Bayesian nets
Meeting notes 1/20/2023

Everybody was there.

I. Zhongming outlined progress on calculating the ‘sample space’ of non-looping networks of n nodes. The strategy is the one he outlined last time: for n nodes, calculate for one link, two links, etc. It sounds like calculation for 3 nodes is in hand, with 4 nodes in progress. The hope is that with those as samples he will be able to figure out a general form of calculation for any n nodes.

An important part of the process is eliminating looped nodes of a certain number of links. This ties in with the general detecting-loops issue further discussed in the meeting.

II. Amber worked further on the graph idea: speed-to-world-detection of various network-change heuristics. She tried to use Patrick’s idea for generating a random non-looped network from last time: For an n-node network, start with a 2-node network. If it has a loop, try another. If it doesn’t, add a node and link. Does that form a loop? If it does, try another. If not, add another node…

But this appeared to be even slower and more computationally expensive than relying on PyAgrum.

III. In one of the purely positive points of the meeting, Dennis has developed both (a) a way of encoding evidence from the world that reflects the ‘time’ at which a node is activated, allowing us to distinguish networks that we couldn’t previously, and (b) the hybrid genetic algorithm we want to explore.

With regard to (b), the question arose of whether and when to introduce mutation along with hybridization. It was agreed that we should build the model so that we could vary these—hybridization without mutation, for example, or with it.

We also agreed that it was more natural build in mutation as something that happens after rather than before hybridization: picking two ‘best scoring’ networks, we first form a hybrid and then mutate the result, rather than picking two ‘best scoring’ networks, mutating each, and then forming a hybrid .

With regard to (a), Amber raised the general question of how we should score a representational network that had all the right links, but perhaps redundant links as well. We decided we could have two measures of ‘success’: (i) whether a network had enough of the links (‘adequacy’) , and whether it didn’t have more than it needed (‘economy’ or ‘parsimony’). Both of these appear to be considerations in building scientific theories, which is one reason to include them both. It may also turn out that we can only maximize one in certain circumstances with sacrifices to the other.

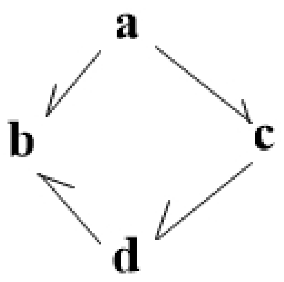
IV. On the lingering question of detecting and eliminating loops, Patrick outlined an idea that had seemed great to him until further thought. The idea was to construct loop-free networks for a set of nodes a, b, c, d, e, for example, by (a) taking all partitions of that set, (b) for each partition, order the subsets that it contains by ‘levels’ in all possible ways, and (c) for each level structure, form all networks possible with directed arrows from any number of nodes on a level to any number of nodes on lower levels.

This would capture lots of networks, all of which would be guaranteed to be non-loop networks because all directed arrows went ‘down.’ It would also capture both ‘branches’ and ‘joins’ in networks.

It also offers the possibility of generating a large set of non-looped 4-node networks, for example, and encoding them once and for all as a look-up table. Then when either asking for a random non-looped 3-node network or asking whether a given 4-node network is looped, we could merely use the look-up table rather than slowly recalculating.

Sophia pointed out that calculation of such a table could be even easier, since it is the structure of the network (rather than labels a, b, c…) that is what makes it looped. One could thus catalog structures, with every substitution of names for nodes counting as part of the list.

Unfortunately, on further thought it became clear that although partitions-and-levels would produce only non-looped networks, it wouldn’t produce them all. The following is a counter-example--


$$\begin{tikzcd}
 & a \arrow[dl] \arrow[dr] & \\
b & & c \arrow[dl] \\
 & d \arrow[ul] &
\end{tikzcd}$$

This is not a looped network, but there is no way of generating it by Patrick’s scheme since the link from d to b goes (harmlessly) ‘upward.’

At this point Patrick promised to think further about whether there might be a way of fixing the idea. We also discussed the idea of working with something like the set of networks it generated—a large and varied set of non-looped networks, with both branches and joins, even if not a set that contains them all. As long as we picked the world from such a set, we could still explore different representation-constructing heuristics. Patrick pointed out that this does go beyond trees (all of which are included), and that Judea Pearl’s initial work was explicitly limited to trees.

V. Since the meeting, Patrick has also found a discussion of loops in directed networks that may or may not be helpful. That is attached as scans of a few pages from Mark Newman’s book *Networks*.

For next week, Amber is going to incorporate the new programming that Dennis developed.

Dennis has sent out directions for accessing and running our files from everywhere. Let's see if we can all make them work.

Patrick is going to try to think more about loops, but with no promises.

Zhongming is going to keep working toward the calculation of n-node non-looped network space.

2:00 last time seemed to work for everyone. We'll aim for 2:00 next Friday as well.