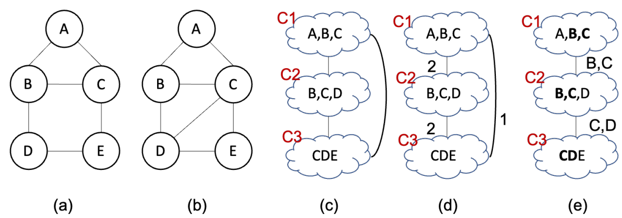

Figure 1: Example non-tree graph translation to a tree

1 Junction Tree Creation for a Cyclic Graph

Generally, to run VEA on graphs, people translate the graph to a structure that has the same characteristics as trees. In other words, a tree of maxcliques is made from the graph, so that every node may have more than one variable. The Running Intersection Property (R.I.P) is one of the characteristics that the tree of maxcliques should hold, as it holds for trees. R.I.P is essential for GJ in dealing with cyclic graphs.

R.I.P of maxcliques: Let $C_1, C_2, ..., C_l$ be an ordered sequence of maxcliques in graph G where l is the number of maxcliques. We say the ordering obeys the R.I.P if for all i¿1, there exists j ¡ i such that $C_i \cap (\cup_{k<i} C_k) = C_i \cap C_j$

In Figure 1 (e), assume the order is $C1, C2$ and $C3$ then the intersection between $C2$ and $C3$ (which is the node (C, D)) is equal to $C3 \cap (C2 \cup C1)$. The union $(C2 \cup C1)$ is called the history of $C3$.

Junction Tree (JT) of maxcliques: JT of maxcliques is a tree of the maxcliques which possesses the R.I.P property. There have exist many algorithms to translate graphs to JTs [KF09, CDLS07, JN07]. Here, we just provide a brief explanation of JT creation.

For a graph, there may be more than one JT. We need to find the best JT where the size of the largest maxclique is smaller than the size of largest maxcliques in other possible JTs from the same graph. Recall that the complexity of finding a marginal is dependent on the size of the largest clique in the graph. For example, we could take all the variables as one maxclique by adding fill-in edges, but in this case, the size of the largest maxclique would be equal to the size of V and the complexity of the inference (VEA) on that tree will be exponentially high. Extracting the best JT is NP-Complete.

It has already been proved that if a graph is triangulated then it has a JT of maxcliques which possesses R.I.P. [WJ08, BKR11]. In a triangulated graph, there should not be a cycle with more than 3 nodes. The triangulated graphs have a perfect elimination order and eliminating any node will not introduce any new fill-in edges.

Not all graphs have a perfect elimination order. For example, in Figure 1 (a), there is a cycle with 4 nodes and eliminating each of them will introduce a fill-in edge. If the graph is not triangulated (e.g. Figure 1 (a)), finding the best elimination order and adding new fill-in edges can lead us to the best triangulated graph, thus we can have the best JT as well. One should check all the combinations to find the best elimination order, hence finding the best elimination order is NP-Complete as well. The good news is that, in our join problem, the number of nodes is small and hence one can find the best elimination order with a small number of fill-in edges, manually. Nonetheless, greedy heuristic algorithms, such as min fill-in, work well even on the graphs with thousands of nodes.

The Min fill-in heuristic: In each step, the min fill-in heuristic adds one node in the elimination order O, and that node is the node which introduces the minimum number of new fill-in edges. If there are more than one node with minimum fill-in edges, it breaks the ties arbitrarily. In Figure 1, A should be eliminated first then among B, C, D and E, one is chosen randomly, and so on and so forth.

The min fill-in heuristic can provide the triangulated graph. For a given graph $G(V, E)$, we find the new fill-in edge set E' with the min fill-in heuristic and make the triangulated graph as $G'(V, E \cup E')$. Figure 1 (b) is one of the possible triangulated graphs for the graph Figure 1 (a). Note that when we add a fill-in edge, it is like we add a potential equal to 1 that can join with any other potentials. The main potentials for fill-in edges are calculated during inference. The min fill-in heuristic can also output all the maxcliques in the graph during triangulation since after elimination of a node, all its neighbors should be a clique. Any pair of maxcliques that have some shared variables are connected to each other so that they make a graph of the maxcliques (e.g. Figure 1 (c)).

So, the output of the min fill-in heuristic contains an elimination order O, a triangulated graph, and a graph of the maxcliques. The issue is how to derive a JT of maxcliques from the graph of maxcliques. There are several ways, and the easiest way is to apply the maximal spanning tree algorithm. This algorithm finds all the separator sets among maxcliques and chooses the edges with maximum separator sizes one-by-one to span the graph of maxcliques. A separator set is a set of nodes in the graph G that if we remove it, G is divided into disconnected sub-graphs.

Figure 1 shows all the steps of translating the graphs to JTs of maxcliques. (a) shows the original graph G, (b) shows the triangulated G, (c) contains the graph of maxcliques. (d) shows the weights (separator sizes) and finally (e) is the JT of maxcliques after applying the maximal spanning tree algorithm.

Once the graph is translated to a JT, the same VEA can be used to calculate the marginals.

For more information about translating graphs to JTs, please refer to [KF09, CDLS07, JN07, WJ08, BKR11].

References

[BKR11] Andrew Blake, Pushmeet Kohli, and Carsten Rother. *Markov random fields for vision and image processing*. MIT press, 2011.

[CDLS07] Robert G Cowell, Philip Dawid, Steffen L Lauritzen, and David J Spiegelhalter. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer Science & Business Media, 2007.

[JN07] Finn V Jensen and Thomas Dyhre Nielsen. *Bayesian networks and decision graphs*, volume 2. Springer, 2007.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[WJ08] Martin J Wainwright and Michael Irwin Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.