Proceedings of the IASTED International Conference
COMMUNICATION SYSTEMS AND NETWORKS
September 9-12, 2002, Malaga, Spain

AUTOMATED TROUBLESHOOTING OF MOBILE NETWORKS USING BAYESIAN NETWORKS

R.Barco[1] , R.Guerrero[2], G.Hylander[2], L.Nielsen[3], M.Partanen[2], S.Patel[4]
[1]Dpt. Ingeniería de Comunicaciones. Universidad de Málaga.
[2]Nokia Networks (Málaga, Spain)
[3]Nokia Networks (Aalborg, Denmark)
[4]Nokia Networks (Cambridge, UK)
C/Severo Ochoa s/n. Edif.Inst. Universitarios, Pl.3, Parque Tecnológico de Anadalucía, Málaga (Spain)

Abstract

In the current telecommunication scenarios operators have to cope with fast technological changes while increasing operational efficiency, i.e. diminishing operational expenditures and, at the same time, maximising performance of the networks. In this paper we present an automated troubleshooting tool for cellular networks, based on Bayesian networks, which will contribute to improve operational efficiency. We propose some Bayesian models for diagnosis in mobile networks and we present a troubleshooting tool, which uses those models to diagnose the cause of problems. A knowledge acquisition tool is also presented, which converts the knowledge of troubleshooting experts into Bayesian models by means of a friendly user interface. The models and tools have been tested in real mobile networks and some results and conclusions are also outlined in this paper.

Key Words: Troubleshooting, diagnosis, Bayesian networks, mobile

1. Introduction

Mobile networks will face deep changes in the coming years, due to the introduction of new technologies, new services, and increased number of subscribers. In the past, operators managed to cope with those changes and the network growth by increasing their personnel. However, this is not a feasible strategy anymore. Furthermore, current expenditures for most operators are mainly focused on operation and maintenance, more than on customer service or on investing in new equipment. Therefore, a viable option to maintain network quality with the existing workforce, whilst integrating new technology at the same time, is to increase the level of automation.

Troubleshooting is part of the network operation and maintenance process. If a cell is temporarily non-operational due to a fault, probably neighbouring cells will also be affected and the final result will be that the whole network performance will be decreased. Therefore, it is crucial to ensure that cells are rapidly brought back into operation.

Currently, troubleshooting is mainly a manual process, in which the person looking into the reasons for the problem has to carry out a series of checks in order to establish the causes of the problem. In this process, several applications and databases have to be queried in order to analyse performance data, cell configuration and hardware alarms. The growing size of cellular networks, together with the increasing complexity of the network elements, creates a need for automated troubleshooting. In order to fulfil this requirement, we have developed a prototype troubleshooting method and tool, which is currently undergoing trials and testing.

The troubleshooting tool automatically collects the required information from the data sources and reasons with this information related to the faulty cell in order to generate a diagnosis of the cause(s) of the problem(s).

Thanks to the automated troubleshooting tool, highly experienced staff can be released from mundane daily troubleshooting tasks and can concentrate on other aspects of network optimisation, thus increasing network performance. One additional benefit is that knowledge from different experts can be stored within the tool, therefore making expert troubleshooting knowledge available to the business at all times.

In this paper we present an application of artificial intelligence to automated troubleshooting of cellular networks. First, we present the current manual troubleshooting procedures and we introduce the required elements for automatic troubleshooting. The proposed solution for automated troubleshooting is based on Bayesian networks, addressed in Section 3.

In Section 4 we propose certain types of Bayesian models with certain structural properties. The main reason for investigating several structures is a trade-off between diagnosis accuracy and model complexity.

We have developed two prototypes: A troubleshooting tool, which is in charge of diagnosing the most probable cause of problems based on automatically collected evidences and Bayesian models; and a knowledge acquisition tool, which converts the knowledge from troubleshooting experts into Bayesian models (which are used by the troubleshooting tool).

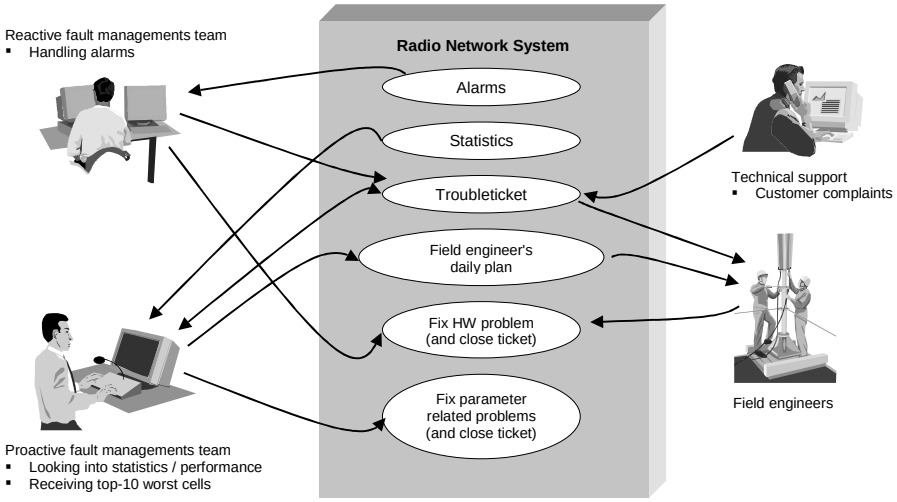

Fig.1. Manual troubleshooting

Both tools have been tested in real cellular networks and the results of the trials are presented in Section 7.

Finally, Section 8 summarises our contributions and discusses future work.

2. Operational Scenario

2.1. Manual troubleshooting

The current scenario for troubleshooting in most operators' networks is shown in Fig.1. In the figure, the main elements involved in troubleshooting can be observed:

- The reactive fault management team is responsible for dealing with alarms generated on the network. They filter the important alarms and raise *troubletickets*, which reflect the problem status. Sometimes, they solve the problem themselves and sometimes they leave the troubleticket for further investigation.
- The proactive fault management team finds poorly performing cells, based mainly on short-term statistics. This team often uses scripts to generate a list of "worst performing BTSs" and takes it as a starting point for their work. Then, they look further into troubletickets raised by the reactive team and raise new troubletickets. They can solve parameter related problems, but they will need to involve field engineers for problems related to HW on the site.
- Field engineers travel to the BTS sites and fix HW problems and any other problem requiring on-site personnel. The field engineers receive a new daily plan every morning containing the list of sites they must visit.
- Finally, a minor part of troubletickets are raised by technical support teams who may receive customer complaints from call centers, management, engineering staff, etc.

2.2. Automated troubleshooting

Automatic troubleshooting improves efficiency of network operation personnel, quickly identifying causes of problems and proposing solutions [1]. Furthermore, it could be integrated with the management and troubleticket system. The scenario for automated troubleshooting is shown in Fig.2.

Automated troubleshooting consists of three steps:

- Fault detection: automatic detection of bad performing cells based on performance indicators, alarms, etc.
- Diagnosis or Cause identification: automatic reasoning mechanisms to identify the cause of the problems and the best sequence of actions to solve them
- Problem solving: executions of the action to solve the problems

Hereafter when speaking about troubleshooting it will be understood that we refer to cause identification, keeping in mind that troubleshooting has a wider scope. Fig.2 explains the automated troubleshooting procedure. First, the knowledge of the troubleshooters has to be transferred into a model for the system by using a knowledge acquisition tool (KAT). The Fault Detection (FD) subsystem indicates which are the cells with problems and the kind of problem. The Troubleshooting

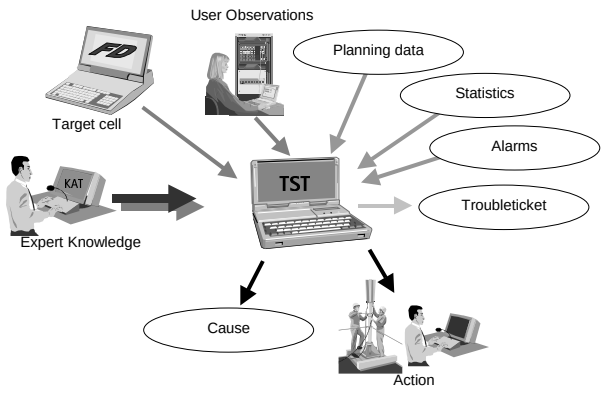

Fig.2. Automated troubleshooting

Tool (TST) chooses the proper model and makes the reasoning based on user observations (e.g. antenna down tilt), configuration data (e.g. frequency reuse setup), statistics from databases (e.g. measured interference levels) and hardware alarms. The output of the troubleshooting tool is a list of possible causes with a probability associated to each cause. The tool also recommends the most cost efficient action to solve the problem. E.g. even if a HW problem is a more probable than a configuration problem, the troubleshooting tool will recommend to change a parameter or to reset the equipment before sending someone to the site in order to check if there is a HW problem. This is due to the fact that the first option is less expensive (time/cost) than sending someone to the site.

3. Bayesian Networks

An expert system simulates the human way of reasoning to infer conclusions from some available information. The automated troubleshooting tool is a decision support system, which is an expert system that rather than trying to completely replace experts, provides support for both experienced and less experienced personnel. Several expert systems techniques have been developed over the last decades, the simpler one being rule-based expert systems [2].

The chosen technique for troubleshooting mobile networks has been Bayesian networks [3], which has several advantages when compared to rule-based systems. Bayesian networks are probabilistic representations for uncertain relations, which have been successfully applied to real-world problems, as diagnosis of medical diseases [4] and troubleshooting of printers [5].

In a Bayesian network, the domain is modelled by means of nodes or variables connected with arrows, which represent causal relations between the nodes. The variables can be continuous or discrete, having a number of exclusive states. Probabilities are incorporated to the model as the "strength" of the connecting arrows.

One important advantage of Bayesian networks is that it has been proved that they are superior to other techniques when dealing with uncertainty within domains. Troubleshooting is an area in which Bayesian networks fit perfectly, as the relation between possible causes of problems and their corresponding symptoms are not deterministic.

There are known algorithms to efficiently infer knowledge from evidences, i.e. knowing the state of some variables, obtaining the probability of each state of other unobserved variables. In that way, knowing some symptoms of a given disease or cause of problems in a mobile network, it is possible to deduce the probability of the possible diseases or causes.

A bottleneck in Bayesian networks is knowledge acquisition, i.e. converting the domain knowledge of experts into Bayesian models. The structure of the Bayesian network must be defined, i.e. the relations between the nodes, and even if some simplifications are made about the structure, still the number of probabilities that has to be specified is usually high. Furthermore, experts in troubleshooting normally do not know how to build a Bayesian model directly. The results of the automated troubleshooting tool depend on how well it was "taught" by the experts. Therefore, having a semi-automated knowledge acquisition tool to help the user seems to be an essential requirement.

Another key aspect of Bayesian networks is that they can learn from experience, e.g. from a database with previous cases, and continuously adapt to changes in the domain. In that case, it is not needed that experts specify all probabilities with a high accuracy.

4. Models for mobile networks

A Bayesian model that can be applied to diagnosis in mobile networks is shown in Fig.3. It consists of three types of nodes:

- Causes: they represent the possible faults that may be causing problems in the network (e.g. HW problem, interference in downlink, etc.)
- Symptoms: they represent manifestations of the causes (e.g. signal level decreased, increase number of HOs, etc.)
- Conditions: they represent factors that can have an impact on the causes (e.g. *cell density*: load problems are more likely in densely populated) or on some symptoms (e.g. the average number of HOs is also different depending on the cell density)

In the example in Fig.3 there are two conditions (*Frequency reuse*, *Cell density*), which have an impact on the causes (*Interference in DL*, *Coverage*) and on some symptoms (*Downlink Quality HOs*, *Uplink Quality HOs*). Furthermore, some symptoms may be combination of other symptoms (e.g. *Downlink or Uplink Quality HOs*).

Once the structure of the model is defined, the states of the nodes should be specified and the probability tables should be filled in. When a node has several parents this becomes a cumbersome task because probabilities for each combination of the parent nodes have to be set. In

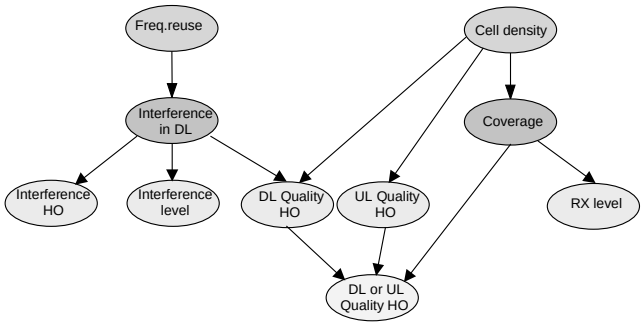

Fig.3. Example of Bayesian network for troubleshooting

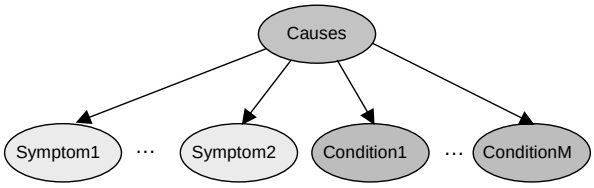

Fig.4. Naïve Bayes model

order to simplify the knowledge acquisition two models are proposed, which consider some assumptions in order to reduce the size of the probability tables: naïve model and causal independence models.

The *naïve model* shown in Fig. 4 has been extensively used in many diagnostic systems. When this model was used in the medical domain, the parent node represented a set of alternative diseases and the children were potential symptoms of the diseases. In the case of troubleshooting mobile networks, the parent stands for the possible causes of problems in the network, whereas the children are the symptoms and conditions. The naïve model has some implicit restrictions: first, single fault assumption, i.e. it supposes that only one cause is present at the same time and, second, the children (symptoms and conditions) are considered to be independent given that the cause is known.

Causal Independence models overcome the limitations of the naïve model [6]. One particular model is Noisy-OR [7], which assumes that each cause C_i will bring about a related symptom S_i to happen unless an "inhibitor" prevents it. While Noisy-Or requires that the causes and symptoms are binary variables, other causal independence models (e.h. Noisy-MAX, Noisy-ADD, etc.) allows any number of states.

The use of causal independence leads to simplifications in probability assessment and inference. For example, in Fig.5, if the probability table of symptom S is built using causal independence assumptions, the number of probabilities to be assessed is linear in relation to the number of parents, instead of exponential.

5. Troubleshooting Tool

An automated troubleshooting tool (TST) has been developed in order to validate the previous concepts. The tool reasons in order to diagnose the cause of the problem(s) in the network, e.g. high number of dropped calls. The conclusions of the TST are based on the following data, which should be introduced to the system before starting the automated troubleshooting:

- Fault cases, e.g. congestion, high drop call-rate, etc., which determines the Bayesian model to use.

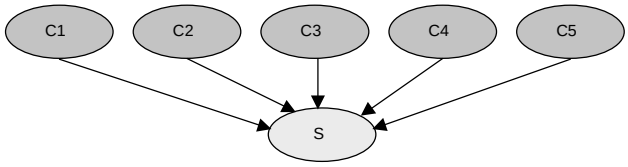

Fig.5. Bayes model where causal independence can be applied to build the probability table of node S

- Cells to be analysed (e.g. top-10 bad performing cells according to statistics collected). Normally this will be automatically fed by the Fault Detection system.
- Dates of the analysis together with the averaging method used to calculate the performance statistics (busy hour, 24 hours data…).

When the previous information is entered into the tool (Fig.6), the user will be asked those data that cannot be automatically collected from the data sources (e.g. weather condition, cell density…).

Next step is to run the analysis or schedule it, for example run TST during early hours of the morning so that analysis is ready when engineers come in the morning.

During the analysis the TST collects data from all sources specified in the model (databases, alarms, etc.). When the data collection finishes, the TST performs the reasoning, based on the selected Bayesian model, and using an inference engine to calculate the probability of each possible cause being the one causing the problem(s).

The conclusions of the troubleshooting tool are displayed when the analysis is finished (Fig.7). The TST presents, for each cell, the list of possible causes of the problem ranked by their efficiency (function of the probability and the cost of the action required to solve the problem).

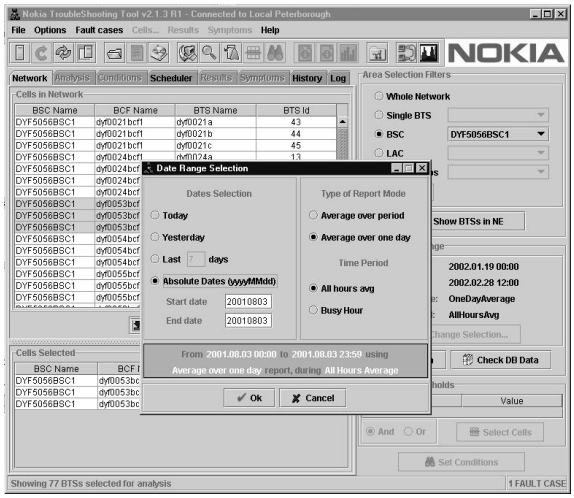

Fig.6. Selection of Target Cells and Dates Options

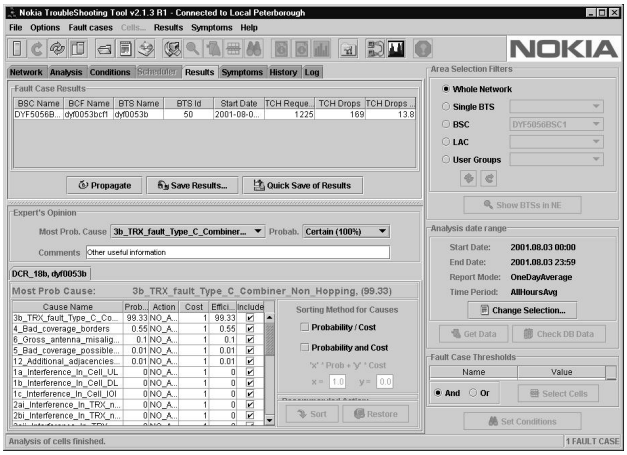

Fig.7. Results of the analysis

All the collected evidence related to symptoms can be displayed or saved locally in text format. The user can also save his own feedback on the problem cause together with the results of the analysis. This information can be utilised by the expert trouble-shooter to verify if the solution provided by the tool was the right one.

6. Knowledge Acquisition Tool

The performance of the troubleshooting tool will depend on the knowledge of the system, i.e. the knowledge of the troubleshooter experts should be precisely transferred to the tool in the shape of Bayesian networks. Normally a knowledge engineer is required to convert the knowledge of the expert into Bayesian models, which should include determining the important variables to consider, states of these variables, relations among the variables, probabilities, etc.

The Knowledge Acquisition Tool (KAT) plays the role of a knowledge engineer by guiding the troubleshooter in creating the network. In order to simplify the knowledge acquisition, some assumptions about the structure of the model are considered (see models proposed in section 4). These assumptions will allow the user to insert a minimal quantity of information that later on will be used by KAT as a seed to automatically complete all the information needed to build the Bayesian network.

The steps for the creation of a network are:

- Definition of causes, symptoms, conditions and actions. They can be reused in different networks. In the case of symptoms and conditions the user has to establish which script (e.g. look up in a database) will be used to collect the data. KAT provides a wide range of scripts grouped by category e.g handovers, interference, congestion, etc.
- Selection of a fault case e.g high drop call rate, and situations that can cause it e.g. Abis problems, bad coverage, etc.
- Selection of symptoms and conditions for each cause in the model.
- Specification of probabilities for causes, symptoms and conditions, including probability of each link among them (conditional probabilities).

KAT will guide the expert through the previous steps and will inform him if there are any inconsistencies in the data (Fig.8). Based on the previous information, KAT will automatically create a Bayesian model (naïve or conditional independence model) that will be used by the Troubleshooting Tool.

When dealing with troubleshooting experts, it has been realised that specifying probabilities for a given structure is quite difficult and different experts can provide completely different values, which can lead to inaccurate results. Normally, operators have system databases containing the history of most variables in the network, which can be used to train the Bayesian network and obtain the probabilities based on those previous cases.

7. Results

The development of TST has been carried out in co-operation with an operator. Testing the accuracy of the model, features of Bayesian networks and the functionalities of the prototype have been performed within a real network environment. There was direct input from expert troubleshooters into the development for both the modelling and the tools functionalities. The fault scenario selected for testing and modelling is built upon troubleshooting cells having high drop call rate. Accuracy and hitrate of solutions are determined from performing manual troubleshooting and comparing with results from TST for the same cells and conditions affecting those cells. An iterative approach to improving the accuracy and hitrate was regarded suitable: the main improvements in the TST hitrate have come from direct fine tuning of the model by the expert troubleshooters.

The TST's performance was derived from recorded cases where the manual analysis was performed and a solution for the majority of the problem cases found. There existed cases where commonly, site resets were

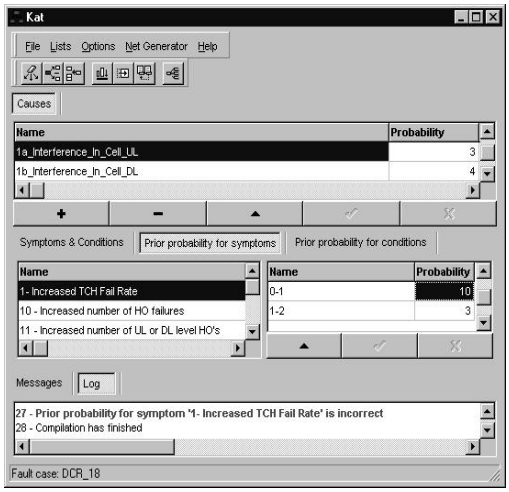

Fig.8. KAT main window

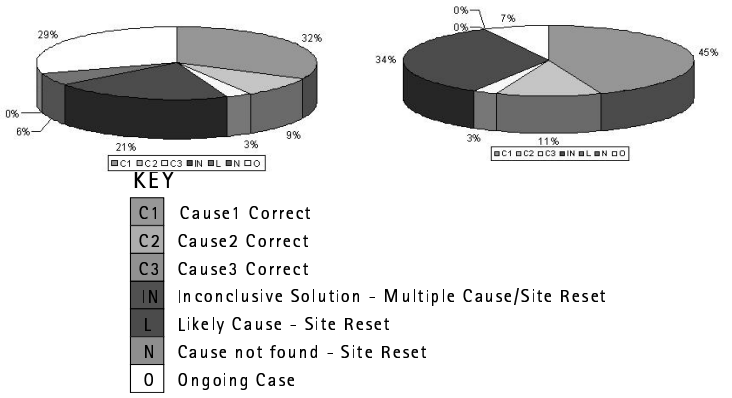

Fig.9. TST hit rate

performed to alleviate the problem and this rendered in-depth investigation impossible as normal service was resumed. The other problem cases were fed into TST and its results stored for analysis and comparison. The resulting hitrate as shown in Figure 9 shows performance %'s over two separate periods, a few weeks apart during testing in 2001.

The results showed that in first tests the output was often a false diagnosis, but the correct cause usually scored quite well, and the highest ranked cause usually made some sense based on the inputs. We also observed that the domain expert could make little changes based on the first trials to make the diagnosis output much more accurate. In a few weeks we got a "hitrate" of around 60-70% on the top-ranked cause.

8. Conclusions and future

This paper presented an application for automated troubleshooting of mobile networks based on Bayesian networks. Appropriate models for the domain were found after an iterative process of interaction with troubleshooting experts. Troubleshooting based on Bayesian networks was proven to be much more efficient than troubleshooting based on rules, due to the intrinsic capacity of Bayesian network of dealing with uncertainty.

A knowledge acquisition tool has also been developed, which helps troubleshooting experts to convert their knowledge into Bayesian models. It has proven to be very useful when dealing with experts in existing operators' networks.

Trials have been carried out in real networks, showing promising results. The obtained hit rate is similar to the one obtained if experts manually diagnose the problems. Moreover, the response time of the tool is much less than the time required by a human expert. Trials are still ongoing, fine tuning the model when new cases are analysed. The performance improvement is quite high every time the model is modified, which makes us think that even better results will be obtained in the future.

Furthermore, there are future research areas that will contribute to further improve the results. Learning will make the model not so much dependent on the experts' accuracy and will lift the workload associated to create and tune the models. Another research area is related to discretization of continuous variables, i.e. defining the adequate number of states and thresholds for each state of a symptom that is continuous by nature.

9. Acknowledgements

This work has been performed as part of the co-operation agreement between Nokia and the University of Malaga. This agreement is partially supported by the Program to promote technical research (Programa de Fomento de la Investigación Técnica, PROFIT) of the Spanish Ministry of Science and Technology.

The authors would like to thank Orange and Sonofon for fruitful discussions and feedback. We also would like to thank Dr.Finn Jensen at University of Aalborg for his valuable advice.

References

[1] Halonen, T., J.Melero, J.Romero, *GSM, GPRS & EDGE Performance: Evolution towards 3G UMTS* (Wiley, 2002)
[2] Russel, S., P.Norvig. *Artificial Intelligence - a modern approach* (Prentice Hall, 1995)
[3] Jensen F., *Bayesian networks and decision graphs* (Springer, 2001)
[4] J.E.Desmedt (Ed.), *Computer-Aided Electromyograpgy and Expert Systems* (Elsevier Science Publishers, Amsterdam 1990)
[5] Heckerman, D., J.Breese, and K.Rommelse. Decision-theoretic troubleshooting. *Communication of the ACM*, 1995, 38 (3), 49-56
[6] Heckerman, D., J.Breese. Causal Independence for probability assessment and interference using Bayesian Networks, *IEEE, Systems, Man, and Cybernetics*, 1995
[7] Kim, J., and J.Pearl. A computational model for casual and diagnostic reasoning in inference engines. *8th International Joint Conference on Artificial Intelligence*, 1983, 190-193