Structure Learning of Bayesian Networks by Evolutionary Algorithms

Based on the article of Larrañaga et al.

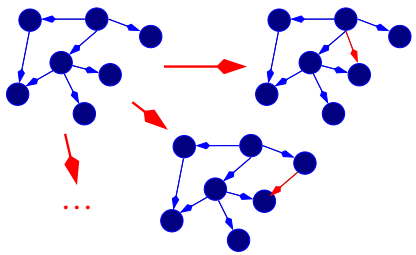
Floran Stuijt & Vincent v. Megen

Outline

- Structure learning
- Approaches to Structure Learning
- Genetic Structure Learning
- Results
- Conclusion
- Questions

Structure Learning (1)

- Given a existing structure, it is easy to generate data
- But how to get from data to the structure which gave rise to it?
- This task is called structure learning (SL)
- Very large search space
- Heuristic methods are necessary to cut down the time necessary to find an optimal graph



Structure Learning (2)

It has been shown that that the number of different structures $f(N)$ is given by

$$f(N) = \sum_{i=1}^{N} (-1)^{i+1} \binom{N}{i} 2^{i(N-i)} f(N-i) \tag{1}$$

N	Number of DAGs
1	1
2	3
3	25
⋮	⋮
8	783,702,329,343
9	1,213,442,454,842,881
10	4,175,098,976,430,598,143

Some Approaches to SL

- Hillclimbing
- Tabu search
- Simulated Annealing
- Evolutionary Algoritms

Hillclimbing: K2

- Probably named after the 2nd highest mountain on earth
- Greedy heuristic
- Starts with node lacking parents
- Adds parent to node if parent increases the score
- Stops if max. nr. of parents is reached, or addition of parent does not lead to higher score
- Algorithm assumes ordering of nodes

Tabu Search

- Essentially the same as hillclimbing, but with tabu list
- Choose an initial solution and determine quality of this solution
- Until quality is sufficient
  - Determine neighbor solutions and determine their quality
  - Choose the best neighbor solution and place the former in the tabu list
  - If the new solution is better, remember the new solution

Simulated Annealing

- Based on technique used in metallurgy, where material is cooled down in a controlled way to improve quality of product.
- Computer simulation of this process resulted in simulated annealing algorithm
- Can be regarded as a stochastic hill climbing algorithm
- A new state is chosen by a probability measure
- Probability is higher if score of neighbor state is higher

Evolutionary Algorithms

- Inspired by biological evolution
- What do we need?

Evolutionary Algorithms

- Inspired by biological evolution
- Initial population
- Fitness or objective function
- We need two operators:
  - Mutation operator
  - Cross-over operator
- Gene representation of state

Abstract Genetic Algorithm

- Make initial population at random
- Until quality is sufficient
  - Select parent from the population
  - Produce children from selected parents
  - Mutate the resulting individuals
  - Extend the population by adding the children to it
  - Reduce the extended population
- Output the best individual found

Genetic Structure Learning

Bayesian network is represented as $n \times n$ matrix C with elements:

$$c_{ij} = \begin{cases} 1 & \text{if } j \text{ is a parent of } i; \\ 0 & \text{otherwise.} \end{cases}$$

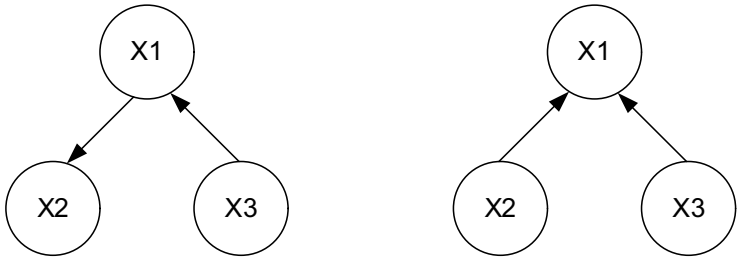
An individual is represented of concatenation of matrix columns:

$$c_{11}c_{21}\ldots c_{n1}c_{12}c_{22}\ldots c_{n2}\ldots c_{1n}c_{2n}\ldots c_{nn}\ldots$$

Example of a BN Representation

Which of the following diagrams is represented by this matrix?

$$C = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$



The Algorithm (1)

- Initial population is generated at random
- Individuals are only allowed to have four parents at max
- Objective function is based on formula of Cooper and Herskovits (1992)

(2) $$P(B_s, D) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!$$

where q_i denotes the number of states over the parents of, and r_i the number of states for a variable X_i.

- $N_{ij} = \sum_{k=1} r_i N_{ijk}$ and N_ijk is the number of cases in data in which variable X_i is in the kth state and parent of X_i in its jth state.
- Our goal is to maximize $P(B_s, D)$

The Algorithm (2)

- Individuals are selected to be a parent according to the following probability

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

where I_t^j denotes the jth individual of the population at time t, and $\mathrm{rank}(g(I_t^j))$ the rank of its objective function.

- This function avoids premature convergence by superindividuals
- Parents generate individuals by 1-point crossover

The Algorithm (3)

- Resulting individuals need to be repaired if no ordering is assumed
- Resulting graphs may be non DAGs
- Repair operator transforms resulting networks by randomly eliminating the edges that invalidate the DAG conditions
- Application of crossover function might generate individuals with > 4 parents
- Two methods of limiting the number of parents:
  - Select random subset of 4 parents
  - Use local optimizer which selects best subset of 4 parents

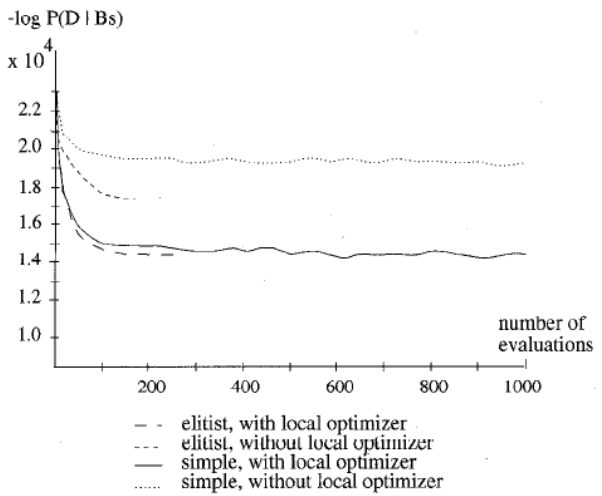
The Algorithm (4)

- Last but not least we must reduce our resulting population
- Again two methods:
  - Simply only regard the children as our new population (simple selection)
  - Choose the λ best graphs among parents and offspring (elitist selection)

Results (1)

- Data sets were generated from two well known networks, namely the ASIA and the ALARM networks
- Genetic approach was tested on resulting data sets
- Simulations were performed with the different parameters
  - Population size λ
  - Crossover probability p_{cr}
  - Mutation rate p_m
  - Ordering restriction (with or without ordering assumption)
  - Hybridization (with or without local optimizer)
  - Reduction criterion (simple/elitist)

Results (2)



Conclusion

- Evolutionary learning of BNs is useful when there is no ordering restriction assumed and a large population size
- The most promising approach is a hybrid algorithm that uses elitist reduction

References

- Cooper, G.F., & Herskovits, E.A. (1992). A Bayesian Method for the Induction of Probabilistic Networks from Data. Machine Learning, vol. 9, no. 4, pp. 309-347.
- Jensen, F. V., & Nielsen, T. D. (2007). Bayesian Networks and Decision Graphs, Second Edition. Springer Press.
- Larranaga, P., Poza, M., Yurramendi, Y., Murga, R. & Cuijpers, C. (1996). IEEE Transactions on Pattern Analysis and Machine Learning, vol. 18, no, 9, pp. 912-926.

Questions?