A quest for formal tools for reasoning about counterfactual causation

Gregor Gössler
Univ. Grenoble Alpes, INRIA, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France

Jean-Bernard Stefani
Univ. Grenoble Alpes, INRIA, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France

Oleg Sokolsky
University of Pennsylvania, USA

In this position paper we discuss three main shortcomings of existing work on counterfactual causation from the computer science perspective, and sketch lines of work to try and overcome these issues. We hope that this paper will spur some fruitful discussion at CREST.

1 Escaping the TEGAR

Research on counterfactual causality analysis has been marked, since its early age [19], by a succession of definitions — in contrast to conjectures, proofs, and theorems as in the natural sciences and mathematics — that are informally (in)validated against human intuition on mostly simple examples. Let us call this approach TEGAR, *textbook example guided analysis refinement*. As pointed out in [10], TEGAR suffers from its dependence on the tiny number of examples in the literature and the lack of stability of the intuitive judgments against which the definitions are validated. This absence of formal tools for evaluating theories of causation is not primarily owed to a lack of formalization: at least since the works of [28, 26], different formal definitions of causality have been proposed. Among the most influential definitions of counterfactual causation are Lewis' possible world semantics [23, 24, 25] and Pearl and Halpern's actual causality [26, 17, 15]. Both definitions have undergone a series of refinements in order to match human intuition on additional examples that were proposed to challenge them. One may doubt that this is the end of the story. It is interesting to note that the understanding of causality and explanations in natural sciences faces a similar lack of objective referential, as witnessed by the dispute reedited in [30].

We believe that a more constructive, reproducible approach to design definitions of counterfactual causality is needed. A first step towards this goal would be to develop an, at least partially, formal — but, as far as possible, model-agnostic — characterization of counterfactual causality, rather than a mere set of concrete definitions. In a sense, we may call this a meta-definition of causation.

Some efforts to axiomatize counterfactual causality have been made. On structural equation models [26] (SEM), [9] introduces three properties that hold in all (recursive and nonrecursive) SEM, two of which characterize manipulation (Pearl's *do* operator) in the recursive case. [14] generalizes these results to an axiomatic characterization of the classes of non-recursive SEM with unique solutions, and arbitrary SEM. With the goal of using counterfactual causation for fault ascription — that is, blaming a system failure on one or more component faults —, [13] proposes general constraints on counterfactuals that are sufficient to entail correctness and completeness. Similarly, a general definition of actual causation is proposed and then instantiated in [2]. While none of these axiom systems is strong enough to

Preliminary Report. Final version to appear in:
CREST 2017

characterize more than basic properties of counterfactual reasoning, we believe that there is still room for progress.

A characterization of counterfactuals boils down to two questions:

- What formal properties should definitions of counterfactual causality satisfy, independently of the modeling framework?
- What are properties of interest that are satisfied only by counterfactual analysis?

In turn, it should help us in answering questions such as:

- How to design a counterfactual analysis satisfying given properties, such as robustness of the causes under equivalence of models, for a given definition of equivalence?
- Can we obtain the same analysis result with other tools than counterfactual analysis?
- Is counterfactual causality analysis inherently NP-complete (as Halpern and Pearl's actual causality) [8, 15]?

2 Native support for system dynamics

It has been pointed out in [10] that Halpern and Pearl's definitions of actual causality, based on SEM over propositions, poorly support reasoning about state changes. Other limitations of SEM — in particular, their inability to distinguish between states and events, and between presence and absence of an event — have also been noted e.g. by Hopkins and Pearl [18], and several other formalisms have been suggested for supporting reasoning about causal ascription (see e.g. [5]). Counterfactual definitions of necessary causality for behaviors over time have been proposed for biochemical reactions in [6] and similarly for programs in [7], and for fault ascription in [12]; some works define variants of actual causality on models of execution traces [4, 20].

Apart from the modeling infelicities of SEM, a key point is that models that allow finitary descriptions of systems dynamics are essential for conducting actual cause analysis. In particular since counterfactual executions may be unbounded it may be necessary to explore a prefix of the counterfactuals whose length is not bounded a priori, in order to evaluate the property. For instance, a system dynamics can be represented by a set of traces or some sort of automata, and actual cause analysis for a property violation during some execution can consist in constructing sets of traces or automata executions that avoid a particular set of violating states but keep at least the antecedent part of the original execution. In order to effectively construct and analyze these counterfactual executions, we then need a symbolic representation, along with symbolic formulations of the counterfactual construction and analysis. Symbolic approaches to causality checking have been proposed e.g. in [3] for Halpern and Pearl's actual causality and in [29, 11] for fault ascription in real-time systems; except for [11] they rely on generating and analyzing bounded counterfactuals.

For systems dynamics, the notion of coalgebra [27] provides a systematic setting, generalizing notions of transition systems. Following the (hyper)set-based formulation of [1], a system can be described coalgebraically as a possibly infinite, mutually recursive, set of equations of the form $x = F(x)$, where F is some operator on sets, and x some variable. For instance, the standard notion of (finitely branching) labelled transition system is given by operator F defined as $F(X) = \mathcal{P}_f(A \times X)$ where $\mathcal{P}_f(S)$ denotes the set of finite subsets of some set S, X is the set of (state) variables and A is the set of labels. One benefit of the coalgebraic approach is its generality. For instance, many different variants of transition systems, including timed, quantitative, and stochastic ones, are instances of coalgebras. Our contention is that it could be beneficial to develop causality analysis in an abstract coalgebraic framework, if only to identify abstractions and constructions (e.g. for counterfactuals) that apply generally irrespective of the actual details of the chosen operators. [13] provides an example of counterfactual analysis developed in an abstract setting – that of configuration structures, which can be understood as a general model for concurrent system executions or unfoldings. It seems to us that general notions of causality and counterfactuals should not depend on the specifics of system or transition system models. Rather, we expect that at least general constraints on counterfactual construction and causal dependencies can be obtained for abstract system models and properties. For instance, a general notion of bisimulation can be defined for coalgebraic systems, that does not depend on the specifics of the chosen operator. Obtaining similarly abstract characterizations of causal dependencies or counterfactuals would be of enormous benefit.

Once we allow for unbounded executions also in the actual world — i.e., the observed execution, — *incremental* causality analysis becomes an issue. Many causality analysis techniques operate on the observed prefix of the execution at the point when an event of interest, such as a failure, is discovered. If the analysis relies on explicit construction of counterfactuals, there is a danger of repeating the same work for different counterfactuals. Moreover, if the analysis has to be performed multiple times over an evolving execution, redundant efforts are even more likely. In this case, incremental analysis can keep partially constructed counterfactuals, hopefully in a symbolic form, and update them as the next observation from the execution arrives. While this approach may not reduce complexity of causality analysis, it may amortize the cost over a long-running execution and reduce analysis latency, once an event of interest is observed.

With this vision, the partial, evolving counterfactual would allow us to answer the question, "if the event of interest is to happen in the next step, what would the causes be?" When there are multiple events of interest — e.g., multiple ways for a failure to occur, — the danger is that the incremental analysis would incur additional cost with bookkeeping for events that never occur.

3 Causation and Abstraction

The impact of modeling choices on counterfactual analysis has long been recognized, see e.g. [16]. In engineering and computer sciences, using counterfactual analysis on hand-crafted models is like modeling a critical system in one formalism and then implementing it in another one from scratch — the semantic gap between the model and the actual system makes it difficult to ensure that the former is faithful with respect to the latter. In order to base causality analysis and its applications, e.g. to establishing liability [22], on firm ground, *accountability* [21] with respect to causality analysis — that is, guaranteeing that all information necessary to elucidate the causes of failures is logged — should become a design requirement for new designs of safety-critical systems.

Pushing this line of thought a little further, software design has been formalized as a series of refinements from a high-level specification down to the implementation. Theories of causation should be able to track causation through these levels of refinement, for instance, to verify causality on a small abstract model and then refine potential causes identified on that level. To this end, theories of causation should have a well-defined behavior under abstraction and refinement, such as correctness (any cause in the abstract model is refined into a cause in the refinement) or completeness (the abstraction of any cause in the refinement is also a cause in the abstract model) of abstraction. One can go even further and ask how causality meshes with system equivalence. The standard benchmark for system equivalences is contextual equivalence: given some notion of observable and some notion of system execution, two systems are equivalent when, placed in the same context, they have the same observables and the same executions. It seems to us plausible to ask of a notion of causality to be robust with respect to contextual equivalence: if causal analysis in a complex system $S[A]$, where $S[\cdot]$ is a context for subsystem A, yields a certain result, then the same analysis performed on $S[B]$, where B is contextually equivalent to A, should yield the same result (e.g. pinpointing some observable event in A or B as the actual cause of some property violation). We are not aware of any work studying abstraction, refinement, or robustness, in the SEM framework. Whenever causation is verified on a causal model other than the actual system model or code, a similar property should also hold for the extraction of the causal model — for instance, existence of a Galois connection between the system model and the causal model. Could a framework formalizing causation as an abstraction of a more complex system behavior address Russell's famous criticism about causality?

Finally, deriving an implementation by refining an abstract specification usually implies that the abstract model encompasses some non-determinism. In order to support multiple levels of refinement, theories of causation have to be able to cope with this non-determinism.

References

[1] J. Barwise and L. Moss. *Vicious Circles*, volume 60 of *CSLI Lecture Notes*. CSLI Publications – Center for the Study of Language and Information, Stanford, California, 1996.

[2] S. Beckers and J. Vennekens. A general framework for defining and extending actual causation using cp-logic. *Int. J. Approx. Reasoning*, 77:105–126, 2016.

[3] A. Beer, S. Heidinger, U. Kühne, F. Leitner-Fischer, and S. Leue. Symbolic causality checking using bounded model checking. In B. Fischer and J. Geldenhuys, editors, *Model Checking Software - 22nd International Symposium, SPIN 2015, Stellenbosch, South Africa, August 24-26, 2015, Proceedings*, volume 9232 of *LNCS*, pages 203–221. Springer, 2015.

[4] I. Beer, S. Ben-David, H. Chockler, A. Orni, and R.J. Trefler. Explaining counterexamples using causality. *Formal Methods in System Design*, 40(1):20–40, 2012.

[5] S. Benferhat, J.-F. Bonnefon, P. Chassy, R. Da Silva Neves, D. Dubois, F. Dupin de Saint-Cyr, D. Kayser, F. Nouioua, S. Nouioua-Boutouhami, H. Prade, and S. Smaoui. A comparative study of six formal models of causal ascription. In *Scalable Uncertainty Management, Second International Conference, SUM 2008, Naples, Italy, October 1-3, 2008. Proceedings*, volume 5291 of *Lecture Notes in Computer Science*, pages 47–62. Springer, 2008.

[6] V. Danos, J. Feret, W. Fontana, R. Harmer, J. Hayman, J. Krivine, C. Thompson-Walsh, and G. Winskel. Graphs, Rewriting and Pathway Reconstruction for Rule-Based Models. In D. D'Souza, T. Kavitha, and J. Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012)*, volume 18 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 276–288. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2012.

[7] A. Datta, D. Garg, D. Kirli Kaynar, D. Sharma, and A. Sinha. Program actions as actual causes: A building block for accountability. In C. Fournet, M.W. Hicks, and L. Viganò, editors, *IEEE 28th Computer Security Foundations Symposium, CSF 2015, Verona, Italy, 13-17 July, 2015*, pages 261–275. IEEE Computer Society, 2015.

[8] T. Eiter and T. Lukasiewicz. Complexity results for structure-based causality. *Artif. Intell.*, 142(1):53–89, 2002.

[9] D. Galles and J. Pearl. An axiomatic characterization of causal counterfactuals. *Foundations of Science*, 3:151–182, 1998.

[10] C. Glymour, D. Danks, B. Glymour, F. Eberhardt, J. Ramsey, R. Scheines, P. Spirtes, C. M. Teng, and J. Zhang. Actual causation: a stone soup essay. *Synthese*, 175(2):169–192, 2010.

[11] G. Gössler and L. Aştefănoaei. Blaming in component-based real-time systems. In *EMSOFT'14*, pages 7:1–7:10. ACM, 2014.

[12] G. Gössler and D. Le Métayer. A general framework for blaming in component-based systems. *Science of Computer Programming*, 113(3):223–235, 2015.

[13] G. Gössler and J.-B. Stefani. Fault ascription in concurrent systems. In P. Ganty and M. Loreti, editors, *Proc. Trustworthy Global Computing - 10th International Symposium, TGC 2015*, volume 9533 of *LNCS*. Springer, 2016.

[14] J. Y. Halpern. Axiomatizing causal reasoning. *J. Artif. Intell. Res. (JAIR)*, 12:317–337, 2000.

[15] J. Y. Halpern. A modification of the halpern-pearl definition of causality. In Q. Yang and M. Wooldridge, editors, *Proc. Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3022–3033. AAAI Press, 2015.

[16] J.Y. Halpern and C. Hitchcock. Actual causation and the art of modeling. *CoRR*, abs/1106.2652, 2011.

[17] J.Y. Halpern and J. Pearl. Causes and explanations: A structural-model approach. part I: Causes. *British Journal for the Philosophy of Science*, 56(4):843–887, 2005.

[18] M. Hopkins and J. Pearl. Causality and counterfactuals in the situation calculus. *J. Log. Comput.*, 17(5), 2007.

[19] D. Hume. *A Treatise of Human Nature*. 1739.

[20] M. Kuntz, F. Leitner-Fischer, and S. Leue. From probabilistic counterexamples via causality to fault trees. In F. Flammini, S. Bologna, and V. Vittorini, editors, *SAFECOMP*, volume 6894 of *LNCS*, pages 71–84. Springer, 2011.

[21] R. Küsters, T. Truderung, and A. Vogt. Accountability: definition and relationship to verifiability. In *ACM Conference on Computer and Communications Security*, pages 526–535, 2010.

[22] D. Le Métayer, M. Maarek, E. Mazza, M.-L. Potet, S. Frénot, V. Viet Triem Tong, N. Craipeau, and R. Hardouin. Liability issues in software engineering: the use of formal methods to reduce legal uncertainties. *Commun. ACM*, 54(4):99–106, 2011.

[23] D. Lewis. Causation. *Journal of Philosophy*, 70, 1973.

[24] D. Lewis. *Philosophical Papers*. Oxford University Press, 1986.

[25] D. Lewis. *Counterfactuals*. Blackwell, 2nd edition, 2000.

[26] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

[27] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science, vol. 249*, 2000.

[28] P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, prediction, and search*. MIT press, 2000.

[29] S. Wang, A. Ayoub, B. Kim, G. Gössler, O. Sokolsky, and I. Lee. A causality analysis framework for component-based real-time systems. In A. Legay and S. Bensalem, editors, *Proc. Runtime Verification 2013*, volume 8174 of *LNCS*, pages 285–303. Springer, 2013.

[30] D.S. Weld and J. de Kleer, editors. *Readings in Qualitative Reasoning about Physical Systems*, chapter 9: Causal Explanations of Behavior. Morgan Kaufmann, 1990.