GBN Cognitive Bribery

- A definition of a Gaussion Bayesian Network:
  - n random variables $X_1, \ldots, X_n$, a DAG $G = ([n], E)$, weights β_i for each $i \in [n$ and β_{ij} for each $ij \in E$.
  - The meaning is that a random variable X_i has mean $\mu_i = \beta_i + \sum_{j \in N^+(i)} \beta_{ji}\mu_j$, where $N^+(i)$ is the in-neighborhood of i
  - there is also a formula for the variances, but let's say that now we don't care about them
  - So, the mean of X_i is an affine combination of the means of the variables "preceding" it
- Optimization problem:
  - Given a GBN where n is a sink node and a threshold t, find a cheapest *change* such that $\mu_n \geq t$
  - A change is a vector $x \in \mathbb{R}^{n+|E|}$ which can be added to the vector β, such that $x + \beta$ are the new (node- and edge-) weights
  - The cost of a change is some function $f(x)$; the most generality I want to look at is where f is separable convex, meaning it's a sum $\sum_{i \in [n]} f_i(x_i) + \sum_{ij \in E} f_{ij}(x_{ij})$ and each f_i and f_{ij} is convex
    - for simplicity think of each f_i as $\| \bullet \|_1$
- Mathematical programming model
  - In the general optimization problem above, it's not difficult to see that it can be written as a mathematical program in variables β.
  - (1) For each $i \in [n]$, we have $\beta'_i = \beta_i + x_i$, and for each $ij \in E$ we have $\beta'_{ij} = \beta_{ij} + x_{ij}$
  - (2) Next, for each $i \in [n]$, we have a constraint $\mu_i = \beta'_i + \sum_{j \in N^+(i)} \beta'_{ji}\mu_j$
  - (3) Next, we have a constraint $\mu_n \geq t$
  - (4) Finally, we have an objective function $\min f(x)$
  - The constraints (1) and (3) are linear, however the constraints (2) are quadratic, because both β'_{ji} and μ_j are variables
  - This means it's a QCP - quadratically constraint program.
  - In the special case where we only allow modifying the vertex variables, constraint (2) becomes linear, and the problem is a linear program
  - Not sure about the edge-only model; it should be simpler, but it doesn't seem linear
- Experiment
  - For a set of "reasonable" or "interesting" instances (to be specified below), do the following
  - Construct the mathematical programming model

- Solve it using multiple solvers: Gurobi, IPopt, logical methods (quantifier elimination / CAD etc.)
- Compare performance (time to solve) and quality of solution (e.g. IPopt is a local solver = may not find a global optimum)
- Also compare how much "power" is lost by forbidding the modification of node or edge variables -- how much more expensive are the optima with these restrictions? But also, how much simpler (or more difficult) is it to solve those restricted variants?
- **Instances:**
    - baseline: random instances -- sample a random DAG (perhaps with pre-specified number of edges), then sample edge- and node-weights randomly
    - upgrade: adversarial instance construction by nevergrad
    - ideal: get real-world instances using a sociological survey (or perhaps from existing surveys; Powell et al. has survey data, but they use it to construct a BN, not a GBN)

- Bottom line of the thesis: experimental evaluation of optimization approaches on a network-modification problem

1 Linked Reference

Sep 16th, 2024

work Aryan

- specifikovat model, experiment [[GBN Cognitive Bribery]]

Unlinked References