Markov approach of Adaptive Task Assignment for Robotic System in Non-Stationary Environments

GABRIELA TONȚ[1], LUIGE VLĂDĂREANU[2], MIHAI STELIAN MUNTEANU [3], DAN GEORGE TONȚ[1]

[1]Department of Electrical Engineering, Measurements and Electric Power Use,
Faculty of Electrical Engineering and Information Technology
University of Oradea
Universității st., no. 1, zip code 410087, Oradea,
ROMANIA,
gtont@uoradea.ro, dtont@uoradea.ro, http://www.uoradea.ro
[2]Institute of Solid Mechanics of Romanian Academy
C-tin Mille 15, Bucharest 1, 010141,
ROMANIA
luigiv@arexim.ro, http://www.acad.ro
[3]Department of Electrical Measurements, Technical University Cluj Napoca, Faculty of Electrical Engineering,
Constantin Daicoviciu st. no 15, 400020 Cluj - Napoca,
mihai.munteanu@et.utcluj.ro, http://www.utcluj.ro
ROMANIA

Abstract: - Adaptive and decentralized task assignment in non-uniform and possibly even non-stationary conditions with the aim of ensuring stability is especially challenging knowledge requirements in order to negotiate within and interact with uncertain and dynamic environments such as robot malfunctions or error propagation. Task assignment addresses to the problem of coordination both in on autonomous and interacting robot. In the scenarios for task assignment robots are embedded in the environment, there are strict constraints on communication, and most importantly tasks that robots should execute are perceived by the robot itself during the mission execution, thus conflicts on the task assignment process might arise. The internal representation of timing constraints on interaction has many implications for the reliability, effectiveness, efficiency, validity, schedulability and robustness of the mobile robot. The task assignment processes and its control implying reasoning about objects and resources and their changing states are dominated by either discrete or stochastic-event dynamics or both. Stationarity is an unrealistic prior assumption for the multi-state components of complex systems. The numerical characteristics of the nonstationary random responses of complex structure are developed in the paper considering the uncertainty and volatility of process transitions states. The appropriate functional levels of multi-state systems components are placed between the two extremes: defect (0%) and the nominal (100%) state, allowing any intermediate state transiting from the range of perfect functionality to complete failure. Time analysis interval of a multi-state system is characterized by intermediate states (states of partial success).
The paper proposes a stochastic model of assessing system probability of unidirectional or bidirectional transition states, applying the non-homogeneous (non-stationary) Markov chain. The capability of time-dependent method to describe a multi-state system is based on a case study, assessing the operatial situation of robotic system. The rationality and validity of the presented model are demonstrated via an example of quantitative assessment of states probabilities of an autonomous robot.

Key-Words: -.Petri nets, Fuzzy sets, mobile robots, non-stationary discrete event, dynamic systems, reasoning

1 Introduction

Modeling system dynamics and concurrency control in a more efficient way requires the development of synchronization-structures with some features need to be added in order to specialize the knowledge representation. To maintain those properties by means of hierarchical fuzzy Petri nets, when the information flow within the system takes the form of process and task and control sequences, may satisfy the Markovian property [1, 11]. The

Markovian property, which affirms that next state of the system depends on the previous one only, and not on the entire history, is satisfied by mobile robot systems in regard of task assignment. The complexity in control and supervision originates from the fact that the lower level of execution control involves both set-point and task oriented controls [1, 10].

Techniques for real-time, Petri Nets allow to build dynamic models which incorporate time information of the process development. In this regard, sequencing and planning actions can be checked and monitored throughout system states that can be related to insecurity conditions.

A step toward quantifying the findings through Petri nets, which are utilized in qualitative analysis, is associating with methods such as Markov, semi-Markov models or Monte-Carlo technique.

Such models have logic mathematical or matrix -based, binary codifying the logical state of each component functions [3, 11]. Petri Nets are graphical techniques that can be utilized to model and analyze safety critical systems relative to the properties as: tangibility, recoverability, fault tolerance. Petri nets identify relations between the systems components as the hardware, software, human actions, or cumulative effects of hardware and software. The Petri nets offer a graphical notation for complex processes that include choice, iteration, and concurrent execution net. By means of Petri net formalism can be represent complex logical interactions between the physical components of the modeled system. Petri nets consider the system state and transitions between them, which occur only if certain conditions are fulfilled [6, 7, 11]. The advantage of this description of discrete event systems is simplicity and the possibility of a system analysis to emphasizes the logical correctness of development, repeatability, extensibility. Modeling with Petri Nets tools differs from most methods of analysis in that it demonstrates, from the start, the dynamic progression transitions states. Petri nets can also be translated in mathematical-logical expressions which can be analyzed with automated tools. Information can be extracted or reformulated with graphs and assisting analysis tables which are relatively easy-to-use tools (eg tangibility charts, graphs inverse Petri Nets, graphs critical condition, etc.). Temporal Petri Nets (TPN) takes into account time-dependent real-time systems;

Inverse Petri nets are necessary, in particular, to analyze the safety and modeling approach uses backward modeling in order to avoid the modeling of all possible tangible states; Petri nets with critical states, which are relatively new technologies and whose implementation is costly explicitly request technical expertise of the analyst.

Some of the potential advantages relatively to other similar techniques include:

- Petri nets can be used for driving time requirements in real-time systems;
- Petri nets allow the user to describe the system using graphical notation, thus the user issued mathematical rigor that requires a complex system;
- Technique can be used across all development phases of the system;
- Using Petri Nets may detect the potential problems related to changes, the stages in which those changes may occur with significantly lower costs than at end stage;
- Nets Petri can be used to determine where the most unfavorable cases for the analysis and risks of failure due to timing;
- Because the same language can shape behavioral hardware, software and human, it is a possible systemic approach;
- Petri nets can be used on different levels of abstraction;
- Petri Nets provides a modeling language that can be used for both formal analysis and simulation;

Adding probability and time for each Petri net can be incorporated in the analysis of probabilistic information and indexing

2. Literature Review

The problem of construction of models of systems and processes by integration (composition) of models of basic components was practically in many works. Many results prove that FPN is suitable to represent and reason misty logic implication relations [2, 5]. FPN is widely applied in knowledge system representation and redundancy reduction. But there exist some main weaknesses when a system is complex:

- The complexity of knowledge system cause a huge fuzzy Petri net model, this hampers the application of FPN;
- Knowledge system is updated or modified frequently.
- Suitable models for them should be adaptable. [3, 6];
- Knowledge can not be classified as well as human cognation in a FPN;

When a knowledge system is made of some substructures, for example, an expert system may be divided into several subsystems according to different type of knowledge.

3. Problem formulation

Any intelligent system has a limited vocabulary of actions that it may take in order to accomplish its goals. The agent must decide which of these actions to perform, and when to perform them. The responsibility for making this decision is shared by the process that creates the knowledge representation and the process that constructs a plan of action based on this knowledge representation. The choice of which representation is used and what knowledge is stored helps to decide the division of this responsibility. Very complex reasoning may be required to condense all of the available information into this single measure [4,16]. The techniques include computation-based closed-loop control, cost-based search strategies, finite state machines, and rule-based systems. Computation-based closed-loop controllers put most of the decision burden on the planning task. In hazardous and populated environments mobile robots utilize motion planning which relies on accurate, static models of the environments, and therefore they often fail their mission if humans or other unpredictable obstacles block their path. Autonomous mobile robots systems that can perceive their environments, react to unforeseen circumstances, and plan dynamically in order to achieve their mission have the objective of the motion planning and control problem. To find collision-free trajectories, in static or dynamic environments containing some obstacles, between a start and a goal configuration, the navigation of a mobile robot comprises localization, motion control, motion planning and collision avoidance. Its task is also the online real-time re-planning of trajectories in the case of obstacles blocking the pre-planned path or another unexpected event occurring. Inherent in any navigation scheme is the desire to reach a destination without getting lost or crashing into anything. A higher-level process, a task planner, specifies the destination and any constraints on the course, such as time. Most mobile robot algorithms abort, when they encounter situations that make the navigation difficult. Set simply, the navigation problem is to find a path from start (S) to goal (G) and traverse it without collision. The relationship between the three subtasks – mapping and modeling of the environment; path planning and selection; path traversal and collision avoidance – into which the navigation problem is decomposed, is shown in Fig. 1.

Being autonomous or partly autonomous-guided systems, the mobile robots need to move in an unstructured environment without having a prior knowledge about it. In order to provide this autonomy, there is a need of sensor-based navigation.

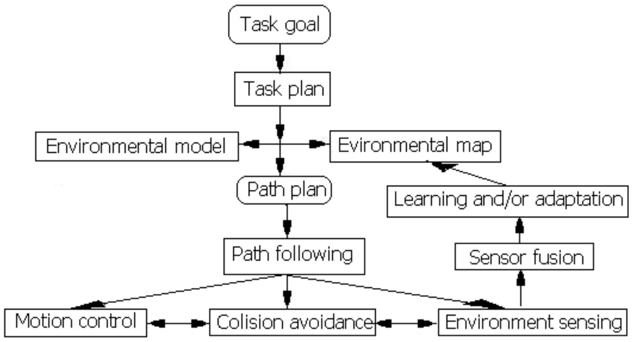

Fig. 1 Mobile robot control hierarchy

A reliable sensor system is the key feature for the perception of the environment. The basic configuration of each mobile robot has its integrated sensors. Depending on the requirements of the task some sensors may be superior to others. For the development of service robots, one usually chooses a combination of different sensors, in order to achieve optimal results in the environmental sensing. First of all, the navigation system needs firmly integrated sensors to each of the actuated wheels or other means of mobility. Generally there are photo-optical encoders coupled to the wheels which give the main control system data for their position and number of turns in some of the rotation directions. Having this information, the control system can follow a certain programmed path. The second group of firmly integrated sensors is those connected to robot safety. The anti-collision sensors are usually attached to the mechanical bumpers surrounding the mobile platform. They are usually tactile sensors or micro switches built in, or, in the more advanced ones the bumper systems are doubled for safety reasons by ultrasonic or laser ones installed around the robot body. There is a large variety of sensors, which can be added to the basic ones, thus adapting the robot for some special applications or extending its abilities. The use of vision to acquire the location of objects requires extensive computing power and robust recognition algorithms. The camera captures a two-dimensional image, from which the vision processing software must extract image features. These features are compared to models of the objects to identify the object of interest and the location to which the robot has to move. The location (position and orientation) of the object relative to the robot, and/or relative to world co-ordinates is then estimated to produce an object location signal. Sensors are varying from simple ones to the very sophisticated sensory systems, applying the artificial intelligence (AI) principles.

A basic problem in robotics is to resolve specified tasks and commands and plan the resulting motions and sub-tasks. The planning system needs to transform a task-oriented problem into a plan, which describes how the given problem can be solved by the robot. For this transformation a detailed knowledge base and world model have to be available. These models give the robot a description of its environment, and thus enable it to construct the necessary operations needed to fulfill the task. The plan generated in this way contains a sequence of action elements (e.g. movement, picking up items, manipulating items) with assigned resources (e.g. the robot or its gripper). The motion control manager determines the start and destination of a path and plans the course and the necessary actions. The resulting motions of a robot are called trajectories or paths and consist of a sequence of desired positions, velocities and accelerations at a given point. The sequence of plan elements is called a task execution sequence. During this planning stage all constraints and restrictions, like closed or impassable areas are considered as well as target times, resources, supplies or the processing of parallel or sequential tasks (Fig. 2.) [2,16].

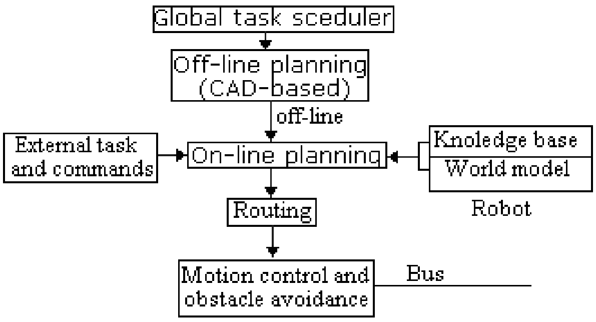

Fig. 2 Task planning of mobile robots

Motion planning of mobile robots in uncertain dynamic environments with no restrictions on the shape of obstacles based on the behavior dynamics of collision-avoidance is transformed into an optimization problem. Applying constraints based on control of the behavior dynamics, the decision-making space of this optimization.

4. Fuzzy Petri Net Formalism

In order to properly represent real world knowledge, fuzzy rules have been used for knowledge representation [5]. A fuzzy rule (FPR) is a rule which describes the fuzzy relation between two propositions. If the antecedent portion of a fuzzy rule contains AND or OR connectors, then it is called a composite fuzzy rule. If the relative degree of importance of each proposition in the antecedent contributing to the consequent is considered, Weighted Fuzzy Rule (WFR) has to be introduced. Let *R* be a set of weighted fuzzy rules R = {R_1, R_2, …,R_n}. The general formulation of the *i*-th weighted fuzzy rule is as follows:

R_i : IF a THEN c ($CF = \mu$), *Th*, *w*

where a =< a_1, a_2,…, a_n > is the antecedent portion which comprises of one or more propositions connected by either AND or OR, *c* is the consequent proposition, μ is the certainty factor of the rule, *Th* is the threshold, *w* is the weight. In general, WFRs are categorized into three types which are defined as follows [7]:

Type 1: A Simple Fuzzy Rule

R: IF a THEN c (CF = μ), λ, w

Type 2: A Composite Conjunctive Rule

R: IF a_1 AND a_2 AND…AND a_n THEN *c* ($CF = \mu$), $\lambda, w_1, w_2, \ldots, w_n$

Type 3: A Composite Disjunctive Rule

R: IF a_1 OR a_2 OR,…,OR a_n THEN *c* ($CF = \mu$), $\lambda_1, \lambda_2, \ldots, \lambda_n, w_1, w_2, \ldots, w_n$

In order to capture more information of the weights, the FN model has been enhanced to include a set of threshold values and weights, it consists of a 13-tuple :

$FN = (P, T, D, Th, I, O, F, W, f, \alpha, \beta, \gamma, \theta)$ (1)

where Th = {$\lambda_1, \lambda_2, \ldots, \lambda_n$} denotes a set of threshold values, $F = \{f1, f2 \ldots fs\}$ denotes a set of fuzzy sets, $W = \{w1, w2, \ldots, wr\}$ denotes a set of weights of WFRs, α : P → F is an association function which assigns a fuzzy set to each place, γ : P → Th is an association function which defines a mapping from places to threshold values. The definitions of *P, T,D, I, O, f* and *β* are knowledge updating. In another word, it has not learning ability.

Definition 1 A fuzzy Petri net with learning ability is a 9-tuple:

FPN = (*P, T, D, I, O, α, β, Th, W*) where P, T, D, I,O, α, β are defined as:

Th : T → [0, 1] is the function which assigns a threshold value λ_i from zero to one {$\lambda_1, \lambda_2, \ldots \lambda_m$}. $W = W_I \cup W_O$. $W_I : I \rightarrow [0, 1]$ *and* $W_O : O \rightarrow [0, 1]$, are sets of input weights and output weights which assign weights to all the arcs of a net. The mappings of the three types of weighted fuzzy rules into this fuzzy Petri net may be represented as follows:

Type 1: A Simple Fuzzy Rule

R: IF a THEN c $Th(t) = \lambda$, $WO(t, pj) = \mu$, $WI(p_i, t) = w$

Type 2: A Composite Conjunctive Rule

R: IF a_1 AND a_2 AND…AND an THEN c, Th(t) = λ,$W_O(t, p_j) = \mu$, $W_I(p_i, t) = w_i$, i = 1,…, n
Type 3: A Composite Disjunctive Rule
R: IF a_1 OR a_2 OR …OR an THEN c, Th(ti) = λi, $W_O(t_i, p_j) = \mu$, $W_I(p_j, t_i) = w_i$, i = 1,…, n
The mapping may be understood as each transition corresponds to a simple rule, composite conjunctive rule or a disjunctive branch of a composite disjunctive rule; each place corresponds to a proposition (antecedent or consequent).
Decomposing fuzzy Petri net into set of linguistic descriptions which form Linguistic Fuzzy Logic Net (LFLN) gets fuzzy Petri Net as an input and creates set of linguistic descriptions corresponding to each output place of fuzzy Petri Net [8, 9]. The algorithm of decomposition is shown below:

```
input : fuzzy Petri net: fpn
output: set of linguistic descriptions: lfln
lfln = Ø;
foreach output place op of fpn do // create linguistic description
    // create set of input variables (places) on whose op depends
    inputs = Ø;
    foreach input transition it of op do
        // add all inputs of transition it to inputs set
        inputs = inputs ∪ it.inputs;
    end
    // construct linguistic description (set of rules)
    rb = Ø;
    foreach input transition it of op do
        // construct rule corresponding to transition it
        rule = Ø;
        foreach element in from inputs do
            if rule ≠ Ø then rule = rule + AND;
            if in 2 it.inputs then
              rule = rule + in.name is edge(in, it):value;
            else
              rule = rule + in.name is UNDEF;
            end
        end
        rule = rule + THEN op.name is edge(it, op).value;
        rb = rb ∪ rule ; // add rule to rule base
      end
        lfln = lfln ∪ rb ; // add rule base to set of linguistic descriptions
  end
```

FPN is a modeling approach rather than a model. It has the following correspondence with real world:

- Objects and Relations: Real world knowledge comes from different resources and their communication;
- FPN portrays knowledge resources as object subnets and join them together by their communication relation;
- Fuzzy Production Rules: Every knowledge resource in real world has its own rules which are relatively independent and may be described by weighted fuzzy nets;
- Concurrency and Conflict: Concurrency occurs within an object subnet and also among multiple objects. Conflicts within FPN are what should be prohibited in knowledge systems, because it causes inconsistency;
- Logic causality: Fuzzy Petri net structure preserves causality, thereby preserves the correct logic relations.

5. Modeling Transitions States Volatility of Non-Stationary Discrete Event. Observability of Transition

Finding the optimal sequence of actions a robotic system should carry out in order to completely fulfill a given set of objectives, such as minimizing a given cost function, or equivalently maximizing a utility function. These are problems not solvable analytically, due to partial and uncertain knowledge about the consequence of the robot actions over the surrounding environment. However, one can use an approximate world model, capturing its main features, and include uncertainty in such a model.
The stochastic discrete event theory [8] is used for modeling and optimal decision making concerning autonomous robots. The probability distributions of event occurrences are strongly dependent on the continuous valued state of the robot and positions of the obstacles in the environment and are not, in most cases, stationary with respect to time. The probability of the state and transition are calculated in this paper by means a stochastic model robotic tasks which may serve as a good guideline to design the decision scheme.
Application of the markovian modeling with continuous parameter, for a realistic analysis of the non-stationary discrete event systems evolution of robotic tasks allows highlighting intermediate states and transitions and quantitative evaluation of probabilities of states occurrence. [11, 12,16].
One of the typical features of stationary long-memory processes is that there appear to be local trends and cycles, which disappear after some time. This property can make it rather difficult to distinguish a stationary process with long memory from a nonstationary process. For short time series, it might be almost impossible to decide this question. Without any additional information, there would be no reason to believe in stationarity. Conclusions based on such short series should, however, be interpreted with caution. Sometimes additional, possibly non-numerical, information is available that favors either nonstationarity or stationarity. If we try to distinguish stationarity from arbitrary trends that persist for arbitrarily short periods, the task becomes impossible. Some

restrictions need to be imposed on the nature of these trends. More generally speaking, there is a vast range of possible nonstationarities. In principle, it is therefore not possible to decide for sure whether the underlying process is indeed stationary. Ultimately, this is rather a philosophical question that might not be of primary practical interest. From the practical point of view, the assumption of stationarity provides a simple framework for parsimonious modeling. Unless there is evidence against it, it tends to be more useful than complicated nonstationary models. We define:

(1)

and

$$I(i) = \lambda_{j_n(i)}^{-1+2H} I(\lambda_{j_n(i)}). \quad (2)$$

where I(i) and $j_n(i)$ are asymptotically independent. The long-memory type behavior can also be caused by complicated global or local non-stationarity. In particular, a phenomenon that often occurs is that a system initially in a nonstationary transient status moves gradually into a stationary equilibrium. There are many other ways how certain long-memory features of a finite data set can be generated. As always in statistics, it is not possible to decide with certainty which model is correct. In fact, it is rather unlikely that any of our models is ever correct exactly. Given a finite data set, the aim can therefore hardly be to find the exactly correct model. Instead, one aims to find a model that serves the intended purpose best.

Criteria for choosing such models are:

– satisfactory fit to the data;
– reliable prediction of future observations; sufficiently accurate parameter estimates;
– simplicity;
– interpretability.

In particular, an important criterion is the principle of parsimony. If the number of parameters is relatively large compared to the number of observations, then parameter estimates and statistical inference based on the model are very inaccurate. Moreover, complicated models with many parameters are difficult to interpret.

In the ARIMA (p, q) models analysis with unknown p and q the dimensions p and q may be estimated by a suitable selection criterion that penalizes high values of p or q.

To model the volatility of transition state is necessary to consider a number of problems and issues associated with stochastic volatility in the assessment of reliability indices. Stochastic volatility has an important effect on the process underlying uncertainty, altering the basic assumption of normal or binomial driving disturbances. To see these effects we consider the simple binomial model used repeatedly and given below:

$$\varepsilon_t = \begin{cases} H & p \\ L & q \end{cases} ; \quad (3)$$

Here, α notes the process constant volatility. If the volatility assumes values of 1 and zero, with probabilities (p_α, q_α) leads to relation (4):

$$\varepsilon_t = \begin{cases} H & p \\ L & q \end{cases} ; \quad \overline{\alpha}_t = \begin{cases} 1 & p_\alpha \\ 0 & q_\alpha \end{cases} \quad (4)$$

For the density function $F_{zt}(z)$, using elementary probability calculations can be written under the form:

$$F_{zt}(z) = \int_0^\infty F_\alpha(\alpha) F_\varepsilon\left(\frac{z}{\alpha}\right)\frac{d\alpha}{\alpha} - \int_{-\infty}^0 F_\alpha(\alpha) F_\varepsilon\left(\frac{z}{\alpha}\right)\frac{d\alpha}{\alpha} \quad (5)$$

We may note that it is not the size of volatility that induces incompleteness but its uncertainty that matters. Assuming that, at the continuous-time limit, the process can be represented by a stochastic differential equation, it is then reasonable to assume that there is some binomial process that approximates the multi-state process. In this approach, a multinomial process is replaced by a binomial process, consisting of as many stages (discretized time) as are needed to replicate.

6 Case study

In order to perform the evaluation Petri net, it is necessary to quantifying the probabilities of states using Markov chains, assuming following hypothesis:

- failure and repair of the component is random, the currently state depends only on the immediately preceding state;
- repartition functions of the operation and repair time is assume exponential, the failure intensity (λ), and repair intensity (μ) are constant time.

Assimilating development over time of system through the various states that may occur as a result of fail and restore elements with a Markov process with continuous time and date, solving is done by the system of differential equations, written under generalized form as a matrix:

$$\left[P^{/}(t)\right] = \left[a_{ij}\right] \cdot \left[P(t)\right] \qquad (6)$$

in which $\left[P(t)\right]$ is the matrix of state probabilities at time t;

[a_{ij}] is transition matrix (matrix of transition intensities from a state to another), whose elements satisfy the following relations:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} = 0, (\forall)\, i = \overline{1,n} \\ a_{ij} \geq 0, (\forall)\, i = \overline{1,n}, i \neq j \\ a_{ij} \leq 0, (\forall)\, i = \overline{1,n}, \end{cases} \qquad (7)$$

where *n* is the possible states number of the system analyzed.

Analytically solving the system of differential equations (7) involves a series of difficulties, for this reason in the engineering assessments is possible its transformation in a system of algebraic equations. In this respect, the fact that for large times $(t \rightarrow \infty)$, the absolute probabilities P (t) are practically independent of the initial state of the studied system and therefore can be considered constant, which implies $P^{/}(t) \rightarrow 0$, and the equation becomes:

$$\left\lfloor a_{ij} \right\rfloor \cdot \left[P\right] = 0 \qquad (8)$$

The matrix $\left\lfloor a_{ij} \right\rfloor$ being singular, the system is undetermined. Transition matrix elements are calculated as follows:

- if the transition from state *i* to state *j* is made by the failure of an element with failure intensity (λ), then: $a_{ij} = \lambda$;

- if the passage of state *i* in state *j* is made by repairing of an item, with the repair intensity (μ), then: $a_{ij} = \mu$

According to relationship (7) the equation becomes:

$$a_{ij} = -\sum_{j=1, j \neq i}^{n} a_{ij}, (\forall) i = \overline{1,n} \qquad (9)$$

Task assignment modeled by two states (operating and fault) are insufficient, imposing the inclusion of one or more intermediate states of partial success. In fig. 3 is considered a robotic system characterized by three states: operating at full capacity (F), defect (D) and intermediate (I).

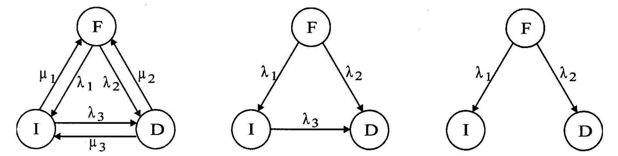

Fig.3 The model with three states for the robotic system

A generalized diagram of states is shown in Fig. 4, which included three intermediate states.

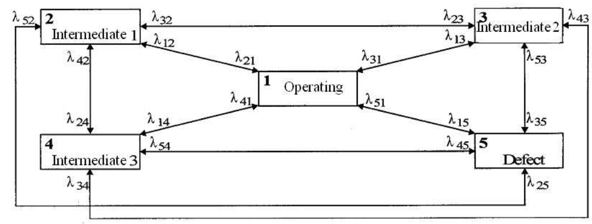

Fig. 4. Generalized diagram of states with three intermediate states

The Markov modeling technique requires to identify each intermediate state (in practice, more neighboring levels can be grouped together), to know the occupancy status of each component (T_i) and the number of transitions between states (N_{ij}), which can calculate as follows:

- occupancy probability of *"i"*state:

$$P_i = \frac{T_i}{T_A}$$

- transition intensity from state *"i"* in *"j":*

$$\lambda_{ij} = \frac{N_{ij}}{T_i}$$

where: $T_A = \sum_i T_i$ is analyzed time interval.

The number of intermediate states to be modeled in order to obtain a more accurate assessment of the reliability group is necessary to consider more than one intermediate state.

Figure 5 presents a model with six states to assess the predictable transitions in a robotic system. The six states of the system are:

1 - operational state of robot;
2 - stopped by demand;
3 - stopped by failure;
4 - stopped by change of task;
5 - stopped by other components failure of the robot;
6 - off for planned maintenance.

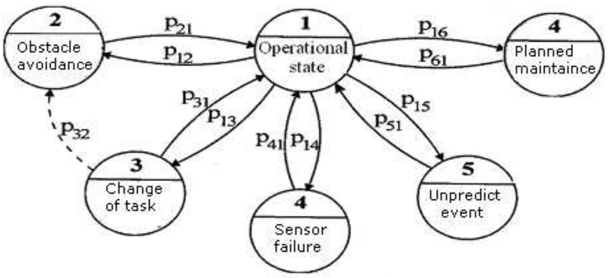

Fig.5. Modeling the states with possible transitions for robot

Based on the surveillance data in operation regime of robot were determined transition probabilities using of the relationship:

$$\hat{p}_{ij} = \frac{n_{ij}}{n_i} \qquad (10)$$

where n_{ij} is the transition from state *"i"* in *"j"* in the analysis time interval;

n_i is the number of all transitions from state *"i"* in any other states.

Values of these transition probabilities are: $\hat{p}_{12} = 0{,}43;$ $\hat{p}_{13} = 0{,}36;$ $\hat{p}_{14} = 0{,}06;$ $\hat{p}_{15} = 0{,}11;$ $\hat{p}_{16} = 0{,}01;$ $\hat{p}_{31} = 0{,}91;$.

By applying the method Markov chains are obtain the occupancy probability of the sates for the robot: P_1=0,81; P_2=0,08; P_3=0,04; P_4=0,005; P_5=0,002; P_6=0,06.

Robotic system can be stochastically modelled under the form of a graph the six states shown in Fig. 6. States ensemble is divided into two areas, defined as:

"The necessary" (N) to meet demand for power;

"The robot is not necessary" (NN), which is under repair or be prepared to enter into service.

Robot can handle the following states:

F1 - in normal operation;

F2 - operational disturbed, forced power reduce;

P - started, but without moving;

R1 - under repair in the N, R_2 - under repair in the NN;

PF - ready for the service;

Pp - the failure probability of the robot during startup;

Po - stopping probability robot immediately after the appearance of a failure of a component;

$\lambda = 1/M\{T_F\}$ - failure;

$M\{T_F\}$ - mean operating time;

$\mu = 1/M\{T_R\}$ - repair rate

$M\{T_R\}$ - mean time to repair

$\delta = 1/M\{T_{F2}\}$ - intensity of transition of F_2 in R_1

$\alpha = \alpha_R + \alpha_S$ - intensity of transition from the (NN) in the (N), where α_R refers to the start of the malfunctions of robot in service;

α_S refers to the start of robot in the case change of navigation task required by environment;

β - Intensity of the transition from the space (N) in the (NN).

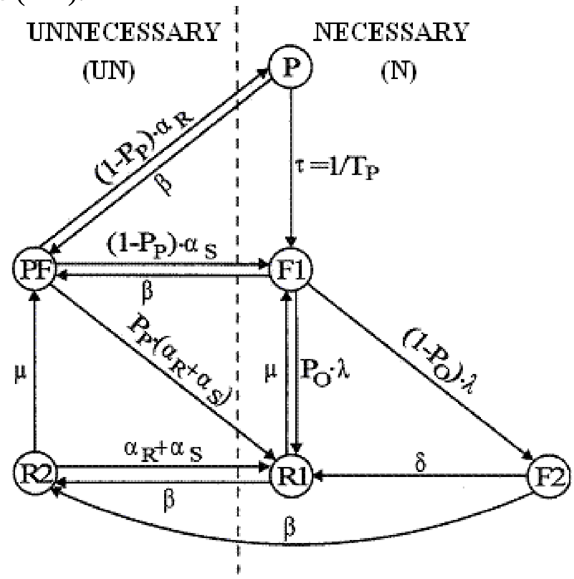

Fig.6. Graph states and transitions for modelling the reliability of a turbogenerator

Markov modeling of the functioning robot, shown in Figure 6, is explained by considering the initial state (PF), as a result of increasing demand to adapt to the environment requirements, or because of changes in the operational state of robot. Through a successful start, the robot goes to the state (F1). If the robotic system is defecting at startup, go in the state (R1). If is forced to reduce power, robot passes in the operating state (P) take account of delays (TP). During the robot is in state (P), the power deficit is covered by turning reserves of the r robot in operation or in neighboring interconnected systems. When the robot on running (F1) is defecting, there are two possibilities of transition: the robot is stopped immediately, by is passing in the state (R1), respectively stopping is deferred, in which case is switched in state (F2). After the period of stopping, the robot will in the pass in the state (R1). After state (R1 robot can return to state (F1), the intensity of the repair (μ).

State (R1) appears not only in space (N) but also in the (NN), the transition between the two areas (N and NN) depends on the variation of demands that appear in the operational state of the robot. Space of all states (N) can move in space (NN) with the intensity of the transition (β): from states (P) and (F1) to move to (PF), and (F2) and (R1) in (R2). Transitions from the (NN) in the (N) are determined by the intensity of transition ($\alpha = \alpha_R + \alpha_S$). For longer intervals of time, probability states are practically independent of the initial state can be considered constant.

A navigation task is accomplished by the cooperation of components such as a position localizer and a path planner [10]. In this example component 1 is functioning (location p_1) or is defective (location p_3), and component 2 is in working order (location p_2) or is defective (location p_4).

The location of the p_5 it was represented the defective state of the system, which consists of two components connected in parallel. Priming t_i transition lead to the failure of component 1 with rate failure $\lambda_1(t)$ and executing the transition t_2 leads to the failure of component 2 with failure rate $\lambda_2(t)$.

Transitions t_3, t_6 and t_7 are transitions without delay (immediate transitions). Priming transition t_4 lead to the restoration component 1 with the rate of recovery $\mu_1(t)$ and priming the transition t_5 leads to the restoration component 2 the rate of recovery $\mu_2(t)$.

Accessibility graph and Markov graph is shown in Fig. 7. Fig.8 and respectively Fig. 9.

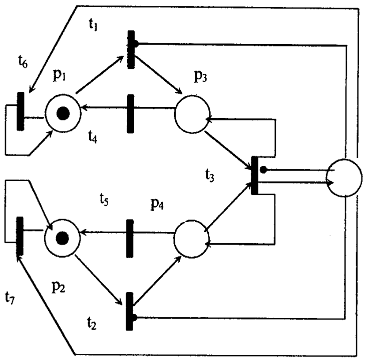

Fig.7 System Petri Net representation

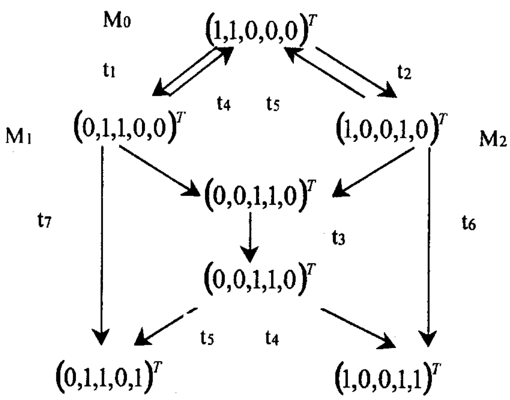

Fig. 8 Accessibility graph for the system

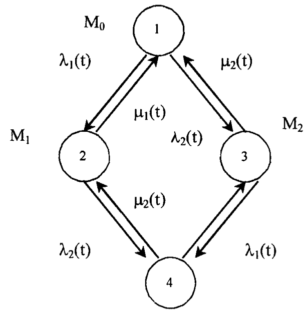

Fig. 9 Associated Markov graph of the system modeled by means of Petri nets

The states 1, 2 and 3 from Markov graph are functioning (Success) states of the system and state 4 is its fault condition (unsuccess/warning). State probability vector P (t) system and matrix of transition rates C (t) are

$$P(t)=[P_1(t), P_2(t), P_3(t), P_4(t)] \qquad (1)$$

$$C(t)=\begin{bmatrix} -\lambda_1(t)-\lambda_2(t) & \lambda_1(t) & \lambda_2(t) & 0 \\ \mu_1(t) & -\lambda_2(t)-\mu_1(t) & 0 & \lambda_2(t) \\ 0 & 0 & -\lambda_1(t)-\mu_2(t) & \lambda_1(t) \\ 0 & \mu_2(t) & \mu_1(t) & -\mu_1(t)-\mu_2(t) \end{bmatrix}$$

(11)

The equations of state for the system are:

$$\begin{cases} P_1^{/}(t) = -P_1(t)[\lambda_1(t)+\lambda_2(t)] + P_2(t)\mu_1(t) + P_3(t)\mu_2(t) \\ P_2^{/}(t) = P_1(t)\lambda_1(t) - P_2(t)[\lambda_2(t)+\mu_1(t)] + P_4(t)\mu_2(t) \\ P_3^{/}(t) = P_1(t)\lambda_2(t) - P_3(t)[\lambda_1(t)+\mu_2(t)] + P_4(t)\mu_1(t) \\ P_4^{/}(t) = P_2(t)\lambda_2(t) + P_3(t)\lambda_1(t) - P_4(t)[\mu_1(t)+\mu_2(t)] \end{cases}$$

(12)

A token is assigned to P_3, and is assumed that the localizer initially knows its position. The Warning event t_5 fires when the localizer fails in estimating robot's accurate position for several steps. Two navigation primitives can be modeled as P_1, P_2, respectively. Initially, the robot selects its motion by a random switch comprising the transitions t_1 and t_2 with corresponds to probabilities P_1' and P'_2, respectively. The transition between them takes place according to the change of localizer states. The immediate transition t_3 means that the robot takes Contour tracking as soon as the localizer Warning event fires. The other transition between two primitives, t_2 and t_4, are modeled as timed transitions in order to express that the robot can change its current navigation primitive during the localizer Success state, if necessary.

7 Conclusion

The modeling of robotic behaviour based on non-stationary discrete event techniques is a necessary step to move toward optimal behaviors of the robot facing its surrounding environment. To obtain the optimal set of actions exponential rates were assigned to each of the uncontrollable events based on empirical considerations. The probabilities absolute value does not matter: what are important are the relative rates of different events, with high transition rates corresponding to short intervened times and low rates associated with long time intervals. These rates values were chosen to simulate a worst-case environment, where the robot need a considerable time recover it back. Fuzzy nets provide a promising solution towards the development quantitative approach of dynamic discreet / stochastic event systems of task planning of mobile robots. For a deeper insight into control and communication governing task assignment of the robot, the entire discrete-event dynamic evolution of task sequential process have to be linguistically described in terms of representations. A comprehensive inference framework is required in analyzing the database of rules of expert systems, Petri nets using a large amount of details for building analysis, even for small systems, which lead to high costs.

Acknowledgements. The authors wish to express their gratitude to the Research and Educational Ministry (MEC) and to the Romanian Academy for their support of the program of work reported herein. The work took place as part of research project no. 263/2007-2010 in the framework of Grants of CNCSIS Program IDEAS, PN II (National Program for Scientific Research and Innovation Technologies).

References:

[1] M.J. Stankovski, M.K. Vukobratovic, T.D. Kolemisevska-Gugulovska, A.T. Dinibutun, *Automation System Redesign Using Manipulator for Steel-pipe Production Line*, Proceedings of the 10th Mediterranean Conference on Control and Automation - MED2002 Lisbon, Portugal, July 9-12, 2002.

[2] R. Zahariev, D. Karastoyanov, *A Navigation System and Task Planningin a Mobile Robot for Inspection*, Bulgarian Academy of Science, Sofia, 2004.

[3] C.G. Looney, *Fuzzy Petri nets and applications, Fuzzy Reasoning in Information, Decision and Control Systems*, Kluwer Academic Publisher, pp. 511-527, 1994.

[4] Vladareanu L., Ovidiu I. Sandru, Lucian M. Velea, Hongnian YU, *The Actuators Control in Continuous Flux using the Winer Filters*, Proceedings of Romanian Academy, Series A: Mathematics, Physics, Technical Sciences, Informantion Science, Volume: 10 Issue: 1 Pg.: 81-90, 2009.

[5] X. Li, W. Yu, *Object Oriented Fuzzy Petri Net for Complex Knowledge System Modeling* Proceedings of the 2001 Conference on Control Application Mexico City, Mexico, 2001.

[6] Leopoulos Vrassidas, *Integrated modeling & simulation of customer order process, production order process and discrete production system, for production planning evaluation using Petri Nets*, pp.237-241, *WSEAS Transactions on Computers*, Issue 1, Volume 2, 2003.

[7] Khalil Shihab, *Simulating ATM Switches Using Petri Nets"*, pp.1495-1502, *WSEAS Transactions on Computers*, Issue 11, Volume 4, 2005.

[8] Cassandras, C. and S. Lafortune. *Introduction to Discrete Event Systems.* Kluwer Academic Publishers, 1999.

[9] Youn-gyou Kook, Woon-yong Kim, Young-keun Choi *A Multi-Agent System based on Specific-Profile for Efficient Distributed Computing*, WSEAS TRANSACTIONS on INFORMATION SCIENCE & APPLICATIONS, Issue 4, Volume 2, April 2005.

[10] Abolfazl Jalilvand, Sohrab Khanmohammadi, Fereidoon Shabaninia *Hybrid Modeling and Simulation of a Robotic Manufacturing System Using Timed Petri Nets, WSEAS TRANSACTIONS on SYSTEMS, Issue 5*, Volume 4, May 2005.

[11] Koseck´a, J. *A framework for modeling and verifying visually guided agents: Design, analysis and experiments*, 1996.

[12] Milutinovic, D. and P. Lima (2002). *Petri net models of robotic tasks.* Proceedings of Conference on Robotics and Automation, 2002.

[13] L. Ngalamou, L. Myers Nepad, *Petri Nets and Fuzzy Sets in Hybrid Controllers Synthesis: The Discrete-Event Aspect*, WSEAS Transaction on Systems and Control, Volume 4, Issue 1, 2009.

[14] H. Motameni, A. Movaghar, M. Ebrahimi, S. Peirovi and A. Khosrozadeh Ghomi, *Designing a Software Tool for Evaluating Qualitative Parameters*, World Applied Sciences Journal, 3 (4, 2008.

[15] V. Panaite, M. O., *Popescu, Calitatea produselor şi fiabilitate*, Matrix Rom Editing House, Bucuresti, 2003.

[16] G. Tont, L. Vladareanu, R.A.Munteanu, D.G.Tont, Fuzzy Petri Net-Based Approach in Modelling Simultaneous Task Assignment for Robotic System, ISI Proceedings, WSEAS Recent Advances in Applied Mathematics, Harvard University, Cambridge, USA, 2010, pg. 510-518, ISBN 978-960-474-150-2, ISSN 1790-2769