Asymptotic Analysis of a Direct Algorithm for Synthesizing a Minimal Joint Graph

Anatolii Maksimov[a], Arseniy Zavalishin[b] and Alexander Tulupyev[b]

[a] *St. Petersburg Federal Research Center of the Russian Academy of Sciences (SPC RAS), St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, 14-th Linia, V.I., No. 39, St. Petersburg, 199178, Russian Federation*

[b] *St Petersburg State University, Universitetskaya nab., 7/9, St. Petersburg, 199034, Russian Federation*

Abstract

Joint graphs are currently the primary way to define the global structure of algebraic Bayesian networks. In this paper, we give an overview of a direct algorithm for the synthesis of a minimal joint graph and carry out its asymptotic analysis. We also describe ways to achieve maximum efficiency. From a theoretical point of view, the problem has not been discussed previously.

Keywords 1

Algebraic Bayesian networks, joint graphs, labeled graph, complexity of algorithms, algorithms on graphs, depth-first search

1. Introduction

Currently, the amount of data being accumulated is increasing rapidly. One effective method of working with their increasing volume is the application of machine learning models [Jasenin et al. 2018]. Algebraic Bayesian networks (ABN) are one such model. They are represented by an undirected graph whose vertices contain knowledge patterns. The mathematical model of the knowledge pattern is the ideals of conjunctions, each element of which is given scalar or interval estimates of the probability of truth. The ability to work with the interval estimates makes the ABN a suitable tool for processing incomplete, inaccurate or not-numerical information.

In machine learning, the ABN is distinguished by several steps, including the stap of constructing a global structure or joint graph. One of the questions in this line of research is the computational complexity of the algorithms involved. They need to be reasonably computational. The aim of this work is to evaluate the asymptotics of the algorithm for constructing the minimal joint graph proposed by Oparin and Tulupyev [1].

2. Minimal joint graph

Before describing the algorithm and studying its complexity, it is necessary to define the joint graphs themselves.

Set the finite alphabet A and the finite set of vertices V. The set of vertices has a labeling function $W\colon V\to 2^A$ which compares each vertex $v \in V$ with its load W_v.

Definition: Two vertices u, v are main connected [2] if $W_u \cap W_v = \emptyset$ or there is a way of P out of u in v that

$$\forall p \in P \quad W_u \cap W_v \subseteq W_p$$

Definition: A graph is the joint graph [2] if any two of its vertices are main connected.

© 2020 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-2782/paper_11.pdf

Multiple joint graphs can be constructed over the same set of vertices with a fixed labeling function. For example, if all vertex loads coincide, the joint graph will be any tree (and generally any connected graph). For logic-probability inference purposes, acyclic joint graphs [2], therefore, ABN theory pays particular attention to the minimal joint graphs.

Definition: The joint graph is minimal [2] if it is minimal by inclusion.

In [3], it was shown that the minimal by inclusion and minimal by number of edges are achieved at the same time.

3. Minimal joint graph algorithm

Definition: A restriction of G_q graph G on a load of q is a pair of $\langle V_q, E_q \rangle$:

$$V_q = \{v \in V | q \subseteq W_v\},$$
$$E_q = \{(u, v) \in E \big| u, v \in V_q\},$$

where W_v is the load on the vertex v.

The algorithm that generates the minimal joint graph was described in [1]. Its pseudocode is shown below. The delegate(S) function returns an arbitrary representative of the set S. The component(G,v,q) function returns the connectivity component of a vertex v in the restriction of graph G to the load of q.

Table 1

Algorithm 1. Minimal joint graph algorithm.

```
Require $V,W$
Ensure $G = \langle V, E \rangle$
  1: $Q=\emptyset$
  2: $G = \langle V, \emptyset \rangle$
  3:  for all $u, v \in V, u \neq v$ do
  4:    if $W_u \cap W_v \neq \emptyset$ & $W_u \cap W_v \notin Q$ then
  5:        $Q \leftarrow W_u \cap W_v$
  6:    end if
  7: end for
  8: while $Q \neq \emptyset$ do
  9:    $q \leftarrow Q$
 10:    $S = \emptyset$
 11:    for all $v \in V$ do
 12:        if $q \subset W_v$ & $v \notin S$ then
 13:            if $S \neq \emptyset$ then
 14:                $d$ = delegate($S$)
 15:                $S$ = $S$ ∪component($G$,$v$,$q$)
 16:                $G.E \leftarrow (d, v)$
 17:            else
 18:                $S$ = component($G$,$v$,$q$)
 19:            end if
 20:        end if
 21:    end for
 22: end while
```

The algorithm sequentially traverses all possible pairwise intersection of loads of q (called separators). For each such intersection, an empty set of S is created. For each vertex of v, the load of which contains the current separator q, but which is not yet in the set S, the set S is combined with the connectivity component of the graph G, which contains v, in restriction to the load q. If the S was not empty, then in the graph G adds an edge between the vertex v and the arbitrary vertex of the set S. As a result, G will contain a minimal joint graph.

4. Introduction

Let's settle first with a known upper estimate of complexity, and then try to improve it.

Let us have sets of A and B. Mark for α, β and γ the time necessary to perform a check of $A \subset B$, to perform a check of $a \in A$ and to perform the operation of $A \cup B$ respectively. The number of vertices and the number of edges, respectively, are also given for *n* and *m*.

Estimate the cycle time in rows 8-22 then. The outer cycle while makes $|Q|$ iterations. Since the set Q contains pairwise interceptions vertex loads of no more than $\frac{(n-1)}{2}$, asymptotically this cycle can be estimated as $O(n^2)$.

The inner cycle for is iterated over all vertices, that is, makes n worth of iterations. The checks from line 12 are for $\alpha + \beta$. The check from line 13 is done in constant time. The delegate(S) function is knowingly performed for *O(n)* by supporting a vector of the logical type that contains information about *i*th vertex is included in S or not. The component*(G,v,q)* function can be performed for $O((n+m)\alpha)$ by modifying the depth first search [4] with an additional check of $q \subset W_v$. The operation of union in the accepted notation is performed for γ. Finally, adding an edge to the graph is performed in constant time. The resulting estimate is $O(n^2 \cdot n \cdot (\alpha + \beta + \gamma + n + \alpha(n+m)))$.

Now let's start improving this score. First of all, get rid of the summand of β, γ and n. We can use the disjoint-set-union (DSU) data structure [4], [5]. Tarjan showed [5] that queries of the species «to choose a representative» and «to unite sets» are executed for *O(a(n)),* where *a(n)* is the inverse Ackermann function (this estimate, however, is discussed [6], [7]). This function grows so slowly that, for example, its value from the Shannon number [8] is less than five, allowing DSU to be neglected in the assessment of asymptotics. Now the estimate is $O(n^2 \cdot n \cdot (\alpha + \alpha(n+m)))$. Asymptotically it is the same as $O(n^2 \cdot n \cdot \alpha(n+m))$.

The next step is to show that not every iteration of the cycle requires a depth-first search of the entire graph. Fix a separator of *q*. The connectivity components in a graph do not intersect. Similarly, the connectivity components in a restriction of q do not intersect. This means that we do not need to make a series of n search for $O(n \cdot \alpha(n+m))$, but only $O(\alpha(n+m))$. In fact, if the search conditions are completed, then the current vertex is not yet in *S*, which means that the connectivity component of the current vertex is not yet in *S*. In turn, the latter means that no vertex or edge will be viewed twice in a series of depth-first searches, meaning that it is necessary to do nothing more than one depth-first search of the entire graph. This is for $O(\alpha(n+m))$. Thus, the asymptotic with $O(n^2 \cdot n \cdot \alpha(n+m))$ decreased to $O(n^2 \cdot \alpha(n+m))$.

Let we give you a little bit of a distraction here. Any permutation can be represented as a product of disjoint cyclic permutations - such a representation is called a cyclic decomposition. The more cycles this decomposition contains, the more of them have a single length. That is, the more elements in a cyclic decomposition, the less transpositions are needed to obtain the identical from the current permutation. This statement is formalised as follows: the minimum number of transpositions required to produce an identical permutation is equal to the difference in permutation length and the number of cycles in the cyclic decomposition [9].

This approach can also be applied in the case under study. Note that in depth-first search for an edge corresponding to the separator counts exactly as many times as the ancestors (not only in the first generation) in the Hasse diagram [10] relative to the «contained» constructed over the set of separators. Vertices are slightly more common: the number of occurrences is the sum of the number of ancestors of all separators in the load of the vertex. The following theorem holds:

Theorem: The height of the Hasse diagram over the set of pairwise intersection of set from multiset the power of *n* is strictly less than *n*.

Prove this by induction. This is true for two vertices simply because the set of separators contains only one element. Now let's make the statement fair for *n*, prove it for *n+1*. Let there be a set of loads that the Hasse diagram has a height of *n+1*. Mark for *t* the end vertex of the path length of *n+1*. Since it is a separator, there are at least two loads of *tx* and *ty* containingt. Consider a third *s* load such that its crossing with *ty* lies in the path of the length of *n+1* in the diagram. Mark this crossing with *w*. Then $w \subset tx \cap s$. Then when you remove a load of *ty*, the height of the Hasse diagram will be reduced by at most one, because the only separator in the *n+1* length path that may cease to exist is the t separator. Then we can construct Hasse's diagram of height n over n vertex, and by induction assumption, that's impossible.

Because knowledge pattern loads consist of not many elements, the Hasse diagram comes out sparse. This means that asymptotically the vertices and edges in the depth-first searches series are counted *O(n)* times. This allows us to further improve the evaluation of the algorithm by reducing the exponent from $O(n^2 \cdot \alpha(n+m))$ to $O(n \cdot \alpha(n+m))$.

Let's estimate now the time of preprocessing 3-7. The cycle makes $O(n^2)$ iterations, each of which performs search for intersection of the sets, test the attachment of the set Q, add the set to the set Q, and test the set for emptiness. The last operation can be performed in constant time, while other operations on sets depend on their size. Limit their size to a constant of k. Then the intersection can be found, for example, for $O(k \log k)$, supporting the sets as self-balancing search trees [11], or for *O(k+k)* by two pointers, supporting the sets as ordered vectors. By supporting Q as a self-balancing search tree, you can add and search items for $O(k \log|Q|)$, where a factor of k is responsible for comparing the two sets. Since the size of Q is less than n^2, you can replace $O(\log|Q|)$ with $O(\log n^2) = O(2\log n) = O(\log n)$. Once summed, we get $O(1 + 2k + 2k\log k)$, which is asymptotically equivalent to $O(k \log k)$. Thus, the preprocessing can be performed for $O(n^2 k \log k)$.

5. Conclusion

The work carried out an asymptotic analysis of the minimal joint graph synthesis algorithm and specified implementation methods to achieve an optimal estimate. The main result is that the algorithm runs for $O(n \cdot \alpha(n+m))$ with a preprocessing of $O(n^2 k \log k)$. The results lead to the conclusion that computational complexity is acceptable. The algorithm seems appropriate.

6. Acknowledgements

The research was carried out in the framework of the project on SPIIRAS governmental assignment No. 0073-2019-0003, with the financial support of the RFBR (project No. 18-01-00626).

7. References

[1] V. V. Oparin, A. L. Tulupyev, Sintez grafa smezhnosti s minimalnym chislom reber: formalizaciya algoritma i analiz ego korrektnosti Tr. SPIIRAN, 11 (2009) 142-157.
[2] A. L. Tulupyev, S. I. Nikolenko, A. V irotkin, Osnovy teorii bajesovskih setej, Izd-vo S.-Peterb. un-ta, 2019.
[3] V. V. Oparin, A. A. Filchenkov, A. V. Sirotkin, A. L. Tulupyev, Matroidnoe predstavlenie semejstva grafov smezhnosti nad naborom fragmentov znanij, Nauchno-tekhnicheskij vestnik informacionnyh tekhnologij, mekhaniki i optiki, 4 (68) (2010).
[4] T. H. Cormen, E. L. Charles, L. R. Ronald, S. Clifford, Introduction to Algorithms, MIT press, 2009.
[5] R. E. Tarjan, A class of algorithms which require nonlinear time to maintain disjoint sets, Journal of computer and system sciences, 18, 2 (1979).
[6] M. Fredman, M. Saks, The cell probe complexity of dynamic data structures, Proceedings of the twenty-first annual ACM symposium on Theory of computing, 1989.
[7] K. Wu, E. Otoo, A Simpler Proof Of The Average Case Complexity Of Union-Find With Path Compression, Lawrence Berkeley National Laboratory, LBNL-57527 (2005).
[8] C. E. Shannon, Programming a computer for playing chess, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 41, 314 (1950).
[9] K. P. Bogart. Introductory Combinatorics. 2nd edition, San Diego: Harcourt, Brace, Jovanovich, 1990.
[10] G. Birkhoff, Lattice theory, American Mathematical Soc, 25 (1940).
[11] L. J. Guibas, R. Sedgewick, A Dichromatic Framework for Balanced Trees, Proceedings of the 19th Annual Symposium on Foundations of Computer Science, 1978.