning and management of earth usage; to take care of integrated supply of infrastructure for environment, for example, water-supply, engineering equipment, sewage, waste collection; to develop sustainable energy and transport systems inside inhabited localities; to ensure planning and management of places for living in vulnerable territories; to promote sustainable construction industry; to create healthy environment in the city.

This research study is aimed to analyse the activities of enterprises, institutions, and organizations according to some components of sustainable requirements dealing with the ecological sustainability, cleaner production manufacturing, and economic growth [5-7]. For these purposes the described decision support system helps in analysing of the development processes of enterprises.

The paper propose that the means should be based on versatile responsibility of enterprises and stimulation of efficiency, paying ever more attention to the requirements of sustainable development and to the issues of environment protection: strategic and tactical planning and control, estimation of economic-social balance, application of information technologies and constant check systems, as well as to legal regulation effect. Novel work organization methods, knowledge management systems, modern information-communication technologies are of great significance in supporting sustainable development management. The proper selection of information technologies and up-to-date methods of their control as well as the skills of their mastering allow us to realize the sustainable development problems of organizations more efficiently and to organize the interstate of inter-departmental and interregional cooperation in a new way. The purpose of this work and problems solved are based on services offered by information systems while estimating proposals in what ways to simulate situations, to make and intellectualised environment pollution estimation by an object. For the improvement of information system structure and services research work is pursued. This caries over to disagreement on what the content of our research should be.

2. Components of Decision-Making System for Evaluation Contamination Processes

Decision-making aimed at the evaluation of the pollution processes of an enterprise deals with: complexity of structures of processes; multiple subsystems with complex mechanism interacting as internal or external parts;

time and space/geographical dependencies; great volume of data acquired from the processes; multi-criteria decision-making; causal, temporal relationships and interaction of processes; complexity of legal information.

The principles of sustainable development consist of a lot of components and make up a totality of requirements [4], [7]. Such requirements can be presented as standards, permission for functioning, taxes for caused pollution, etc. These principles give rise to very important problems of legal regulation, the significance of which and relation to the legal system are topical and may be properly implemented.

Continually we have to evaluate the situation according to the sustainability in the given region. For this purposes we deal with the decision support integrating principle components of decision support system as presented in schema (Fig. 1).

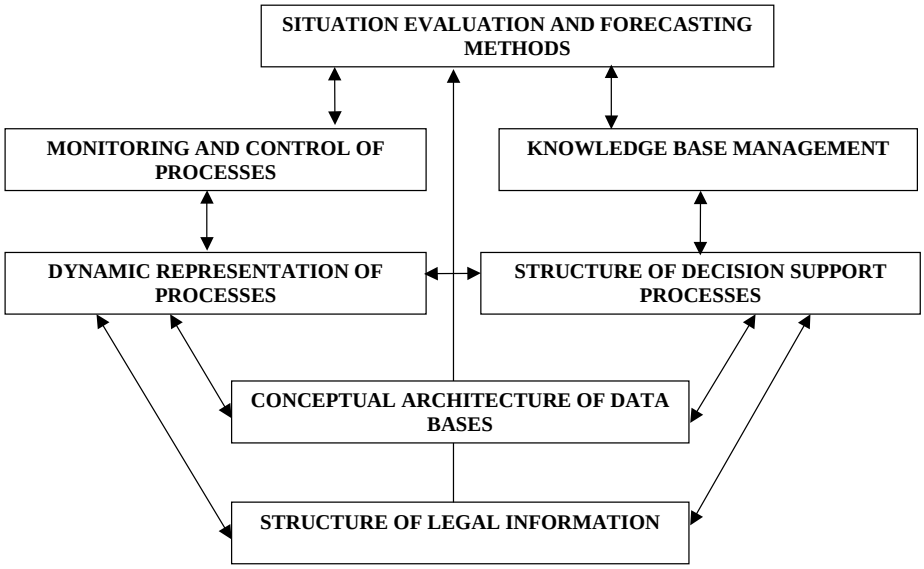

Fig. 1. *Components of decision support for evaluation of environment contamination processes*

Every of components need more precise description [6], [8], [9]. If we analyse the developing of information infrastructure in the field of environment protection, it covers the following:

- the structures of the main laws enforcing the environmental protection;
- the forms of sources of legal information and their impact in the formation of environmental protection information systems;
- the representation of knowledge content of environment application domain.

A key part of law enforcement incorporates understanding of those activities, through the development and use of methods, models for collecting and further on interpreting the large volume of data available in real time.

A distinctive feature of rapidly changing systems is the dynamism property related with a state changing in time and space measurements. The environment of this type is usually characterized by the space of states changing, where each state is steady in time for a short period. The complexity of structures of processes, multiple subsystems with their own complex mechanism interacting as internal or external parts, time and space/geographical dependencies, a great volume of data acquired from the processes, and multiple-criteria decision-making are essential features for the analysis and representation of such an application domain.

3. The Formal Description of Water Contamination Processes

The main purpose of the advisory system is to assist in environment protection control processes, creating of suspect profiles by giving computer-aided instructions, planning and situation recognition techniques. The use of advisory systems may also improve legal training, for example, providing environment protection agencies with advice on the type of information required by inspection to reduce the pollution activities. As an example, we consider the activities of enterprises, firms, and organizations, i.e. the main stationary objects, to estimate pollution of water bodies. In line with the object functioning nature the project reflects information on the activities pursued while the license defines the limit in which environment pollution is allowed, i.e. limits of pollutants cast are drawn. Besides, the objects must give reports according to their activities pursued and in line with statistical account ability forms.

The level of representation of dynamical aspects shows the dynamics of observable processes. The multiple objective decision-making deals with the analysis of information obtained from the static sub-model

taking into account all possible measurement points revealed in dynamic sub-model of such a system. Further actions, operations, etc. are determined through the mechanism of cooperation of agents that are working by using the temporal information registration window.

A dynamically changing environment imposes time constraints. Many problems are to be solved simultaneously [10], [12]. The values of the observed parameters may change dynamically, depending on time and the events occurring. Solution of different problems is interfered with one another. For instance, the high concentration of harmful material thrown out into the air is related with the risk factors referring prevention of links that are of biological significance and time-dependent, etc. Another essential aspect of such an application domain is its spatial dimension. While in many other application domains the problems of study are within a very precise and, usually, narrow frameworks. For instance, the contamination problem of an enterprise (e.g. manufactory, firm, and plant) deals with spatially varying phenomena of unbounded limits.

The complexity of environment research problems consists in the complexity of criteria and differences of attitudes.

The knowledge representation framework supports organizational principles of information in a static semantic model. The model of behavioural analysis of the target system shows the dynamics of observable processes. One of its characteristics is a need for a lot of data to properly model and verify these problems.

The Evaluation nets (i.e., E-nets are the extension of Petri nets) are introduced by [13]. The structure and behavioural logic of E-nets give new features in conceptual modelling and imitation of domain processes and decision-making processes. Apart from time evaluation property, E-nets have a much more complex mechanism for description of transition work, some types of the basic transition structures, a detailing of various operations with token parameters. In addition to Petri nets, two different types of locations are introduced (peripheral and resolution locations). The exceptional feature is the fact that the E-net transition can represent a sequence of smaller operations with transition parameters connected with the processes.

It is possible to consider the E-net as a relation on (E,M_0,Ξ,Q,Ψ), where E is a connected set of locations over a set of permissible transition schemes, E is denoted by a four-tupple $E=(L,P,R,A)$, where L is a set of locations, P is the set of peripheral locations, R is a set of resolution locations, A is a finite, non-empty set of transition declarations; M_o is an initial marking of a net by tokens; $\Xi=\{\xi j\}$ is a set of token parameters; Q is a set of transition procedures; Ψ is a set of procedures of resolution locations.

The E-net transition is denoted in [13] as $a_i=(s_i, t(a_i),q_i)$, where s_i is a transition scheme, $t(a_i)$ is a transition time and q_i is a transition procedure. In order to represent the dynamic aspects of complex processes and their control in changing environment it is impossible to restrict ourselves on the using only one temporal parameter $t(a_i)$ which describes the delaying of the activity, i.e. the duration of transition.

A concrete parameter of token obtains a concrete value according to its identification, when the token is introduced into the location $b_j(\xi_k)$. Such a combination of locations with the tokens in them, the parameters of which obtain concrete values, describes a situation for process execution.

Such an understanding of the transition procedure enables us to introduce the time aspects into procedure of control of processes and determine operations with token parameters in time dimension. The exceptional feature is the fact that the E-net transition can represent a sequence of smaller operations with transition parameters connected with the event/process. Operations are described in the transition procedure with these parameters.

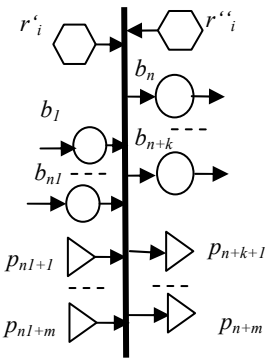

Fig. 2. *The common transition schema of E-net representation*

The E-nets support a top down design in graphical representation manner. The hierarchical construction of dynamic model is simplified by representing macro-transition and macro-location constructions. The input locations L_i' of the transition correspond to the pre-conditions of the activity (represented by the transition on Fig. 2). The output locations L_i'' correspond to post-conditions of the activity. The complex rules of transition firing are specified in the procedures of resolution locations Ψ and express the rules of process determination.

More in details we consider the example of the analysis of water resources and the pollution of sewage of the enterprise. The pollutants from the production are entering into the water in some types of cases. Such cases we find out by the construction of E-net distribution processes of sewage in the enterprise (Fig. 3).

The harmful materials that are represented by peripheral locations of the E-net (on Fig. 3) are very important for evaluation of water pollution of an enterprise:

$p_{1,1},\ldots,p_{1n}$ are materials included in water efflux;
$p_{2,1},\ldots,p_{2,n}$ are waste materials from the primary sewage purification plant;
$p_{3,1},\ldots,p_{3,n}$ are waste materials from the common sewage purification plant;
$p_{4,1},\ldots,p_{4,n}$ are materials entering into open reservoirs, that are not detained in the sewerage system of the enterprise;
$p_{5,1},\ldots,p_{5,n}$ are materials entering into open reservoirs, if there is no rainwater collection system;
$p_{6,1},\ldots,p_{6,n}$ are utilized wastes from primary purification plants;
$p_{7,1},\ldots,p_{7,n}$ are materials entering into rain water if they are stored openly in the territory of enterprise;
$p_{8,1},\ldots,p_{8,n}$ are materials entering into the external reservoir from the primary sewage purification plant.

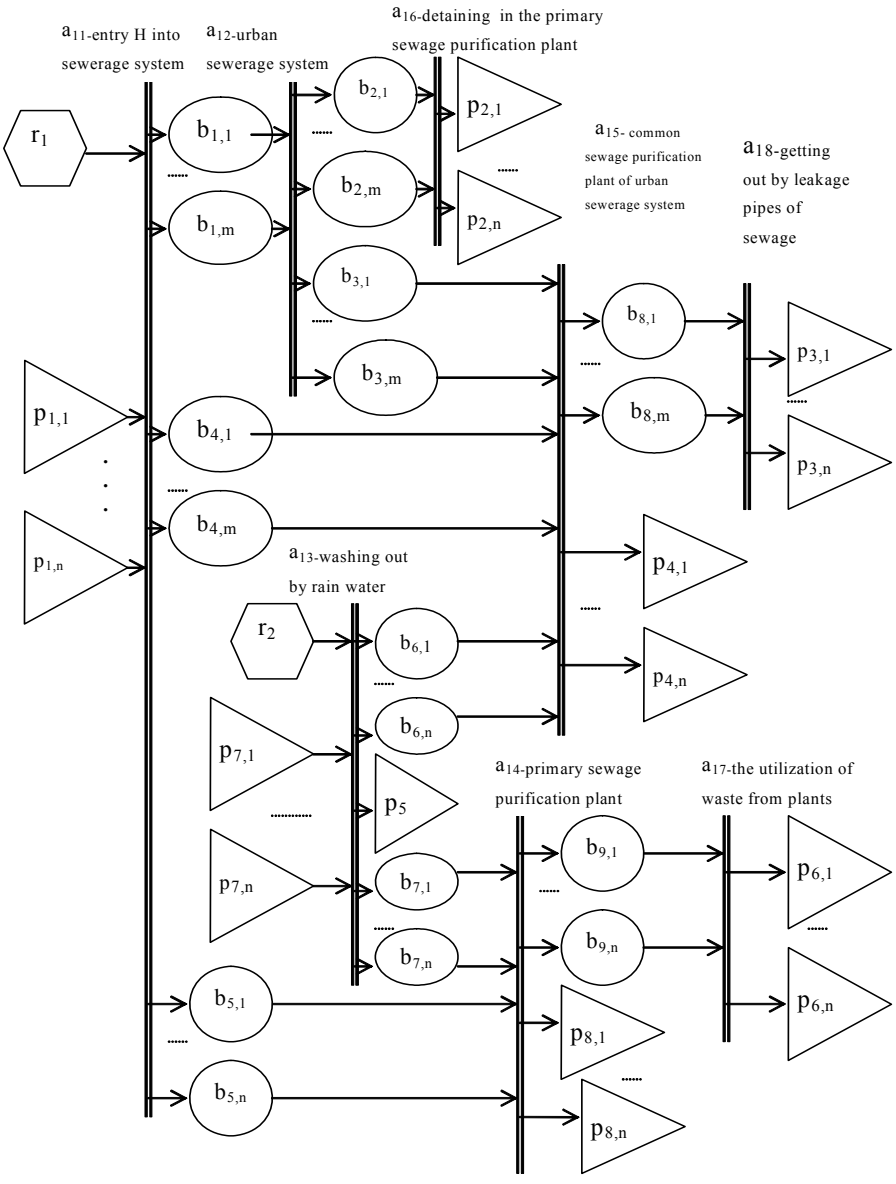

Fig. 3. *The E-net of distribution processes of harmful materials in the water of enterprise*

The E-net structure, which describes the decision-making process, gives visually the parameters needed for control and the control structure relation with tasks and decisions.

A real situation is the result of system development activities (history) and, in order to evaluate the situation in a decision support system, a retrospective analysis of the activities is made.

In fact decisions bear some risk elements, and, naturally, the goal is posed to decrease the error probability in a decision made by ensuring the information completeness and reliability. First of all, the problem and its

structure are identified in the decision-making process and a survey of the necessary data performed. Later on, other possible models for solving the problem are analysed. Specific operation performing variants are correlated with these models. In the next stage, an optimal plan is selected out of a set of possible alternative decisions.

Real-time subsystem is embedded in the target system as a concurrent computing system related with the monitoring of data. The monitoring subsystem connected with expert subsystem must detect the faults of process performance. The time for obtaining a solution is often strictly limited. These conditions impose strict deadlines on the obtaining a decision and maintaining the functioning correctness.

4. Complexity of Evaluation of Environmental Pollution Processes

At the stage of analysis and evaluation of the enterprise performance, the use of this meta-model could allow as follows:

- to recognize what changes in the environment may induce changes in decision goals;
- to decide if the situation is relevant for the ready application of the existing rules or not;
- to specify the process of identification of possible courses of actions and alternatives and to control the choice of concrete variant of these actions by evaluating attractiveness of the consequences of each action.

The multiple objective decision deals with the analysis of information obtained from the static sub-model taking into account all possible measurement points revealed in dynamic sub-model of such a system. The task structure relationship with information elements, the course of decision-making processes and presentation of alternative variants of decisions are represented in sub-models.

An important aspect of enclosed oceanic basins is the limited rate of exchange (renewal) of water that lies within them with the adjacent ocean. The Baltic Sea outflow and water exchange is restricted through the contorted Skaggerak and the passages around the islands that are a part of the country of Denmark. The new Oresund Bridge between Denmark and Sweden may have further restricted the flow of water here. In either case, these restricted passages for the exchange of water mean that nutrients and other forms of pollution are not purged rapidly from these bodies of water. Increasing concentrations of nutrients foster larger and more prolonged phytoplankton blooms; other forms of pollution may enter the ecosystem and affect it detrimentally.

Development of search systems for law acts and standard document bases is one of the preconditions in creating as good as possible legal information usage conditions at the Internet and Intranets, we can expect that law acts are properly systematized and perfected, realization of law acts speeds up, and law-making contradictions are avoided. Comprehension of information is associated with computer application in information activities, new kinds of hardware, modern information processing, storage, and transmission technologies. Information becomes an economic fund of knowledge. It is the product of intellectual activities of most skilled and creative population. Information and knowledge accumulated help in saving sources of social, material, and intellectual work.

The information of legal environment protection is structured according to the content of distinguished rubrics that covers all fields in environmental areas, such as: natural sources, territory planning, cadastres and registers, etc. The environment protection field is structured in particular themes according to the main regulation areas. This structure influences the hierarchy of legal documents. The percentage evaluation of accepted legal acts related with economic activities in the environmental protection rubrics in Lithuania is presented in [11].

The analysis revealed the disturbances of legal acts according to the different environment protection areas based on the data stored in data warehouses and provided by Inspection of Environment Protected Agencies in Lithuania (Fig. 4).

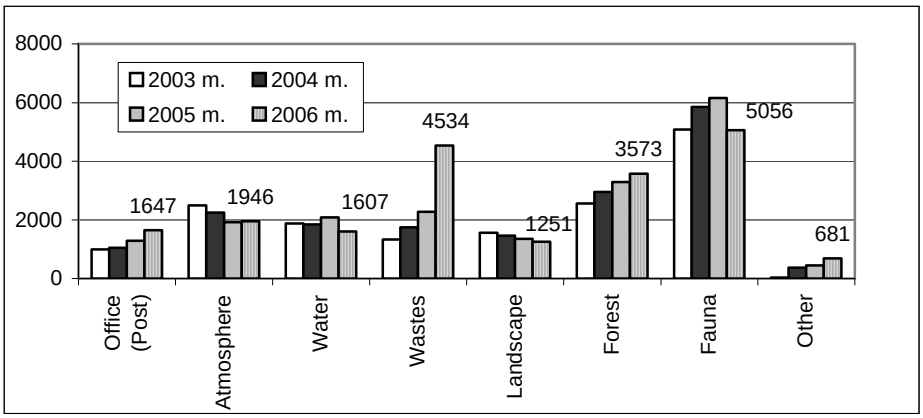

Fig. 4. *The results of statistical analysis of legal act's disturbances in the environment protection domain according to the data of Inspection of Environment Protection in Lithuania*

According to the Report on the 6th Baltic Sea NGO Forum, 5-7 October 2006, Stockholm, Sweden, with Focus on the Radioactive Pollution of the Baltic Sea Region the attention was attracted to the following – that "the Baltic Sea is the most radioactive sea in the world. The Chernobyl accident, atmospheric nuclear bomb tests and Sellafield's releases are the main contributing factors. We cannot influence these historical contributions, done is done, but we can influence the contributions of current nuclear activities in the region. At the top of our agenda at present is the question of nuclear waste management and storage. Both Sweden and Finland are planning final repositories for their spent nuclear fuel – and plan to locate these final repositories at the shore of or below the Baltic Sea…".

A dry waste storage facility has been built – without a building permit and without an EIA (environmental impact assessment) only 90 meters from the Gulf of Finland. Uranium tailings are being imported through the Baltic to St. Petersburg from Germany and France – 9740 tones from Germany alone in 1996-2001, about 6 thousand tones in 2001-2005. The final waste stays in Russia; it is a form of importing radioactive wastes, etc. MORS is the division of HELCOM that deals with monitoring of radioactivity and radiation protection. HELCOM-MORS has several monitoring programs: discharges from sources to the sea, environmental levels of radioactivity, indicator levels, a sediment baseline study (spatial distribution of nuclides). The unit conducts recurrent validations of measuring stations and produces thematic reports. The sediment baseline project is currently focused on improving inventory estimates.

Consequently, intelligence programs have been developed, leading to recognition of a field of activity called contamination analysis, which has been described as the identification of and the provision of insight into the relationship between environment data and other potentially relevant data with the view to specialist experts.

A key part of this approach enforcement is to understand those activities, through the development and use of methods, models and tools for collecting and then interpreting the large volume of data available in real time for environment protection investigation. Some issues for qualitative information representation including legal information and statistical analysis are considered.

5. Conclusions

The consideration of real time process control and enterprise functioning is organized by the integration sustainable development requirements for retrospective analysis of contamination processes. The components of decision support information infrastructure are described for aims of assistance in contamination evaluation. The level of representation of dynamical aspects shows the dynamics of observable processes. The multiple objective decision-making level deals with the analysis of information obtained from the static sub-model taking into account all possible measurement points revealed in temporal window of dynamically changed information.

It is very important that the means of enterprise development should be based on versatile responsibility of enterprises and stimulation of efficiency, paying ever more attention to the requirements of sustainable development and to the issues of environment protection: strategic and tactical planning and control, estimation of economic- social balance, application of information technologies and constant check systems, as well as to legal regulation effect.

References

1. Analysis of Summary of Results of Environment Protection Control. 2006. Data of State Environment Protection Inspection – http://vaai.am.lt/VI/index.php#a/458
2. Baltic 21, 2003, Report 2000-2002: Towards Sustainable Development in the Baltic Sea Region. Baltic 21. Series No. 1. Poland, Drukarnia. MISIURO.
3. Burinskiene, M., Dzemydiene, D. Rudzkiene, V. An Approach for Recognition of Significant Factors for Sustainable Development Strategies. In: *Proc. of Intern. Conference "Modelling and Simulation of Business Systems" / H. Pranevicius, E. Zavadskas, B. Rapp (Eds.),* KTU Technology, 2003, pp. 90-96.
4. Crespo, A., Botti, V., Barber, F., et al. A Temporal Blackboard for Real-time Process Control. In: *Engineering Applications of Artificial Intelligence / L. Motus (Ed.).* Vol.7. No 3, 1994, pp. 255-266.
5. Dzemydiene, D. Temporal Information Management and Decision Support for Predictive Control of Environment Contamination Processes. In: *Advances in Databases and Information Systems: Proc. of Fifth East-European Conference / J. Eder, A.Caplinskas (Eds.).* Vilnius, 2001, pp. 158-172.
6. Dzemydiene, D. An Approach to Modelling Expertise in the Environment Pollution Evaluation System. In: Databases and Information Systems / J. Barzdins, A. Caplinskas (Eds.). Dordrecht/Boston/London: Kluwer Academic Publishers. 2001, pp. 209-220.
7. Dzemydiene, D., Pranevichius, H. *Description of a dynamically changing environment in a decision support system: Proceedings of the Baltic Workshop on National Infrastructure Databases: Problems, Methods, Experiences, Vol. 2.* Vilnius, 1994, pp. 102-111.

8. Dzemydiene, D., Naujikiene, R. Structural Analysis of Legal Environment Protection Information in the Estimation of Pollution Factors. In: *Proc. of International Conference "Modelling and Simulation of Business Systems"/ H. Pranevicius, E. Zavadskas, and B. Rapp (Eds.).* KUT Technologija Press, 2003, pp. 324- 328.
9. Dzemydiene, D., Jakobsen, K., Maskeliunas, S. Intelligent Decision Support for Water Resource Management via Web Services. In: *Proceedings of Selected papers of 4 th. International Conference "Citizens and Governance for Sustainable Development" CIGSUD'2006 / W. Leal Filho, D. Dzemydiene, L. Sakalauskas, E.K. Zavadskas (Eds.).* Vilnius. Technika, 2006, pp. 184-189.
10. Environment 2010: Our future, our choice. The Sixth EU Environment Action Programmer 2001-2010. Luxembourg. 2001. (http://www.environment.com).
11. Legal information system LITLE – http://www.litlex.lt/portal/ml/start.asp?lang=eng. 2006.
12. *Future Cities: Dynamics and Sustainability / F. Moavenzadeh, K. Hanaki, P. Baccini (Eds.). Series: Alliance for Global Sustainability Book Series, Vol.1*. 2002. 248 p.
13. Noe, J.D. and Nutt, G.J. Macro E-nets for representation of parallel systems, *IEEE Transactions on Computers*, C-22(5), 1973, pp. 718-727.
14. Swanson, D.A., Pintér, L., Bregha, F., Volkery, A., Jacob, K. *National Strategies for Sustainable Development: Challenges, Approaches and Innovations in Strategic and Co-ordination Actions*. IISD, 2004.

REINFORCEMENT LEARNING WITH FUNCTION APPROXIMATION: SURVEY AND PRACTICE EXPERIENCE

Yuriy Chizhov

Department of Modelling and Simulation
Riga Technical University
Kalku 1, Riga, LV-1658, Latvia
Phone: +371-26499192. E-mail: jurij.ch@gmail.com

Classical Reinforcement Learning with tabular value function form is unable to cope successfully with real world tasks which suppose continuous or large space of states and actions. Value Function Approximation and Policy Gradient allow solving the mentioned problem. In most papers the methods are described theoretically, but suffer from the lack of details of practical part. The aim of this article is to make an overview of mentioned methods and meet a lack. For this purpose some aspects of the implementing a couple of algorithms related to Value Function Approximation are shown: Tile Coding and Gradient Descent with Back-propagation Artificial Neural Network. The Mountain Car task is used to demonstrate results of experiments of Tile Coding.

Keywords: *Reinforcement Learning, Value Function Approximation, Gradient Policy, Tile Coding, Neural Network*

1. Introduction

Simple and effective idea for intelligence agents is advised by Reinforcement learning (RL) for automated exploration of unknown environment and goal achieve. As many other AI algorithms, RL should be exceedingly upgraded to cope with real world tasks. To work with continuous or large spaces of states two basic approaches were suggested: Value Function Approximation and Gradient Policy. Both algorithms are broadly investigated, but some practical details are not clear yet, for example, the influence of tiling size in Tile Coding. In one's turn, the adaptation of Neural Networks to Reinforcement Learning is not trivial task due to requirements of the problem, its properties, selecting of activation function for each hidden layer and so on. The complexity is indorsed by the words of Richard Sutton (the expert and researcher in reinforcement learning): "It is a common error to use a back-propagation neural network as the function approximator in one's first experiments with reinforcement learning, which almost always leads to an unsatisfying failure. The primary reason for the failure is that back-propagation is fairly tricky to use effectively, doubly so in an online application like reinforcement learning" [1]. Nevertheless the method was successfully applied in a series of individual works.

The paper provides a survey of value function approximation methods and describes a few technical details gained from self experience.

2. Reinforcement Learning

Reinforcement Learning is defined as the problem of an agent that learns to perform a task through trial and error interaction with an unknown environment which provides feedback in terms of numerical reward [2]. The agent and the environment interact continually (see Fig. 1) within discrete time.

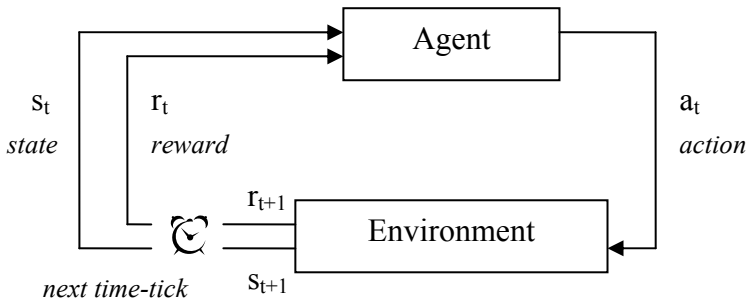

Fig. 1*. The agent-environment interaction in reinforcement learning*

At time t the agent senses the environment to be in state s_t; based on its current sensory input s_t the agent selects an action a_t in the set A of the possible actions; then action a_t is performed in the environment. Depending on the state s_t, on the action a_t performed, and on the effects of a_t in the environment, the agent receives a scalar reward r_{t+1} and a new state s_{t+1}. The agent's goal is to maximize the amount of reward it receives from the

environment in the long run. This is usually expressed as the discounted expected payoff (or expected return as in [3]) which at time t is defined as follows:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+1+k} \quad , \tag{1}$$

where γ is the discount factor ($0 \leq \gamma \leq 1$) that specifies the importance of future reward. The larger is γ, the more important and more distant future rewards.

In common case reinforcement learning algorithms use tabular functions for estimating utility (value) of current properties. Two kinds of value functions are exist in classic RL:

- functions of states – V(s) – estimates "how good" it is for the agent to be in a given state *s*;
- functions of state-actions pairs – Q(a, s) – estimates "how good" it is to perform a given action *a* in a given state *s*.

The notion of "how good" here is defined in terms of future rewards that can be expected in terms of expected return. Accordingly, value functions are defined with respect to particular policies [3]. Thus, for example, optimal value of a state is the expected infinite discounted sum of rewards which agent will gain if it starts in current state and implements optimal policy.

In reinforcement learning the agent learns how to maximize the incoming reward be developing an action-value function Q(a,s) or a state value function V(s) that maps state-action pairs or states into the corresponding expected payoff value (Equation 1).

2.1. Drawbacks of tabular RL

By the present days, researchers faced to many problems peculiar to RL (not only tabular). Most essential are the following:

- huge amount of trials – the main principle of RL require to execute certain action by agent to explore a reward for each allowable state;
- exploration and exploitation dilemma – the problem rises if the exploitation of agent is not separated from its learning. In that case the amount of exploration is another parameter including to system;
- picking up constants and parameters – usually each algorithm requires custom values of constants and parameters per each task or environment. Often it is done by expert's manual setting up;
- adopting "reality" into RL concept – obviously the problem is nature for all AI algorithms,

At last, the tabular-RL specific drawback is the "curse of dimensionality". Exactly to that problem is devoted the approximation. Classic way of value functions (state-value function or action-value function) representation in reinforcement learning is tabular form. Hence, value storing and updating is simple, intuitive and fast, in other hand, the way is only capable to cope with small number of states and actions. Simple toy tasks, like walking in grid worlds, Windy Gridworld (mentioned in [3]), Pick-and-Place Robots etc, are successfully used by researches for demonstrating principals of RL functioning. Real-world tasks, often requiring taking in account complicated physics in real time, should be neither oversimplified nor solved by other methods.

More over, it is unable to work in tasks with continuous spaces of states or actions. Due to possibly large state-action spaces, it has become clear that tabular-based reinforcement learning scales-up poorly.

For example, Q-learning's Q-table (which is $|S| \times |A|$) grows exponentially in the problem dimensions [2]. The "curse of dimensionality" implies growing of experiences required to converge to an enough estimate of the optimal V- or Q-table, and requires more memory to store the table.

2.2.2 Existing solutions

Two fundamental ways to cope with large space of states are known today wide: value function approximation and policy gradient methods. In Figure 2 the methods are shown in hierarchical structure.

First of all it is important to point out, that we can't cope with the "curse of dimensionality" simply by using local linear features. Simply because of the number of features which grows exponentially with the number of dimensions of the model state space [4].

In the function approximation technique the action-value function Q(a,s) is seen as a function that maps state-action pairs into real numbers (i.e., the expected payoff); gradient descent techniques are used to build a good approximation of function Q(a,s) from on-line experience [2]. In other words, function approximation takes examples from a desired function (in our case a value or action functions) and attempts to generalize from them to construct an approximation of the entire function [3]. So, function approximation allows represent value functions for large state spaces. We can interpret it as compressing. But the main benefit is that thanks to function approximation agent might generalize self experience from "visited" states to unknown. In [5] the authors point out some of the drawbacks of value function estimation (not including residual gradient algorithm).

Most implementations lead to deterministic policies even when the optimal policy is stochastic, meaning that probabilistic action policies are ignored even when they would produce superior performance [6]. Further, because these methods make distinctions between policy actions based on arbitrarily small value differences, tiny changes in the estimated value function can have disproportionately large effects on the policy.

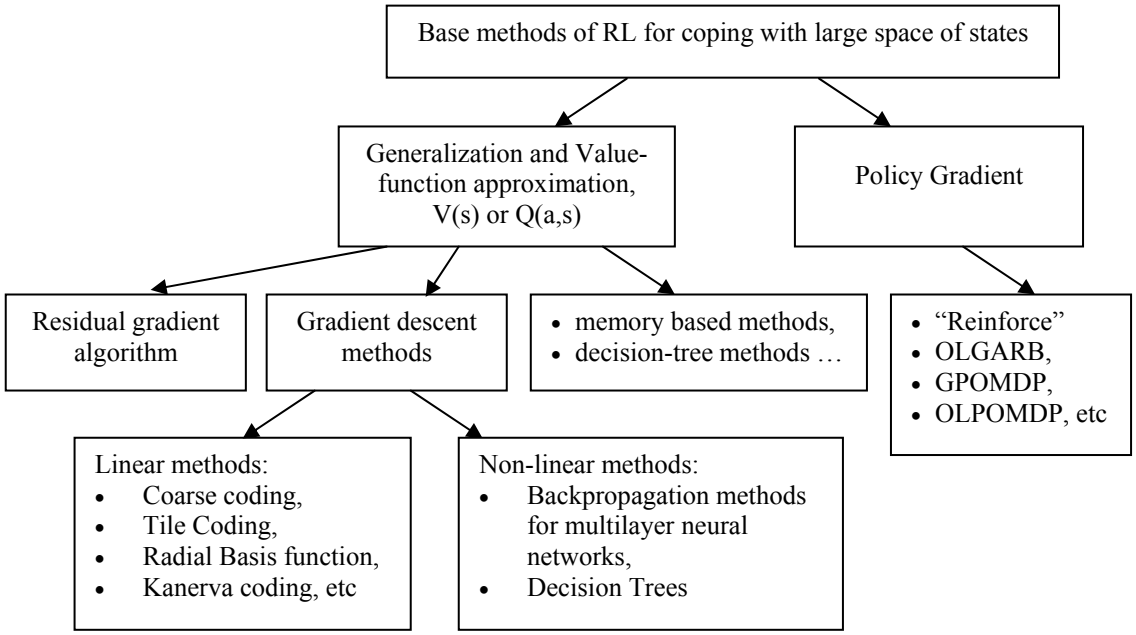

Fig. 2. *Algorithms and methods to cope with continuous or large spaces of states*

In turn a policy-gradient approach is able to bypass drawbacks of directly mentioned above. In this approach, instead of learning an approximation of the underlying value function and basing the policy on the expected reward indicated by that function, policy-gradient learning algorithms maximize the long-term expected reward by searching the policy space directly. In addition, being able to express stochastic optimal policies and being robust to small changes in the approximation, under certain conditions policy gradient algorithms are guaranteed to converge to an optimal solution [7], [8].

Interesting is that value function approximation despite some theoretical drawbacks (mentioned above), demonstrates in practice greatly better results than policy gradient. Exhaustive experiment named "*Policy Gradient vs. Value Function Approximation: A Reinforcement Learning Shootout*" executed by [6] demonstrates that Sarsa(λ) armed with function approximation is able perform better than OLGARB[1] in continuous, stochastic, partially-observable, competitive multi-agent environment.

The residual gradient algorithms (proposed in [9]) are a new class of algorithms, which perform gradient descent on the mean squared Bellman residual, guaranteeing convergence. It is shown, however, that they may learn very slowly in some cases.

Let's consider a special case of gradient-descent function approximation when approximate function $V_t(s)$ (which is value of state s) is a linear function of the parameter vector $\vec{\theta}_t$. The general gradient-descent method for state value prediction is as follows:

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha[v_t - V_t(s_t)]\nabla_{\vec{\theta}_t} V_t(s_t), \tag{2}$$

where α is a positive step-size parameter, and v_t is target output of the *t*-th training example. For details see [3]. Corresponding to every state s, there is a column vector of features $\vec{\phi}_s = (\phi_s(1), \phi_s(2), ...\phi_s(n))^T$. Thus, the linear approximate state-value function is given by

$$V_t(s) = \vec{\theta}_t^T \vec{\phi}_s = \sum_{i=1}^{n} \theta_t(i)\phi_s(i). \tag{3}$$

[1] OLGARB is initials from "On-Line GPOMDP with an Average Reward Baseline", in one's turn GPOMDP is initials from "Gradient of a Partially Observable Markov Decision Process".

Due to linear case the gradient of the approximate value function with respect to $\vec{\theta}_t$ simply is

$$\nabla_{\vec{\theta}_t} V_t(s_t) = \vec{\phi}_s . \tag{4}$$

Finally, our goal is to find the parameter vector $\vec{\theta}$. In one's turn to convert state into features representation the Tile Coding is used. Notice that for control tasks the action-value $Q_t(s_t, a_t)$ is used instead of $V_t(s_t)$.

3. Tile Coding Implementation

Tile Coding is an algorithm of generalization value function having linear representation by a set of parameters. In our case the software implements on-policy Sarsa(λ) control method using linear, gradient-descent function approximation with binary features via Tile Coding. Some parts of the software are based on [3] works.

Let's see several details how to implement value function approximation by Tile Coding algorithm. We will use Mountain Car as the experimental task due to some difficulty: gravity is stronger than the car's engine and even at full throttle the car cannot accelerate up the steep slope when starting (at zero velocity) at the bottom. The only solution is at first to move away from the goal and up the opposite slope on the left. This is a simple example of a continuous control task where things have to get worse in a sense (farther from the goal) before they can get better. The actions available to the car are full throttle forward (+1), full throttle reverse (-1) and zero throttle (0). The car moves according to a simplified physics.

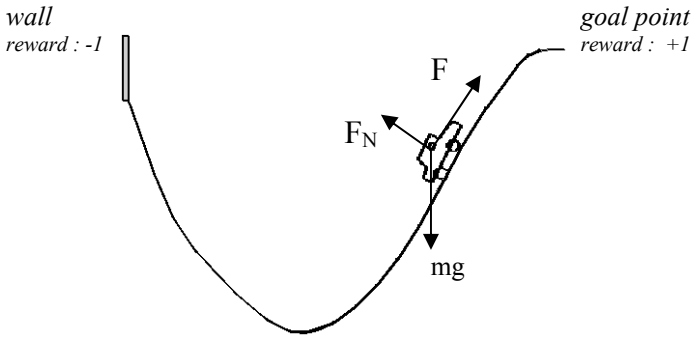

Fig. 3. *The Mountain Car task*

Its position x_t and velocity v_t are updated by:

$$x_{t+1} = bound[x_t + v_{t+1}]$$
$$v_{t+1} = bound[v_t + 0.001a_t - 0.0025\cos(3x_t)]^{'}$$

where the bound operation enforces -1.2 <= x_{t+1} <= 0.5 and -0.07 <= v_{t+1} <= 0.07. When x_{t+1} reaches the left bound it has crashed into the wall and its velocity v_{t+1} is reset to 0. When x_{t+1} reach the right boundary it has reached the goal and the episode is terminated.

The central idea of Tile Coding is that the all continuous space of search (bounded by task's parameters) is divided on pieces called tiles. In other words, each tile represents corresponding feature $\phi_s(i)$. Each tile have own weight. There might be (and should to be!) different ways of partitioning, thus each partition calls tiling. Due to it, the tiling is overlapped (see Figure 4, for example). For a given point in a search space the approximate value is sum of the weights of the tiles (one per tiling, in which it is contained). A number of equal tiling overlapped with offset (shift) usually is enough to avoid generating of different tiling. It well simplifies the implementation. The square on the search space (for 2-dimension case) bounded by each tiling calls resolution. The resolution is one of the generalization's significant properties.

It is important to point out, that high resolution is not a guarantee of best result. The explanation is given below in the text. Nevertheless the Tile Coding successfully deals with continuous variables (a proof sketch is in [10]).

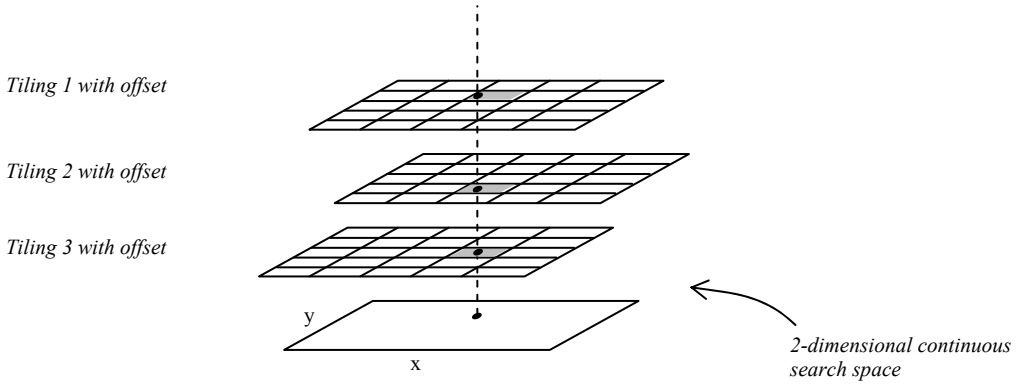

Fig. 4. *Example of overlapped tiling for 2-dimensional space*

Using built software additionally to research of [3] let's investigate dependence of convergence on tiling partitioning. The experiment supposes practically find the optimal tiling partitioning for obtaining the policy which fast (with minimal number of agent's actions) leads the agent to the goal point. Figure 5 represents last 19 observations (total 100) of 14 different partitioning (tiling).

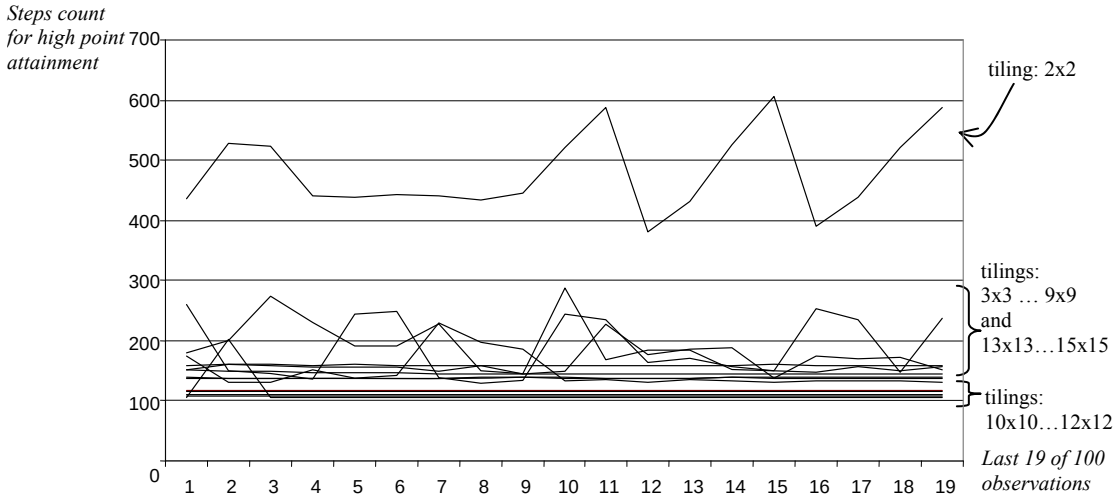

Fig. 5. *Influence of discrimination on time of convergence*

Thus, according to Figure 5, the speed of convergence is not in linear dependence on size of tiling discretization. Most optimal values are 10x10, 11x11 and 12x12. In the same time 8x8, 9x9,13x13, 14x14 unable to give best convergence. Such peculiarity occurs due to result accuracy of approximation to desired function.

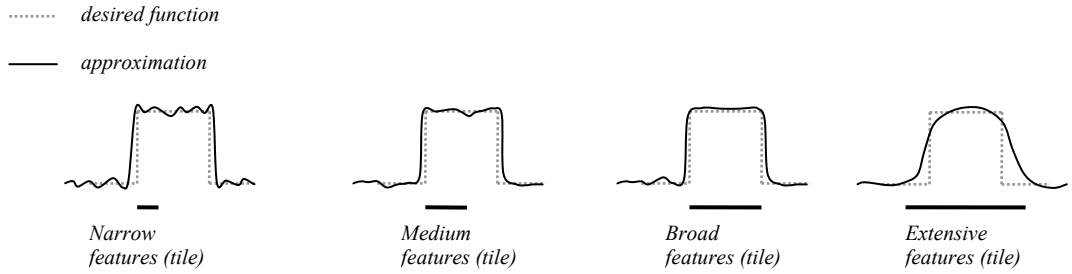

Fig. 6. *Example of features width's*

The width of feature (tile) should be selected taking in account the nature of the desired function. If it is impossible, then the work [10] purposed to automated parameter choosing will be helpful.

4. Gradient Descent with Back-propagation Neural Network

Main motivations to choose the artificial neural networks in task of approximation value function are ability to work with non-linear functions and simple adaptation of features.

Back-propagation property of neural network is used to compute the gradient of the squared TD(λ) error with respect to the network weights. According to [3] the backward view of the action-value method the following expressions take a place. The equation (in terms of RL) of multi-layered perceptron becomes the following:

$$Q(s_t, a_t^k) = g\left(\sum_{j=0}^{N} \theta_i g(\sum_{i=0}^{M} \theta_{ij} s_i) + \theta_0 b_0 \right), \tag{5}$$

where g is activation function of perceptron, a_t^k – an action *k*.

The update of gradient presented in terms of eligibility traces is the following:

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \delta_t \vec{e}_t, \tag{6}$$

where δ_t is the TD-error and computing as:

$$\delta_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t). \tag{7}$$

Finally eligibility traces becomes to

$$\vec{e}_t = \gamma \lambda \vec{e}_{t-1} + \nabla_{\vec{\theta}_t} Q_t(s_t, a_t) = \gamma \lambda \vec{e}_{t-1} + \frac{\partial Q_t(s_t, a_t)}{\partial \vec{\theta}_t}, \tag{8}$$

where α – learning rate, γ – discount rate. Initial value $e_0 = 0$.

The weights of neural networks represent the vector $\vec{\theta}$ (parameter vector). By adjusting the weights, any of a wide range of different functions Q_t (or V_t) can be implemented by the network [3]. The input of the network is the state s_t. Usually the input layer size is equal to state variables count. It works only for discrete state representation. Thus, for 2-dimension task with M×N states the network should be equipped with M+N input neurons. In task with continuous state spaces this way of representation does not satisfy. Using of a gaussian distribution over the input nodes is alternative representation of continuous input state [11].

The output of network is a value of action-value function Q_t. Often output layer size is equal to number of available actions if output neurons are coded in binary mode. Thus, each output neuron is interpreting as a flag to implement or not corresponding action for a current state. Some labours advice to use the same number of networks as many the actions are possible [12] (action per network). The structure of neural network for 1 output value might be implemented as it presented in Figure 7.

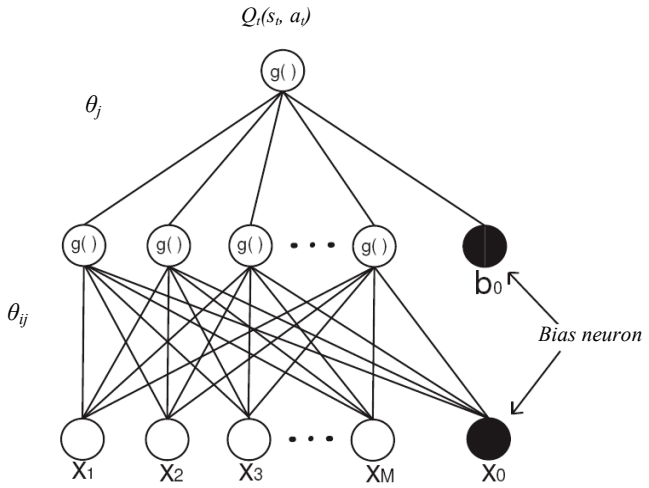

Fig. 7. *The multi-layered perceptron*

As usually, size of hidden layer is free for experiments and searching an optimal value between speed of convergence, occupied memory and quality of output value. The preferred activation function is sigmoid function:

$$g(a) = \frac{1}{1+e^{-a}}, \tag{9}$$

which is the most used activation function for back-propagation networks partly due to its simple derivative. The activation function g(a) is normally a monotonic and non-linear function. Algorithm of combining RL with function approximation is described in [3].

5. Conclusions

In this paper, we have discussed the problem of value function and action-value function approximation in Reinforcement Learning. The problems of tabular form of value function are described. To deal with them, two base methods are expounded: Tile Coding and Gradient Descent with Back-propagation Artificial Neural Network. Both methods successfully deals with continuous state space, nevertheless Tile Coding suffer of "curse of dimensionality". In one's turn binary coding of input layer neurons of neural network might be replaced by gaussian distribution over the input nodes to deal with continuous space.

In Tile Coding, the accuracy of approximating might be increased by tuning up of tiles size. Too large or too small partitioning of state space causes slack approximation (see Figure 5).

Value approximation gives opportunity to RL to be implemented in real-world tasks. In a certain sense a serious work should be done before a turn of agent's policy exploitation starts. State and action description should be transformed to corresponding method's input structure. For finding the optimal working parameters a mass of experiments should be done.

References

1. Sutton, R. *Frequently Asked Questions about Reinforcement Learning* – http://www.cs.ualberta.ca/~sutton/RL-FAQ.html. Initiated 2001.08.13. Last updated 2004.04.02. Visited 2008.04.03.
2. Butz, M.V., Goldberg, D.E., Lanzi, P.L. Gradient Descent Methods in Learning Classifier Systems: Improving CXS Performance in Multi-step Problems*, Evolutionary Computation, IEEE Transactions,* Vol. 9, Issue 5, Oct. 2005, pp: 452-473.
3. Sutton, R.S., Barto, A.G. *Reinforcement Learning. An Introduction*. Cambridge, MA: MIT Press, 1998. 342p.
4. Baxter, J., Bartlett, P.L. *Direct Gradient-Based Reinforcement Learning: I. Gradient Estimation Algorithms.* Research School of Information Sciences and Engineering, Australian National University, July 29, 1999. 24 p.
5. Sutton, R.S., McAllester, D., Singh, S., Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: *Advances in Neural Information Processing Systems 12,* Cambridge, MA: MIT Press, 2000. pp. 1057-1063.
6. Beitelspacher, J., Fager, J., Henriques, G. and Amy McGovern. *Policy Gradient vs. Value Function Approximation: A Reinforcement Learning Shootout: Technical Report No. CS-TR-06-001*. School of Computer Science University of Oklahoma Norman, OK 73019, Feb. 2006.
7. Fager, J. *Online Policy-Gradient Reinforcement Learning using OLGARB for Space-War*. University of Oklahoma, 660 Parrington Oval, Norman, OK 73019 USA, 2006.
8. Baxter, J., Bartlett, P. L. Infinite-horizon policy-gradient estimation*, Journal of Artificial Intelligence Research*, Vol. 15, Nov. 2001, pp. 319-350.
9. Baird, L. *Residual Algorithms: Reinforcement Learning with Function Approximation*. Department of Computer Science, U.S. Air Force Academy, CO 80840-6234. 1995.
10. Sherstov, A.A., Stone, P. Function Approximation via Tile Coding: Automating Parameter Choice. In: *Symposium on Abstraction, Reformulation, and Approximation (SARA-05)*. Edinburgh, Scotland, UK, 2005, p. 12.
11. Bishop, Ch.M. *Neural Networks for Pattern Recognition.* USA: Oxford University Press, 1995, p. 504.
12. Jakša, R., Sinčák, P., Majerník, P. Back-propagation in Supervised and Reinforcement Learning for Mobile Robot Control. In: *Computational Intelligence for Modelling, Control & Automation (CIMCA'99)*. Vienna, Austria, 1999, p. 6.