U.P.B. Sci. Bull., Series C, Vol. 86, Iss. 4, 2024 ISSN 2286-3540

ANT COLONY CORRELATION OPTIMISATION OVER BAYESIAN NETWORKS

Cătălin-Mihail CHIRU[1], David Traian IANCU[2]

We propose a novel way of combining two canonical models used in Artificial Intelligence (AI): Bayesian Networks (BN) and Ant Colony Optimisation (ACO) in order to obtain a fast graph-traversal algorithm that establishes the highest correlation path between the nodes of a BN and the target node, similarly to a variable independence test.

Keywords: Bayesian Network, Ant Colony Optimisation, Self Organising Systems, Inference, High correlation path

1. Introduction

Bayesian networks are probabilistic graph models used to represent problems with a degree of uncertainty through directed acyclic graphs (DAGs) that capture the probabilistic events (the nodes) and their structure and conditional dependency (the edges). Before the emergence of Deep Learning (DL) and Artificial Neural Networks (ANNs), Bayesian Networks were among the predominant models in research and IT production for tasks in multiple fields, such as classification in car assurances and creditor analysis [1], modelling crime linkage [2], clinical decisions - diagnosis and treatment planning [3]. The main benefits of those models were their speed of inference and clear explainability: the user can see that changing the probabilities of one node affects the other dependent nodes. However, the time complexity for exact inferences scales exponentially with the number of nodes in the network: $O(k^n)$, where k is the number of possible valuables for categorical nodes (k = 2 for boolean variables).

The Ant Colony Optimisation (ACO) is an optimization algorithm part of the swarm intelligence family, used to alleviate the exponential complexity of pathfinding to a polynomial one through heuristic exploration of the solutions' space using multiple agents (ants) and converging to an optimal solution using the pheromones of the ancestors from previous iterations. The algorithm mimics the way in which ants guide each other from the nest as the starting point towards the food, representing the end point.

[1] Master student, Dept. of Computer Science, Artificial Intelligence, UNST POLITEHNICA Bucharest, Romania, e-mail: cata_chiru@yahoo.com

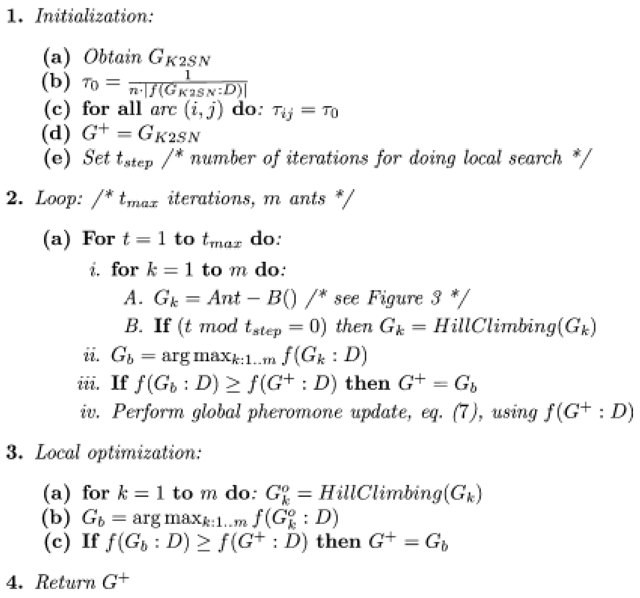
[2] Prof., Dept. of Computer Science, Artificial Intelligence, UNST POLITEHNICA Bucharest, Romania, e-mail: david_traian.iancu@upb.ro

In the literature, the only way in which ACO was used over Bayesian Networks [4 - 6] was to determine the structure of the graph by pruning nodes and edges that were redundant. Our approach initially differed from this idea: Based on the inputs necessary to run ACO; we set up the algorithm to obtain fast queries from a source node to an effect, finding the highest correlation path along the way. However, through the behaviour of the algorithm and the assumptions made, the algorithm functioned much more similarly to the canonical approaches mentioned above. Through their walk on the graph, the ants discover high correlation paths from source to leaves, which can be interpreted as non-removable components when pruning down the network.

2. State-of-the-art

As expert systems, Bayesian Networks need a proper configuration of their graph topology when this structure is not already given, and it is known that the deterministic exploration of this procedure is NP-Hard [10, 11]. Self-organising algorithms represent an unsupervised heuristic way of reducing the exploration space, such that solvers of NP-hard problems take polynomial time at the cost of obtaining approximate solutions (local optima).

The first article in which ACO was employed to determine the Bayesian graph's form automatically was De Campos' "Ant colony optimization for learning Bayesian networks" [4]. In this paper, the authors introduce an alternative to greedy hill climbing when learning a Bayesian Network's structure: ACO-B, an ACO-based metaheuristic scoring-based learning algorithm for BNs.



```
1. Initialization:
    (a) Obtain $G_{K2SN}$
    (b) $\tau_0 = \frac{1}{n \cdot |f(G_{K2SN}:D)|}$
    (c) for all arc $(i, j)$ do: $\tau_{ij} = \tau_0$
    (d) $G^+ = G_{K2SN}$
    (e) Set $t_{step}$ /* number of iterations for doing local search */
2. Loop: /* $t_{max}$ iterations, $m$ ants */
    (a) For $t = 1$ to $t_{max}$ do:
        i. for $k = 1$ to $m$ do:
            A. $G_k = Ant - B()$ /* see Figure 3 */
            B. If $(t \bmod t_{step} = 0)$ then $G_k = HillClimbing(G_k)$
        ii. $G_b = \arg\max_{k:1..m} f(G_k : D)$
        iii. If $f(G_b : D) \geq f(G^+ : D)$ then $G^+ = G_b$
        iv. Perform global pheromone update, eq. (7), using $f(G^+ : D)$
3. Local optimization:
    (a) for $k = 1$ to $m$ do: $G_k^o = HillClimbing(G_k)$
    (b) $G_b = \arg\max_{k:1..m} f(G_k^o : D)$
    (c) If $f(G_b : D) \geq f(G^+ : D)$ then $G^+ = G_b$
4. Return $G^+$
```

Fig. 1. ACO-B Algorithm [4]

The algorithm starts from an empty graph for each ant and the 'B'-part constructs the edges in a greedy manner to maximise a decomposable scoring metric while avoiding creating cycles in the graph. The 'B'-part converges when no possible edge-addition improves the scoring metric. For the scoring metric, K2 was chosen, as it is also a performance evaluation metric. K2 measures the joint probability between a BN and the database on which it was generated.

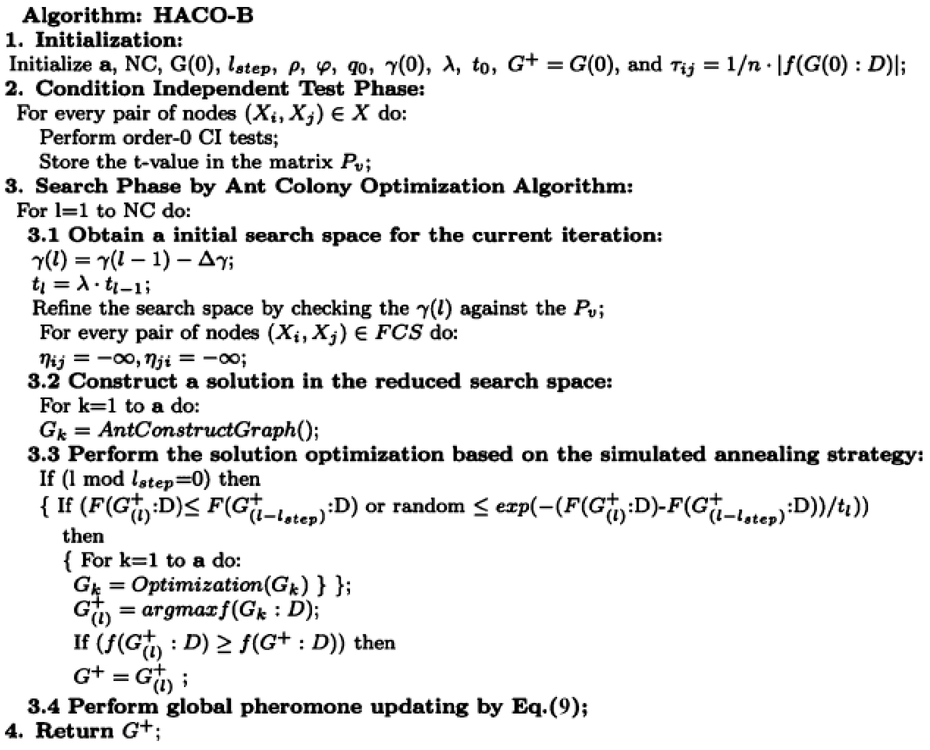
To ensure that the ants don't get stuck in local optima, the authors periodically change the graphs on which ants operate with ones obtained with Hill Climbing. The ant that receives the best score reinforces and propagates the pheromones for its graph construction. The pheromone production is also controlled by a sub unitary evaporation factor - ρ

Based on the last iteration ants, another optimization rounds proceeds. Each ant is enhanced using hill climbing to try to outscore the current maximum. The final graph returned by ACO-B is the one that maximizes the scoring metric.

After ACO-B, numerous variations improved over it [5 - 9]. Out of those, we mention HACO-B [5] and ABC-Miner [6]. The first uses another self-organising principle (simulated annealing), while the latter represents the most recent progress in terms of Bayesian Learning with Ant Colony optimisation, and it is also used in classification tasks as it can build Bayesian Classifiers (BNCs).

In the HACO-B paper [5], the authors' main optimizations over the original ACO-B are:

- Reducing the search space in which edges are considered by the ants when constructing the graph by computing a relaxed conditional independence test between two variables given a set of fixed conditions. The authors use the order-0 independence test to reduce the computational complexity, meaning that the conditional set is empty. This change corresponds to calculating the mutual information for each edge of the fully connected graph. The constraints are relaxed through iterations by a factor γ so ants don't get stuck in local optima.
- Adapting the heuristic function on which ACO-B builds an ant's graph. The HACO-B algorithm introduces a multiplicative factor, $\omega = 1 + Inf(X_i, X_j)$, that shows the intensity of adding an arc by considering the conditional dependency introduced in the structure by it.
- Simulated annealing scheduling of the optimisation strategy: HACO-B compares the current best graph with the last iteration's best (ΔF), and there are two possibilities: If the previous best outperforms the current one, the current graph requires the optimisation stage, otherwise, the optimisation stage is probabilistically triggered, based on simulated annealing procedure of temperature t: $\mathrm{P} = \mathrm{e}^{-\frac{\Delta \mathrm{F}}{\mathrm{t}}}$



```
Algorithm: HACO-B
1. Initialization:
Initialize a, NC, G(0), $l_{step}$, $\rho$, $\varphi$, $q_0$, $\gamma(0)$, $\lambda$, $t_0$, $G^+ = G(0)$, and $\tau_{ij} = 1/n \cdot |f(G(0) : D)|$;
2. Condition Independent Test Phase:
 For every pair of nodes $(X_i, X_j) \in X$ do:
    Perform order-0 CI tests;
    Store the t-value in the matrix $P_v$;
3. Search Phase by Ant Colony Optimization Algorithm:
 For l=1 to NC do:
   3.1 Obtain a initial search space for the current iteration:
   $\gamma(l) = \gamma(l-1) - \Delta\gamma$;
   $t_l = \lambda \cdot t_{l-1}$;
   Refine the search space by checking the $\gamma(l)$ against the $P_v$;
    For every pair of nodes $(X_i, X_j) \in FCS$ do:
    $\eta_{ij} = -\infty, \eta_{ji} = -\infty$;
   3.2 Construct a solution in the reduced search space:
    For k=1 to a do:
    $G_k = AntConstructGraph()$;
   3.3 Perform the solution optimization based on the simulated annealing strategy:
    If (l mod $l_{step}$=0) then
    { If ($F(G^+_{(l)}$:D)$\leq F(G^+_{(l-l_{step})}$:D) or random $\leq exp(-(F(G^+_{(l)}$:D)-$F(G^+_{(l-l_{step})}$:D))$/t_l$))
       then
       { For k=1 to a do:
         $G_k = Optimization(G_k)$ } };
         $G^+_{(l)} = argmax f(G_k : D)$;
         If ($f(G^+_{(l)} : D) \geq f(G^+ : D)$) then
         $G^+ = G^+_{(l)}$ ;
   3.4 Perform global pheromone updating by Eq.(9);
4. Return $G^+$;
```

Fig. 2. HACO-B Algorithm [5]

By looking at the quantitative results of HACO-B, in terms of K2, the algorithm obtains similar optima to ACO-B, but it outperforms it in terms of stability and speed. HACO-B has less variance by at least a factor of 5, and it also at least halves the computation time for the Bayesian networks, no matter their size.

The ABC-Miner article [6] enhances ACO-B by:

- Building upon HACO-B, using the mutual information of two nodes that will be linked as the heuristic function.
- Automatically computing the maximum number of parent dependencies a node can have by looking at each node independently.
- Starting the graph from a Naive-Bayes structure - parent to all its children and expands to a Bayesian Augmented Naive-Bayes structure.
- Building the graph using ants with personality: each ant has its own α and β parameters from the original ACO algorithm.
- Calculating the pheromone update on two complementary terms: initially predominates the update given by the iteration-best, and over time, the update corresponding to the best network becomes more impactful. Evaporation of the pheromones is obtained through normalization.

Algorithm 1 Pseudocode of ABC-Miner

```
Begin
$BNC_{\mathrm{bsf}} = \phi;\ t = 1;$
$InitializePheromones();$
$InitializeHeuristicValues();$
repeat
    $BNC_{t\mathrm{best}} = \phi;\ Q_{t\mathrm{best}} = 0;$
    for $i = 1 \rightarrow$ colony_size do
        $BNC_i = CreateSolution(ant_i);$
        $Q_i = ComputeQuality(BNC_i);$
        if $Q_i > Q_{t\mathrm{best}}$ then
            $BNC_{t\mathrm{best}} = BNC_i;$
            $Q_{t\mathrm{best}} = Q_i;$
        end if
    end for
    $PerformLocalSearch(BNC_{t\mathrm{best}});$
    $UpdatePheromone();$
    if $Q_{t\mathrm{best}} > Q_{\mathrm{bsf}}$ then
        $BNC_{\mathrm{bsf}} = BNC_{t\mathrm{best}};$
        $Q_{\mathrm{bsf}} = Q_{t\mathrm{best}};$
    end if
    $t = t + 1;$
until $t =$ max_iterations or $Convergence($conv_ iterations$);$
return $BNC_{\mathrm{bsf}};$
End
```

Fig. 3. ABC-Miner Algorithm [6]

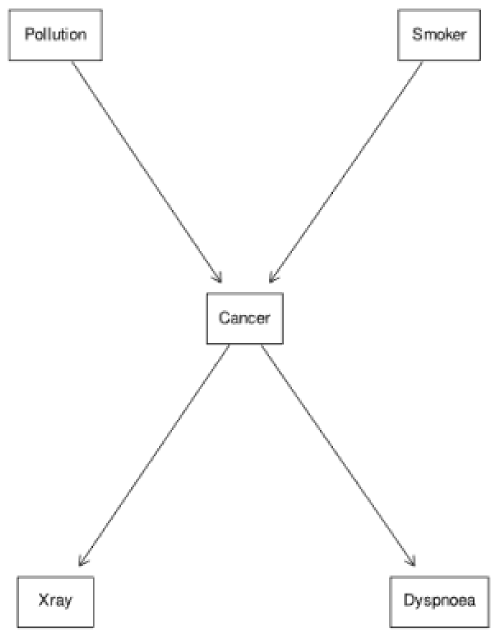
Other approaches presented in the literature are: ChainACO [11], where the authors propose a trade-off from the K2-based ACO by reducing the computational time, at the expense of less exploration in the BN structure space and higher chance to get stuck in a local best.. MMACO [12] claims state-of-the-art results in determining BN's structure, by applying max-min parent-child (MMPC) to establish the backbone of the Bayesian network and ACO to determine the direction in which the edges point. Both mentioned articles find their source of inspiration from other self-organising methodologies; ChainACO [11] builds upon an idea derived from genetic algorithms (GA), whereas MMACO's [12] idea starts from gradient hill climbing (GHC).

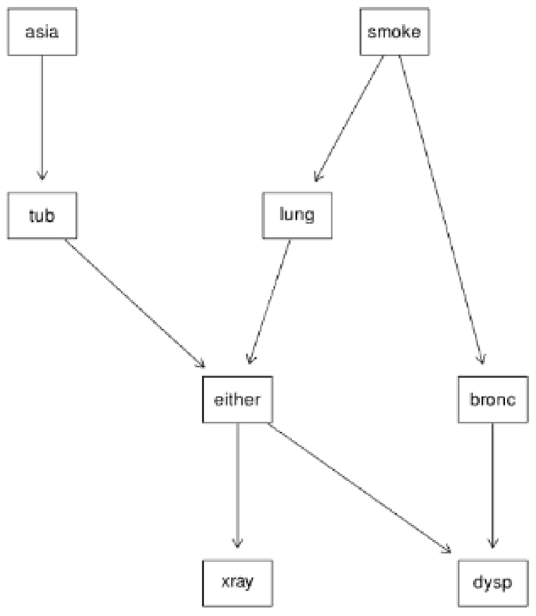
3. Methodology

3.1. Datasets - Bayesian Networks

As presented in the previous chapter, the networks from BNlearn [13] have been used as benchmarks in the field of Bayesian Networks, with the majority of studies reporting their findings on those open-source data.

As a result, the studies done in this work were done using three datasets and Bayesian nets from BNlearn [13]: CANCER (small - 5 nodes, 4 arcs), ASIA (small - 8 nodes, 8 arcs), ALARM (medium - 37 nodes, 46 arcs).





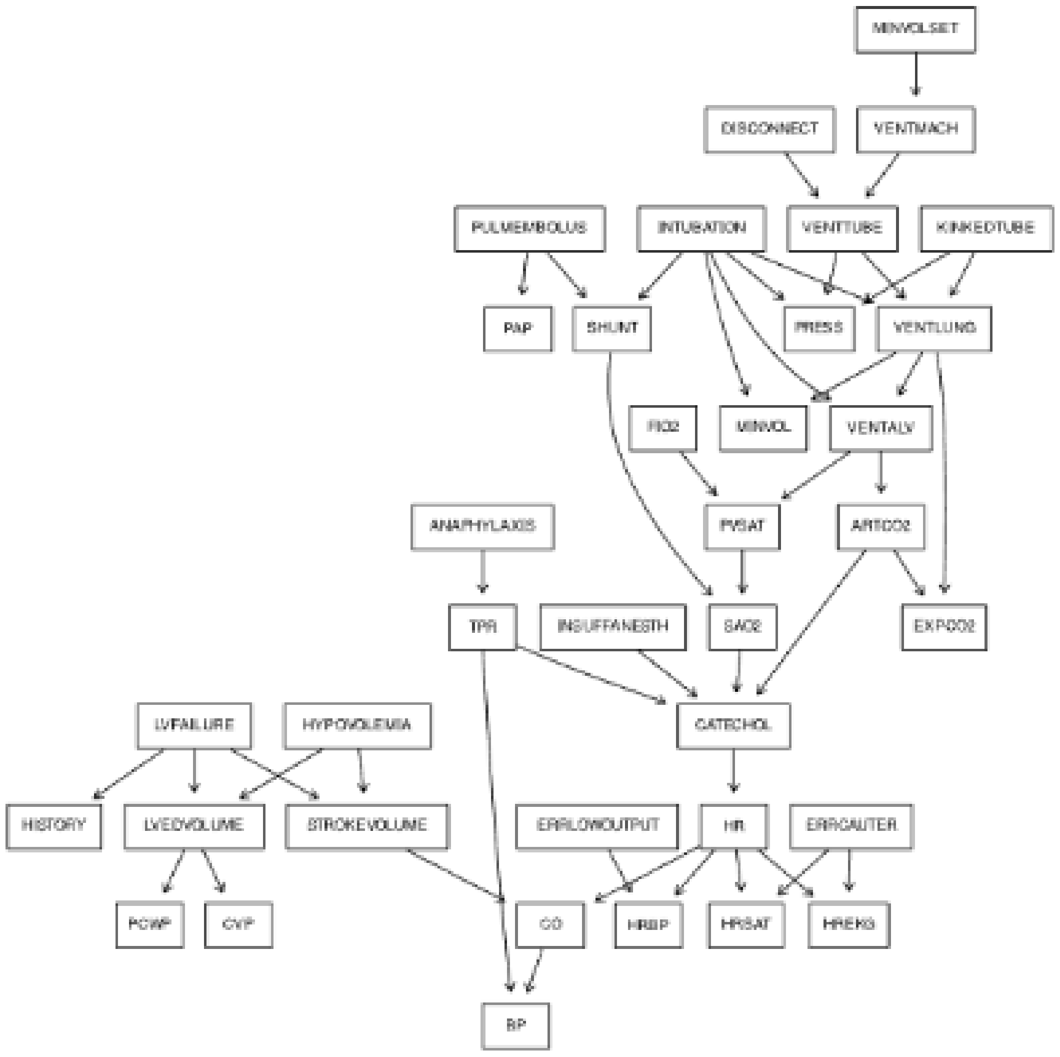


Fig. 4. Explored Bayesian Networks (CANCER, ASIA and ALARM)

The general observations and walkthrough of the algorithm will be discussed on the ASIA dataset in order to keep the article compact while still maintaining visual coherence and intuition.

3.2. Datasets - Exploratory Analysis

Based on the state-of-the-art and Taskensen's Github implementation [13], we have computed BN graphs using structure learning from the dataset, with Hill Climbing and K2 as a scoring type. Afterwards, we computed the chi-square test of variable independence for the obtained graph and plotted with highlights the nodes with significant dependencies between them (Figure 5).

Our ACO algorithm aims to determine this sub-graph without the explicit probabilistic apparatus but implicitly through the pheromone trails left by the ants (Section: 3.3. ACO-Corr). The novelty of our approach is represented by overlapping ACO over Bayesian Networks trying to obtain the highest correlation route, similarly to the concept of stigmergy [14] and drastically shifting from the way the literature's paradigm of establishing Bayesian Network's structure using ACO variants.

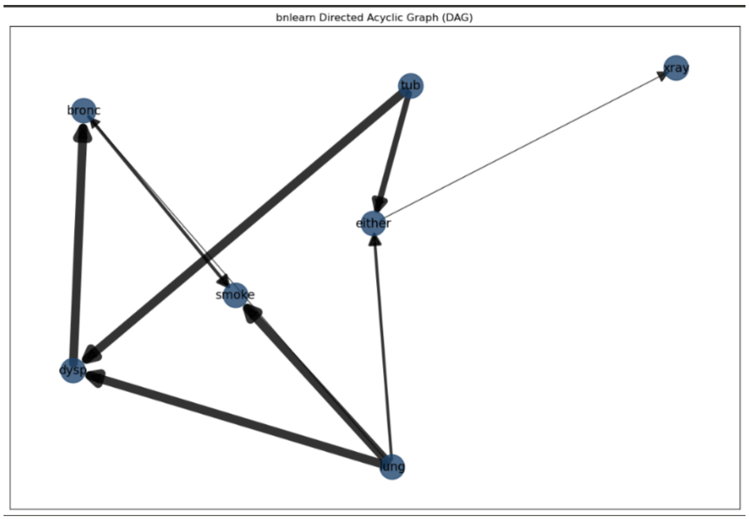


Fig. 5. chi-square test on ASIA BN

3.3. ACO-Corr

We start our algorithm by applying a step of Variable Elimination (VE) to obtain the cumulative probabilities for each node.

We introduce the following changes in the original ACO algorithm [15]:

- We assume only binary variables in the Bayesian Network and assign the "True" value obtained after Variable Elimination to each graph node. In the case of a categorical variable, our algorithm makes the assumption "One versus all" we will further discuss this assumption in the Limitations chapter.
- We instantiate the pheromones on the edges as the product between each node's "True" probabilities. The η_{ij} factor used in updating the pheromones is static and corresponds to $cost_{ij}^{-1}$ where $cost_{ij}^{-1}$ denotes the initial cost associated with that edge.
- For computing the cost, we must remember that ACO solves shortest path problems. In order to transform the probabilities accordingly, we use the logarithm function to scatter those from the unit segment and

inverse the results to obtain positive costs, as the logarithm of a subunitary value is negative. This setup has a clear concordance between high probability and low cost/resistance path.

- Evaporation is controlled using a subunitary multiplicative factor. Based on a hyperparameter search, we set the pheromone factor value to 0.99.
- The structure of BNs imposes another constraint to the original use of ACO: An ant cannot visit a node from every other node of the graph, as the BN has a Directed Acyclic Graph (DAG) structure, and the order of traversal is important and asymmetrical. As a result, we have forced the following constraints: Ants' building paths start from a clause node (in-degree of the node= 0) and end with an effect (out-degree of the node = 0), similarly to Wu's approach [11]. An ant can only traverse the BN downwards, and when calculating the next step, an ant only looks at the viable options from the current node it is sitting in (allow_k function), similarly to HACO-B [5].
- Based on the above-mentioned behaviour, the ants learn to cheat the maze and find unbalanced walk paths, going from a clause to an effect by visiting as few nodes as possible. In small, balanced networks, such as CANCER or ASIA, this type of walk correctly finds the most important connections. Still, in larger settings (ALARM), the ants misidentify good clauses and converge too soon (Figure 6). To solve this problem, we tried the alternative of Softmax Normalization instead of simple summation when computing the next step probabilities, to force ants to choose longer, highly-correlated paths in contrast to single-edge, not as correlated routes. The ants still get stuck in local optima, choosing at most length 3 chains of nodes instead of one edge, but still not exploring enough interesting parts of the graph.

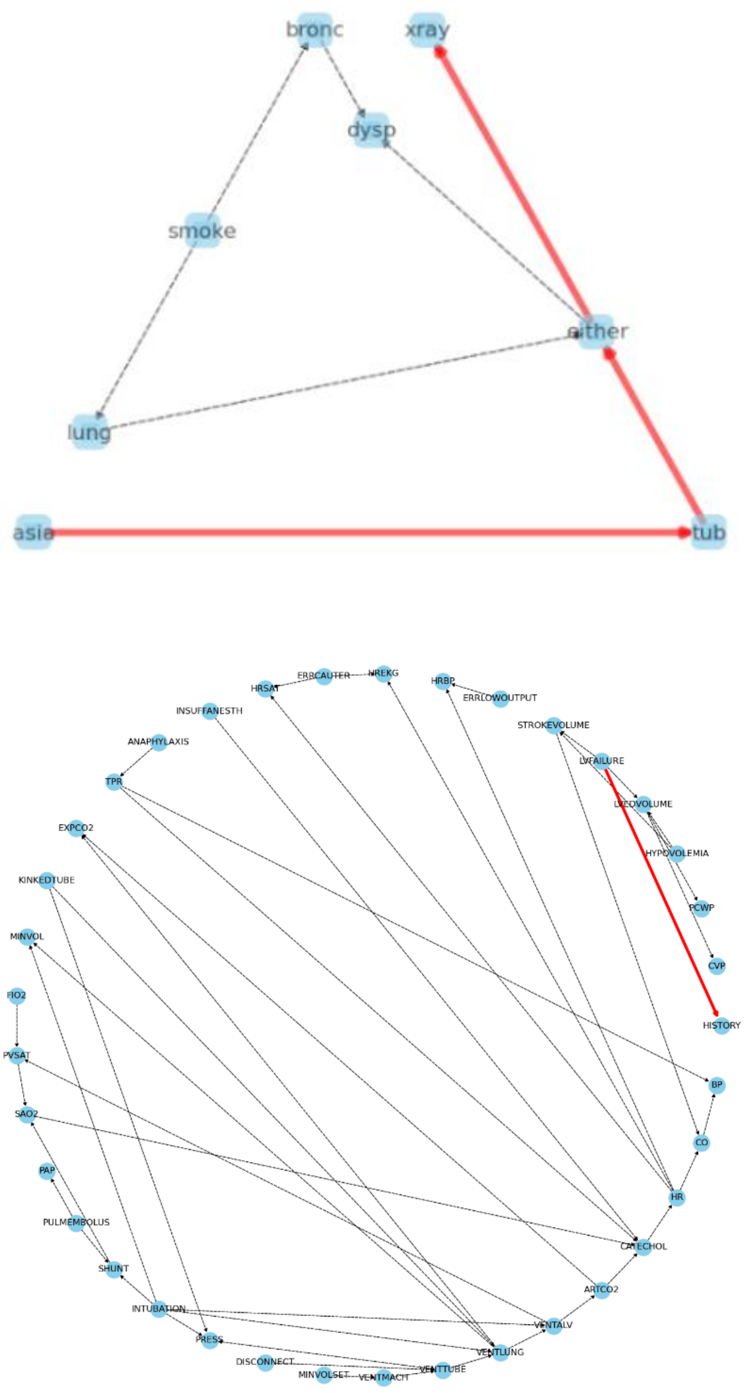


Fig. 6. Comparative path walk between ASIA and ALARM

3.4. Example on CANCER BN

We will take a small example to clarify how ACO-Corr derives its best paths in-place, without working with dataset splits:
For the Cancer Bayesian:

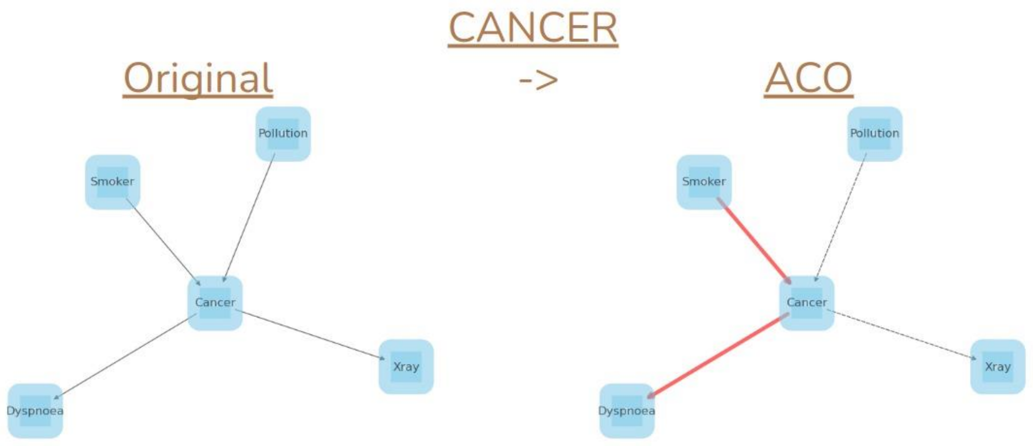


Fig. 7. ACO-Corr completion of "CANCER" Bayesian Network

We rename the binary attributes to 0 and 1 to have an unitary representation

After using variable Elimination, we get the complete probability distributions for all the nodes: (p(Cancer = 1) = 0.9884, p(Xray = 1) = 0.7919, p(Dyspnoae = 1) = 0.6959), besides Polution and Smoker, which are root nodes, and their total probabilities are already determined.

We consider the pheromones equal in both directions, the source and the destination receiving pheromones equal to the product of p(Node = 1), for example:

$$\mathrm{f(Cancer, Dyspnoea)} = \mathrm{f(Dyspnoea, Cancer)} =$$

$$\mathrm{p(Cancer} = 1) * \mathrm{p(Dyspnoea} = 1) = 0.9884 * 0.6959 = 0.6878.$$

The cost for this route would then be equal to $-log(f(Cancer, Dyspnoea)) = 0.3742$ on both directions, and similarly η of the route would be:

$$\eta = \frac{1}{-log(f(Cancer, Dyspnoea))} = 2.672$$

After finishing those initializations, we begin the ACO algorithm: Each ant starts from a root node, in this example's case, Pollution and Smoker. Move to next state with a transition probability, first in Cancer, as it is the only child, and then on either Dyspnoea or Xray based on a probability distribution given by the probability formula in the original ACO:

$$\mathrm{p(source, destination)} = \frac{\tau(\mathrm{source, destination})^{\alpha} \cdot \eta(\mathrm{source, destination})^{\beta}}{\sum_{a} \tau(\mathrm{source, destination})^{\alpha} \eta(\mathrm{source, destination})^{\beta}}$$

If we want to apply softmax, for faster convergence towards the current most promising nodes, we exponentiate the numerator and sum up those exponentiation in the denominator. Based on this, more ants will favour Dyspnoea as it has bigger pheromone factor and η.

We compute all ants' costs by summing all the routes from source to destination and updating our current best ant and its cost. If later iterations have a worse cost or identify the same path, we stop, as the algorithm has converged to this local optimum ant.

Following the given example, the convergence path and cost: (Smoker -> Cancer -> Dyspnoea, 0.7425)

In Figure 7, we have coloured this path red.

For Bayesian networks with more nodes, we have sorted the ants based on the cost and the length of the path from a root node to a leaf for hyperparameter tuning purposes and to highlight the trade-off between the best cost and the length of an ant's route.

4. Results

The hyperparameter search was rather shallow, as we have started the experiments on the small BNs where the algorithm converged without much trouble, and when switching to larger examples, the issues came from the structural suppositions:

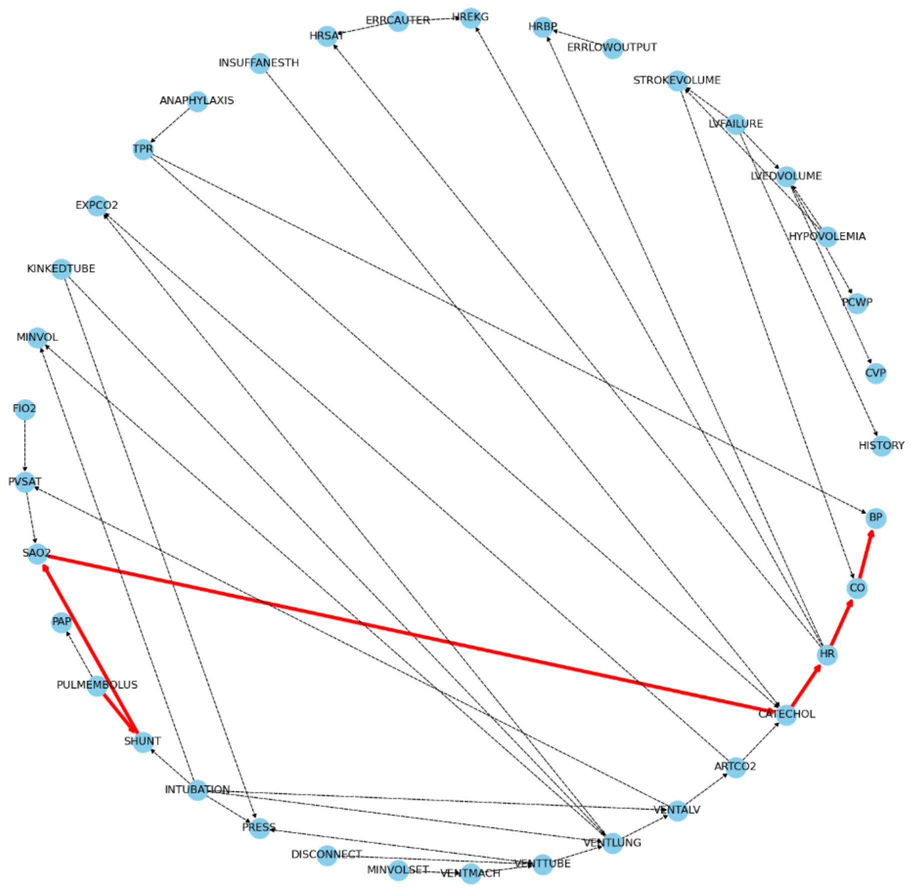
The evaporation factor is set to 0.99 as mentioned in the Methodology.

$\alpha = 2$ and $\beta = 3$ were chosen small, as each ant has to do products based on those 2 hyperparameters, and we have considered that the pheromones may oversaturate and $\beta > \alpha$ keeps the ants from fully discarding some initially unpromising paths. Looking back, these choices might have been detrimental, favoring early convergence.

Q value was set to 1 for simplicity to keep the pheromones' update formula coherent with the initial products with $\eta_{ij} = \mathrm{cost}_{\mathrm{ij}}^{-1}$.

At last, the results were computed for 1000 ants and 1000 epochs. The number of ants was constantly increased to make sure they visited the larger networks, as for fewer ants, they would overlap fast and fail to explore longer routes. This choice was updated when operating on the ALARM dataset, because, in the smaller settings, there were no more than tens of possible paths to explore and less than 50 ants with randomization would be able to cover completely the search space.

Analyzing the qualitative results, when ordering based on their costs and lengths, we see that the first best longer ants explore ALARM nicely (Figure 8), although their cost is two orders of magnitude higher than the best found one (Table 1).



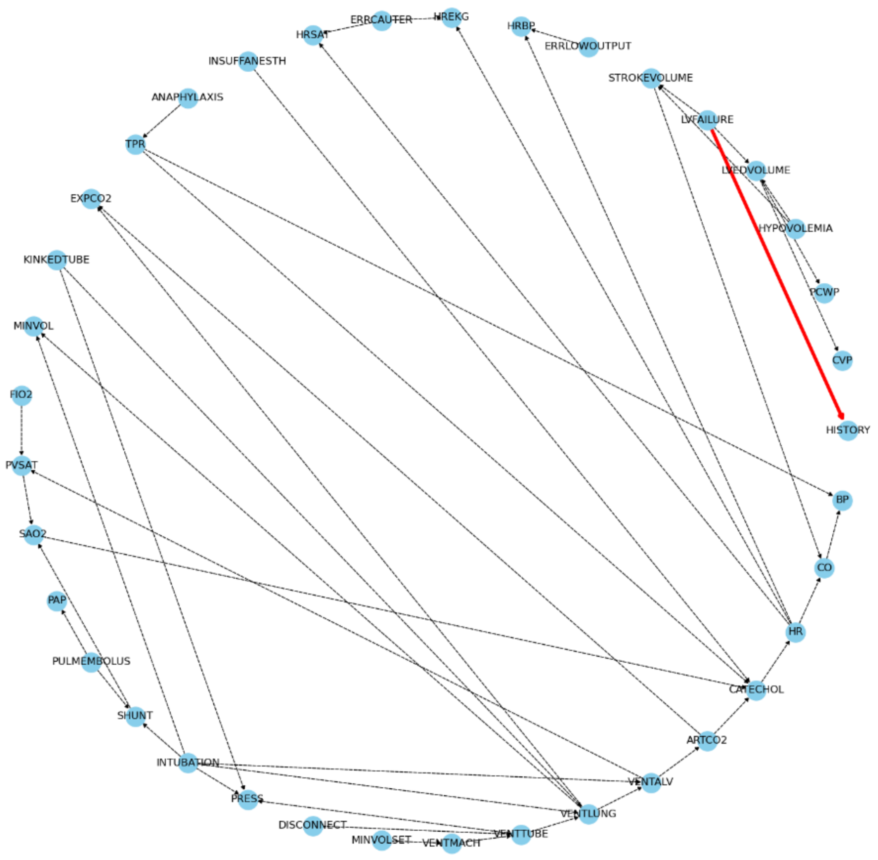


Fig. 8. 35th best ant vs best ant chain walk in ALARM

A qualitative analysis justifies the choice of small values for α and β by illustrating the fast impact in computation when adding complexity to the algorithm: Softmax Normalization affects the running time of the algorithm by one order of magnitude, while keeping the costs comparable and giving better qualitative paths. However, we see that the running time does not explode with the network size. A positive aspect that validates our hypotheses, but that might be influenced by the early convergence of the ants.

Table 1

ACO-Corr relationship between dataset, solution's speed and best cost

BN	Normalization	Time for solution	Best cost
CANCER	Sum	8.11e-04	6.13e-01
	Softmax	1.41e-03	7.43e-01

ASIA	Sum	7.50e-02	2.82e-01
	Softmax	1.10e-01	7.38e-01
ALARM	Sum	1.16e-01	1.07e-01
	Softmax	1.47e-01	1.41e0

5. Conclusions

Our algorithm combines two canonical ideas in self-organising systems (ACO) and AI (Bayesian Networks) to facilitate the rapid finding of high-correlation vertices in a Bayesian graph without suffering the downfall of vanishing probability. The costs presented in Table 1 have a magnitude of 1e-1 to 1e0, whereas, by applying probability multiplications with Variable Elimination, for each node, we diminish the cumulative probability with an order of magnitude 1e1, thus getting to smaller values in only two nodes distance from the source.

However, our approach still has structural flows that impose applying ACO-Corr on a niche of BNs or require coming up with variations:

Bayesian Networks can deal with categorical variables, whereas ACO-Corr needs binary probabilities to be initialised and run. In this case, to not destroy the meaning of information, we should modify the graph such that we create k binary nodes with probabilities of "class k" versus the rest and with connections between parents and all of them, them fully connected one to another, and from them to the following effects. This would lose the benefit of applying ACO as the spatial complexity would become exponential: Nodes spatial complexity would rise from $O(N)$ to $O(\frac{2N * 2^{k}}{k})$ and, consequently it would exponentiate the number of edges and the exploration steps.

In the current format, ACO-Corr has to start from an initial node. In contrast, Variable Elimination, Junction Trees or Statistical Independence Tests can walk through the BN in both directions to obtain the needed query or correlation value.

Our algorithm is well-behaved for BNs with similar lengths from root causes to leaf effects, as seen in CANCER and ASIA, where the ants did not cheat exploring the network because there was no incentive to bypass nodes. However, paths' imbalance towards a target effect contributes to ants' greedy behavior, resulting in them cheating the Bayesian network. In ALARM, even with the Softmax change in place, the first ten ants are of small lengths.

As future work, we plan on improving current limitations and applying ACO-Corr to BNs used in industry, as we presented in the introduction that they are a well-established prediction tool in car assurances, credit analysis and clinical decisions.

R E F E R E N C E S

[1]. Masmoudi, K., Abid, L., & Masmoudi, A. (2019). Credit risk modeling using Bayesian network with a latent variable. Expert Systems with Applications, 127, 157-166.
[2]. de Zoete, J., Sjerps, M., Lagnado, D., & Fenton, N. (2015). Modelling crime linkage with Bayesian networks. Science & Justice, 55(3), 209-217.
[3]. Ma, S. X., Dhanaliwala, A. H., Rudie, J. D., Rauschecker, A. M., Roberts-Wolfe, D., Haddawy, P., & Kahn Jr, C. E. (2023). Bayesian Networks in Radiology. Radiology: Artificial Intelligence, 5(6), e210187.
[4]. De Campos, L. M., Fernandez-Luna, J. M., G´amez, J. A., & Puerta, J. M. (2002). Ant colony optimization for learning Bayesian networks. International Journal of Approximate Reasoning, 31(3), 291-311.
[5]. Ji, J., Hu, R., Zhang, H., & Liu, C. (2011). A hybrid method for learning Bayesian networks based on ant colony optimization. Applied Soft Computing, 11(4), 3373-3384
[6]. Salama, K. M., & Freitas, A. A. (2013). Learning Bayesian network classifiers using ant colony optimization. Swarm Intelligence, 7, 229-254
[7]. Daly, R., & Shen, Q. (2009). Learning Bayesian network equivalence classes with ant colony optimization. Journal of Artificial Intelligence Research, 35, 391-447.
[8]. Jun-Zhong, J. I., ZHANG, H. X., Ren-Bing, H. U., & Chun-Nian, L. I. U. (2009). A Bayesian network learning algorithm based on independence test and ant colony optimization. Acta Automatica Sinica, 35(3), 281-288.
[9]. Wang, C., Liu, S., & Zhu, M. (2012). Bayesian network learning algorithm based on unconstrained optimization and ant colony optimization. Journal of Systems Engineering and Electronics, 23(5), 784-790.
[10]. Chickering, D., Geiger, M., & Heckerman, D. (1994). Learning Bayesian networks is NP-hard (Technical Report). Advanced Technologies Division, Microsoft Corporation, Redmond, WA
[11]. Y. Wu, J. McCall and D. Corne, "Two novel Ant Colony Optimization approaches for Bayesian network structure learning," IEEE Congress on Evolutionary Computation, Barcelona, Spain, 2010, pp. 1-7, doi: 10.1109/CEC.2010.5586528.
[12]. Pinto, P. C., Nagele, A., Dejori, M., Runkler, T. A., & Sousa, J. M. (2009). Using a local discovery ant algorithm for Bayesian network structure learning. IEEE transactions on evolutionary computation, 13(4), 767-779.
[13]. E. Taskensen, https://github.com/erdogant/bnlearn?tab=readme-ov-file, https://bnlearn.com/bnrepository, Last access: 17th October 2024
[14]. Raileanu, Silviu, et al. "Open-control paradigm: holonic implementation." ESM’09, International conference on industrial engineering and systems management. 2009.
[15]. Dorigo, M., Maniezzo, V., & Colorni, A. (1991). The ant system: An autocatalytic optimizing process.