Assem SHAYAKHMETOVA[1,2], Natalya LITVINENKO[1] ,Orken MAMYRBAYEV[1,2] , Waldemar WÓJCIK[3]

Institute of Information and Computational Technologies, Almaty, Kazakhstan (1),
Al-Farabi Kazakh National University Almaty, Kazakhstan (2),
Lublin Technical University, Poland (3)

doi:10.15199/48.2020.12.55

Graph models with multiple Bayesian networks

Abstract. *This paper is devoted to some issues of using multiple Bayesian networks in the various applied problems. Sometimes we deal with applied problems that are difficult to describe with a model that is represented by only one Bayesian network. At the same time, the considered problem may contain blocks with various types of uncertainties that can be well described by multiple Bayesian networks. Even if the problem can be described by only one Bayesian network, the size of this network could be so large that it will be impossible to find the solution with the help of existing software products. In this case, it is better to decompose in some way this large Bayesian network into several smaller ones. However, existing software products are poorly adapted to work with several Bayesian networks simultaneously. In this project, we develop and describe a software product that allows us to work with several Bayesian networks simultaneously.*

Streszczenie.
Artykuł ten jest poświęcony niektórym zagadnieniom związanym z wykorzystaniem szeregu sieci bayesowskich w różnych obszarach zastosowań. Czasami mamy do czynienia z zagadnieniami stosowanymi, które są trudne do opisania za pomocą modelu, który jest reprezentowany przez pojedynczą sieć Bayesa. Jednocześnie, rozważany problem może zawierać bloki z różnymi rodzajami niepewności, które mogą być dobrze opisane przez wiele sieci Bayesa. Nawet jeśli problem może być opisany tylko przez jedną sieć Bayesa, rozmiar tej sieci może być tak duży, że niemożliwe będzie znalezienie rozwiązania przy pomocy istniejącego oprogramowania. W tym przypadku lepiej jest rozłożyć w jakiś sposób tę dużą sieć na kilka mniejszych. Istniejące oprogramowanie jest jednak słabo przystosowane do pracy z kilkoma sieciami jednocześnie. W tym celu opracowaliśmy i opisaliśmy oprogramowanie, które pozwala nam na pracę z kilkoma sieciami Bayesa jednocześnie. ***Grafy z wieloma sieciami bayesowskimi.***

Keywords: Bayesian networks, graph models, uncertainties, machine learning.
Słowa kluczowe: Sieci bayesowskie, modele wykresów, niepewności, uczenie maszynowe.

Introduction

Sometimes we need to describe rather complicated processes with uncertainties. In this case, a Bayesian network is a very convenient tool. Bayesian network theory is based on the probability theory and graph theory. The basic definitions from the probability theory can be found, for example, in [1,2]. The definitions and concepts of graph theory used in Bayesian Network theory can be found in [3,4,5]. The basic principles of the Bayesian network theory can be found in [6, 7, 8, 9,10]. The main principles of work with Bayesian networks in popular software product BayesiaLab can be found in [11] and in software product AgenaRisk in [12].

The capabilities of artificial intelligence are widely used to solve various applied problems in different research areas. One of the leading tools in the field of artificial intelligence is the use of Bayesian networks. Many authors consider and utilize Bayesian networks as a tool for studying models with uncertainties. A very detailed consideration of Bayesian networks was done in [13,14]. Bayesian networks can use the ideas of machine learning. Models based on Bayesian networks are capable of self-learning and self-improvement as the new data are assimilated. These models are usually insensitive to incomplete and redundant data. Bayesian networks allow the use of heterogeneous data.

Although the mathematical apparatus for working with Bayesian networks is well developed, implementation of theoretical features in algorithms and development of the programming code is a nontrivial task. Nevertheless, a sufficient number of successful algorithms were developed, and these algorithms were successfully implemented in popular software products.

Despite some success in the development of efficient algorithms, the use of Bayesian networks may require massive computations involving quite complex algorithms, and therefore be very time-consuming. Very often, in such cases, a large amount of data has to be processed. This fact limits the usage of Bayesian networks. And it motivates the Bayesian network community and us to develop new, more efficient techniques and software products. Examples of good software products and tools which are working with Bayesian networks include well-known BayesiaLab, AgenaRisk, Hugin Expert, etc.

The main difficulties in the process of algorithm development for the Bayesian network tools are due to a large amount of data and high computational cost. This data are obtained during the construction and training of Bayesian networks. The amount of data and the computational cost are growing exponentially w.r.t. the number of nodes and the number of connections between nodes. The algorithm developers are trying to reduce both the amount of data and the number of computations in various ways. Some of these ideas are briefly described below.

Calculations in Bayesian Networks

The computational algorithms in Bayesian networks in the absence of evidence and the presence of evidence are fundamentally different. In the process of development of algorithms, the concept of “Generation” is often used. This concept allows conveniently systematize the nodes of the network (graph) for the development of various algorithms.
One can distinguish between two types of generations – generations of descendants and generations of ancestors. For a generation of descendants, this is a set of vertices that have parents only from earlier generations (or do not have parents at all) and have children only in later generations (or do not have children at all). For a generation of ancestors, this concept is similar. In essence, it is a set of vertices that have children only in later generations (or do not have children at all) and have parents only from earlier generations (or do not have parents at all). The only difference is that the construction of generations of descendants begins with nodes that do not have parents, and the construction of generations of ancestors begins with nodes that do not have children. Generations of descendants are constructed, starting from vertices that do not have parents.

Definition. Generations of descendants are defined as follows:

- Nodes without parents belong to the 0th generation of descendants.

- Nodes with parents from only the 0th generation belong to the 1st generation of descendants.
- Nodes with parents from 0th and 1st generation belong to 2nd generation of descendants.
- …….
- Nodes with parents from 0th, 1st, 2nd, … K-th generation belong to $(K+1)$-th generation of descendants.

Generations of ancestors are constructed starting from vertices which do not have children.

Definition. Generations of ancestors are defined as follows:

- Nodes without children belong to the 0th generation of ancestors.
- Nodes with children from only 0th generation belong to 1st generation of ancestors.
- Nodes with children from 0th and 1st generation belong to the 2nd generation of ancestors.
- …….
- Nodes with children from 0th, 1st, 2nd, … K-th generation belong to $(K+1)$-th generation of ancestors.

…….

- Initialization of a Bayesian network, i.e. calculation of values in the nodes from the values of unconditional variables (nodes without parents) and conditional probability tables is a simple task. Below we briefly describe the initialization algorithm with the concept of "Generation".

1. We separate the vertices of the Bayesian network into generations of descendants, as described above. Recall that the 0th generation consists of vertices without parents. For nodes of 0th generation calculations are not needed.
2. We calculate the values for all nodes of the 1st generation, using the values of the nodes of the 0th generation and the formula for total probability.
3. We calculate the values of the nodes of the 2nd generation, using the values of the nodes of the 0th generation and the calculated values of the nodes of the 1st generation, as well as the formula for the total probability.
4. Similarly, we calculate the values of the nodes of the third generation.
5. … etc.

At some point, some nodes in the Bayesian network get evidence. It is required to recalculate the values of the network nodes considering the obtained evidence. This task is much more complicated than the initialization task. One of the difficult issues is determining the order of calculation of Bayesian network nodes. Obviously, the nodes that obtained the evidence do not need calculations. Also, the nodes that do not have parents and satisfy to the following condition do not need calculations. The condition is that among the descendants of these nodes there are no nodes that have obtained evidence.

It would be useful to collect all the nodes for which we do not need to make calculations in one set. This set will be called the zero level (set) of the Bayesian network nodes. In the next set, we will collect all the nodes for which calculation we need information from only the nodes of the zeroth level. This set will be then called the first level of the Bayesian network nodes. In the next set, we will collect all the nodes, for which values calculation we need the information from the nodes of the zero and first levels. This set will be called the second level of the Bayesian network nodes. In the next set, we will collect all the nodes, for which calculation we need information from the nodes of the zero, first and second levels. This set will be called the third level of the Bayesian network nodes, and so on.

Most existing Bayesian software products suggest that the problem you are studying is described by the only Bayesian network you need to build. The constructed network can have a sufficiently large number of nodes and probabilistic connections between nodes. This approach has several disadvantages. We list some of them below:

- A general mathematical graph model of the studied process is not always adequately reflecting the studied process. Often it is more convenient to present the mathematical model of the whole process in a completely different form. With the use of a Bayesian network, it is possible to conveniently simulate only a certain, although very important, part of the studied process. In such cases, developers of Bayesian networks software products offer specially developed libraries and modules for various program environments, using which the user can write their own modules for working with Bayesian networks.
- The studied model may contain several blocks rather independent from each other. Each of these blocks is well described by a graph model based on a Bayesian network. Existing Bayesian networks software products are not really suitable for simultaneous work with several networks.
- The complexity of the data presentation in Bayesian networks and the computational complexity grow exponentially and depend on
  - the number of nodes;
  - the number of connections between nodes;
  - the number of parents of individual nodes;
  - the probabilistic scale of nodes.

Let us consider in more details the difficulties that arise with growing the Bayesian network.

Nodes with parents are characterized by the so-called Conditional Probability Table (CPT). The computational cost and storage requirement can be estimated from a certain combination of the largest CPTs. The size of a CPT can be computed by the formula:

$$V_{CPT} = U_0 \cdot U_1 \cdot U_2 \cdot U_3 \cdot \cdots \cdot U_N$$

where U_j denotes the number of all possible values, which the considered node and its parents can take.

Let us evaluate the capabilities of a typical personal computer (PC) for handling Bayesian networks. Typically, most researchers use a 32-bit Windows operating system, which allows addressing no more than 4GB of RAM. The operating system itself and some related programs usually occupy from 1 to 1,5GB of RAM. The available rest is approximately 2.5 GB of RAM.

Let us consider an abstract Bayesian network. Let, for example, the vertex U_0 has 10 parents: $U_1, U_2, U_3, \ldots, U_{10}$. Let each of these 11 vertices can take 5 different values, for example, $\{1, 2, 3, 4, 5\}$. In this case, the number of elements in CPT for the vertex U_0 is equal to 5^{11} = 48828125.

The storage cost is approximately 400MB of RAM. If we consider 5 such vertices, we need already 2GB of RAM. As we can see, 5 vertices need almost the whole amount of available RAM. This example shows that by developing of algorithms for large Bayesian networks it is necessary to take into account a possible shortage of RAM. The storage of the data on external hard disks may dramatically increase the computing time.

The second problem is the need to fill the CPT. CPT is filled either manually or automatically, based on some expert data. To fill 48 millions of elements in the CPT manually for the problem above is impossible. To find an expert data for constructing 48 million elements of CPT automatically is also impossible. Practically this problem can be solved in the following ways:

- The first one is reducing the number of values the nodes can take. Usually, every node can take only two values, e.g., (Y, N) or (Yes, No) or (0, 1). The disadvantages of this method include the loss of accuracy in estimating the factor due to reducing the number of possible values (coarsening of the scale). However, if we consider the previous example, the number of elements in the CPT will already be 2^{11} = 2048. The amount of required memory is approximately 16 KB. As we can see, these are quite acceptable RAM requirements.
- The second one is a reasonable simplification of the Bayesian network structure. Elimination of not important connections between BN nodes quite effectively reduces the RAM requirements. If in the previous example, the number of parents does not exceed five, then the number of elements in the CPT will not exceed: 5^6 = 15625 elements and will require not more than 125 KB of memory. With such improvements, it will be possible to process networks with hundreds of nodes.
- Reducing the number of nodes may also significantly reduce the required storage.
- Often, it is possible to simplify the topology of the Bayesian network to use fast and efficient computing algorithms in the Bayesian network. For example, one can try to build a "Markov Blanket".

The considered methods do not always construct a Bayesian network correctly, reflecting the studied problem and, at the same time, allowing effective processing. We are trying to fill this gap and propose the following approach:

1. Identify and select independent blocks
2. For each such block, build its own Bayesian network.
3. Build an assessment system for each block. For example, consideration of a job applicant's higher education level can be reduced to two estimates - the average score and the average score for specialized subjects.
4. Add to each BN new nodes which correspond to the constructed estimates. Build dependencies between new nodes and previously constructed nodes.
5. Build a general Bayesian network in which selected independent nodes are represented only by the constructed earlier estimates.

This split of the general network dramatically reduces the size of the given network and makes it possible to use the Bayesian network tool quite effectively. In the following example, we decompose the given Bayesian network into three independent networks and estimate the new memory storage.

Variant 1. Let us consider the Bayesian network, which consists of 300 nodes. For the sake of simplicity, we assume that each node can take three values, e.g., $\{0, 1, 2\}$. We assume that there are 90 nodes, where each one has 10 parents. There are 90 nodes which do not have parents at all. There are 60 nodes, where each one has 5 parents. Every from the last 60 nodes has 3 parents. The CPT requires about 130MB of RAM:

$$K_{CPT} = 90 \cdot 3^{11} + 90 \cdot 3 + 60 \cdot 3^6 + 60 \cdot 3^4 = 15992100$$
$$V_{CPT} = 15992100 \cdot 8 = 127936800$$

Variant 2. Assume we have 3 Bayesian networks. Every Bayesian network consists of 100 nodes. As in previous Variant 1, we assume that each node has a probability scale with 3 points {0, 1, 2}. We assume that there are 30 nodes, where each one has 10 parents. Every node from another 30 nodes has no parents. There are another 20 nodes, where each has 5 parents. Every node from the last 20 nodes has 3 parents. The CPT requires about 43MB of RAM.

$$K_{CPT} = 30 \cdot 3^{11} + 30 \cdot 3 + 20 \cdot 3^6 + 20 \cdot 3^4 = 5330700$$
$$V_{CPT} = 5330700 \cdot 8 = 42645600$$

When we calculate one of three Bayesian networks, information about the other two networks can be stored on an external hard disk. Computations with small Bayesian networks can be finished faster. Thus, one can get many advantages when splitting a Bayesian network into several networks.

Unfortunately, the most popular software products for Bayesian networks are not always convenient for complex simultaneous research of several networks. Of course, these programs can be used, but it is inconvenient and requires some effort. It would be handy to have a common platform on which it would be possible to study several Bayesian networks simultaneously. We addressed this problem in this work, and as a result, we have developed a software product for simultaneous work with several Bayesian networks.

The software

The main logical unit of a software product is a *Project*. At the same time, up to five Projects can be opened in the main window (see Figure 1).

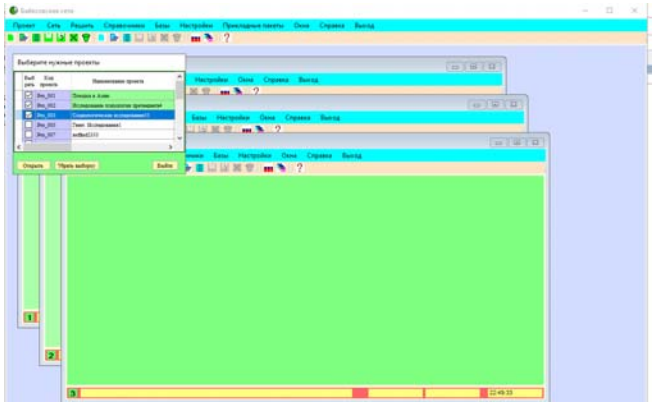

Fig.1. Five Projects are opened in the main window

Each *Project* is opened in its window. The windows are numbered. The number from 1 to 5 can be found on the left-hand side at the bottom. Any of the existing *Projects* can be opened in each window. If the *Project* is already opened in one of the windows, the second time this *Project* can no longer be opened in another window. Everything related to this *Project*, for example, Bayesian networks, can only be opened in the window in which the *Project* is open. We also use the concept of the current *Project*. This is a *Project* in which the user performs some actions. The corresponding window will be marked with a darker colour. The user can perform some standard actions (see Figure 2) with each *Project*.

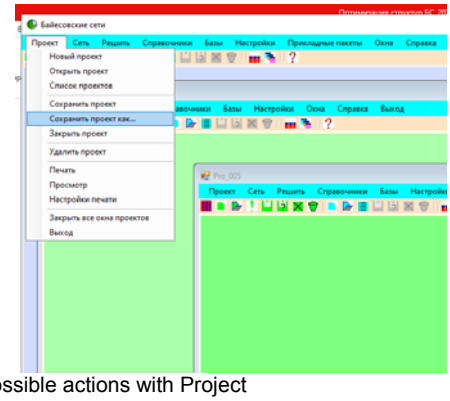

Fig. 1. Possible actions with Project

If the studied problem is well described by a single Bayesian network, as it is assumed in other software products, then in the main window, the user can open up to 5 Bayesian networks that do not belong to any *Project*.

Figure 3 shows three independent Bayesian networks and one *Project.*

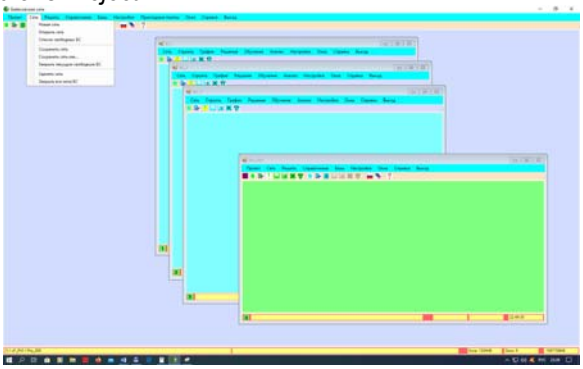

Fig.2. Three independent Bayesian networks

With each open Bayesian network, the user can also perform some typical actions.

We assume that each *Project* can contain any number of Bayesian networks. The only limitation is the storage cost. In the right corner in the bottom, the main window shows the following information:

- the current size of the heap,
- total number of currently opened windows,
- size of presently available RAM.

Let us summarize the main features of the developed tool below.

- The basic logical unit of research is the *Project.* A project may contain several Bayesian networks, as well as windows to work with data. The menu of the main window presents standard features for work with the *Project*. Some of the menu items of the main window are duplicated by the “hot” keys of the main window. The main window of each open *Project* also contains the main menu that relates directly to the open *Project.* Some items of this menu are also duplicated by “hot” keys.
- Up to 5 Projects can be opened simultaneously in the main window of the program. Each Project (and everything that belongs to this Project) is contained in its window. Any Project window can be minimized or maximized to the main program window.
- Two versions of the Project can be created for the same problem under investigation. These two versions can be opened simultaneously in the main program window. This can be useful for comparison to find an optimal Project.
- It is possible to reuse the current Project for a new problem. Bayesian networks in an existing Project can be adjusted for a new problem. This can significantly speed up the development of a new Project.
- Up to 5 independent Bayesian networks can be opened simultaneously in the main program window. Each independent Bayesian network is contained in its window.
- Everything that belongs to this Bayesian network is contained only in this window. The main menu of the main window contains several features to perform typical operations. A part of the menu items is duplicated by the “hot” keys. Each window of an open Bayesian network has its own main menu and its own set of “hot” keys to work only with this Bayesian network.
- Bayesian networks opened in the main window can be minimized or maximized to the size of the main window.
- Windows with Bayesian networks opened in the Project can be minimized or maximized in the window of the corresponding.
- The Project may contain quite a large number of Bayesian networks. The memory is required only for open Projects and Bayesian networks.
- Various manipulations with nodes and edges of a network are possible.
- Various manipulations with the network are possible.
- Various manipulations with the Project are possible.

Conclusions

In modelling of a real process, the corresponding graph model may contain several Bayesian networks. Existing Bayesian network tools are not well suited for working with multiple networks. It motivated us to develop an approach that allows us to work with multiple Bayesian networks simultaneously. As a result, we implemented the new software tool, which requires much less memory, and, therefore, allows us to handle much larger Bayesian networks.

Acknowledgements. This work was supported by the grant “Development and software implementation of a Bayesian networks package for solving applied problems”. The project number is AP05131293.

***Authors**: Shayakhmetova Asem PhD, senior research scientist of the Institute of Information and Computational Technologies CS MES RK, 125 Pushkin St, 050000 Almaty, Kazakhstan, e-mail: asemshayakhmetova@mail.ru.; Litvinenko Natalya, Junior Researcher of the Institute of Information and Computational Technologies CS MES RK, 125 Pushkin St, 050000 Almaty, Kazakhstan, e-mail: n.litvinenko@inbox.ru; Mamyrbayev Orken, PhD, assoc.prof. Deputy General Director of the Institute of Information and Computational Technologies CS MES RK, 125 Pushkin St, 050000 Almaty, Kazakhstan, e-mail: morkenj@mail.ru; Waldemar Wójcik, Professor at the Lublin University of Technology, Institute of Electronics and Information Technology, Nadbystrzycka 38A, 20-618 Lublin, Poland, e-mail: waldemar.wojcik@pollub.pl.*

REFERENCES

[1] Gmurman V.E. Probability Theory and Mathematical Statistics: *Manual – Moscow.* (2003), 479.
[2] Kolmogorov A.N. Basic Concepts of Probability Theory*: Manual. – Moscow: Science*, (1974), 412 p. (in Rus).
[3] Karpov D.V., Graph theory, (not edited) *https://logic.pdmi.ras.ru/~dvk/graphs_dk.pdf* (in Rus)
[4] Ore O. Graph theory. *Moscow: Science*. (1980), (in Rus)
[5] Kharari Ph. Graph theory. *Moscow: Mir. (*1973), 300. (in Rus)
[6] Jensen F.V., Nielsen T.D. Bayesian Networks and Decision Graphs. *Springer.* (2007).
[7] Barber D., Bayesian Reasoning and Machine Learning. (2017), *http://web4.cs.ucl.ac.uk/staff/D.Barber/textbook/020217.pdf*
[8] Neapolitan R.E. Learning Bayesian Networks, Pearson Prentice Hall (2004). *http://www.cs.technion.ac.il/~dang/books/ Learning%20Bayesian%20Networks(Neapolitan,%20Richard). pdf.*
[9] Litvinenko A., Litvinenko N., Mamyrbayev O., Shayakhmetova A., Generations in Bayesian Networks, *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska, 9 (2019), no 3, 10-13. https://doi.org/10.35784/iapgos.228*
[10] Litvinenko A., Mamyrbayev O., Litvinenko N., Shayakhmetova A. Application of bayesian networks for estimation of individual psychological characteristics. *Przeglad Elektrotechniczny*, 95 (2019), no. 5, 92-97.
[11] Litvinenko N., Litvinenko A., Mamyrbayev O., Shayakhmetova A. Work with Bayesian Networks in BAYESIALAB. *Almaty: IPIC*, (2018), 311 (in Rus). *ISBN 978-601-332-206-3.*
[12] Litvinenko N., Litvinenko A., Mamyrbayev O., Shayakhmetova A. AgenaRisk. Work with Bayesian Networks. *Almaty: IPIC*, (2019), 236. *(in Rus). ISBN 978-601-332-335-0.*
[13] Pearl J. How to Do with Probabilities what People Say You Can't, *IEEE, North Holland*. (1985), 6–12.
[14] Pearl J. Probabilistic Reasoning in Intelligent Systems. *San Francisco: Morgan Kaufmann Publishers*. (1988), 552.