16 Structure Learning of Bayesian Networks by Hybrid Genetic Algorithms

Pedro Larrañaga, Roberto Murga, Mikel Poza and Cindy Kuijpers

Department of Computer Science and Artificial Intelligence
University of the Basque Country,
P.O. Box 649, 20080 San Sebastián, Spain.

ABSTRACT This paper demonstrates how genetic algorithms can be used to discover the structure of a Bayesian network from a given database with cases. The results presented, were obtained by applying four different types of genetic algorithms – SSGA (Steady State Genetic Algorithm), GAeλ (Genetic Algorithm elistist of degree λ), hSSGA (hybrid Steady State Genetic Algorithm) and the hGAeλ (hybrid Genetic Algorithm elitist of degree λ) – to simulations of the ALARM Network. The behaviour of these algorithms is studied as their parameters are varied.

16.1 Introduction

In recent years, the search for the structure of a Bayesian network able to reflect all existing relations of interdependence in a database of cases has constituted a research topic of fundamental importance. Although the first algorithms were related to tree and polytree structures (e.g. [Chow68, Rebane89]), research has been concentrated upon multiple connected structures [Fung90, Herskovits90, Cooper92, Bouckaert93, Wedelin93, Lauritzen93, Chickering95, Bouckaert94]. For an introduction on Bayesian networks, see [Pearl88].

In this article we propose to obtain Bayesian network structures with the help of an intelligent search process based on genetic algorithms.

Evolutionary algorithms are probabilistic search algorithms which simulate natural evolution. They were proposed about 30 years ago. Their application to combinatorial optimization problems has, however, only recently become an actual research topic. Three different types of evolutionary algorithms exist: genetic algorithms [Holland75, Goldberg89, Davis91], evolution strategies [Schwefel67] and evolutionary programming [Fogel62]. This paper, however, focusses upon genetic algorithms (GAs). GAs are search algorithms based on the mechanics of natural selection and genetics. They combine "survival of the fittest" among string structures with a structured yet randomized information exchange to form a search algorithm which, under certain conditions, evolves to the optimum with probability 1 [Eiben90, Chakraborty93, Rudolph94].

In GAs the search space of a problem is represented as a collection of individuals. The individuals are represented by character strings, which are often referred to as *chromo-*

[1] *Learning from Data: AI and Statistics V.* Edited by D. Fisher and H.-J. Lenz. ©1996 Springer-Verlag.

[2] We thank Gregory F. Cooper for providing his simulation of the ALARM Network.

This work was supported by the Diputación Foral de Gipuzkoa, under grant OF 94/0762, by the Fondo de Investigación Sanitaria, Ministerio de Sanidad y Consumo, under grant 94/1370, and by the grant PI94/78 from the Gobierno Vasco - Departamento de Educación, Universidades e Investigación.

somes. The purpose of a GA is to find the individual from the search space with the best "genetic material". The quality of an individual is measured with an objective function. The part of the search space to be examined in each iteration is called the *population*.

A genetic algorithm works approximately as follows. First, the initial population is chosen at random, and the quality of each of its individual is determined. Next, in every iteration parents are selected from the population. These parents produce children, which are added to the population. For all newly created individuals of the resulting population a probability near zero exits that they "mutate", i.e. that they change their hereditary distinctions. Later, some individuals are removed from the population according to a selection criterion in order to reduce the population to its initial size. One iteration of the algorithm is referred to as a *generation*.

The operators which define the child production process and the mutation process are called the *crossover operator* and the *mutation operator*, respectively. Mutation and crossover play different roles in the GA. Mutation is needed to explore new states and helps the algorithm to avoid being trapped on local optima. Crossover should increase the average quality of the population. By choosing adequate crossover and mutation operators, as well as a reduction mechanism, the probability that the GA results in a near-optimal solution in a reasonable number of iterations is enlarged.

In Figure 1 we show the basic structure of a genetic algorithm (GA).

```
BEGIN GA
    Obtain the initial population at random.
    WHILE NOT stop DO
        BEGIN
            Select parents from the population.
            Produce children from the selected parents.
            Mutate the children.
            Add the children to the population.
            Reduce the population to its original size.
        END
END GA
```

FIGURE 1. Pseudo code of a genetic algorithm (GA).

16.2 Proposed Approach

We represent a Bayesian network structure by a *connectivity matrix* $C = (c_{ij})_{i,j=1,..,n}$, where

$$c_{ij} = \begin{cases} 1 & \text{if } (j \text{ is a parent node of } i) \text{ and } (i > j), \\ 0 & \text{otherwise.} \end{cases}$$

The inequality $i > j$ originates in the assumed ancestral order between the variables. Because of the inequality the crossover and mutation operators to be used are closed operators.

We consider four different genetic algorithms to which we refer as the SSGA (Steady State Genetic Algorithm), the GAeλ (Genetic Algorithm elitist of degree λ), the hSSGA (hybrid Steady State Genetic Algorithm) and the hGAeλ (hybrid Genetic Algorithm elitist of degree λ). Here the word *hybrid* indicates the addition of a *local optimizer* to the algorithm.

In an iteration of the SSGA and the hSSGA only one new individual is created, while in the GAeλ and the hGAeλ the generation replacement has a global character. In all algorithms, the reduction criterion is elitist. In the SSGA and the hSSGA, the created individual is compared with the worst existing individual at the time of creation. In the GAeλ, and the hGAeλ, however, the population at time $t+1$ consists of the λ best individuals drawn from the union of two sets: (1) the set of the λ individuals existing at time t, and (2) the λ children created from the individuals at time t.

The behaviour of all algorithms is studied with the help of three different *population sizes* λ ($\lambda = 10$, $\lambda = 50$, $\lambda = 100$).

The *objective function* used to evaluate the quality of a structure, is based on the formula proposed by Cooper and Herskovits [Cooper92], for a joint probability $P(B_S, D)$ of a Bayesian network structure B, and a database D, expressed in terms of the natural logarithm. Therefore, our aim is to find the structure with the highest joint probability.

The *selection function* is based on the rank of the objective function. If we denote by I_t^j the j-th individual of the population at time t, and by $\text{rank}(g(I_t^j))$ the rank of its objective function, the probability $p_{j,t}$ that individual I_t^j is selected to be a parent is equal to

$$p_{j,t} = \frac{\text{rank}(g(I_t^j))}{\lambda(\lambda+1)/2}.$$

The *reproduction function* to be used is the so-called 1-point crossover operator. Following the selection of two parents, the probability that these parents are crossed is 1. This probability makes it feasible to compare the algorithms. The *mutation operator* consists of the probabilistic alteration of the bits, which represent the connectivity matrix. This alteration is performed with a probability near to zero. We consider two different mutation probabilities p_m, namely $p_m = 0.001$ and $p_m = 0.01$.

The algorithms *stop* when either, 10,000 structures have been evaluated or when in 1000 successive iterations of the algorithm, the evaluation of the best structure in the current population corresponds with the average evaluation over the structures in that population.

The initial population of network structures is generated at random, subject to the restriction that a node in a network structure never has more than m parent nodes (in our case, $m = 4$).

After applying the crossover and mutation operators, the created structures do not necessarily satisfy the restriction that each node of a network structure has at most m parent nodes. To enforce this restriction, the SSGA and GAeλ algorithms select n parent nodes at random, ($0 \leq n \leq m$), for each node of a created network structure. This approach will give poor results. Therefore, we try a second approach that is the hybridization of each of SSGA and GAeλ algorithms with the help of a *local optimizer*. This optimizer selects the best subset of at most m parent nodes for each node in a network structure. The process of generating structures and the application of the local optimizer, is repeated in every iteration of the algorithm.

TABLE 16.1. Evaluation of the ALARM Network structure.

Number of cases ALARM Network	$logP(D\|B_S)$
100	-6.3860e02
200	-1.1413e03
500	-2.6461e03
1000	-5.0345e03
2000	-9.7291e03
3000	-1.4412e04
10,000	-4.7086e04

16.3 Results

We describe the results of an experiment in which a database generated by simulation of a Bayesian network is used to search for the structure which has a maximal joint probability. This joint probability is compared with the corresponding value of the structure of the initial network structure. Also the Hamming distance between both structures, and the number of evaluations needed to obtain convergence are considered. All results were obtained with a SPARCserver 1000 under operating system Solaris 2.3.

We applied the algorithms to a database of 10,000 cases generated with the ALARM Network, which was constructed by Beinlinch *et al.* [Beinlich89] as a prototype to model potential anesthesia problems in the operating room. The simulation of the 10,000 cases of this network has been achieved with the help of a Monte Carlo technique developed for Bayesian networks by Henrion [Henrion88]. It corresponds with the first 10,000 cases generated by Herskovits [Herskovits91]. We have considered different subsets consisting of the first 100, 200, 500, 1000, 2000, 3000, and 10,000 cases from the original database. The evaluations of the initial structures for the different databases can be seen in Table 16.1.

All algorithms are evaluated with respect to the population size and the mutation rate. For every possible combination of parameters 10 executions were carried out. Therefore, the total number of evaluations performed for every database of cases was $2 \times 2 \times 3 \times 2 \times 10 = 240$.

In Table 16.2 and Table 16.3 the results are presented. BOF refers to the best value found of the objective function and AOF is the average value of the objective function. HD refers to the Hamming distance between the ALARM Network structure and the one with the best objective function, while the number between parentheses is the average Hamming distance.

By comparing Table 16.2 with Table 16.3, we see the importance of the local optimizer. Table 16.2 indicates that the evaluation function of the ALARM Network was improved for small population sizes (λ=10) only, but all results of the hybrid algorithms hSSGA and hGAeλ (see Table 16.3) were better than the ones presented in Table 16.1.

Another remarkable point is the small variability in the results found by the hybrid algorithms (see Table 16.3) with respect to the ones obtained by the SSGA and the GAeλ (see Table 16.2).

TABLE 16.2. Results obtained with the SSGA and the GAeλ.

		SSGA			GAeλ		
		$\lambda = 10$	$\lambda = 50$	$\lambda = 100$	$\lambda = 10$	$\lambda = 50$	$\lambda = 100$
	BOF	-6.2014e02	-6.2363e02	-6.3223e02	-6.2090e02	-6.2650e02	-6.4073e02
100	AOF	-6.3952e02	-6.4680e02	-6.5899e02	-6.3993e02	-6.5037e02	-6.6497e02
	HD	31 (48.2)	32 (51.7)	34 (56.5)	37 (49.4)	38 (55.0)	51 (65.7)
	BOF	-1.1256e03	-1.1307e03	-1.1432e03	-1.1253e03	-1.1348e03	-1.1587e03
200	AOF	-1.1556e03	-1.1673e03	-1.1911e03	-1.1543e03	-1.1755e03	-1.2038e03
	HD	19 (38.5)	19 (42.4)	34 (52.9)	17 (38.1)	22 (45.5)	37 (57.9)
	BOF	-2.6350e03	-2.6438e03	-2.6741e03	-2.6354e03	-2.6492e03	-2.7069e03
500	AOF	-2.6918e03	-2.7186e03	-2.7740e03	-2.6908e03	-2.7262e03	-2.7946e03
	HD	11 (34.8)	18 (41.2)	29 (52.9)	12 (33.2)	22 (45.1)	41 (58.8)
	BOF	-5.0279e03	-5.0404e03	-5.0799e03	-5.0286e03	-5.0611e03	-5.1533e03
1000	AOF	-5.1117e03	-5.1741e03	-5.2491e03	-5.1137e03	-5.2026e03	-5.3185e03
	HD	4 (30.2)	15 (40.8)	28 (53.7)	2 (30.2)	20 (45.3)	38 (60.5)
	BOF	-9.7200e03	-9.7440e03	-9.8159e03	-9.7200e03	-9.7538e03	-9.8769e03
2000	AOF	-9.8504e03	-9.8890e03	-9.9481e03	-9.8436e03	-9.9188e03	-10.0068e03
	HD	3 (30.7)	11 (41.4)	28 (54.9)	3 (30.1)	16 (47.6)	43 (66.1)
	BOF	-1.4404e04	-1.4425e04	-1.4485e04	-1.4405e04	-1.4450e04	-1.4649e04
3000	AOF	-1.4578e04	-1.4722e04	-1.4957e04	-1.4580e04	-1.4810e04	-1.5173e04
	HD	1 (29.0)	12 (41.4)	29 (56.2)	2 (30.3)	22 (50.5)	48 (68.8)
	BOF	-4.7079e04	-4.7118e04	-4.7279e04	-4.7079e04	-4.7163e04	-4.7362e04
10,000	AOF	-4.7462e04	-4.8019e04	-4.8506e04	-4.7531e04	-4.8264e04	-4.9083e04
	HD	2 (31.1)	15 (46.4)	34 (61.2)	2 (32.5)	26 (56.6)	42 (72.3)

Because of the mentioned considerations, we decided to analyze the algorithms SSGA and GAeλ separately from the hybrid algorithms.

The analysis of the 1680 (240×7) runs has been carried out using the Kruskal-Wallis test, which looks for differences statistically significants.

16.3.1 Analysis of the results of the SSGA and the GAeλ

Objective Function

The average behaviour of the SSGA is similar to the GAeλ. There are no statistically significant differences in any of the 7 databases. However, statistically significant differences exist with respect to the population size, obtaining the best performance with λ=10. The results found with the mutation rate p_m=0.001 are significantly better than the ones found with p_m=0.01.

Number of evaluations needed until convergence

For the population sizes λ=50 and λ=100, none of the algorithms was able to converge under the stop criterion earlier described. For λ=10, the convergence velocity, both of the SSGA as well as of the GAeλ, is significantly larger for p_m=0.001 than for p_m=0.01 for

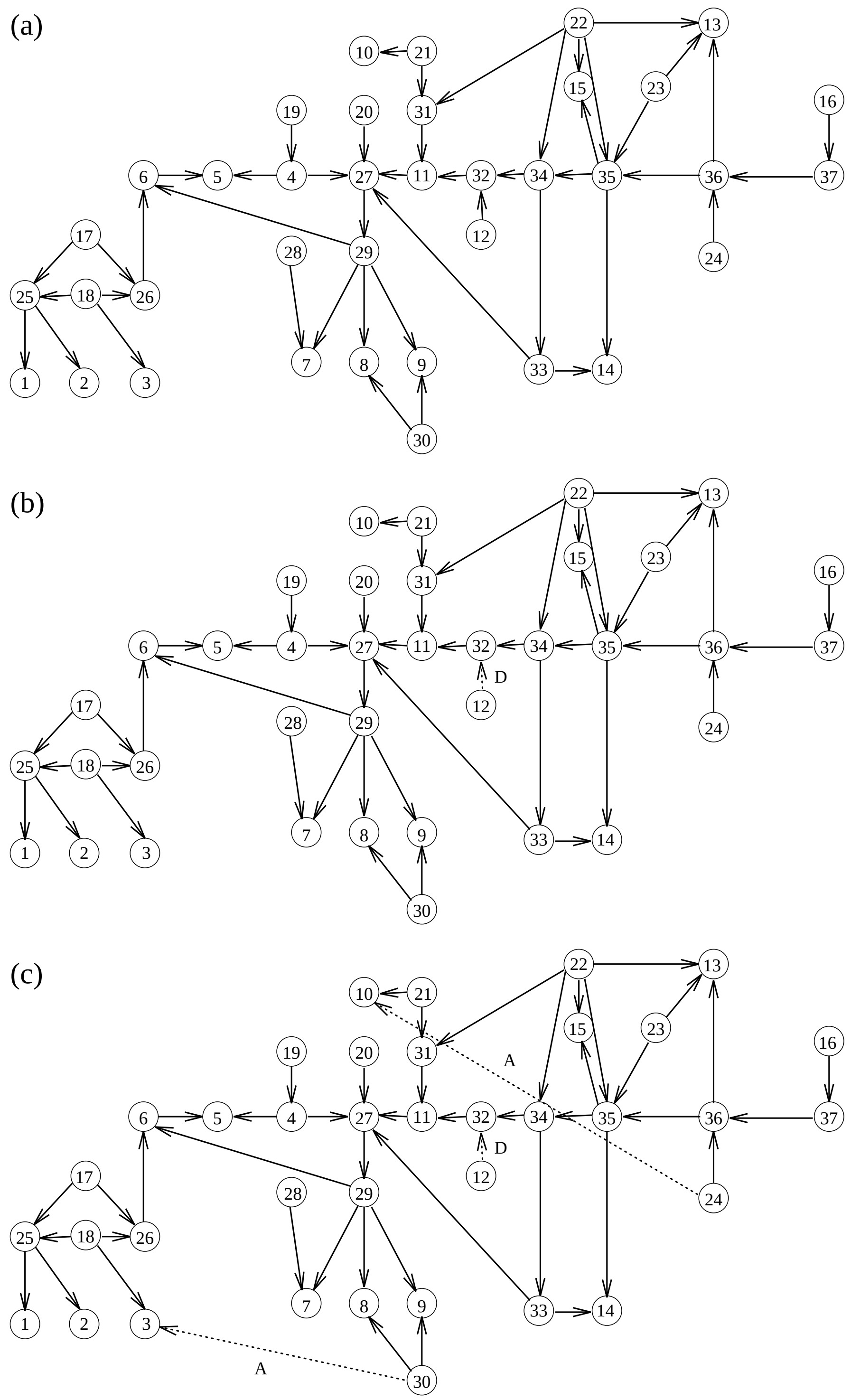

FIGURE 2. (a) The ALARM network structure. (b) The network structure learned by the hSSGA and the hGAeλ algorithms from a 10000-case and 3000-case database generated from the ALARM network. (c) The network structure learned by the hSSGA and the hGAeλ algorithms from a 2000-case database generated from the ALARM network. Arcs that are added or deleted with respect to the ALARM network are indicated with A and D respectively.

TABLE 16.3. Results obtained with the hSSGA and the hGAeλ.

		hSSGA			hGAeλ		
		$\lambda = 10$	$\lambda = 50$	$\lambda = 100$	$\lambda = 10$	$\lambda = 50$	$\lambda = 100$
	BOF	-6.1901e02	-6.1901e02	-6.1901e02	-6.1901e02	-6.1901e02	-6.1901e02
100	AOF	-6.1928e02	-6.1932e02	-6.1921e02	-6.1945e02	-6.1927e02	-6.1920e02
	HD	34 (35.6)	34 (35.6)	33 (35.6)	34 (35.7)	34 (35.6)	32 (35.3)
	BOF	-1.1249e03	-1.1249e03	-1.1249e03	-1.1249e03	-1.1249e03	-1.1249e03
200	AOF	-1.1249e03	-1.1253e03	-1.1249e03	-1.1249e03	-1.1249e03	-1.1250e03
	HD	19 (19.9)	19 (20.0)	20 (20.0)	19 (19.9)	18 (19.6)	17 (19.4)
	BOF	-2.6350e03	-2.6350e03	-2.6350e03	-2.6350e03	-2.6350e03	-2.6350e03
500	AOF	-2.6360e03	-2.6353e03	-2.6350e03	-2.6350e03	-2.6350e03	-2.6350e03
	HD	12 (13.0)	11 (12.5)	12 (12.3)	12 (12.6)	12 (12.1)	11 (12.1)
	BOF	-5.0279e03	-5.0279e03	-5.0279e03	-5.0279e03	-5.0279e03	-5.0279e03
1000	AOF	-5.0279e03	-5.0279e03	-5.0279e03	-5.0281e03	-5.0303e03	-5.0279e03
	HD	4 (4.0)	4 (4.0)	4 (4.0)	4 (4.1)	3 (4.1)	4 (4.1)
	BOF	-9.7200e03	-9.7200e03	-9.7200e03	-9.7200e03	-9.7200e03	-9.7200e03
2000	AOF	-9.7206e03	-9.7200e03	-9.7200e03	-9.7205e03	-9.7200e03	-9.7200e03
	HD	3 (3.1)	3 (3.0)	3 (3.0)	3 (3.1)	3 (3.0)	3 (3.0)
	BOF	-1.4404e04	-1.4404e04	-1.4404e04	-1.4404e04	-1.4404e04	-1.4404e04
3000	AOF	-1.4404e04	-1.4404e04	-1.4404e04	-1.4404e04	-1.4404e04	-1.4407e04
	HD	1 (1.0)	1 (1.0)	1 (1.0)	1 (1.0)	1 (1.0)	1 (1.2)
	BOF	-4.7078e04	-4.7078e04	-4.7078e04	-4.7078e04	-4.7078e04	-4.7078e04
10,000	AOF	-4.7078e04	-4.7082e04	-4.7078e04	-4.7078e04	-4.7083e04	-4.7078e04
	HD	1 (1.0)	1 (1.1)	1 (1.0)	1 (1.0)	1 (1.1)	1 (1.0)

all databases considered.

The poor results found with this first approach we attribute to the "blind" parent selection process used for maintaining the restriction on the maximum number of parent nodes.

16.3.2 Analysis of the results of the hSSGA and the hGAeλ

Objective Function

For the small databases (100, 200 and 500 cases) we found statistically significant differences for the mutation rate, obtaining the best performance for p_m=0.01. Differences due to population size were only significant for the 500-case database, where performance improved as the population size became larger. The large databases (1000, 2000, 3000, 10,000 cases) did not give statistically significant differences with respect to any of the three parameters considered (the type of the GA, the population size and the mutation rate).

Number of evaluations needed until convergence

The stop criterion was sufficient for guaranteeing the convergence of the hybrid algorithms. We found, for all databases, that the hSSGA converges significantly faster than the hGAeλ. Moreover, the algorithms converged faster as the population size became smaller. Finally a mutation rate equal to 0.01 resulted in a faster convergence than a mutation rate of 0.001.

The best structure obtained by the hybrid algorithms coincided for both the algortihms and was found with both the 3000-case database as well as the 10,000-case database. If we compare this structure (see Figure 2(b)) with the ALARM Network (see Figure 2(a)), we see that the only difference between the two structures is the arc from node 12 to node 32, which is missing in the best structure found by the hybrid algorithms. The best structure found by the hybrid algorithms with the 2000-case database is shown in Figure 2(c). This structure has, in comparison with the ALARM Network, two additional arcs (the arc from node 24 to node 10, and the arc from node 30 to node 3) and one missing arc (the arc from node 12 to node 32).

The obtained improvements using the local optimizer, we interprete as an empirical demonstration of the validity of our hybrid approach. The local search related to every node involves that unimportant parts of the search space are not examined.

16.4 Conclusions and Future Research

We have illustrated how the genetic approach can be used in the Structure Learning of Bayesian networks from a database of cases.

First, we have tried an approach in which a “blind” selection process was used to “repair” created structures with nodes which have too many parent nodes. Second, we have followed a hybrid approach.The results of the latter approach are far better than the results of the former approach. In this case the results are independent of the generation gap and, in outline, also of the mutation rate and the population size.

In the future we plan to tackle the more general problem in which the assumption of the ancestral ordering between the variables is not assumed. Other potential research is related to the use of different evaluation functions, some of which appear in [Bouckaert93, Bouckaert94, Chickering95].

We expect that our approach can also be applied to dynamical Bayesian networks. Also, it would be interesting to investigate the use of other heuristical search methods, like i.e. tabu search.

16.5 References

[Beinlich89] Beinlich, I.A., Suermondt, H.J., Chavez, R.M., & Cooper, G.F. (1989). The ALARM monitoring system: A case study with two probabilistic inferences techniques for belief networks. *Proceedings of the Second European Conference on Artificial Intelligence in Medicine* (pp. 247–256).

[Bouckaert93] Bouckaert, R.R. (1993). Probabilistic network construction using the minimum description length principle. In M. Clarke, R. Kruse & S. Moral (Eds.) *Symbolic and Quantitative Approaches to Reasoning and Uncertainty – ECSQARU-93*, No. 747, Lectures Notes in Computing Science, pp. 41-48, Springer-Verlag.

[Bouckaert94] Bouckaert, R.R. (1994). Properties of Bayesian belief network learning algorithms. *Uncertainty in Artificial Intelligence, Tenth Annual Conference* (pp. 102–

109). San Francisco, CA: Morgan Kaufmann.

[Chakraborty93] Chakraborty, U.K., & Dastidar, D.G. (1993). Using reliability analysis to estimate the number of generations to convergence in genetic algorithms. *Information Processing Letters, 46*, 199–209.

[Chickering95] Chickering, D.M., Geiger, D., & Heckerman, D. (1995). Learning Bayesian networks: Search methods and experimental results. *Preliminary Papers of the Fifth International Workshop on Artificial Intelligence and Statistics* (pp. 112–128).

[Chow68] Chow, C.K., & Liu, C.N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory, 14*, 462–467.

[Cooper92] Cooper, G.F., & Herskovits, E.H. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning, 9*, 309–347.

[Davis91] Davis, L. (Ed.) (1991). *Handbook of Genetic Algorithms.* New York: Van Nostrand Reinhold.

[Eiben90] Eiben, A.E., Aarts, E.H.L., & Van Hee, K.M. (1990). *Global convergence of genetic algorithms: An infinite Markov chain analysis.* Computing Science Notes, Eindhoven University of Technology, The Netherlands.

[Fogel62] Fogel, L.J. (1962). Atonomous automata. *Ind. Res., 4*, 14–19.

[Fung90] Fung, R.M., & Crawford, S.L. (1990). CONSTRUCTOR: A system for the induction of probabilistic models. *Proceedings of AAAI* (pp. 762–769).

[Goldberg89] Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, MA: Addison-Wesley.

[Henrion88] Henrion, M. (1988). Propagating uncertainty in Bayesian networks by probabilistic logic sampling. In J. Lemmer & L. Kanal (Eds.) *Uncertainty in Artificial Intelligence, 2*, pp. 149–163, North-Holland.

[Herskovits91] Herskovits, E. H. (1991). *Computer based probabilistic-network construction.* Doctoral dissertation, Medical Information Sciences, Stanford University.

[Herskovits90] Herskovits, E. H., & Cooper, G.F. (1990). *Kutató: An entropy-driven system for construction of probabilistic expert systems from databases.* Report KSL-90-22, Knowledge Systems Laboratory, Medical Computer Science, Stanford University.

[Holland75] Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems.* The University of Michigan Press.

[Lauritzen93] Lauritzen, S.L., Thiesson, B., & Spiegelhalter, D.J. (1993). Diagnostic systems created by model selection methods-A case study. *Fourth International Workshop on Artificial Intelligence and Statistics* (pp. 93–105).

[Pearl88] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems.* San Mateo, CA: Morgan Kaufmann.

[Rebane89] Rebane, G., & Pearl, J. (1989). The recovery of causal polytrees from statistical data. In L. Kanal, T. Levitt & J. Lemmer (Eds.) *Uncertainty in Artificial Intelligence, 3*, pp. 175–182, North-Holland.

[Rudolph94] Rudolph, G. (1994). Convergence analysis of canonical genetic algoritms. *IEEE Transactions on Neural Networks, vol. 5, no. 1*, 96–101.

[Schwefel67] Schwefel, H.-P. (1967). *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie.* Basel: Birkhäuser.

[Wedelin93] Wedelin, D. (1993). *Efficient algorithms for probabilistic inference combinatorial optimization and the discovery of causal structure from data.* Doctoral dissertation, Chalmers University of Technology, Göteborg.