Diagnosis of Complex Systems Using Ant Colony Decision Petri Nets

Calin Ciufudean, Adrian Graur, Constantin Filote, Cornel Turcu, and Valentin Popa
"Ştefan cel Mare" University of Suceava, 9 University str., RO-720225, Romania
calin@eed.usv.ro, adriang@eed.usv.ro, filote@eed.usv.ro, cturcu@eed.usv.ro, valentin@eed.usv.ro

Abstract

Failure diagnosis in large and complex systems is a critical task. A discrete event system (DES) approach to the problem of failure diagnosis is presented in this paper. A classic solution to solve DES's diagnosis is a stochastic Petri nets. Unfortunately, the solution of a stochastic Petri net is severely restricted by the size of its underlying Markov chain. On the other hand, it has been shown that foraging behavior of ant colonies can give rise to the shortest path, which will reduce the state explosion of stochastic Petri net. Therefore, a new model of stochastic Petri net, based on foraging behavior of real ant colonies is introduced in this paper. This model can contribute to the diagnosis, the performance analysis and design of supervisory control systems.

Key words. *Stochastic Petri nets, discrete-event systems, Ant Colony Optimization algorithm.*

1. Introduction

Diagnosis is a crucial and challenging task in the automatic control of complex systems, e.g., in flexible manufacturing systems. In this paper a discrete event system (DES) approach to the problem of diagnosis of complex system is presented. The property of diagnosability is introduced in the context of the failure diagnosis problem, e.g., in the context of the availability of the DES. We propose a systematic procedure for diagnosis implemented with a new class of stochastic Petri nets (GSPN's) ,i.e., ant colony decision Petri Nets (ADPN. The acceptance of such high-level formalism is due to their ability to represent complex systems in a compact and convenient way, while still describing an underlying continuous-time Markov chain (CTMC) [1]. This method suffers from the well-known state explosion problem: a GSPN can determine an underlying CTMC with a large number of states. This problem severely limits the size of models for which an exact analysis can reasonably be attempted. Stochastic Petri nets (SPN) were developed by associating transitions/places with exponentially distributed random time delays [2]. Generalized SPN [3] allowed the inclusion of immediate transition and inhibitor arcs. These formalisms are all based on results obtained from the underlying Markov chain for such systems models. In [4] Sampath et al. proposed a diagnosis approach for discrete event systems. They introduced the notion of diagnosability and gave a necessary and sufficient condition to test it. Ant Colony Optimisation (ACO) is a recently developed approach that takes inspiration from the behavior of real ant colonies to solve NP - hard optimisation problems. The ACO meta-heuristic was first introduced by Dorigo [5], and was defined by Dorigo, Di Caro and Gambardella [6]. It has been successfully applied to various hard combinatorial optimization problems. In this paper we present the first application of ACO to Petri nets formalism, in order to simplify the models achieved with GSPN for solving the diagnosis of complex systems. In section 2 we briefly introduce the Ant Colony Optimization algorithm. Then we describe the structure of our diagnoser in section 3. In section 4 we present the experimental results implemented on an FMS. Finally, we summarize our findings and conclude with some discussion.

2. Ant Colony Optimization Algorithms

A Bayesian network (BN) is a directed acyclic graph where nodes represent random variables and edges represent conditional dependencies (e.g., probability distributions) between random variables [7]. Although the distributions in a BN can be discrete or continuous, we shall consider discrete ones. Search algorithms have been studied extensively in combinatorial optimization. Researches have applied various search strategies, for example, the best first search [5], linear programming, stochastic local search, genetic algorithms [7], etc. Ant algorithms were inspired by the foraging behavior of real ant colonies, i.e., how ants can find the shortest path between food sources and nest. Ants deposit on the ground a chemical substance called pheromone while walking. This forms pheromone trails through which ants can find the way and, also provides indirect communication among ants. It has been shown experimentally [5] that this foraging behavior can give rise to the emergence of the shortest path when employed by a colony of ants. Based on this ant colony foraging behavior, ACO algorithms using artificial ant systems to solve hard discrete optimization problems have been developed. In an ant system, artificial ants are created to explore the search space simulating real ants searching their environment. The objective values to be optimized usually correspond to the quality of the food and the length of the path to the food. The artificial ants can make use of some local heuristic functions to help choose among a set of feasible solutions. In an ant system, artificial ants build solutions by moving on the Bayesian network from one node to another. When an ant visits node x_i, it must take a conditional branch which is a number in the CPT. For evidence nodes A, ants are only allowed to take the branches that agree with A. Each node in BN has three tables: the Pheromone Table (PT), the Heuristic Function Table (HFT), and the Ant Decision Table (ADT). The PTs store pheromone values accumulated on each conditional branch. HFTs represent heuristics used by ants. ADTs are used by ants to make the final decision which branch to take. The ADT, $A_i = [a_{ijk}]$, of node x_i is obtained by the composition of the local pheromone trail values ph_{ijk} with the local heuristic values h_{ijk} as follows [6]:

$$a_{ijk} = \frac{\left(ph_{ijk}\right)^{\alpha} \cdot \left(h_{ijk}\right)^{\beta}}{\sum_{j} \left(ph_{ijk}\right)^{\alpha} \cdot \left(h_{ijk}\right)^{\beta}} \quad (1)$$

Where j is the j^{th} row and k is the k^{th} column of the corresponding ADT at the i^{th} node. Parameters α and β control the relative weight of pheromone trails and heuristic values. We also know [5], [6] the probability with which an ant chooses to take a certain conditional branch:

$$p_{ij} = \frac{a_{ij\pi_i}}{\sum_{j} a_{ij\pi_i}} \quad (2)$$

where π_i is the column index of the ADT and its value is conditioned on the values of parent nodes of i^{th} node. After ants have built their tour (a diagnosis), each ant deposits pheromone Δph_{ijk} on the corresponding pheromone trails (i.e., the conditioned branches of each node of the tour). For us, the pheromone value represent the probability to cover the selected tour (e.g., by anticipation of the next section, we show that the pheromone value i represent the probability of firing transition i in the SPN), as follows:

$$\Delta ph_{ijk} = \begin{cases} P(x_1, \ldots, x_n), & j = x_i,\ k = \pi(x_i) \\ 0, & otherwise \end{cases} \quad (3)$$

where $P(x_1, \ldots, x_n)$, is:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P\left(x_i / \pi\left(x_i\right)\right) \quad (4)$$

Where, $\pi(x_i)$ denotes the parent nodes of x_i.
Each ant drops pheromone to one cell of each PT at each node, i.e., the j^{th} row, k^{th} column of the PT at i^{th} node. After dropping the pheromone, the ant dies.

3. The Ant Colony Decision Petri Net Diagnoser

In our assumption the diagnoser is a stochastic Petri net (SPN), where the places are marked with the availability of the correspondent production cell. The availability of a production cell is calculated with a Markov chain, where the transitions reflect the gradual importance of the failures in the cell. We may say that the diagnoser is an extended observer where we append to every state estimate a label. The labels attached to the state estimates carry failure information and failures are diagnosed by checking these labels. A diagnoser is a deterministic finite state machine whose transitions correspond to observations and whose states correspond to the set of system states and failures that are consistent with the observations. The transitions of the diagnoser are labelled with observable events, and the states of the diagnoser are labelled with sets of pairs (v,l) denoting a state and a failure label of the abstracted model. In our approach, the diagnoser efficiently maps observations to sets of possible system states and failures, and it is modelled with a new class of Petri nets, called here Ant Colony Decision Petri Nets (ADPN), which are an extension of our previous work [8] where we introduced the Stochastic Coloured Petri Nets (SCPN). Here, the colour of tokens in ADPN, represents the colour of the ants, grouped in families. We suppose that in our model there are different ant families (e.g., red ants, black ants, s.a.), each kind of ant has a specific pheromone; an ant will sense the pheromone in the nodes of the net and will follow only the specific path that was marked with the pheromone of its family. In the initial marking of the Petri net we know the number of the test ants, by colour. Considering that after firing a transition in the net, the ant leaves its pheromone in the control place of the respective transition (see fig. 1), and then dies, after the first ant reaches the end of the graph we count the number of the ants remained in the first place of the net. We conclude which is the shortest way in the net, i.e.,, which family of ants found the optimum path, considering that a family of ants will never follow the same way as another ant family.

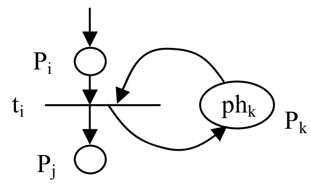

Fig. 1 The basic structure of ADPN

In Fig.1. one can see that the control place, p_k, of transition t_i memorize the pheromone of the ant which burns first the transition t_i. We say that transition t_i will be fired only by ants with colour ph_k, where ph_k has the same signification as that given in relation (1).

The firing rates of transitions in ADPN are given by the next relation:

$$f_i = \frac{(ph_i)^{\alpha_i} \cdot (h_i)^{\beta_i}}{(ph_i)^{\alpha_i} + (h_i)^{\beta_i}} \tag{11}$$

In relation (11) ph_k is the pheromone dropped in the control place by the first ant, that burns the transition t_i; h_i is the classic exponential firing rate of a transition in a stochastic Petri net; probabilities α_i and β_i control the failure rate, respectively the repair rate of elements (machines, electronic devices, etc) of a complex system, such as a Flexible Manufacturing System (FMS). We define our ADPN as follows:

An ADPN is a fire-tuple (P,T,k,m,V), where:

$P = \{p_1, p_2, ..., p_n\}$, $n > 0$, and is a finite set of places;

$T = \{t_1, t_2, ..., t_s\}$, $s > 0$, and is a finite set of transitions with $P \cup T \neq \emptyset$, $P \cap T = \emptyset$;

$K = \{Pk_1, Pk_2, ... Pk_s\}$, $s > 0$, and is a finite set of pheromone - control places;

$m : P \rightarrow N$, and is a marking whose i^{th} component is the number of tokens in the i^{th} place. An initial marking is denoted by m_0;

$V : T \rightarrow R$, is a vector whose component is a firing time delay with an ant decision function.

In our work we assumed that when a device, sensor, transducer or any other hardware component of the analyzed system, (e.g., a FMS) fails, the system reconfiguration (after repairing it) is often less than perfect. The notion of imperfection is called imperfect coverage, and it is defined as probability c that the system successfully reconfigures given that component fault occurs. The imperfect repair of a component implies that when the repair of the failed component is completed it is not "as good as new". A dependability model for diagnosability of flexible manufacturing systems is presented. The meaning of dependability here is twofold:

- System diagnosability and availability
- Dependence of the performance of the FMS on the performance of its individual physical subsystems and components.

The model considers the task-based availability of an FMS, where the system is considered operational as long as its task requirements are satisfied; respectively the system throughput exceeds a given lower bound. We model the FMS with ADPN. We decompose the FMS in productions cells. In our assumption the availability of a cell j (j=1.2…..n, where n is the total number of part type cells in the FMS) is calculated with a Markov chain which includes the failure rates, repair rates, and coverability of the respective devices in the production cell i. The colour domains of transitions that load cell i include colours that result in a value between 0 and 1, and the biggest value designates the cell (respectively the place in the ADPN model) which ensures the liveness of the net, respectively which will validate and burn its output transition. We assume that the reader is familiar with Petri nets theory and their applications to manufacturing systems or we refer the reader to [7]. Each part entering the system is represented by a token. The colour of the token associated with a part has two components [8]. The first component is the part identification number and the second component represents the set of possible next operations determined by the process plan of the part. It is the second component that is recognized by the stochastic colours Petri net model, and the first component is used for part tracking and reference purposes. Let B_i be a (1xm) binary vector representing all the operations needed for the complete processing of part type i. Let E_i be a (mxm) matrix representing the precedence relations among the operations of part type i, where m is the number of operations that are performed in the respective cell j (j=1.2…., n). For a part to be processed in the cell j it requires at least one operation that can be performed in the cell, that implies $B_j > 0$. Also, for a part type where there is no precedent relationship between required operations, E_i is a matrix of zeros. For a part with identification x and part type y, the initial colour of the corresponding token is:

$$V_{yx} = \lfloor yx, B_y - (B_y \cdot E_y) \rfloor \tag{12}$$

Where $\left(B_y \cdot E_y\right)$ is a matrix of multiplication.

For example consider the process plan of part type L_1 and L_2 shown in Fig.2

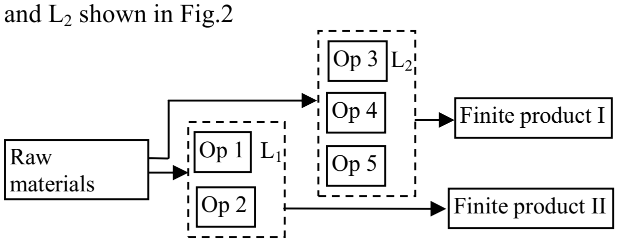

Fig.2. Process plan of part type L_1 and L_2

Our process plan first requires operation op1 and then operation op2 for complete processing. We assume that our FMS can complete 5 different types of operations (e.g., for simplicity we consider only 5 different types of operations). For part type L_1, we have: B_{L1} = [00011].

$$E_{L_1} = \begin{array}{c} \\ op5 \\ op4 \\ op3 \\ op2 \\ op1 \end{array} \begin{pmatrix} op5 & op4 & op3 & op2 & op1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & A_2 \\ 0 & 0 & 0 & A_1 & 0 \end{pmatrix}$$

Where A_1 is the availability of production cell 1 (which performs operation 1), and A_2 represents the availability of production cell 2 at time t. The availability A_i of cell i is calculated, as shown below, with Markov chains. We notice that A_i is re-evaluated at each major change in the process plan of FMS (such as occurrence of events: damages of hardware equipments, changes of process plan, etc). Assuming that $A_1 > A_2$, then we assign to A_1 value 1 and to A_2 value 0, so that applying relation (12), the initial color of the token corresponding to a part that belongs to part type L_1 with identification mark 1, would be $V_{L1.1}$ = ($L_{1.1}$, 00001). Note that the information carried by the color of the tokens in the SCPN indicates the next operation to be performed by the FMS. Generally, we may say that V is the set of colors that represent all the possible combinations of operations that can be performed in the FMS. Each member of the set V is a vector with m components, where m is the maximum number of operations to be performed in the cells of the FMS. For example, in an FMS with 5 operations to be performed, we may have V = {00000, 00001, … 11111}. For simplicity, we assume that operations in FMS are maped to places in the SCPN model, places which are labeled with the operation identification number. The requirement for a production cell j (j=1, …, n) which have N_i (i=1, …,m) devices of type i, is that at least k_i of these devices must be operational for the FMS to be operational. To determine the system availability which includes imperfect coverage and repair, a failure state due to imperfect coverage and repair was introduced [4]. To explain the impact of imperfect coverage, we consider the system given in Fig.3 which includes two identical manufacturing devices M_1 and M_2.

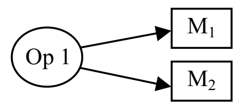

Fig.3. Example of operation performed by two identical devices

If the coverage of the system is perfect, i.e. c=1, then operation op1 is performed as long as one of the devices is operational. If the coverage is imperfect, then operation op 1 fails with probability 1-c, if one of the devices M_1 or M_2 fails. We may say that, if operation op 1 has been scheduled on device M_1 that has failed, then the system in Fig.3 fails with probability 1-c.

The Markov chain for manufacturing cell j is shown in Fig.4. In Fig.4 the parameters λ, μ, c, r denote respectively the failure rate, repair rate, coverage factor and the successful failure repair rate of devices in the cell. The first part of the horizontal transition rate with the term 1-c represents the failure due to imperfect coverage of an alternative equipment. The second part, with the term 1-r represents imprecise repair of the devices. The vertical transitions reflect the failure and repair of the equipments. We assume that only one device fails at a time, in a certain operation cell. At state N_i cell i is functioning with all N_i devices operational. At state k_i there are only k_i devices oparational. The state of cell i changes from working state w_i, for $k_i \leq w_i$ N_i, where w_i is the number of operational devices at a certain moment, to failed state F_i, either due to imperfect coverage (1-c) or due to imperfect repair (1-r). If the fault coverage of the system and repair of the components are perfect, the Markov chain in Fig.4 reduces to one-dimension model. The solution of the Markov chain model given in Fig.4 is a probability that at least k_i devices are working at time t.

The status of this graph (e.g., the ADPN) at different moments t_k, gives us the diagnosis of the FMS.

IEEE COMPUTER SOCIETY

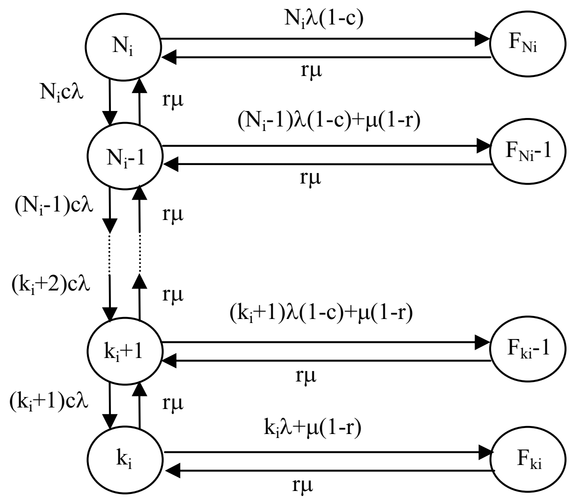

Fig.4. Markov model for cell i

The availability of cell i is given by the next relation [8]:

$$A_i(t) = \sum_{w_i=k_i}^{N_i} P_{ki}(t), \text{ for i=1,2,…,n} \qquad (13)$$

Where $A_i(t)$=the availability of cell i at moment t;
$P_{ki}(t)$=probability of k_i devices being operational in cell I at time t;
N_i=total number of devices of type j in cell i;
K_i=required minimum number of operational devices in cell i.
After a Markov chain for each cell of the measuring system is constructed and desired probabilities $A_i(t)$, i=1,2,…,n corresponding to each manufacturing cell are determined, the ant colony decision Petri net (ADPN) can be initialized and the simulation process of the FMS begins.

4. Illustrative example

In this section, we exemplify the above presented approaches on a flexible manufacturing system. We give the relative error in aggregated measures, such as the mean number of tokens in a given place or a throughput of transitions. Markov chain was solved using Gauss-Seidel with iterations continuing until the relative element-wise difference between subsequent probability vectors was less than 10^{-6}. The flexible manufacturing system consists of two cells linked together by a conveyor system. Each cell consists of a machine to handle within-cell part movement. Work-pieces enter the system at the Load/Unload station, where they are released from two buffers, *A* and *B*, and then are sorted in cells (pieces of type "*a*" in one cell, and pieces of type "*b*" in the other cell). We notice that in the buffer *A* there are pieces of types "*a*", "*b*", and others. In buffer *A* the number of pieces "*a*" is greater than the number of pieces "*b*". In the buffer *B* are pieces of types "*a*", "*b*", and others, where the number of pieces "*b*" is greater than the number of pieces "*a*". The conveyor moves pieces between the Load/Unload station and those two cells. The finished (sorted) work-piece leaves the system, and a raw work-piece (unsorted piece) enters the system, respectively in one of those two buffers *A* or *B*. The maximum number of work-pieces permitted inside a cell at any given time is limited, owing to the finite storage capacity available within the cell. The conveyor along with the central storage incorporates a sufficiently large buffer space so that it can be thought of as possessing infinite storage capacity. Thus, if a work-piece routed to a particular cell finds that the cell is full, it refuses entry and it is routed back to the centralized storage area. If a work-piece routed by the conveyor is different from the required types to be sorted respectively, "*a*" and "*b*", it is rejected. We notice that once a work-piece is blocked from entry to a cell, the conveyor does not stop service; instead it proceeds to the other work-pieces waiting for transport. We also assume that within a cell no further blocking is caused once a work-piece is admitted. At the system level, we assume that the cells are functionally equivalent, so that each cell is sufficient to maintain production (at a reduced throughput). We say the manufacturing system is available (or operational) if the conveyor and at least one of the cells are available. A cell is available if its machine is available. Over a specified period of operation, owing to the randomly occurring subsystem failures and subsequent repairs, the cellular automated manufacturing system will function in different configurations and exhibit various levels of performance over the random residence times in these configurations. Table 1 gives the interpretation of the places and transitions of the ADPN model build with the previously presented algorithm, is given in Fig.5. In the initial marking, we have *n* tokens in P_{HC}, and k_i, i = 1,2 tokens in each $P_{Free\ Buffers\ i}$. We notice that places $P_{Buffer\ Ai}$ and $P_{Buffer\ Bi}$ allow the admission of the work – pieces from the corresponding buffers (*A*, and respectively *B*) in the moments when it is needed to sort more work-pieces from one category or from another one (*A* or *B*).We mention that the processing time (part moving time in the case of the buffers A and B) at machine *i* is exponentially distributed with mean $1/\mu_i$, i = 1, 2. In all cases, the part moving time on the conveyor is assumed to be exponentially distributed with mean $1/\mu_0$.

The queuing model of the system is produced under the First Come First Served queuing assumption, in spite of the blocking phenomena present in the model (if the cell capacity is fully occupied, the work-piece is blocked and re-enters in buffer *A* or *B*).

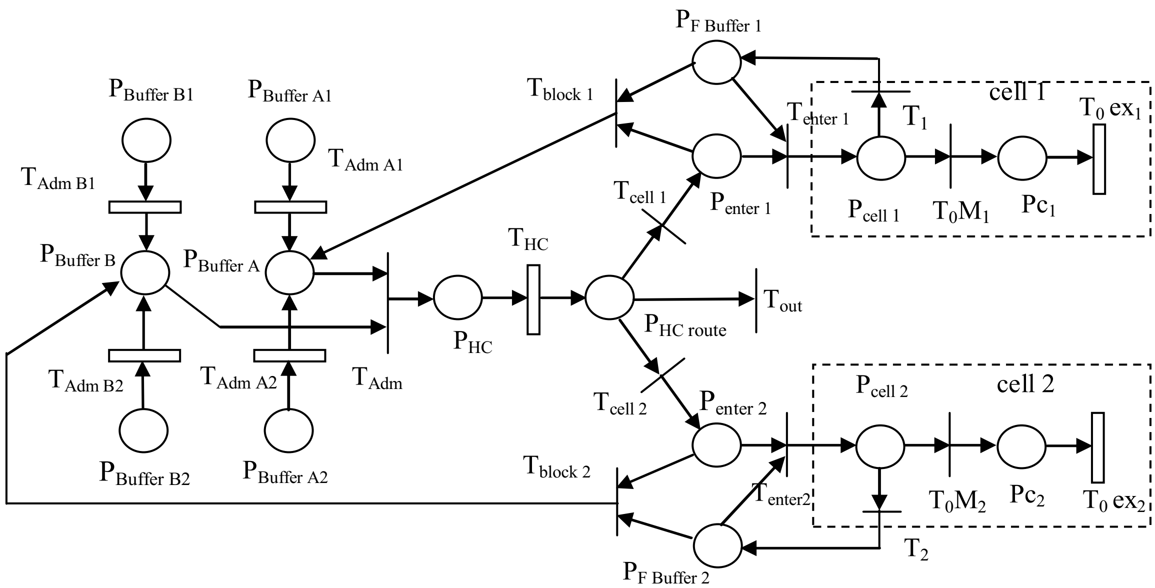

Fig.5. Petri net model for performance analysis of the FMS

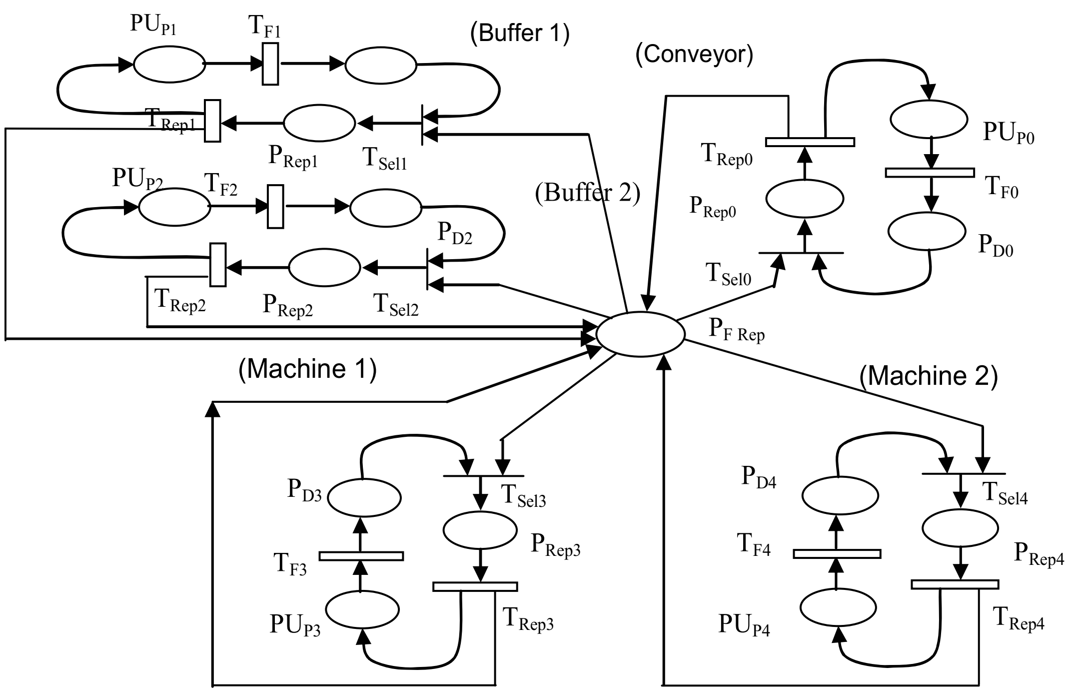

Fig.6. Petri net model of the diagnoser of the FMS

0-7695-2567-9/06 $20.00 © 2006 IEEE

Table 1 Legend for the Petri net model of FMS

Symbol	Places
P_{HC}	Work-pieces being admitted to the conveyor
$P_{HC\ route}$	Work-pieces being routed by conveyor
$P_{F\ Buffer\ i}$	Free buffer spaces in cell *i*
$P_{enter\ i}$	Work-piece attempting to enter cell *i*
$P_{cell\ i}$	A work-piece in cell *i* is being sorted by machine *Mi*
P_{ci}	Work-piece waiting to leave the cell *i*
$P_{Buffer\ A,B}$	Work-pieces in buffer *A*, respectively *B* waiting for admission to conveyor
$P_{Buffer\ Ai,Bi}$	A work-piece of type *a*, respectively *b* is expected to be sorted by machine *i*.
Symbol	**Transitions**
T_{HC}	Timed; firing rate (no. of tokens in P_{HC}) x μ_0; models the infinite server operation of conveyor
$T_{adm\ Ai,Bi}$	Timed; firing rate (number of sorted work-pieces *A* or *B*) x μ_0.
T_{adm}	Immediate; a work-piece is admitted (released) to conveyor
$T_{Black\ i}$	Immediate; a work-piece trying to enter cell *i* is blocked
$T_{cell\ i}$	Immediate; a work-piece is routed to cell *i* by machine *i*
$T_{enter\ i}$	Immediate; a work-piece is allowed into cell *i*
$T_{o\ Mi}$	Immediate; a work-piece is routed by machine *i*
T_i	Immediate a work-piece is sorted in cell *i*
$T_{o\ ex\ i}$	Timed; firing rate μ_i; a sorted work-piece is routed to exit the system
T_{out}	Immediate; a piece is rejected from the system

4.1 A Petri net model of the availability of the AMS

We assume an independent failure model. Let α_i, i=1,2 denote the mean failure rate of machine M_i and let its mean repair time be $1/\beta_i$. Also, let α_0 and β_0 denote respectively the failure and repair rate of the conveyor and buffers *A* and *B* material handling system seen as a whole. A team of technicians is available to work on the failed subsystem at a time. A failed subsystem can use the services of only one technician. Fig.6. shows the Petri net model of the availability of FMS; it models the failure and repair of the different subsystems. Table 2 gives the interpretation of the places and transitions of this Petri net. Initially, all the subsystems are assumed to be functioning and all technicians are idling. Thus, in the initial marking, a token is placed in each of the places PU_{pj}, j=0,...,4; the number of tokens in the place $P_{F\ Rep}$ equals N_R (i.e., the number of technicians in the system). Evaluation of the performance and availability of the FMS is obtained by calculating the mean cycle time of the Petri net showed in Fig.5, respectively of the Petri net showed in Fig.6, using the algorithm given in the previous section.

Table 2 Legend for the Petri net model of the availability of the FMS

Symbol	Places
PU_{pi}	Subsystem *i* is working
P_{Di}	Subsystem *i* has failed, and is awaiting repair
$P_{Rep\ i}$	Subsystem *i* under repair
$P_{F\ Rep}$	Idle repairmen
Symbol	**Transitions**
T_{Fi}	Timed; models breakdown of subsystem *i*
$T_{Rep\ i}$	Timed; models repair of subsystem *i*
$T_{Sel\ i}$	Immediate; failed subsystem *i* taken up for repair

The mean times obtained from the related graph gives us solutions for the analysis of the FMS over a finite time horizon. From Petri net showed in Fig.6 we calculate the probability of obtaining uptime of "u" hours during a finite interval of time containing a multiple of eight – hours shifts with one or more technicians. From Petri net showed in Fig.5 we calculate the performability production index such as: expected cumulative production, coefficient of variation of cumulative production (i.e., standard deviation divided by mean), etc. For example, our system has the following conditions: cell 1 can permit at most 15 work-pieces to be present at a time (k_1 = 15). Cell 2 allows a maximum of 16 work-pieces (k_2 = 16), $\alpha_1 = \alpha_2 = ½\alpha_0 = ½$ (failure rate, i.e., firing rate of corresponding transitions, of machine M_i and conveyor), $\beta_1 = \beta_2 = ½\beta_0 = ½$ (repair rate, i.e., firing rate of corresponding transitions, of machine M_i and conveyor). In such conditions, the expected cumulative production over a 96-hour period, over a six-day week, working two eight-hours shifts a day, with one, two, and three technicians, will increase with the number of technicians employed. The use of more than three technicians has not increased the cumulative production. Table 3 reports the minimum and maximum number of Gauss-Seidel iterations required by the 16 CTMCs. The maximum observed relative error on the throughput of transitions was less than 4% for n > 8. Table 4 gives size of the state space, the range of continuous Markov chain sizes CTMC (we notice that since in Fig.6, there are 16 CTMCs, we only list the size of the smallest and the largest ones) as a function of n, the number of jobs in the system.

Table 4 Iterations in the CTMC for the FMS, n = 8

fixed pt. iterations	GSD iterations (min)	GSD iterations (max)
1	40	2,100
2	24	565
8	22	214
20	18	121
40	12	73
60	1	27
110	1	1

Table 3 State space in the CTMC for the FMS in Fig. 3

n	min state space M_k	max state space M_k
5	6	683
8	14	7,194
12	20	68,348
16	28	91,775

5. Conclusions

In this paper we have proposed a new architecture for the diagnosis of complex systems such as FMS, using a special class of stochastic coloured Petri nets entitled here ant colony decision Petri nets (ADPN).

The advantages of this approach are:

Alternate sequencing of operations is allowed during processing; device assignments for operations are made dynamically during processing; the model of FMS captures all operation sequences in the system, as exemplified above for a part type, e.g. L_1, of the process plan.

An analytical technique for the availability evaluation of FMS was also presented in this paper. The advantages of this approach are:

The construction of large Markov chains is not necessary; it reveals when the system coverage and the component repair are not satisfactory; it allows determination of the timing of a major repair of the system.

The disadvantages of this approach are:

It necessitates qualified personnel for modelling properly the production plan and the coverage of the system; the calculus of availability necessitates performing computers because complex FMS`s require large Markov chains. One may mention that in FMS study usually it is assumed that the failure and repair times of the machines are exponential random variables, but in industrial systems the time distributions are often arbitrary and semi-Markov processes can be handled. Therefore, failure and repair times are no longer constrained to be exponentially distributed. Our approach deals with Markov chains and with semi-Markov chains. If we compare it with other approaches, one may see that the above-proposed model is much more versatile, and it can be easily simulated in Matlab 6 support. Others classic software dedicated for the study of Markov chains like SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator) [9], and HARP (Hybrid Automated Reliability Predictor) [10] implement more expensive simulations than ours, where deviations of availabilities calculated with relation (13) are less than 4%. We mention that classic Markov chain models for FMS`s do not include the concept of coverability which explains the mentioned differences between availabilities. The model presented in this paper can be extended to include other components, e.g. hardware equipments, throughput systems, etc. However, increasing the number of subsystems will increase the number of Markov chains and transitions sub-matrices, which implies inherent difficulties in simulation of the system, often without significant differences between the initial simplified model and the new, augmented one. This work establishes a necessary relation between diagnosis and availability of manufacturing system. Further research will focus on modelling FMS with semi-Markov processes.

References

[1] P. Bucholz, “Numerical solutions methods based on structured description of Markovian models”, In *Computer performance evaluation*, pp. 251-267, *Elsevior Science Publishers B.V.,* 1991

[2] P. Bucholz, G. Ciardo, S. Donatelli, P. Kemper, “Complexity of memory-efficient Kroneker operations with applications to the solutions of the Markov models*”, Informs J. Comp.,* no. 12(3), pp. 203-222, Summer 2000.

[3] M. K. Malloy, “Performance analysis using stochastic Petri nets*”, IEEE Trans. Comput*., vol. C-3, no. 9, pp. 913-917, 1992

[4] M. Sampath, et. Al., “Diagnosability of discrete-event systems”, *IEEE Trans. Automat. Contr*., vol. 40, pp. 1555-1575, Sept. 1995

[5] M. Dorigo, “Optimization, Learning and Natural Algorithms”, Ph. D. Thesis, Polytechnico di Milano, Italy, 1992

[6] M. Dorigo, G. Di Caro, L. Gambardella, “Ant Algorithms for Discrete Optimization”, *Artificial Life*, no. 5(2), pp. 137-172, 1999

[7] S. Russell, P. Norvig, Artificial Intelligence : A Modern Approach, N. J., 2003

[8] C. Ciufudean, s.a., “Scheduling Diagnosis of Flexible Manufacturing Systems”, In *Proc. Int. Conf. on Autom, Contr. And Syst. Eng. (ACSE'05),* Cairo, Egispt, Dec. 2005

[9] R.Sahner, K.Trivedi, “Reliability modelling using SHARPE”, *IEEE Trans. Reliab*., vol.36, pp.186-193, 1997

[10] S.Bavuso, J.Dugan, K.Trivedi, E.Rothmaun, W.Smith, “Analysis of typical fault-tolerant architecture using HARP”, *IEEE Trans. Reliab.,* vol.36,pp.176-185,1997.

0-7695-2567-9/06 $20.00 © 2006 IEEE

IEEE COMPUTER SOCIETY