SYM-AM-18-097

PROCEEDINGS
OF THE
FIFTEENTH ANNUAL ACQUISITION RESEARCH SYMPOSIUM

THURSDAY SESSIONS
VOLUME II

Acquisition Research:
Creating Synergy for Informed Change

May 9–10, 2018

March 30, 2018

Prepared for the Naval Postgraduate School, Monterey, CA 93943.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL

Further Causal Search Analyses With UCC's Effort Estimation Data

Anandi Hira—is currently a PhD student under Dr. Barry Boehm at the University of Southern California's (USC's) Computer Science Department. Her research interests lie in cost estimation and models, software metrics in relation to project management, and process improvement. Hira has been a part of the Unified Code Count (UCC) development effort at USC's Center for Systems and Software Engineering (CSSE) for the past six years and has been collecting and analyzing the data to compare the effectiveness of functional size metrics. [a.hira@usc.edu]

Barry Boehm—is the TRW Professor in USC's Computer Sciences, Industrial and Systems Engineering, and Astronautics Departments. He is also the Director of Research of the DoD-Stevens-USC Systems Engineering Research Center and the founding Director of the USC Center for Systems and Software Engineering. He was a Director at DARPA-ISTO, TRW, Rand Corporation, and General Dynamics. His contributions include the COCOMO family of cost models and the Spiral family of process models. He is a Fellow of the primary professional societies in computing (ACM), aerospace (AIAA), electronics (IEEE), and systems engineering (INCOSE), and a member of the U.S. National Academy of Engineering. [boehm@usc.edu

Robert Stoddard—is a Software Engineering Institute Principal Researcher at Carnegie Mellon University. His research includes machine/causal learning, applied statistics, Bayesian probabilistic modeling, Six Sigma, and quality/reliability engineering. Stoddard achieved an MS in Systems Management and significant doctoral progress in reliability and quality management. He is a Fellow of the American Society for Quality and a senior member of the IEEE. Stoddard holds five ASQ certifications and is a Motorola-certified Six Sigma Master Black Belt. [rws@sei.cmu.edu]

Michael Konrad—is a Principal Researcher at the SEI, providing analytic support to various projects using statistics, machine learning, and most recently, causal learning. Since 2013, Konrad has contributed to research in requirements engineering, software architecture, and system complexity measurement. From 1998 to 2013, he contributed to CMMI in many technical leadership roles. Prior to 1998, Konrad was a member of the teams that developed the original Software CMM and ISO 15504. He is coauthor of the main Capability Maturity Model Integration for Development (CMMI-DEV) books. Konrad received his PhD in mathematics from Ohio University in 1978. [mdk@sei.cmu.edu]

Abstract

Correlation does not imply causation. Though this is a well-known fact, most analyses depend on correlation as proof of relationships that are often treated as causal. Causal search, also referred to as causal discovery, involves the application of statistical methods to identify causal relationships using conditional independences (and/or other statistical relationships) within data. Though software cost estimation models use both domain knowledge and statistics, to date, there has yet to be a published report describing the evaluation of a software dataset using causal search. In a previous paper, the authors ran a PC causal search algorithm on Unified Code Count's (UCC's)[1] dataset of maintenance tasks and compared them to correlation test results. This paper builds on the previous paper to introduce causal discovery to software engineering research by exploring additional causal search algorithms (PC-Stable, fast greedy equivalent search [FGES], and fast

[1] http://ucc.usc.edu

adjacency skewness [FASK]) and comparing their results to the traditional multi-step regression analysis.

Introduction

Though analysts seek causation, “data, on their own, only communicate associations” (Elwert, 2013). Two variables with high correlations, or associations, may have causal or non-causal relationships (Elwert, 2013). Correlations, by themselves, cannot generally determine which relationships are causal (Cook, Campbell, & Shadish, 2002). Hence, statisticians emphasize correlation does not imply causation. Judea Pearl (2001) stated, “Behind every causal conclusion there must lie some causal assumption that is not testable in observational studies,” suggesting that one cannot gain complete causal knowledge through only observational studies or data alone. Experiments that manipulate causal variables can help identify causal relationships (Cook et al., 2002). However, there are “practical and ethical considerations that limit the application of controlled experiments in many cases” (Spirtes, 2010).

In software engineering, it is impractical to run software development effort experiments. Developing the same software with various personnel attributes or manipulating the project's size and product attributes with the same team would become very expensive for software development teams. Hence, most data in software engineering are observational versus from controlled experiments. Though causal inference (which includes causal search) is characterized as “finding answers to questions about the mechanisms by which variables come to take on values, or predicting the value of a variable after some other variable has been manipulated” (Spirtes, 2010), estimation model developers have not used such statistical methods to confirm or reject causal assumptions. Until recently, estimation model developers lacked the tools to systematically evaluate whether the size and effort drivers identified by software project experts do, in fact, have significant (both in the statistical and effect size sense) causal effects on effort and, therefore, should be preferred for selection over other candidate drivers for effort estimation.

Using the theory of causal inference, one can now perform a causal search “based on unmanipulated data” (Spirtes, 2010). According to Spirtes, causal inference consists of two parts: “search for a causal graph, and estimation of the free parameters from sample data and the causal graph.” In this paper, the authors run causal search algorithms that return causal graphs (the first part defined by Spirtes) with the intent to use the causal discovery results to estimate the parameters (the second part defined by Spirtes).

Previously, the authors reported the differences between correlation test results and commonly-used causal search algorithm results on UCC’s dataset of maintenance tasks (Hira et al., 2018). This paper builds upon the first paper in several ways. First, the authors take the opportunity to discuss the differences between constraint-based versus score-based search algorithms as both are used in this paper. Second, the authors run additional causal search algorithms (of both types) in order to better illustrate some of the other search algorithms and their capabilities given a relatively small dataset, and to further confirm or modify findings of the first paper in terms of factors having direct or indirect causal effects on software development efforts. Third and last, the authors more clearly contrast the traditional approach of correlation and multiple regression with that of the causal search approach. Interestingly, not all causal search algorithms will necessarily result in the same set of causal relationships.

Unified Code Count

Development Environment and Dataset Summary

The University of Southern California (USC) maintains Unified Code Count (UCC), a small- to medium-sized tool that provides source lines of code (SLOC) counting metrics for about 30 programming languages, such as logical SLOC (Park, 1992) and cyclomatic complexity (McCabe, 1976). UCC is an object-oriented project written in C++, and each year, development teams work on enhancements or extensions that range in size from 45 to 1,425 equivalent source lines of code (ESLOC, defined in the subsection titled Dataset Attributes). USC releases an updated UCC annually with new language parsers, additional features, and/or additional metrics. UCC's current dataset covers recent projects consisting of six new language parser projects, five new features projects (such as GUI interface or additional input options), and 19 projects researching and adding cyclomatic complexity metrics to UCC's outputs. Data for analysis came from the developed code, weekly timesheets, test case documentation with corresponding test data, and explanatory reports summarizing the steps taken and the results of projects that began and completed between 2010 and 2014.

Dataset Attributes

Along with size and effort, UCC's dataset contains project and personnel characteristics as defined by Constructive Cost Model® (COCOMO®) II. COCOMO® II is a parametric software cost estimation model that requires size, product, and personnel attributes as input, and outputs the estimated effort in Person-Months (PM). A summary of the attributes included in UCC's dataset of maintenance tasks are as follows, where items numbered 6 to 13 are effort factors defined by COCOMO® II:

1. Equivalent Logical Source Lines of Code (ESLOC): Logical SLOC was developed and defined by the Software Engineering Institute (SEI) to standardize SLOC measurement (Park, 1992). Equivalent logical SLOC (ESLOC) makes modifications to reused code equivalent to new code, which has been calculated using Nguyen's modification to COCOMO® II's reuse model (Nguyen, 2010; Boehm, Madachy, & Steece, 2000).
2. IFPUG Function Points (FPs): Each project in the dataset has been sized using version 4.3.1 of IFPUG's FPs method.
3. IFPUG Software Non-Functional Assessment Process (SNAP) Points: Each project in the dataset has been sized using version 2.3 of IFPUG's SNAP method.
4. COSMIC Function Points (CFPs): Each project in the dataset has been sized using version 4.0 of COSMIC's FPs (CFPs) method.
5. Total Effort: Effort in terms of hours, including time spent on training, requirements gathering, coding, testing, and documenting.
6. Applications Experience (APEX): Most of the personnel that join UCC's development team do not have prior industry experience in similar application types, though they have sufficient computer science education. Therefore, the Low rating is used for APEX on all data points (as opposed to Very Low or Nominal, etc.).
7. Platform Experience (PLEX): The development personnel have little experience in the graphical interface platform and building cross-platform applications. Hence, a Low rating for PLEX best describes the development teams for all data points.

8. Use of Software Tools (TOOL): Currently, the UCC development environment only uses tools corresponding to the Very Low rating.
9. Personnel Continuity (PCON): COCOMO® II's highest personnel turnover rating is 48% per year. On average, UCC faces a 90% turnover over four months. Two of the authors had to adjust the rating value for this parameter in a previous study (Hira, Sharma, & Boehm, 2016).
10. Documentation Match to Life-Cycle Needs (DOCU): All teams are required to document the requirements of the project and summarize the work completed and decisions made. However, a couple of projects had substantially more documentation with respect to the requirements and earned a High rating for DOCU, and one project had less than the required documentation and earned a Low rating.
11. Analyst Capability (ACAP): “Analysts are personnel who work on requirements, high-level design, and detailed design” (Boehm et al., 2000). Some teams showed high and very high analyst capability, whereas very few teams showed low capability in analysis and design.
12. Programmer Capability (PCAP): Though most of the programming personnel were sufficiently capable, some developers had especially proficient programming skills.
13. Product Complexity (CPLX): Although most of the UCC maintenance tasks fall within the Nominal or Average rating for CPLX, some were rated Low based on the types of control operations and computation operations implemented.

The applicable COCOMO® II effort factors and their corresponding values for each of the projects were evaluated by reviewing the source code, deliverables, and Hira's weekly notes on teams' progress at the time of data collection. Since items 6–9 are rated the same across all data points, they are not included in the causal search algorithm runs.

Causal Search Algorithms Further Explained

Causal search algorithms typically take a dataset and hyper-parameters governing the search and output a set of graphs whose nodes are the variables appearing in the dataset (and depending on the algorithm, may include latent variables) and whose edges indicate some kind of direct causal connection between the pair of nodes they join. (Optionally, the algorithms also take sets of required and prohibited direct causal relationships between pairs of variables, which the user can use to encode the results of experiments or elicited domain knowledge.) There are many variations on this simple theme among the dozens of search algorithms, but in terms of understanding how they function, and thus something of their relative strengths and limitations, it will help to organize them into two broad categories: constraint-based and score-based search algorithms (Spirtes, 2010).

For both categories of searches, pointwise-consistent convergence has been proven. In other words, with increasing datasets drawn from the same population, the algorithm will eventually find the correct causal graph(s). Unfortunately, uniformly-consistent convergence has not been proven, which could provide the rate of convergence and level of confidence for particular causal relationships (Spirtes, 2010).

Constraint-Based Search Algorithms

The first practical constraint-based search algorithm developed was the PC search algorithm, the algorithm used in the previous paper by the authors (Hira et al., 2018). In its simplest form, constraint-based search involves two stages: Adjacency Search and Edge Orientation. Starting with a complete undirected graph, edges are iteratively removed by testing for the conditional independence of joined nodes given a subset of neighboring nodes. If conditional independence is found, the edge is removed and the conditioning set employed is noted for later use in the Edge Orientation stage. This process is continued until all edges have been evaluated in this way. The result of this first stage, Adjacency Search, is thus an undirected graph. Edge Orientation starts with an undirected graph and iteratively orients edges according to a few rules that make use of the conditioning sets noted during the Adjacency Search stage. The result is an equivalence class of graphs, called a Markov Equivalence Class (MEC), rather than a single graph, because it is often impossible to determine the orientation of all the edges in the undirected graph that is output from the Adjacency Search stage (Spirtes, 2010).

For example, suppose we have a dataset featuring three variables, X1, X2, and X3, and the only independence discovered among them is X1 is independent of X3 conditioned on X2. We also suppose we have no additional knowledge to encode about X1, X2, and X3, only the dataset. Then the Adjacency Search stage will output the undirected graph X1 – X2 – X3 (as well as some kind of note that the conditioning set that made X1 and X3 independent is {X2}). Then, given that particular independence, it necessarily follows that during the Edge Orientation stage, the direction of orientations for the edges of this undirected graph will not be able to be determined uniquely. Indeed, any of the following three pairs of orientations are valid, constituting the MEC: { X1 → X2 → X3, X1 ← X2 ← X3, X1 ← X2 → X3 }. Note that the following sequence of orientations is not part of the MEC: X1 → X2 ← X3. This type of relationship among variables is referred to as a collider. In a collider, the independence conditioning set is the empty set, because X1 is independent of X3 unconditionally. Hence, if the only independence found among X1, X2, and X3 is that X1 and X3 are unconditionally independent, then the MEC would consist of exactly one graph: X1 → X2 ← X3. Thus, colliders provide important clues for orienting edges during the Edge Orientation Stage (Spirtes, 2010).

While the idea of a set of graphs being the output of a causal search may disappoint, it is important to note that all graphs in an MEC have the same set of colliders and are built on top of the same undirected graph. Thus, all graphs in an MEC manifest the same set of correlations present in the dataset but may vary as to the causal orientations for some edges.

The hyper-parameters of a constraint-based search algorithm typically include but are not limited to

1. type of independence test used (e.g., Fisher Z Test, Conditional Correlation Test)
2. confidence level for conditional independence testing (cutoff for p values)
3. maximum size of condition set

Constraint-based search makes very significant use of independence tests, and what type of independence test to use for what purpose is an ongoing area of research to help achieve both accuracy and speed across a range of different assumptions (e.g., non-Gaussian univariate distributions). Another area of research is how to achieve both accuracy and speed in determining edge orientations. The need to conduct search on enormous

datasets and a very large number of variables (e.g., one million cases and tens of thousands of variables) motivates research for better algorithms.

Score-Based Search Algorithms

To those readers more familiar with machine learning, score-based search algorithms employ a familiar mechanism: a maximum likelihood-based score (such as Bayesian information criterion [BIC]). Like constraint-based search, there are two stages, both are iterative, and in each iteration of each stage there is both a currently-considered MEC (see the previous section for an explanation of this term, but it is important to note that all graphs in an MEC share the same underlying undirected graph and the same colliders) and a set of neighboring MECs and that each either possesses an additional edge (first stage of search) or has one edge removed (second stage of search; Spirtes, 2010).

In each iteration of the first stage, from the currently-considered MEC, the algorithm scores all neighboring MECs that have one additional edge. The best-scoring neighboring MEC then becomes the currently-considered MEC in the next iteration. The algorithm continues to iterate, building graphs one edge at a time, until a better score cannot be attained. In the second stage, the algorithm proceeds similarly but in reverse, considering only those MECs having one edge removed. Again, the algorithm halts when no better score can be attained, and the resulting MEC is returned as the output (Spirtes, 2010).

The advantages of score-based search algorithms over constraint-based search algorithms are as follows:

- They can obtain more accurate adjacencies.
- They will typically output only directed or undirected edges. There are no bi-directed edges because equivalence class scoring will almost always favor one orientation over the other (or make the rarely-required arbitrary selection). A bi-directed edge signifies that there may be an unmeasured variable affecting the two variables.

A limitation of score-based search algorithms is that they can be slow and might not scale as well as constraint-based searches.

Applying Three Additional Search Algorithms (Beyond the First Paper)

As mentioned earlier, the authors ran the PC search algorithms earlier (Hira et al., 2018). PC is often cited by data analysts experienced with performing causal search as the "go to" causal search algorithm, given its generally high accuracy (generally better at getting adjacencies right) and high scalability. In this paper, the authors run three additional search algorithms on the same dataset: PC-Stable (constraint-based), FGES, and FASK (both score-based), in order to better illustrate some of the other search algorithms and their capabilities given a relatively small dataset.

Here is a short description of the particular niche where each algorithm has some relative strengths over the others:

1. PC-Stable is a variant of PC that addresses the problem that the causal graphs output by many search algorithms depend on the order of the variables within the dataset (Colombo & Maathuis, 2014).
2. FGES is a score-based search algorithm and perhaps best qualifies as the causal-search data analysts' favorite "go to" search algorithm after PC (particularly if they do not like to deal with bi-directed edges; Center for Causal Discovery, 2017).

3. FASK, a score-based search algorithm, addresses the problem that oftentimes variables have asymmetric distributions. The previously listed algorithms assume that direct causal relationships are linear up to an error term that is Gaussian, whereas FASK actually exploits skew in error terms' distributions to determine how best to orient edges (Sanchez-Romero et al., 2018).

Tetrad

As part of a National Institutes of Health (NIH) Big Data initiative, the University of Pittsburgh, Carnegie Mellon University (CMU), and Pittsburgh Supercomputing Center serve as founding members of the Center for Causal Discovery (CCD). The CCD develops and maintains causal algorithms, software, and tools, including the Tetrad[2] program with its GUI, API, and command-line interfaces (referred to as Tetrad in this paper). Tetrad allows users to run causal search algorithms on a dataset as well as estimate and evaluate parametric models. The authors loaded the data from the UCC's dataset (described in the section titled Unified Code Count) and ran causal search algorithms, which returned causal graphs representing the cause-effect relationships discovered in the dataset.

PC-Stable Causal Search Results

The PC-Stable causal search is run on UCC's dataset first with all size-related variables (ESLOC, CFPs, FPs, and SNAP), and second, individually with only one size-related variable at a time. The authors run the algorithm with the simultaneous inclusion of all size-related variables to determine whether the size metrics might work together to characterize total effort, as represented by the variable TotalEffort.

The graphical search results from PC-Stable are displayed in Figures 1–5, which conclude that product complexity (CPLX) is consistently identified as the single causal factor of TotalEffort except for the search with all size metrics included. When all size metrics are included, the size metric COSMIC Function Points (CFPs) is identified as the sole causal factor on TotalEffort. Interestingly, ESLOC, FPs, and SNAP have an undirected edge with CPLX, which thus has a potential causal effect on TotalEffort through CPLX (Figures 2–4), while CFPs' role with CPLX is flipped (see Figure 5). Lastly, each of the figures shows an undirected edge between analyst capability (ACAP) and programmer capability (PCAP).

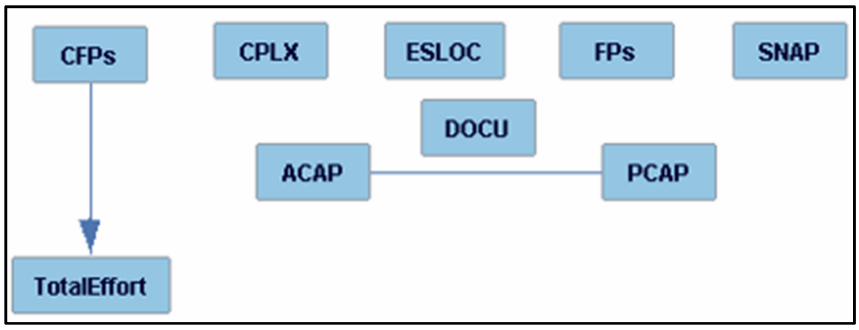

Figure 1. PC-Stable Result When All Size Metrics Are Included

[2] https://github.com/cmu-phil/tetrad

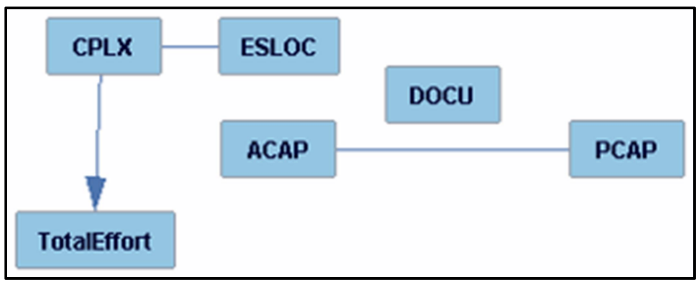

Figure 2. PC-Stable Result When Only ESLOC Is Included as a Size Metric

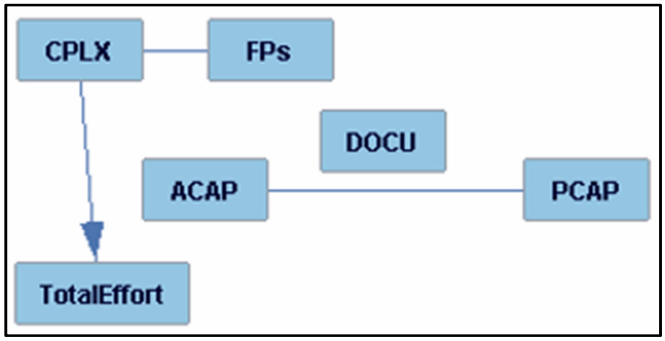

Figure 3. PC-Stable Result When Only IFPUG FPs Is Included as a Size Metric

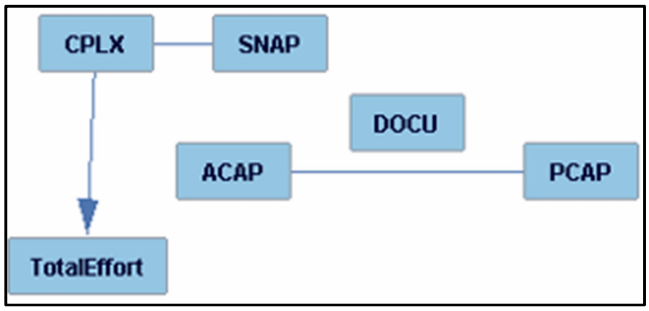

Figure 4. PC-Stable Result When Only IFPUG SNAP Is Included as a Size Metric

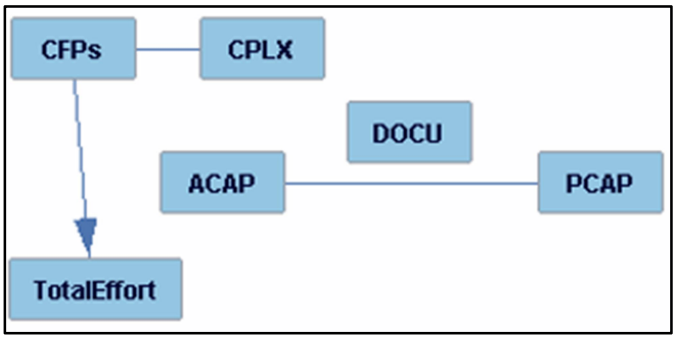

Figure 5. PC-Stable Result When Only CFPs Is Included as a Size Metric

FGES Causal Search Results

The graphical search results from running FGES may be seen in Figures 6–10. The causal graphs returned by FGES have some different conclusions than those returned by PC-Stable. The PCAP factor shows up in all cases as having a causal effect on TotalEffort (and an undirected edge with ACAP), while CFPs also have a causal factor on TotalEffort (consistent with PC-Stable results). Only with ESLOC is there an undirected edge between the size metric and CPLX (see Figure 7; consistent with PC-stable result).

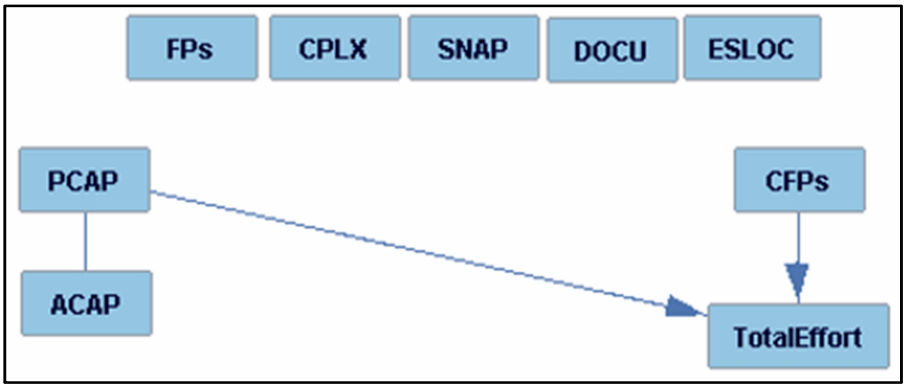

Figure 6. FGES Result When All Size Metrics Are Included

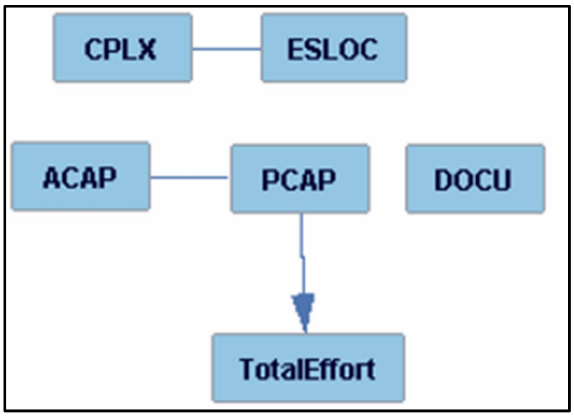

Figure 7. FGES Result When Only ESLOC Is Included as a Size Metric

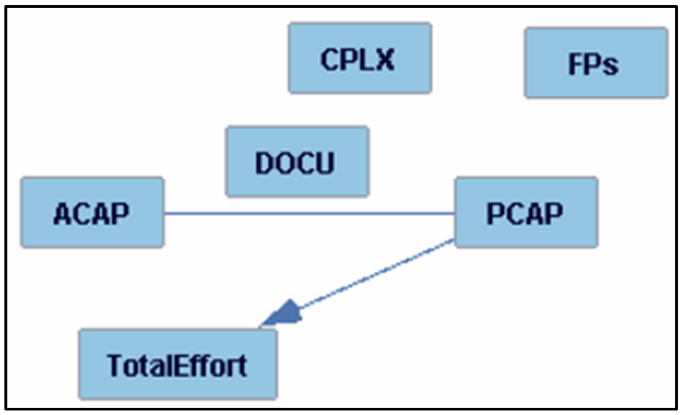

Figure 8. FGES Result When Only IFPUG FPs Is Included as a Size Metric

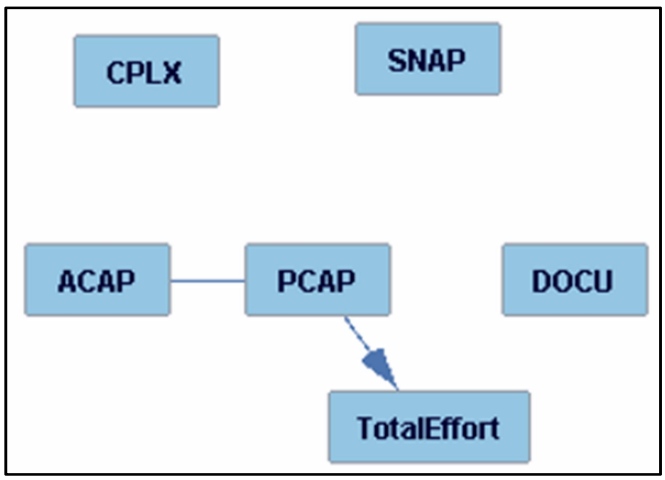

Figure 9. FGES Result When Only IFPUG SNAP Is Included as a Size Metric

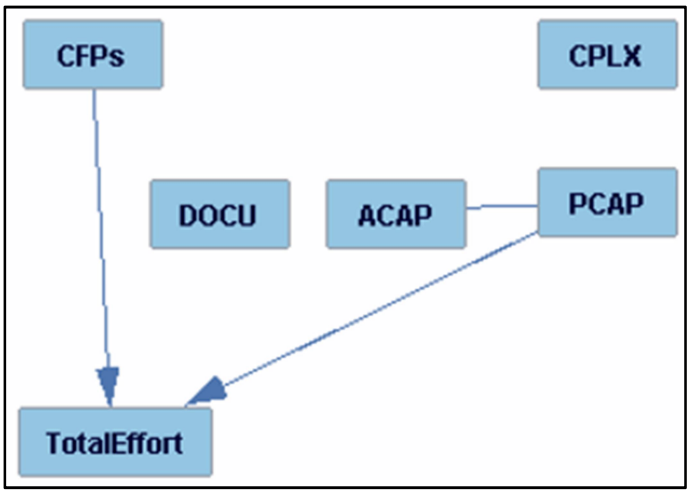

Figure 10. FGES Result When Only CFPs Is Included as a Size Metric

FASK Causal Search Results

Before conducting the FASK causal search, the authors revisited the distribution of each factor as FASK achieves improved edge orientation when distributions are skewed. Table 1 summarizes the results of checking for skewness. Since most variables are skewed, the authors proceeded to run FASK, aware that the lack of full skewness might render some edge orientations incorrectly.

Table 1. Summary of Variables' Skewness

Variable	Skewed
CPLX	YES
CFP	YES
FP	NO
SNAP	YES
ESLOC	YES
ACAP	NO
PCAP	NO
DOCU	YES
Total Effort	YES

The graphical search results from FASK, Figures 11–15, show that the algorithm returned some similar and some very different results compared to both PC-Stable and FGES. Most interestingly, the algorithm returns the factor Documentation Match to Lifecycle Needs (DOCU) as having a causal effect on TotalEffort, along with some of the size metrics. Except when all size metrics are included in the analysis, DOCU is identified as having a causal effect on ACAP, and there is an undirected edge between DOCU and PCAP. Additionally, SNAP is identified as having causal effects on ACAP and DOCU, and FPs as having a causal effect on DOCU.

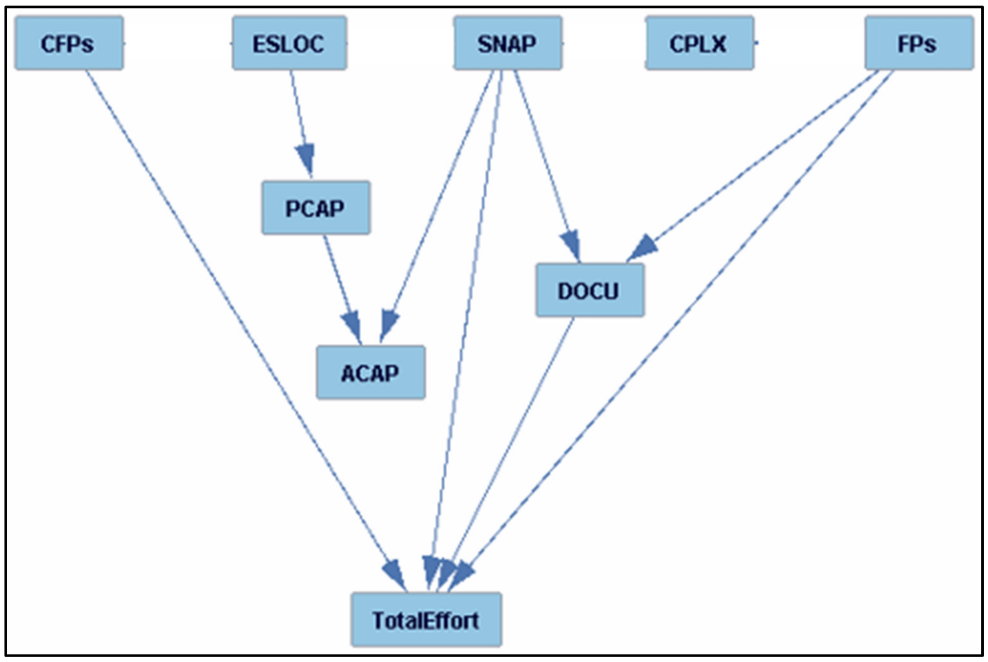

Figure 11. FASK Result When All Size Metrics Are Included

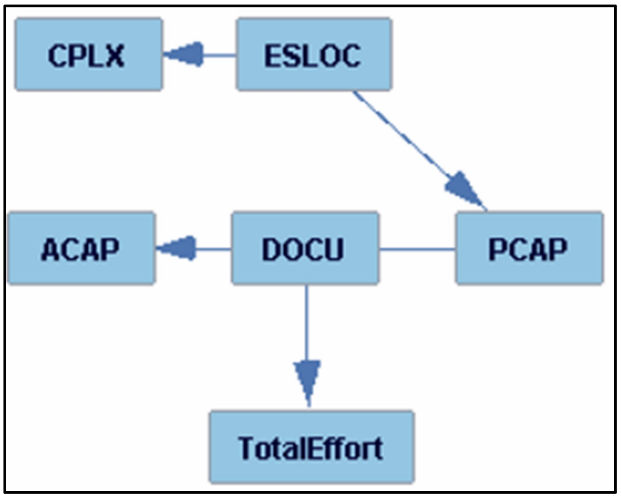

Figure 12. FASK Result When All Size Metrics Are Included

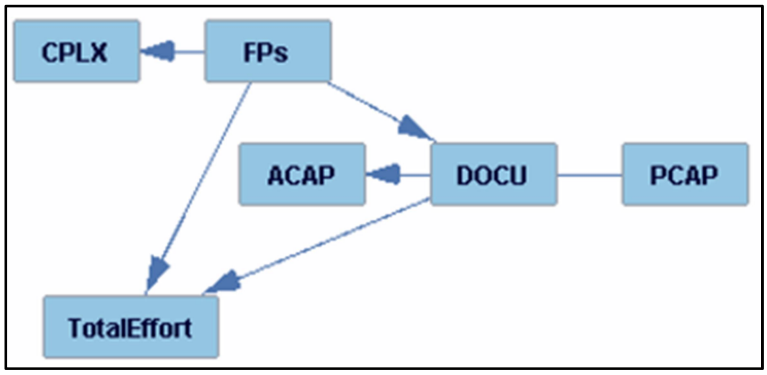

Figure 13. FASK Result When All Size Metrics Are Included

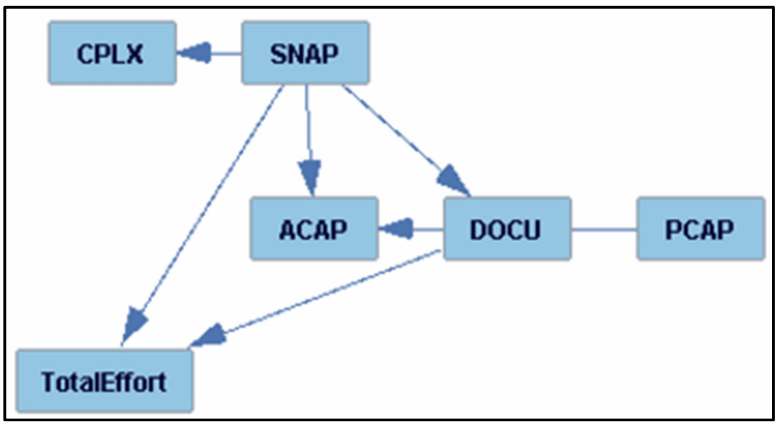

Figure 14. FASK Result When All Size Metrics Are Included

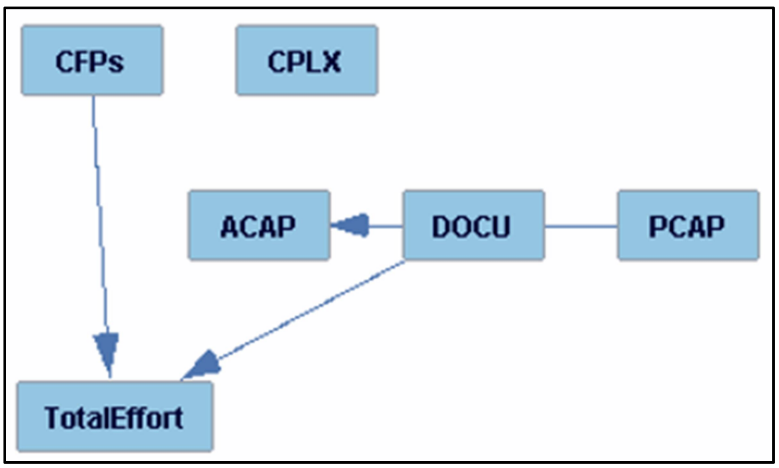

Figure 15. FASK Result When All Size Metrics Are Included

Traditional Stepwise Multiple Regression

In order to determine the most influential factors and an acceptable prediction model, data analysts typically use the stepwise multiple regression approach. As displayed in Table 2, stepwise regression produces results indicating that PCAP and CFPs remain significant in predicting TotalEffort. Although models with three or more factors achieve a reasonably high Adjusted R-Squared, the simplified, two-factor model with PCAP and CFPs produces similar Adjusted R-Squared results. The details of the resulting prediction model are detailed in Table 3, and Figure 16 displays that the residuals of the model are normally distributed and do not have multi-collinearity.

Table 2. Stepwise Regression Results Summary With Variables Considered During Each Run

Vars	R-Sq	R-Sq (adj)	R-Sq (pred)	Mallows Cp	S	CFPs	ESLOC	FPs	SNAP	ACAP	PCAP	DOCU	CPLX
1	57.6	56.1	51.0	35.0	231.50						X		
1	54.6	53.0	47.9	39.3	239.65	X							
2	84.9	83.8	78.3	-2.3	140.54	X					X		
2	78.5	76.9	67.4	6.9	167.87				X		X		
3	85.3	83.6	76.1	-0.9	141.35	X					X		X
3	85.1	83.4	77.2	-0.6	142.30	X		X			X		
4	85.4	83.0	73.0	1.1	143.98	X			X		X		X
4	85.3	83.0	68.6	1.1	144.13	X				X	X		X
5	85.4	82.3	61.9	3.0	146.83	X			X	X	X		X
5	85.4	82.3	67.0	3.0	146.89	X	X		X		X		X
6	85.4	81.6	55.2	5.0	149.92	X	X		X	X	X		X
6	85.4	81.6	59.9	5.0	149.99	X			X	X	X	X	X
7	85.4	80.8	53.0	7.0	153.29	X	X		X	X	X	X	X
7	85.4	80.8	52.0	7.0	153.29	X	X	X	X	X	X		X
8	85.4	79.8	49.5	9.0	156.90	X	X	X	X	X	X	X	X

Table 3. Coefficients and Corresponding P-Values for the Regression Model Predicting TotalEffort

Term	Coef	SE Coef	T-Value	P-Value	VIF
Constant	-1791	244	-7.33	0.000	
PCAP	1948	264	7.38	0.000	1.12
CFPs	76.3	10.9	7.00	0.000	1.12

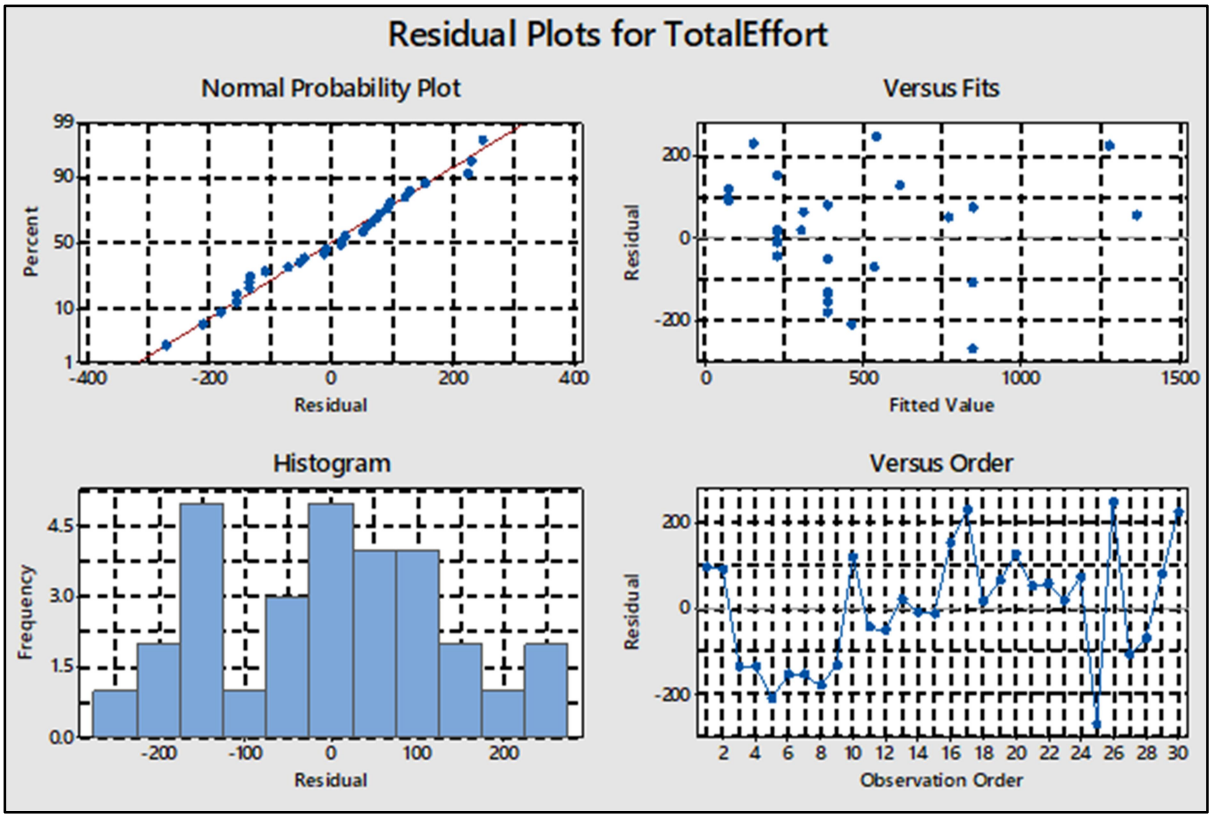

Figure 16. Residual Plots for the Regression Model Predicting TotalEffort Displaying the Fulfillment of Linear Regression Assumptions

Summary of Results

Table 4 depicts a summary of the results of the PC (Hira, Sharma, & Boehm, 2018), PC-Stable, FGES, and FASK causal searches in which the direct and indirect causal factors are noted. Contrary to the traditional stepwise multiple regression approach which identified PCAP and CFPs as significant factors of TotalEffort, the causal search has uncovered direct causal evidence, depending on the algorithm, of FP, SNAP, CFP, CPLX, PCAP, and DOCU. The algorithms also identified ESLOC, FP, SNAP, ACAP, and PCAP as indirect causal drivers of TotalEffort.

There remains a number of ways in which this summary may be interpreted. One approach would be to look for the complete absence or presence of causal relationships as noteworthy new knowledge. Another approach would be to look for direct and indirect causal factors showing up across a majority of causal algorithms. Lastly, one could choose to be much more inclusive and look at all direct and indirect factors that show up at least once.

Acknowledging the small size of the data set and the impact of the small sample size on the causal search results, one can still see that causal search can inform a researcher of alternative independent factors that may actually be the causal force on the dependent factor. As such, given the small sample size, there is evidence that CPLX, PCAP, DOCU, and several size measures (CFP, FP, and SNAP) may be considered as direct causal factors. It is interesting to also note that PCAP only showed up as a direct causal factor on TotalEffort using the FGES score-based algorithm, while DOCU showed up as a direct causal factor using FASK. Among the size measures, the strongest candidate for a direct causal factor for TotalEffort would seem to be CFP. In conclusion, these causal search

results should motivate the researcher to collect additional data and continue analyzing beyond what a traditional regression approach might offer toward the ultimate goal of estimating a quantified actionable model for TotalEffort—an actionable model that is suitable for use in deciding how to intervene and change a project's course toward achieving desired target outcomes. Correlation alone really doesn't provide sufficient insight.

Table 4. Identified Direct and Indirect Causes of TotalEffort by Algorithm and Scenario

	Direct Cause of Total Effort								Indirect Cause of Total Effort							
Algorithm	ESLOC	FP	SNAP	CFP	CPLX	ACAP	PCAP	DOCU	ESLOC	FP	SNAP	CFP	CPLX	ACAP	PCAP	DOCU
PC-AllSizeMetrics				YES			YES				YES			YES		
PC-ESLOC							YES							YES		
PC-FP							YES							YES		
PC-SNAP			YES				YES							YES		
PC-CFP				YES			YES									
PC-Stable-AllSizeMetrics				YES			YES									
PC-Stable-ESLOC					YES				YES							
PC-Stable-FP					YES					YES						
PC-Stable-SNAP					YES						YES					
PC-Stable-CFP				YES									YES			
FGES-AllSizeMetrics				YES			YES							YES		
FGES-ESLOC							YES							YES		
FGES-FP							YES							YES		
FGES-SNAP							YES							YES		
FGES-CFP				YES			YES							YES		
FASK-AllSizeMetrics		YES	YES	YES				YES								
FASK-ESLOC								YES	YES						YES	
FASK-FP		YES						YES							YES	
FASK-SNAP			YES					YES							YES	
FASK-CFP				YES				YES							YES	

Note. Black cells are Not Applicable as factor not present in model.

Implications for Acquisition Research

The authors believe the implications of adding causal searches and ultimately, causal estimations of causal influence, could transform acquisition research in the following ways:

1. The need to pursue expensive and, more likely, prohibitive experiments of acquisition factors could be obviated by use of causal methods appropriate for observational data. (Or at a minimum, causal research findings should be used to more efficiently focus such experiments.)
2. Revisiting research data affiliated with the acquisition research arena could quickly help filter likely causal factors from the merely correlated factors, thereby reducing researcher distraction and accelerating progress in acquisition research.
3. Different acquisition researchers could more easily begin to unite causal conclusions into a more holistic causal model of acquisition research outcomes.
4. The acquisition research community could identify and prioritize constrained research funding towards causal research outcomes worthy of investment in repeatability and reproducibility studies.

5. Causal research findings would be more confidently tested by real-world acquisition program interventions and risk less in wasteful interventions.
6. SEI's acquisition experts indicate specific acquisition outcomes and associated factors worthy of causal learning include
    a. delivering required warfighter capability at an affordable price and on schedule, driven by potential factors such as lack of competencies in the workforce, greater system complexity, and underestimation of cost and schedule,
    b. unrealistic assumptions made by decision-makers very early in the acquisition lifecycle,
    c. program acquisition strategy (and changes in the strategy), the program structure, and the technical challenges facing the program,
    d. leadership incentives to conduct critical thinking and apply evidence-based knowledge and practice early in a program,
    e. people, process, requirements, and incentives in general,
    f. the overall contracting process in context of changes that are often difficult to make and take a lot of time, leading to issues on the contractor side,
    g. the technical strength of the government team,
    h. measurement of agile versus more traditional development, and
    i. measured effectiveness of the Department of Defense (DoD) 5000 mandated reviews.

Next Steps

The authors welcome research collaboration making use of causal search and estimation, especially acquisition research focused on impacts to acquisition cost and schedule. Research collaboration of this nature would complement existing research funded through the SEI as part of a three-year (FY 2018–2020) DoD research project titled "Investigating the Feasibility of an Integrated Causal Model for Software Cost Control (SCOPE)." Many forms of acquisition research collaboration exist, including (1) providing subjective and objective research data, (2) connecting our research team with acquisition managers and organizations who might provide research data, (3) helping to identify the nature of acquisition outcome measures worthy of study along with insights to the potential causal factors of those outcomes, and (4) learning how to conduct causal search and estimation and working with our research team as a direct contributing member or as a reviewer of causal results.

References

Boehm, B. W., Madachy, R., & Steece, B. (2000). *Software cost estimation with Cocomo II with Cdrom.* Prentice Hall.

Center for Causal Discovery. (2017). Fast greedy equivalence search (FGES) algorithm for continuous variables. Retrieved from http://www.ccd.pitt.edu/wiki/index.php?title=Fast_Greedy_Equivalence_Search_(FGES)_Algorithm_for_Continuous_Variables

Colombo, D., & Maathuis, M. H. (2014). Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research, 15*(1), 3741–3782.

Cook, T. D., Campbell, D. T., & Shadish, W. (2002). *Experimental and quasi-experimental designs for generalized causal inference*. Boston, MA: Houghton Mifflin.

Elwert, F. (2013). Graphical causal models. In *Handbook of causal analysis for social research* (pp. 245–273). Dordrecht: Springer.

Hira, A., Sharma, S., & Boehm, B. (2016, May). Calibrating COCOMO® II for projects with high personnel turnover. In *Proceedings of the International Conference on Software and Systems Process* (pp. 51–55). ACM.

Hira, A., et al. (2018). Preliminary causal discovery results with software effort estimation data. In *Proceedings of the 11th Innovations in Software Engineering Conference* (pp. 1–11). ACM.

McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on Software Engineering, 4,* 308–320.

Nguyen, V. (2010, September). Improved size and effort estimation models for software maintenance. In Pro*ceedings of the 2010 IEEE International Conference on Software Maintenance* (ICSM; pp. 1–2). IEEE.

Park, R. E. (1992). *Software size measurement: A framework for counting source statements* (No. CMU/SEI/92-TR-20). Pittsburgh, PA: Carnegie-Mellon University, Software Engineering Institute.

Pearl, J. (2001). Causal inference in the health sciences: A conceptual introduction. Health *Services and Outcomes Research Methodology, 2*(3–4), 189–220.

Sanchez-Romero, R., Ramsey, J. D., Zhang, K., Glymour M. R. K., Huang, B., & Glymour, C. (2018). Causal discovery of feedback networks with functional magnetic resonance imaging. Retrieved from https://www.biorxiv.org/content/early/2018/01/10/245936

Spirtes, P. (2010). Introduction to causal inference. *Journal of Machine Learning Research, 11*(May), 1643–1662.

Acknowledgments

This material is based upon work supported in part by Cyber Security and Information Systems Information Analysis Center (CSIAC) and in part upon work funded and supported by the DoD under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the SEI, a federally-funded research and development center. The authors would like to thank Michael McClendon and Julie Cohen (both of the SEI) for their acquisition insight to high value acquisition targets of causal learning described at the end of the paper. The authors would also like to thank David Zubrow (SEI) for his encouragement and support and for sharing his insights for the work in this paper. Additionally, the authors thank David Danks, Kun Zhang, Madelyn Glymour, and Joe Ramsey for their help in understanding causal search, the search algorithms, and the Tetrad tool.

ACQUISITION RESEARCH PROGRAM
GRADUATE SCHOOL OF BUSINESS & PUBLIC POLICY
NAVAL POSTGRADUATE SCHOOL
555 DYER ROAD, INGERSOLL HALL
MONTEREY, CA 93943

www.acquisitionresearch.net