January 2023

Root Cause Analysis Using Graph Representation of Constraints

Ming-Ho Stewart Siu

Root Cause Analysis Using Graph Representation of Constraints

ABSTRACT

When several quantitative variables are related through constraints and objectives, it can be difficult to understand why a certain quantity changes in magnitude after certain changes, or why a certain number seems larger or smaller than expected (as compared to a reference value). Large organizations which seek to optimize very large numbers of parameters to achieve constraints such as supply-demand matching face such problems, where the relationships between variables are controlled by a mixture of human processes and software algorithms. This disclosure describes scalable, flexible frameworks and searching techniques that improve transparency, e.g., enable root-cause analysis, for large families of variables. The techniques enable the understanding of the difference between two data-generating flows (versions) with comparable inputs, intermediate variables, and output variables.

KEYWORDS

- Root cause analysis
- Graph representation
- Business constraint
- Exact reasoning systems
- Machine learning
- Sensitivity analysis
- Heteroskedasticity
- Bayesian networks
- Linear programming
- Constraint solver

BACKGROUND

When several quantitative variables are related through constraints and objectives, it can be difficult to understand why a certain quantity changes in magnitude after certain changes, or why a certain number seems larger or smaller than expected (as compared to a reference value).

Large organizations which seek to optimize very large numbers of parameters to achieve constraints such as supply-demand matching face such problems, where the relationships between variables are controlled by a mixture of human processes and software algorithms. Even dedicated tools set up to provide specific explanations are difficult to maintain as the business logic evolves.

Exact reasoning systems and traditional knowledge bases represent data in terms of discrete semantic relationships such that exact logical deductions can be performed on them [1, 2]. However, these are less useful for generating explanations for numerical values. Constraint solvers such as linear programs can help in understanding numerical values, and indeed, it is possible to calculate reduced costs of solver variables [3] through sensitivity analysis to understand the causal impact of different variables. However, sensitivity analysis requires a combinatorial number of full simulation runs with all the exact details of the solver and is therefore computationally expensive [4].

While machine learning techniques [5] can be used to learn entity relationships, make inferences from learned results, and explain decisions, these don’t work well in situations with little data. Once a graph or network is defined, causal inference can be traced from one node to another. For example, if full conditional distributions are available, Bayesian-network techniques [6] can be applied. However, conditional distributions tend to come from data and are difficult to specify from domain knowledge.

Another type of explanatory inference comes from attributing neural network outputs to individual neurons [7, 8]. However, neural network layers are directional, with each neuron's dependence on others explicitly laid out, whereas general business variables can have hidden relationships. Also, neutral network attribution typically requires that the attributed weights sum

to unity, while in practice overlapping reasons, as long as they are properly ranked, can provide a satisfactory explanation.

DESCRIPTION

This disclosure describes scalable and flexible frameworks and searching techniques that improve transparency, e.g., enable root-cause analysis, for large families of variables as found in large business use cases. The techniques enable generating an understanding of the difference between two data-generating flows (versions) with comparable inputs, intermediate variables, and output variables. One-shot, approximate explanations provide useful insights even under incomplete information. Reasoning based on human-defined knowledge is permissible, even in situations with little data or with heteroskedastic data.

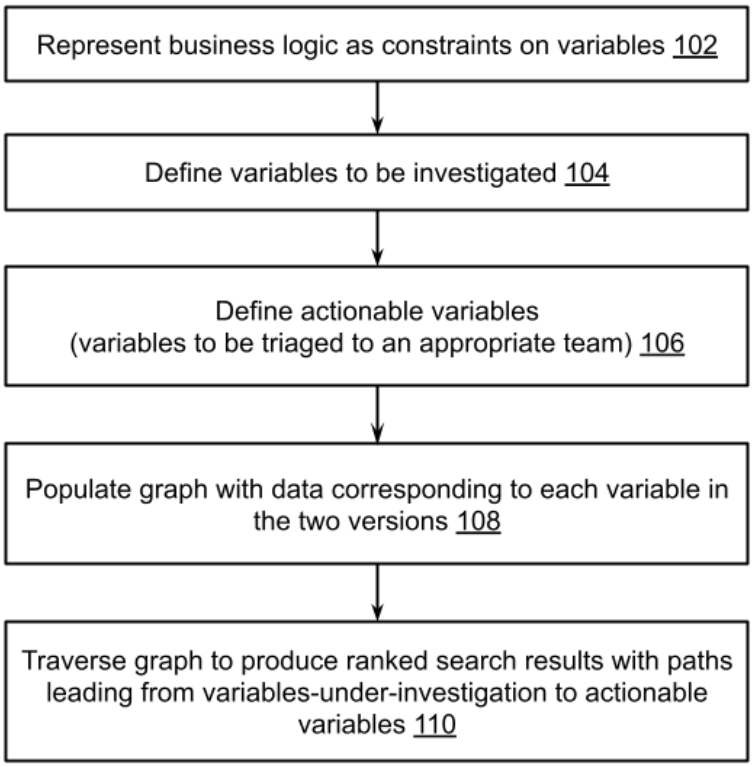

Fig. 1: Root cause analysis using a graph representation of constraints

Fig. 1 illustrates root cause analysis using a graph representation of constraints. Key business logic is represented by constraint formulas in the variables of interest with annotations about causal direction (102). Only linear constraints with constant coefficients are considered, as more general constraints can be approximated by a first-order Taylor expansion.

Variables to be investigated are identified or defined (104). For example, these are variables covered by the constraint formulas which have (perhaps substantially) different values between the two versions, which are differences that users would like to understand. Such starting point variables are denoted as X_0. Actionable variables, e.g., variables whose values can trigger a triage to an appropriate team, are defined (106). Such variables are referred to as root causes and denoted by X_r.

The graph is populated with data corresponding to each variable in the two versions (108). The graph is traversed to produce ranked search results for paths leading from each variable-under-investigation to actionable variables (110). Optionally, users can further their understanding by manually traversing the graph starting from any variable node in the search result and inspecting neighboring variables. The constraint formulas can be updated as the business evolves.

Definitions

To describe in greater detail root cause analysis using a graphical representation of constraints, certain definitions are made. Without loss of generality, at the first run, all variables are assumed to be zero. The delta between the first run and an arbitrary second run is denoted by the random variables X_0, X_1, X_2, ..., X_n. The actual delta values for a particular second run, e.g., an instance of the random variables, are denoted x_0, x_1, x_2, etc. A subset $\{X_r\}$ of the variables is defined as *actionable*, if

- they can be directly influenced by the user, e.g., they are not necessarily determined by other variables; and
- there is enough associated metadata with them for the users to rank them by importance, understand how to change them, and be assured of whether changes caused by them are working as intended.

The *impact* on x_0 attributed to x_r along an ordered sequence of variables, denoted $a(x_0, x_1, x_2, ..., x_r)$, is defined recursively as follows:

$$a(x_r) = x_r$$

$$a(x_{r-i}, x_{r-i+1}, ..., x_r) = x_{r-i} - E[X_{r-i} \mid X_{r-i+1} = x_{r-i+1} - a(x_{r-i+1}, ..., x_r)].$$

Intuitively, the above recursion captures what the value of X_{r-i} would have been if the impact from X_{r-i+1} along the path had been taken away. An expectation operator captures the fact that the values of the hypothetical run are not fully known and can potentially be non-deterministic.

Further assuming that the change of one variable with respect to the next one is linear, we can write $k_{r-i,\, r-i+1} = (x_{r-i} - E[X_{r-i} \mid X_{r-i+1} = x_{r-i+1} - a(x_{r-i+1}, ..., x_r)]) / a(x_{r-i+1}, ..., x_r)$ as only depending on x_{r-i} and x_{r-i+1}, in which case we have:

$$a(x_{r-i}, x_{r-i+1}, ..., x_r) = k_{r-i,\, r-i+1}\, a(x_{r-i+1}, ..., x_r)$$

The *excess positive impact* a_+ is defined as follows only for paths with $a(x_0, ..., x_i)>0$ for all i up to r:

$$a_+(x_r) = x_r$$

$$a_+(x_{r-i}, x_{r-i+1}, ..., x_r) = \min(x_{r-i}, x_{r-i} - E[X_{r-i} \mid X_{r-i+1} = x_{r-i+1} - a_+(x_{r-i+1}, ..., x_r)])$$

$$= \min(x_{r-i}, k_{r-i,\, r-i+1}\, a_+(x_{r-i+1}, ..., x_r))$$

The excess positive impact captures the impact along the variables contributing to x_0 in the positive direction after cancellation with other factors contributing to X_{r-i} if they are net-

negative. This is useful to capture the impact in one direction if some other factors are expected to cancel out the impact and the excess after cancelation is to be found. The *excess negative impact* is defined similarly.

Any ordered sequence of variable values $\{x_0, x_1,...,x_r\}$ is referred to as a path, denoted *p*.

Setup

To enable root cause analysis, a *bipartite graph* is created from the relationships of the variables as expressed within the constraints. *Substantial and verifiable causes* are identified, and *causal directions* of influence of one variable upon another indicated. Variables are *grouped by correlation*. These are described in greater detail below.

Bipartite graph

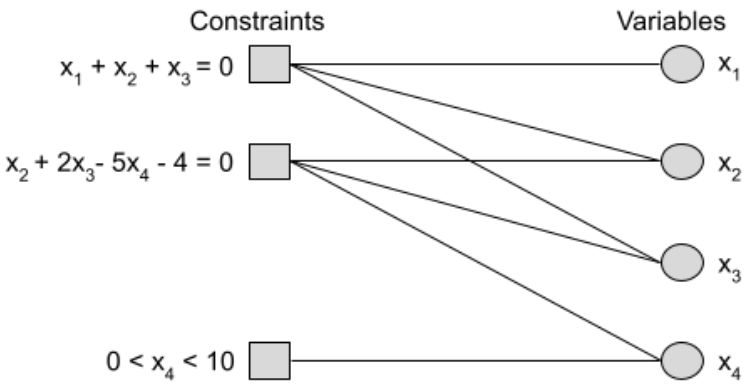

Fig. 2: A bipartite graph created out of the constraints between variables

A bipartite graph (factor graph), illustrated in Fig. 2, is created by arranging constraints in the form of a column of nodes and variables in another column of nodes. A column node representing a constraint is connected to a variable node if the constraint includes the variable. Constraints can be definitions of variables or actual restrictions (explicit or implicit in solvers) on

values that variables can take. As explained earlier, constraints are linear or can be made so using Taylor approximations.

A path connected by constraints is denoted as $\{x_0, C_{01}, x_1, C_{12}, \ldots, x_r\}$. In this context, a constraint does not necessarily indicate that variables are restricted to take only certain numerical values. The variables may include, for example, solver objectives, such as a total monetary cost being minimized; a constraint merely relates the cost to constituents contributing to the cost.

Substantial and verifiable causes

Substantial and verifiable causes are paths of interest p that connect x_0 to an actionable node x_r in the graph such that

$$|a(p)| >= a_{THRESHOLD};$$
$$\text{length of path} <= L; \text{ and}$$

the path is verifiable by users in constant time.

A substantial and verifiable cause is a requirement for the graph to be sufficiently complex to include an interesting root cause without necessarily specifying all relationships completely. This enables users to find useful results with only partially complete relational specifications.

Causal direction

Constraints can indicate whether a variable is causally influenced by another. For example, a constraint $Z = X_1 + X_2 + X_3$ with no limits on the values taken by Z can imply that Z is influenced by X_1, X_2, and X_3, while X_1 isn't influenced by Z or by X_2. Such causal directionality can be indicated in the graph. Causal directionality can be value-dependent and

approximate: for example, X_1 can be influenced by Z or by X_2 when Z is close to its lower/upper bound but not otherwise.

Correlation grouping

Variables that are a priori correlated (either positively or negatively) can be grouped together. For example, if X_1 and X_2 are not independent of each other and are negatively correlated, a constraint $Z = X_1 + X_2$ can be added, and $X_1 + X_2$ replaced throughout by Z.

Searching techniques

With the above definitions and setup, the graph can be searched to produce ranked search results for paths leading from each variable-under-investigation to actionable variables. Two example search techniques include *basic searching* and *strict searching*, described in greater detail below.

Basic searching

For every constraint $C_{r\text{-}j,\ r\text{-}j+1}$ in the graph, evaluate $\partial_{r\text{-}j+1}X_{r\text{-}j}$ such that if the constraint $C_{r\text{-}j,\ r\text{-}j+1}$ is

$$X_{r\text{-}j} - kX_{r\text{-}j+1} + \ldots = 0, \text{ then}$$

$$\partial_{r\text{-}j+1}X_{r\text{-}j} = k,$$

unless the causal direction of the constraint clearly indicates that $X_{r\text{-}j}$ cannot be influenced by $X_{r\text{-}j+1}$, in which case

$$\partial_{r\text{-}j+1}X_{r\text{-}j} = 0.$$

Under this condition, a naive chain-rule estimated impact $a_{NAIVE}(x_0, x_1, \ldots x_{r\text{-}i+1})$ is defined as

$$a_{NAIVE}(x_0, x_1, \ldots x_{r-i+1}) = x_{r-i+1} \prod_{j=i..r} \partial_{r-j+1} X_{r-j}.$$

The basic search technique returns every path without cycle in the bipartite graph starting from x_0 such that:

- it ends at (causally starts from) an actionable node x_r with path length <= L; and
- every intermediate node x_{r-i+1} is such that $|a_{NAIVE}(x_0, x_1, \ldots x_{r-i+1})| > \varepsilon$.

A breadth-first search on the graph that prioritizes $|a_{NAIVE}(x_0, x_1, \ldots x_{r-i+1})|$ at every intermediate node gives a ranking on the search results. A path p such that $|a(p)| > a_{THRESHOLD} >> \varepsilon$ (as described above in substantial and verifiable causes), can be found within the returned paths with probability $1 - P_{random\ cancel}(\varepsilon, L)$, where $P_{random\ cancel}(\varepsilon, L)$ is the probability of at least one node x_{r-j+1} along the path being canceled by other terms in the constraint $C_{r-j,\ r-j+1}$, such that the net effect of all terms, X_{j-1}, is less than ε. The parameters L and $a_{THRESHOLD}$ can be used to control the size of $P_{random\ cancel}(\varepsilon, L)$.

Strict searching

Strict searching uses a stricter requirement, e.g., $a_{NAIVE}(x_0, x_1, \ldots, x_{r-i+1}) > a_{THRESHOLD}$ for all intermediate nodes. Strict searching returns all and only the paths with excess positive impact $a_+(p) > a_{THRESHOLD}$, should any exist. From $a_{THRESHOLD} < a_+(p)$, it follows that

$$\begin{aligned} a_{THRESHOLD} &< a_+(x_0, x_1, \ldots, x_r) \\ &= \min(x_0, k_{01} \min(x_1, k_{12} \min(x_2, \ldots)) \\ &= \min(x_0, k_{01} x_1, k_{01}k_{12} x_2, \ldots), \end{aligned}$$

which is the same as requiring $a_{NAIVE}(x_0, x_1, \ldots x_{r-i+1}) > a_{THRESHOLD}$ for all i if $k_{r-i,\ r-i+1}$ can be extracted from $C_{r-i,\ r-i+1}$ as the coefficient on X_{r-j+1}. The naive approach can fail if there are other terms depending on X_{r-j+1}, but failures can be addressed as follows:

- If the other terms contribute to X_{r-j} in the same sign as X_{r-j+1}, treat those as separate paths and set $a_{THRESHOLD}$ appropriately. In the extreme cases, this may end up with a large number of paths with $a_+(p) << x_0$, but such paths can still be useful for human understanding. Further, variables can be redefined to reduce such paths.
- If the other terms contribute in the opposite sign, it may be that the true combined contribution is less than $a_{THRESHOLD}$, but correlation grouping can be relied upon to ensure that an intermediate variable with the net contribution appears instead. If the other terms contribute in a way that exactly cancels the effect of X_{r-j+1}, causal directionality can be relied upon to skip them. For example, given a constraint $Z = X_1 + X_2 + X_3$, by definition, a large and identical change in both Z and X_2 has no effect on X_1. Causal directionality enables the tracing of Z to X_1, X_2, X_3, but prevents tracing from X_1 to other variables, e.g., $\partial_Z X_1 = 0$, etc.

Practical considerations

To understand a set of numbers, there may in practice be only one version of values for variables, whereas the above formulation assumes two versions of values. In case a single version of values is available, a *synthetic comparison* can be executed, e.g., variable values can be compared against a naive expectation. Artificial values are synthesized throughout the bipartite formula graph to construct a reference for comparison.

For example, suppose a variable X is translated via a black box into a variable Y with the constraints

$$Z = Y - 3X \text{ and}$$
$$Z >= 0.$$

A synthetic reference can be obtained by setting Y = 3X and Z = 0, e.g., the deviation of the variables from the base case of Z = 0 is analyzed.

Actionable nodes can be flexibly defined so long as the definition is useful to the investigator. If actionable nodes cannot be clearly defined, enabling users to traverse the bipartite graph starting from a path that ends with a node such that |a(p)| > can already be useful. In the special case where the outputs of solvers are to be understood, the objective function and its constituents can be useful actionable nodes. For example, a variable x_0 is large because the projected profit, an objective that is being maximized, has gone up. The change in projected profit, along with the metadata around the variables contributing to the projected profit, may provide sufficient context for the users to take action. It also qualifies as an actionable node.

Some advantages of the described techniques of root cause analysis include:

- Compared to custom curated views for explanations, the framework described herein is declarative and automates human reasoning processes.
- The described framework is more maintainable: as the business evolves, only formulas in the bipartite graph are updated, rather than step-by-step instructions to find explanations.
- Checking for data correctness and for logical consistency is greatly facilitated by the bipartite graph.
- Compared to approaches to explanations based on statistical and machine learning, the described techniques require little data to be useful. The techniques focus on capturing human domain knowledge, which can be the result of collaboration by many teams, and are particularly useful for cases when the data size is small or highly heteroskedastic.
- The techniques can be utilized by any organization with interconnected quantitative data driven by human processes to organize information and make it more understandable.

CONCLUSION

This disclosure describes scalable, flexible frameworks and searching techniques that improve transparency, e.g., enable root-cause analysis, for large families of variables. The techniques enable the understanding of the difference between two data-generating flows (versions) with comparable inputs, intermediate variables, and output variables.

REFERENCES

[1] "Reasoning system," https://en.wikipedia.org/wiki/Reasoning_system accessed Aug. 19, 2022.

[2] "Knowledge graham" https://en.wikipedia.org/wiki/Knowledge_graph accessed Aug. 19, 2022.

[3] Martin, Richard Kipp. "Large scale linear and integer optimization: a unified approach." *Springer Science & Business Media*, 2012.

[4] Kwisthout, Johan, and Linda C. Van Der Gaag. "The computational complexity of sensitivity analysis and parameter tuning." *arXiv preprint arXiv:1206.3265* (2012).

[5] Ji, Shaoxiong, et al. "A survey on knowledge graphs: Representation, acquisition, and applications." *IEEE Transactions on Neural Networks and Learning Systems* 33, no. 2 (2021): 494-514.

[6] Pearl, Judea. "Causality." Cambridge university press, 2009

[7] Sun, Yi, and Mukund Sundararajan. "Axiomatic attribution for multilinear functions." In *Proceedings of the 12th ACM conference on Electronic commerce*, pp. 177-178. 2011.

[8] Sundararajan, Mukund, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks." In *International conference on machine learning*, pp. 3319-3328. PMLR, 2017.

[9] "Factor graph," https://en.wikipedia.org/wiki/Factor_graph accessed Aug. 19, 2022.