2024-05-01

Automated Composition of Multivariable Scientific Workflows Considering Scientific Assumptions

Raul Alejandro Vargas Acosta
University of Texas at El Paso

AUTOMATED COMPOSITION OF MULTIVARIABLE SCIENTIFIC WORKFLOWS

CONSIDERING SCIENTIFIC ASSUMPTIONS

RAUL ALEJANDRO VARGAS ACOSTA

Doctoral Program in Computer Science

APPROVED:

Natalia Villanueva Rosales, Ph.D., Chair

Monika Akbar, Ph.D.

Deana D. Pennington, Ph.D.

Stephen L. Crites, Jr., Ph.D.
Dean of the Graduate School

To my mother, Dilia Maria Guadalupe Acosta Barreñada. I feel you e-v-e-r-y-d-a-y.

To my wife Hilda. Your continuous support made this possible.

To Diego and Axel, your innocence has been the most beautiful part of this journey.

AUTOMATED COMPOSITION OF MULTIVARIABLE SCIENTIFIC WORKFLOWS CONSIDERING SCIENTIFIC ASSUMPTIONS

by

RAUL ALEJANDRO VARGAS ACOSTA

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

THE UNIVERSITY OF TEXAS AT EL PASO

May 2024

Acknowledgements

As a first acknowledgement, I would like to thank and recognize my advisor, Dr. Natalia Villanueva Rosales. Her support did not extend only to research advising and financial support. Muchas gracias Natalia.

Secondly, I would like to give a warm acknowledgment to M.S. Luis Garnica. His research advice gave the needed direction to this work. I have no doubts that his experience and work ethic enabled me not to claudicate in the pursuit of my objective. Gracias Luis!

To my committee, Dr. Deana Pennington and Dr. Monika Akbar. Both of them shared their experience in the making of this work. Being able to use our knowledge from Computer Science and apply it in a different field, with a special focus on helping the environment has been an exciting experience, especially in water sustainability. I would also like to extend my gratitude to Dr. David Gutzler and Dr. Alex Mayer for their guidance with domain knowledge in this work.

To my family, they were my pillar for this journey; without them, this would not have been possible. To my cousins Eduardo “Popeye” Carreon and Erika Melendez, as they supported me in any possible way I give my gratitude. To my friend M.S. Ismael Canales, as he always found a way to support me, first with Newcastle University and then at UTEP; to M.S. Francisco Lopez as he allowed those gears to move, I shall remember what he said to me “The world is out there, go get it”.

To all the friends I made while being at UTEP, inside and outside of classes. To my lab-mates, their feedback and critique improved this work, and their emotional support helped me to overcome challenges as a student.

Lastly, I would like to extend my gratitude to Dr. Kelvin Cheu, and the Department of Computer Science at UTEP, for their financial support as a Research Assistant, Teaching Assistant, and Assistant Instructor.

Workflow composition, as described in the “Automating Multivariable Workflow Composition for Model-to-Model Integration” e-Science 2022 conference paper, was based upon work supported by the National Science Foundation. This material is based upon work supported by the National Science Foundation under Grant No. 1835897. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Abstract

Many ground-breaking scientific experiments require the execution of multiple complex scientific computations. Thus, scientific workflows (i.e., a sequence of scientific computations) have received significant attention, more specifically, the automated composition of scientific workflows. Scientific workflows that repurpose data may have unique scientific assumptions that need to be considered when composing a workflow. Workflow composition tools have enabled a wider range of stakeholders (e.g., policymakers, the general public, and researchers) to create and execute workflows; however, domain expertise is still required for these tasks. The overarching goal of this work is to further improve the automatic composition of scientific workflows by validating if the scientific assumptions taken during the creation of a dataset are aligned with the scientific assumptions required to use these datasets for a specific scientific computation. This work aims to answer the following research questions: How can metadata and provenance be used to describe scientific assumptions of data consumed by scientific computations for the improvement of automated scientific workflow composition and repurposing of data? and to what extent can current Artificial Intelligence (AI) planning techniques with a heuristic function be used to formulate a scientific workflow that considers scientific assumptions in a hydrology domain?

Our initial work focused on exploring automatic workflow composition with components that require and produce multiple scientific variables for an abstract case study (i.e., domain-free) using graph traversal. In addition, a second case study was conducted for a real-world hydrology scenario, which provided us with insights into how scientific assumptions could be described to enable model-to-model integration. In both cases, abstract and real-world scenarios, we use domain-independent vocabulary to represent a workflow for interoperability between different workflow management systems. We extended existing and widely used ontologies and

vocabularies for describing scientific assumptions that are used in the automated composition of workflows. In addition, we propose a heuristic function for optimizing the algorithm. Our work aims to support scientific decision-making by enabling a wider range of stakeholders (e.g., policymakers, the general public) to automatically generate scientific workflows leveraging additional domain knowledge captured in metadata that can be executed in frameworks compatible with the standard workflow language used in this work.

Table of Contents

Acknowledgements v

Abstract vii

Table of Contents ix

List of Tables xi

List of Figures xii

Chapter 1: Introduction 1

1.1 Motivation 2

1.2 Scope of the Research 3

1.3 Out of Scope of the Research 4

1.4 Research Questions 5

1.5 Goal and Objectives 5

1.6 Contributions 7

Chapter 2: Background and Related Work 8

2.1 Knowledge Representation 8

2.2 Informed Search Strategies 10

2.3 Composing Scientific Workflows 11

2.4 Enabling Variable Reconciliation 14

2.5 Component Selection: A Tie-Breaking Strategy 15

2.6 Annotating Computational Components 16

Chapter 3: Research Methodology 17

3.1 Automating Single Variable Workflow Composition 18

3.1.1 Automated Single Variable Workflow Composition: Approach 18

3.2 Automating Multivariable Workflow Composition 19

3.2.1 Automated Multivariable Workflow Composition: Approach 21

3.3 Ontology-Driven Scientific Variable Reconciliation 24

3.3.1 Variable Reconciliation: Approach 25

3.4 Target Variables: A Tie-Breaking Strategy 27

3.4.1 Tie-Breaking Strategy: Approach 27

Chapter 4: Results 30
4.1 Automatic Workflow Composition 30
4.1.1 The Workflow Composer Service 30
4.1.2 The Workflow CWL Service 31
4.1.3 Case Study 32
Overview 32
4.1.4 SWIM Workflow Input 34
4.1.5 SWIM Workflow Output 35
4.1.6 Automatic Workflow Composition: Results 36
4.2 Variable Reconciliation 37
4.2.1 Ontology 37
4.2.2 Ontology Population 39
4.2.3 Automated Reasoning 40
4.2.4 Variable Reconciliation: Results 42
4.3 Target Variables: A Tie-Breaking Strategy 48
4.3.1 Tie-Breaking Strategy: Results 48
Chapter 5: Evaluation and Discussion 49
5.1 Automatic Workflow Composition 49
5.2 Variable Reconciliation 50
5.3 Tie-Breaking Strategy 52
Chapter 6: Conclusions and Future Work 55
References 61
Appendix 73
Appendix A – Test cases for workflow composition using our tie-breaking strategy 73
Curriculum Vitae 76

List of Tables

Table 1 Variable Annotation Example 9

Table 2 Pseudocode for Multivariable Workflow Composition © 2022 IEEE 22

Table 3 Possible Workflow Paths © 2022 IEEE 24

Table 4 Variable Annotation Example 26

Table 5 Pseudocode for Populating the VaR-O with Scientific Variables 26

Table 6 Pseudocode Modification for Multivariable Workflow Composition © 2022 IEEE 28

Table 7 Pseudocode for Tie-Breaking Strategy Part I 28

Table 8 Pseudocode for Tie-Breaking Strategy Part II 28

Table 9 SWIM Workflow Input Excerpt. 34

Table 10 SWIM Workflow Output Excerpt. 35

Table 11 Super Class Description Example 38

Table 12 Super class description example 39

Table 13 Example of Scientific Variable Described in our Ontology 40

Table 14 Equivalent Individuals Serialized as JSON 41

Table 15 Equivalent Individuals Ignoring Scientific Assumptions 42

Table 16 Annotation Example as JSON. 44

Table 17 Continuous Value Annotation Example as JSON. 45

Table 18 Available Data for Abstract Case Study in JSON. 45

Table 19 Component Catalog for Abstract Case Study in JSON. 46

Table 20 Expected Output for our Variable Reconciliation. 48

Table 21 Output Obtained for our Variable Reconciliation. 48

List of Figures

Figure 1 Graphical Representation of Our Methodology 18

Figure 2 Each Node in the Abstract Graph Digests and Produces a Single Data Element, Which is Digested by the Next Node in the Graph. 18

Figure 3 DFD of a Multivariable Workflow Example. The Dashed Path Passes Through Candidate Nodes, the Blue Solid Path is Selected for the Workflow. © 2022 IEEE 21

Figure 4 DFD of a Multivariable Workflow Example After Implementing our Tie-Breaking Strategy. Green Nodes are not Discovered Given the Heuristic Used. 29

Figure 5 DAG of a SWIM Model-to-Model Integration Scenario Derived from the Case Study. © 2022 IEEE 33

Figure 6 Surface Water Evaporation Depth Output of the WBM © 2022 IEEE. 37

Figure 7 Reservoir Evaporation Rate Input to the HEM © 2022 IEEE. 37

Figure 8 Elephant Butte Reservoir Storage Projected by the WBM © 2022 IEEE. 37

Figure 9 Surface Water Storage Projected by HEM. This Output is a Sum of the Two Regional Reservoirs, Elephant Butte and El Caballo © 2022 IEEE. 37

Figure 10 Data Flow Visual Representation 44

Figure 11 Graphical Depiction of Models Used as Part of the Case Study. 51

Chapter 1: Introduction

Current ground-breaking scientific experiments are taking advantage of executing multiple complex scientific computations sequentially (Carrillo et al., 2019; Davies et al., 2020; Maechling et al., 2005; Pavlovikj et al., 2014; Reed et al., 2007; Riedel et al., 2018; Vahi et al., 2018; Zia et al., 2016). The referred scientific computations might have unique scientific assumptions that need to be considered when executing sequential computations. As a result, scientific workflows (i.e., the representation of "complex distributed scientific computations" (Gil et al., 2007a) have received significant attention, more specifically on the composition of scientific workflows (Gil et al., 2010; Kasalica & Lamprecht, 2020; Kim et al., 2006) with a focus on data repurposing.

Scientific workflow composition deals with defining a sequence of complex scientific computations for producing a desirable output. Although diverse approaches to the composition of scientific workflows have been proposed (Gil et al., 2010; Kasalica & Lamprecht, 2020), there is an interest in identifying and chaining compatible scientific computations taking into consideration their unique assumptions. We define scientific assumptions as all design decisions made during the creation of a computational component that affects data produced by a scientific computation. These scientific assumptions are of relevant interest as they might impact when datasets should be used (or should not).

For example, Holmes et al., (2022) describe a water balance model used to better understand the hydrologic effects of climate change in the local reservoirs located in the Rio Grande river region (R. N. Holmes et al., 2022). This water balance model requires inflows to the region to project water storage in local reservoirs. The U.S. Bureau of Reclamation, using the VIC hydrologic routing model (Brekke et al., 2014), provides region-specific streamflow without human-made alterations (R. N. Holmes et al., 2022). If the output dataset from the U.S. Bureau of

Reclamation is used as input to a regional surface water model then erroneous conclusions might be generated due to a mismatch of spatial scale (global vs. regional) and not considering anthropogenic impairments within the region. Therefore, validating that input data, either provided by a third party (e.g., governmental agencies) or self-collected, is suitable for consumption by a specific scientific computation is of vital importance.

This dissertation focuses on further improving the automatic composition of scientific workflows with a special focus on repurposing and reuse of data. We state that our work will serve as an initial step for stakeholders to sketch out an initial workflow. In this chapter, we elaborate on our motivation to further improve the composition of workflows as well as the scope of the research and objectives; contributions are described at the end of the chapter. In Chapter 2 we present related work that is used as the foundation for this work. Our methodology and approach are described in Chapter 3. We present our results and discussion in Chapter 4 and Chapter 5 outlines our conclusions and future work. Happy reading.

1.1 MOTIVATION

Understanding and predicting environmental phenomena is an urgent need in today's modern society as it enables stakeholders (e.g., scientists, and policymakers) to make informed decisions (Carrillo et al., 2019). Being able to predict or to make projections of this type of phenomenon, assists stakeholders in the uptake of scientific conclusions fostering better management of non-renewable natural resources, (e.g., water (Ward et al., 2019a)), or understanding our environment, (e.g., the study of celestial bodies (Gil et al., 2010; Vahi et al., 2018)).

Current practices on scientific research take advantage of computing power to accelerate their calculations, thus the design, development, and sharing of scientific computations are

becoming a common practice among the scientific community as shown in the collaborative effort led by the HydroShare team to enable scientists to discover hydrologic data and models (Tarboton et al., 2014). Moreover, chaining the execution of several scientific computations is becoming a requirement among diverse scientific experiments.

With this goal in mind, workflows are adopted and used as scientific workflows to capture and communicate the steps taken in a scientific experiment as well as the location of the datasets used. Scientific workflows specify, without ambiguity, which computational tasks must be performed, and the scientific assumptions data should comply with to be used in the experiment (Carvalho et al., 2017; Deelman et al., 2018; Gil et al., 2007a). However, composing scientific workflows currently requires manual selection of scientific computations from domain experts. It is therefore of interest in this work to aim to further improve the composition of scientific workflows by automatically selecting scientific components.

Within this work, we will refer to computational tasks as scientific components or scientific computations, an example of these components is a scientific model. We focus on creating workflows of components that digest and produce multiple data values, within our work we refer to those values as scientific variables, therefore the name of our work is denominated as the automated composition of multivariable scientific workflows.

1.2 Scope of the Research

Our work relies on identifying the capabilities of knowledge representation languages for describing the scientific assumptions adopted when generating scientific data in scientific computations (i.e., a semantically annotated model catalog) for the further improvement of scientific workflow composition based on scientific assumptions. Our catalog will focus on

describing scientific variables and their assumptions when generated/consumed by scientific components.

Our work is evaluated using an abstract case study and a real-world case study in the environmental sciences, more specifically, hydrology. It is of interest to local scientists in the El Paso region to model the hydrologic behavior of the region. The Sustainable Water through Integrated Modeling (SWIM) (Garnica Chavira et al., 2022) is a framework that contributes to this objective; therefore, the SWIM framework is part of our case study and enables us to implement and evaluate our approach.

1.3 Out of Scope of the Research

The automatic generation of annotations of scientific variables and computations is out of the scope of this project. Scientific variables produced and required by scientific computations need to be manually annotated to generate a workflow. The process of how to manually or automatically annotate scientific assumptions is not covered in this dissertation.

In addition, we envision that our work will be an initial step to automatically compose workflows that consume data contained in files, rather than data contained in a web service request. This type of workflow will require further representation of data digested/produced by the workflow.

Evaluating the portability of our approach will include the use of relevant existing workflow management systems. The implementation of these tools is provided by third-party agents.

While executing two or more scientific components within a workflow, it can be found that two or more scientific variables produced by different scientific components must follow a specific scientific pattern (i.e., to identify if there is a correlation or causation between variables). This type

of validation involves the analysis of the computations within the scientific components, i.e., scientific component validation, therefore this is indicated as future work.

1.4 Research Questions

1. How can metadata and provenance be used to describe scientific assumptions of data consumed by scientific computations for the improvement of automated scientific workflow composition and repurposing of data?
2. To what extent can current Artificial Intelligence (AI) planning techniques with a heuristic function be used to formulate a scientific workflow that considers scientific assumptions evaluated in a scientific domain?

1.5 Goal and Objectives

The overarching goal of this research is to further improve the automatic composition of scientific workflows by validating if the scientific assumptions taken during the creation of a dataset are equivalent to the scientific assumptions data must comply with if used as input in a subsequent scientific computation. The following specific objective addresses research question one.

O1. to describe and enhance current approaches to scientific workflow composition considering scientific assumptions

A1.1. investigate advantages and limitations of current approaches to scientific workflow composition

A1.2. extend current approaches to scientific workflow composition using a method that considers scientific assumptions

A1.3. investigate state-of-the-art approaches for representing scientific workflows using domain-independent specific vocabulary

For addressing research question two, we define the following specific objective.

O2. to define how knowledge representation languages can assist in describing the scientific assumptions adopted when generating scientific data in scientific computations

A2.1. investigate current approaches on describing scientific assumptions of data generated in scientific computations using knowledge representation languages

A2.2. extend current approaches using knowledge representation languages to describe scientific assumptions of data generated in scientific computations for the automatic composition of scientific workflows

Finally, we propose a third objective for evaluating our work.

O3. to evaluate the role of semantically annotated data in enhancing the composition of scientific workflows using scientific assumptions

A3.1. validate the behavior of the workflow composer when creating workflows on different operating parameters

A3.2. validate the portability of the composed scientific workflow in diverse workflow management systems

A3.3. evaluate the correctness of the scientific workflows used for evaluation with domain experts in a scientific domain.

1.6 CONTRIBUTIONS

The contributions of this dissertation are:

1. A methodology for the automatic composition of scientific workflows composed of multiple sequenced input/output scientific computations that consider scientific assumptions in data generation.
2. Knowledge representation vocabularies for describing the scientific assumptions when generating data from scientific computations.

Chapter 2: Background and Related Work

2.1 Knowledge Representation

One approach used to formally describe a specific domain is an ontology. Hitzler et al. (2020) describe an ontology as "a description of knowledge about a domain of interest, the core of which is a machine-processable specification with a formally defined meaning" (Hitzler et al., 2010). In the context of this work, we describe variables as individuals; their relationship to other individuals, referred to as object properties; attributes, which are called data properties; and their types, referred to as classes in an ontology.

Given the domain of interest, it might be required to have a more specific definition of the class. In other words, a class might be decomposed into subclasses, the parent class is referred to as superclass in these situations. An example is given by Noy and McGuinness (Noy & McGuinness, 2001) in which they define the set of all wines to be classified into red, white, and rosé. In this example, red, white, and rosé are examples of subclasses, and the parent class, wine, is the superclass.

Ontologies can be created with the use of the Web Ontology Language (OWL) (World Wide Web Consortium, 2012). OWL is a standardized language that extends the Resource Description Framework (RDF). RDF is a "general-purpose language for representing information in the Web" (Brickley et al., 2004), a domain of interest is then described using a triple: subject-predicate-object using RDF syntax. An example of how RDF is used for describing a domain is seen in Table 1, the "prefix" annotation denotes that we will use the vocabulary described by the RDF vocabulary[1] . The triple is composed of the subject: data5, the predicate: rdf:type, and the

[1] http://www.w3.org/1999/02/22-rdf-syntax-ns#

object: WaterFlow. For the purpose of this work, we make use of Protégé (Musen, 2015) as a tool for describing our domain of interest, which is scientific components.

Table 1 Variable Annotation Example

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

data5 rdf:type WaterFlow .
```

While RDF can be used to serialize ontologies, the expressiveness of RDF is limited in comparison with OWL. OWL allows the creation of hierarchies, a class can be defined as the intersection of other classes, and it also allows the definition of properties (Horrocks et al., 2003). It is for these reasons that OWL is the preferred method of serializing our ontology.

With the formal description of the domain of interest, logical conclusions can be computed through classes, data properties, and/or object properties. To further explain our definition of a logical conclusion we use a typical example as follows: Assume that statement 1 states that “All humans have a birthdate”; statement 2 establishes that “Diego is human”; in consequence, we can logically conclude that “Diego has a birthdate”. A logical conclusion is described as an inference, given the widespread adoption of ontologies there are different types of inference engines, known as reasoners, to facilitate the automatic generation of logical conclusions.

Reasoners make use of formal knowledge representation languages for the process of deriving logical conclusions. Description Logics (DL) is a formal knowledge representation language (Baader et al., 2007) used by the automatic reasoner Hermit (Shearer et al., 2008). RDF and DL work in tandem to describe a domain and for the making of inferences. We rely upon the utilization of ontologies and reasoners to identify equivalence between scientific variables. Our proposed variable reconciliation approach, described in 3.3 Ontology-Driven Scientific Variable Reconciliation, makes use of ontologies to formally describe scientific components and their

scientific variables (i.e., input and output scientific variables), Hermit then allows the identification of equivalent variables using superclasses. We rely upon the capabilities of Hermit instead of other reasoner tools, for example, Pellet (Sirin et al., 2007), as its technical abilities and learning curve are more suitable for our needs to perform subsumptions and inferences operations.

2.2 INFORMED SEARCH STRATEGIES

In the broad context of searching, diverse strategies are used. An uninformed search is based on using a breath or depth-first search (Russell & Norvig, 1995). However, a strategy using an uninformed search might lead to exploring the totality of the search space to find a specific target, as shown in our work. To reduce the exploration of the totality of the search space, an informed search strategy is typically used. This type of strategy makes use of a heuristic function, Russell and Norvig (1995) define a heuristic function as an "estimated cost of the cheapest path from the state at node n to a goal state" (Russell & Norvig, 1995), in our context, a heuristic function aids in guiding our algorithm to compose a scientific workflow.

In terms of our very own heuristic function, our algorithm is classified as a greedy search algorithm. Per Russell & Norvig (1995), greedy search "tries to expand the node that is closest to the goal, on the grounds that this is likely to lead to a solution quickly". That is, our approach involves the making of the best logical choice given with local information. In other words, when a problem is divided into smaller subproblems, a decision is taken based on the information available to the subproblem, not the problem overall. An example is shown in our research, in which we aim to choose a scientific component to analyze based on the data it generates (local information) described in 3.4 Target Variables: A Tie-Breaking Strategy.

2.3 Composing Scientific Workflows

Manual and automatic scientific workflow composition have been addressed in previous work. In this section, we review approaches for manually describing workflows in different scientific domains, which require a domain expert to sketch out how the scientific components should be executed. In contrast, we outline how automatic approaches for the composition of workflows can use graph theory, abstract workflows, or SAT solvers (i.e., to evaluate if there is a set of values that yield a propositional formula to true (Zhang & Malik, 2002)).

Automatically composing a type of workflow is a challenge addressed and described in diverse areas. In software engineering, for example, efforts were made to define a sequence of services to achieve a specific task. In (Oberhauser & Stigler, 2017), Oberhauser and Stigler describe their approach for automatically composing workflows of microservices. Their generated workflow, defined as a microflow, describes the execution of microservices (i.e., an element in which every functionality of a solution is divided into diverse components allowing to upgrade or replace them as required (Jamshidi et al., 2018)).

Oberhauser and Stigler's approach consists of creating workflows of microservices in which every microservice generates a single element or kind of data; this allows for a directed cycle graph to be pre-defined. Having a pre-defined graph enables the retrieval of a path to execute a specific task using a graph algorithm, such as the shortest path (i.e., finding the shortest path between two nodes). Our work described in 3.2 Automating Multivariable Workflow Composition, leverages Oberhauser and Stigler's work by following their approach to manually annotate microservices and by implementing a graph traversal algorithm; however, our approach focuses on composing a path of computational components that digest and produce multiple scientific variables.

Other efforts to compose workflows are found in diverse scientific domains. In (Carrillo et al., 2019; Davies et al., 2020; Maechling et al., 2005; Pavlovikj et al., 2014; Reed et al., 2007; Riedel et al., 2018; Vahi et al., 2018; Zia et al., 2016) several examples of workflows and their implementations are described. Another example of workflow composition is found in (Del Rio et al., 2013; Villanueva-Rosales et al., 2015), who make use of semantic web technologies for composing workflows that allow users to make projections of species distribution based on specific climate scenarios.

Del Rio et al., (2013) also make use of Semantic Health and Research Environment (SHARE) (Vandervalk et al., 2009), through the Semantic Automated Data Integration framework (SADI) (Wilkinson et al., 2011), for composing workflows to enable the transformation of data to satisfy the input requirements of models (Del Rio et al., 2013).

The data transformation workflows defined by Del Rio et al., (2013) are manually composed and are not computational component specific as they do not specify which components are to be executed. Instead, the workflows operate to define the characteristics and restrictions of the components to be executed as well as the order in which they must be executed. These specifications are then fed to SHARE for retrieval of components that satisfy the restrictions specified (Vandervalk et al., 2009) and for building a working workflow of computational components.

The approach taken by SHARE requires the definition of the classification, restrictions, and the sequence of execution of the components. This characteristic requires a domain expert to manually define a pre-processed workflow. This approach is also adopted by Gil et al., in (Gil et al., 2010) for composing a scientific workflow.

Gil et al., approach is focused on representing scientific computations (Gil et al., 2007a). Their approach, denoted Wings, requires input descriptions and a workflow template (i.e., a sketch of the scientific experiment to be executed) to generate an abstract workflow or workflow instance. This instance contains the data location of the scientific experiment to be then mapped to an executable workflow. This executable workflow is then used as input by a workflow enactment solution (e.g., Pegasus (Deelman et al., 2019a)) for its execution (Kim et al., 2006).

The literature review described in the next two paragraphs was originally published in (Vargas-Acosta et al., 2022) © 2022 IEEE.

Subsequently, the MINT project was proposed by Gil et al. to assist users with cross-disciplinary model integration building on existing tools, including CSDMS (Peckham, 2014), BMI (Ferreira da Silva et al., 2018), GSN (Garijo et al., 2018), WINGS (Gil et al., 2011), Pegasus (Deelman et al., 2019b), Karma (Gupta et al., 2015), and GOPHER (Karpatne et al., 2016). MINT's approach includes using semantic representations to describe model requirements and data characteristics, automatic planning through abductive reasoning techniques, a data discovery and integration framework, and machine learning algorithms for model parameterization (Gil et al., 2018). Their current implementation is being used to explore the role of weather on food availability in select regions of the world (Gil et al., 2021).

Kasalica et al. present the Automated Pipeline Explorer (APE), a synthesis-based workflow discovery framework for automated workflow composition. APE captures technical domain knowledge in taxonomic and functional tool annotations. The user intent of required output data is then modeled through temporal constraints using Semantic Linear Time Logic (SLTL) and then translated as a propositional logic instance that can be solved with a SAT solver. This tool has

been applied to the geospatial domain to map waterbird movement patterns in the Netherlands (Kasalica & Lamprecht, 2020).

Approaches taken by SHARE (Vandervalk et al., 2009) and Wings (Gil et al., 2010) require a domain expert to define abstract computational components and their execution sequence as well as their specific constraints (i.e., a workflow template). The former approach digests the pre-processed workflow serialized as a SPARQL query (Vandervalk et al., 2009) while the latter provides a graphical user interface that allows an expert to sketch out a workflow template. We propose the automated composition of a workflow without manual guidance, contrary to the previously mentioned approaches.

2.4 Enabling Variable Reconciliation

In our approach, it is required for variables to be analyzed and identify those that are equivalent between each other given their classification and representation of scientific assumptions. This step will enable component-to-component integration, in other words, identifying that an output variable can be used as an input variable for another model will enable the creation of workflows. In this section, we describe previous efforts in describing variables and identify areas of opportunities.

Madin et al., (2007) present the Extensible Observation Ontology (OBOE) for describing and discovering scientific variables and their units. They describe discovery as "the process of locating relevant and available data related to a specified topic of interest." Their approach is to first classify variables in classes (e.g., weight) and query the ontology for variables using the classification. Their approach enables the description of scientific variables of different domains, as explained in their work. Their framework allows us to reconcile scientific variables by extending

their ontology to describe scientific assumptions with a custom classification and facilitating component-to-component integration (i.e., composing a workflow).

2.5 Component Selection: A Tie-Breaking Strategy

Matching algorithms are used in different contexts. Chen et al., (2022) propose a matching algorithm with an adaptative tie-breaking strategy to match food orders with food riders. In e-commerce, more specifically in the online-to-offline business, online food delivery has become popular. This type of service enables customers to order food by using technology (e.g., smartphones). Logistics of delivering food proves to require a matching algorithm for matching food orders and riders, and a tie-breaking strategy. Chen et al., (2022) besides proposing their matching algorithm, propose a heuristic based on matching orders to the best riders available. This strategy enables to ensure quality of service (i.e., delivery on time) by evaluating riders and prioritizing food orders. However, this strategy is not perfect as Chen et al., recognize a need to break ties arises if two or more food orders are matched to a single rider. It is then that they further propose a tie-breaking strategy based on operators.

Their proposed strategy consists of selecting one out of five greedy methods for matching a food order to a rider. Their different tie-breaking operators consist of computing dispatching costs (i.e., the cost of dispatching a rider with a set of food orders), time, and distance. This type of heuristics enables them to formulate a solution based on previous information (e.g., a catalog of riders, and traveling distance). Chen et al., (2022) provide a similar problem to the one we describe in this work but applied in a different domain. We recognize similarities between their work and ours for example their approach to matching food orders to best riders can be implemented as ranking computational components (e.g., scientific models) based on precision,

accuracy, correctness, resources needed, and time to execute, among others. However, we do not consider those attributes as, in our case study, similar information is scarce.

A second likeness identified is that their tie-breaking strategy consists of a greedy approach based on known data, for example, traveling distance or riders' performance. While we must adhere to the fact that our information is limited, we utilize variables produced by components to prioritize them and therefore, break the tie.

2.6 Annotating Computational Components

The development of a computational component can be an important step for research projects. In (Harpham et al., 2019; Khattar et al., 2021), an approach for sharing models and data is presented. Furthermore, (Harpham et al., 2019) propose a tool for documenting computational components. They elaborate that their tool can be incorporated into any component without a need to modify the code of the component.

The Open Modeling Interface (OpenMI), as defined by (Harpham et al., 2019), consists of a Software Development Kit that can be incorporated into the code of a component, therefore creating a wrapper for a component. This approach can be of assistance in creating machine-readable descriptions of the components.

The OpenMI standard has been adopted by different frameworks. Adoption examples are (Buahin & Horsburgh, 2018; Bulatewicz et al., 2010; Fotopoulos et al., 2010). The effort made by (Harpham et al., 2019) allows to use a standardized approach to document components and enables the discovery by workflow composition frameworks. This approach requires documentation in the code while our approach requires annotations (i.e., metadata) that are not included in the code.

Chapter 3: Research Methodology

This section describes our methodology for addressing the composition of multivariable workflows, a graphical representation is found in Figure 1. First, we address the composition of single variable workflows, in other words, chaining components that consume and produce a single variable. Addressing this challenge is suitable to be addressed as it presents the challenges of chaining components. In this work, we learned that a machine-readable description of components is needed for analyzing components and that a directed acyclic graph can improve the composition of this type of workflow.

As an incremental step, we designed an algorithm for composing workflows of components that consume and produce more than one variable. The challenge in this type of workflow lies in the uncertainty of having all the required data to execute a component. Given this uncertainty, a directed graph for all possible cases is not a suitable approach, therefore we propose the exploration of the components catalog for dynamically composing a directed acyclic graph that represents the workflow. In this work, two challenges are identified: an automatic step for variable reconciliation and a tie-breaking strategy for selecting a component to be analyzed.

In the third and fourth subsections of this chapter, we present our approach for identifying equivalent variables given their type and their scientific design decisions (i.e., scientific assumptions), and our strategy for selecting a component.

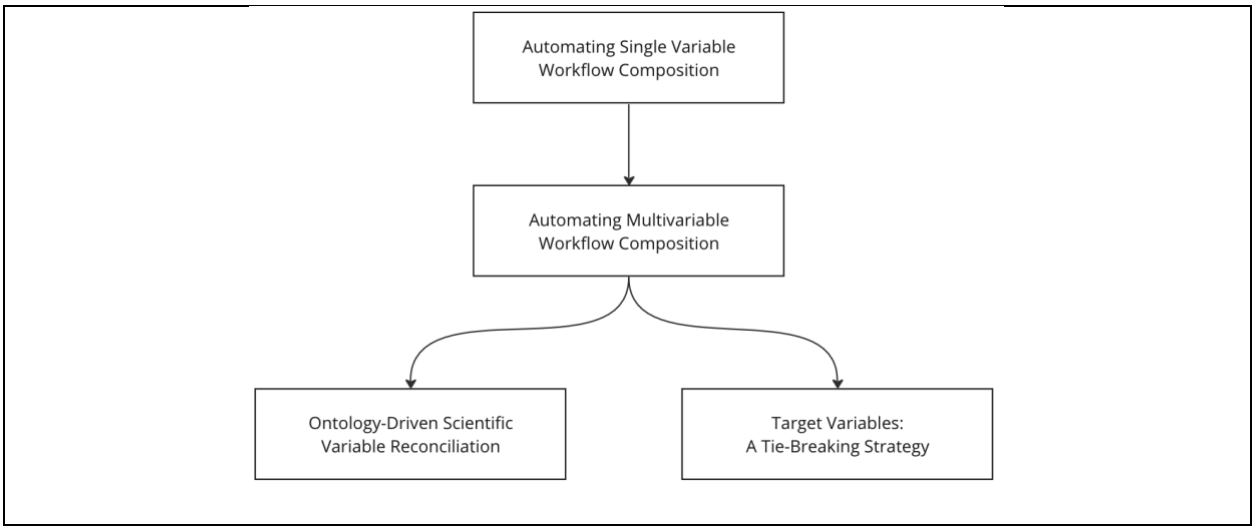

Figure 1 Graphical Representation of Our Methodology

3.1 AUTOMATING SINGLE VARIABLE WORKFLOW COMPOSITION

In our first approach to workflow composition, we explored the creation of a workflow to perform unit transformation. Unit transformation is sometimes required to perform operations or to correctly display the information so end-users can better understand the results. This case study proved to be a good initial exploration as the components in the workflow consume and produce a single variable.

3.1.1 Automated Single Variable Workflow Composition: Approach

An abstract graph is shown in Figure 2 in which every node, a computational component (circle), digests and produces a single scientific variable (e.g., b, c, d). An edge between two nodes denotes that executing a node produces enough data to execute the next node.

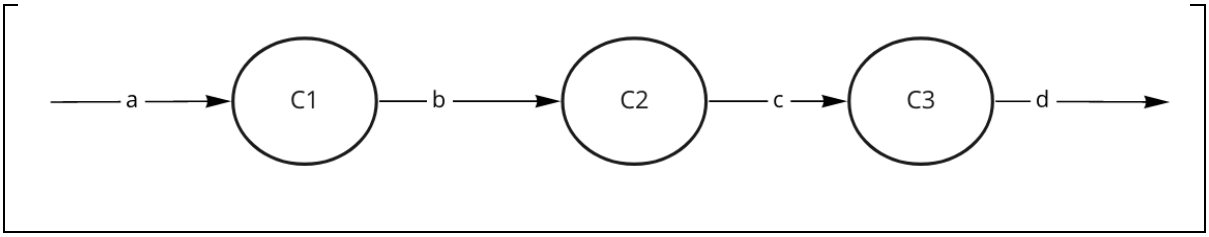

Figure 2 Each Node in the Abstract Graph Digests and Produces a Single Data Element, Which is Digested by the Next Node in the Graph.

We focused on implementing an algorithm to compose a workflow for performing unit transformation. Our solution involved the design and implementation of a directed graph in which every node represents a Typescript function, a unit transformation function (e.g., convert from kilometers to miles), and every edge in the graph implies that a transformation function digests a unit generated by another transformation function.

The directed graph is then traversed by an algorithm to find the shortest path between a source node (i.e., a transformation function that digests the same unit given as the input of the problem) and a target node (i.e., a transformation function that produces the same unit given as the desired output of the problem). If such a path exists, then this path is stored in temporary memory to be reused if needed again.

This first attempt to enable the automated workflow composition allowed us to identify key aspects and challenges in this area, i) semantic annotations are needed for identifying suitable computational components, and ii) a directed graph enables the identification of a path in a single input/output scientific component. However, identifying a path in a graph is not a trivial task if the scientific variables required by a node are not available. This type of workflow in which a component produces and digests a single variable, will have predefined paths. In the next subsection, we describe our approach to composing workflows of components that produce/consume multiple variables.

3.2 Automating Multivariable Workflow Composition

This section is composed of an explanation of the importance of building scientific workflows, followed by our approach to automatically construct workflows, and finally a description of the required components to implement our workflow composer as well as executing scientific workflows.

This section details our approach described in the 2022 IEEE 18th International Conference on eScience (Vargas-Acosta et al., 2022) © 2022 IEEE for automatically composing scientific workflows as well as our initial efforts to incorporate the mentioned approach into an existing research software solution.

Computational workflows, particularly scientific workflows, address the challenge of integrating data sources, methods, and computational models across different domains (Carrillo et al., 2019). At the core level, computational workflows capture the computational steps and data dependencies required to execute computational experiments (Gil et al., 2007a). Computational workflows are widely used across domains, including those that require expensive computational processes.

Creating a workflow usually starts at a conceptual level with the use of abstract representations such as Data Flow Diagrams (DFD) and Directed Acyclic Graphs (DAGs). Once scientists conceptualize and capture their computational process as declarative workflow structures, they can use workflow-management systems (WMS) to support their scientific endeavors by "creating, merging, executing, and reusing these processes" (Gil et al., 2007b). Declarative workflows can then be serialized to a target WMS system using programmatic libraries or following tool-specific syntax and structure. The workflow specification may also require metadata to locate data and jobs across distributed computational environments, along with data transfer protocols and credentials. The disparity of workflow specification languages across WMSs was addressed by standardization efforts such as the Common Workflow Language (CWL) (Crusoe et al., 2022).

Despite broad access and standardization efforts, WMSs still require users to have a high level of computational expertise and collaboration across scientific teams (Gil et al., 2007b). This

may require a high learning curve for the public (i.e., non-scientists) and users who lack the necessary expertise to use these tools, limiting informed decision-making and potential scientific breakthroughs. Even with the proper domain expertise, building a workflow plan for a WMS by hand can be a time-consuming and cumbersome task. The design of workflows can be supported by systems that can potentially accelerate the process to automate the creation and reproduction of workflows.

3.2.1 Automated Multivariable Workflow Composition: Approach

Our approach to the automated composition of workflows uses a breadth-first search strategy; in other words, we start building a directed acyclic graph in which the expansion of nodes is performed first with siblings instead of child nodes, opposite to how the depth-first search is performed (Russell & Norvig, 1995). In Figure 3, a computational process (p) is represented as a node that needs to be expanded. Scientific variables produced by this computational process enable the exploration of other computational processes that can consume the generated data as part of their inputs. In our scenario, computational processes can produce and consume multiple data elements (e), which we also refer to as variables in the rest of the manuscript.

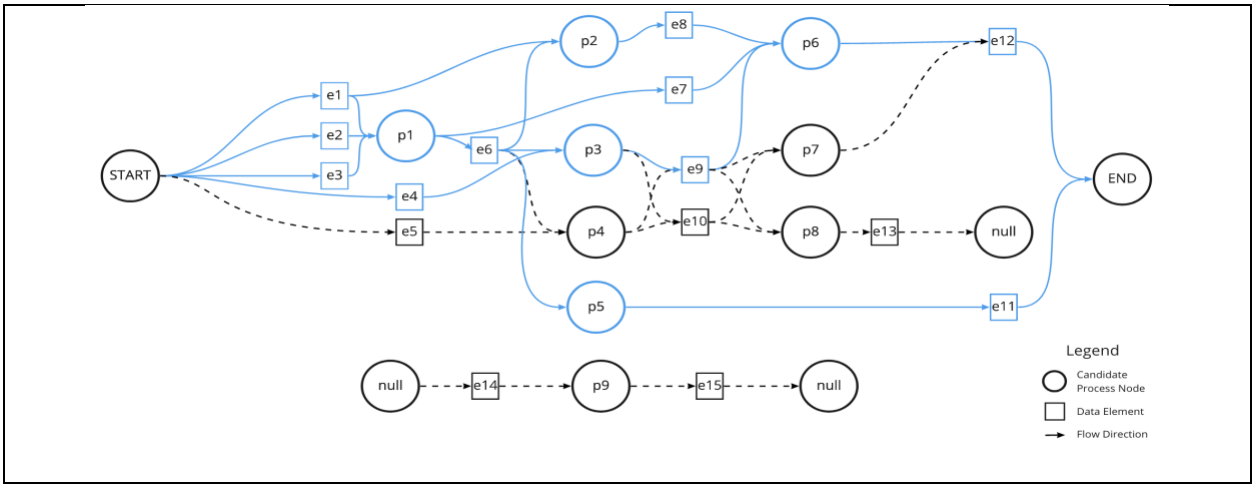

Figure 3 DFD of a Multivariable Workflow Example. The Dashed Path Passes Through Candidate Nodes, the Blue Solid Path is Selected for the Workflow. © 2022 IEEE

Table 2 shows the core algorithm used by the workflow composition. This algorithm can automatically compose a workflow of processes that consume and produce multiple data elements. The algorithm composes a scientific workflow by iteratively identifying candidate processes that can be executed based on the initially available data elements (line 10). For each candidate process, data variables generated from each process are added to a set of collected data variables and a record of the iteration (i.e., step) and the process that produced it (lines 14 – 20). The algorithm continues to iterate until all desired data elements (i.e., target variables) are collected, or there are no processes available to explore (lines 11 – 13). Multiple paths can lead to the generation of the target variables; however, only one path is needed. Figure 3 shows a DFD that represents a multivariable workflow example. If a process does not contribute to the final solution (e.g., the process identified as "p8" in Figure 3), it is pruned (lines 24 – 34). Processes that cannot be executed due to a lack of data elements (e.g., the process identified as "p9") are not included in the analysis.

Table 2 Pseudocode for Multivariable Workflow Composition © 2022 IEEE

1:	*targetVariable = desired variable provided in the request*
2:	*collectedVariable = available variable provided in the request*
3:	*iterationNumber = 1*
4:	*workflowMap* = new Map(Integer, Process set)
5:	**for** each *variable* ∈ *collectedVariable* **do**
6:	*variable*.addOrigin("request")
7:	*variable*.addStep(0)
8:	**end for**
9:	**while** !*collectedVariable*.contains(*targetVariable*) **do**
10:	*executableProcesses* = ProcessCatalog.getProcessBy(*collectedVariable*)
11:	**if** *executableProcesses*.isEmpty() && !*targetVariable*.isEmpty() **do**
12:	**return** Exception("Target variable cannot be collected with available processes and initial variables")
13:	**end if**
14:	**for** each *process* ∈ *executableProcesses* **do**
15:	**for** each *variable* ∈ *process*.getOutputs() **do**
16:	*variable*. addOrigin(*process*)
17:	*variable*. addStep(*iterationNumber*)
18:	*collectedVariable*.add(*variable*)
19:	**end for**
20:	**end for**
21:	*iterationNumber++*
22:	**end while**
23:	*variableToBeCollected = desired variable provided in the request*
24:	**for** each *variable* ∈ *variableToBeCollected* **do**
25:	**if** *workflowMap*.contains(*variable*.getStep()) **do**
26:	*processSet* = *workflowMap*.get(*variable*. getStep())

27:	**end if**
28:	*process* = *variable*.getOrigin()
29:	**for** each *variableInput* ∈ *process*.getInputs() **do**
30:	*variableToBeColleted*.add(*variableInput*)
31:	**end for**
32:	*processSet*.add(*process*)
33:	*workflowMap*.put(*variable*.getStep(), *processSet*)
34:	**end for**

In this example (Figure 3), the target data elements are labeled as e11 and e12, and user-provided data elements are e1, e2, e3, e4, and e5. The target data elements and available data elements form part of a user request. In addition, to the user request, metadata for available processes is also required as input. All processes need to be previously described in the process catalog. The process catalog includes the data elements consumed by a process and the data elements produced. The process catalog used in our example contains nine processes depicted in Figure 3; the blue path shows a candidate workflow path for generating the target variables.

The multivariable workflow composition algorithm generated the blue path by first analyzing all desired and available initial data variables (lines 1 and 2 in Table 2) for the creation of a target and collected data sets, a record of the origins of the initial data variables is created in lines 5 – 8. Process nodes are collected in line 10, resulting in p1 being selected. Data variables generated by this process are added to the collected data set, and the record of data origins is updated. This iteration is repeated, and now line 10 will expand processes p2, p3, p4, and p5; data variables generated by these processes are added to the collected data set, and the record of data origin is updated. In the third iteration, the processes p6, p7, and p8 are retrieved from the model catalog by line 10, extracting all target variables. In the fourth iteration, line 11 will identify that the target variable set is empty and stop iterating. Table 3 shows the possible paths for this abstract example. Each “step” can include a set of processes that can be executed in parallel.

Table 3 Possible Workflow Paths © 2022 IEEE

Path	Step 1	Step 2	Step 3
1	p1	p2, p3, p5	p6
2	p1	p2, p4, p5	p6
3	p1	p3, p5	p7
4	p1	p4, p5	p7

At this stage, the process graph will contain multiple paths for generating target variables and a process that doesn't contribute to the final output. Lines 24 – 34 will then use the record of data origin to backtrack at what iteration the data was generated and what process generated it. The cycle will continue for every data collected, given the inputs required by the processes.

This algorithm, as described before, will enable the creation of a candidate path for generating the scientific variables. The implementation of this algorithm as a stand-alone web service is described in 4.1 Automatic Workflow Composition.

3.3 Ontology-Driven Scientific Variable Reconciliation

Modeling environment behavior as a scientific model (i.e., scientific component) is most of the time performed by domain experts. During this process, domain experts generally develop computational components, to the best of their abilities, that might yield to using non-standard vocabulary for naming the scientific variables produced and consumed by their models. Documentation for computational components and their corresponding variables would ideally be found in the form of software documentation and/or publications to foster the reproducibility of workflows. However, this practice is uncommon among domain experts. It is of interest then, to identify equivalencies between scientific variables among third-party computational components

so they can be integrated, and a scientific workflow composed, this process is described in this work as variable reconciliation.

We hypothesize that ontologies and automated reasoners can allow us to identify equivalencies between scientific variables to automatically orchestrate workflows and support the reuse of scientific models. This would require providing additional information about scientific variables including their type (e.g., category) and any scientific assumption used when describing and using these variables. A category provides more information about the type of data the variable will hold, e.g., flow of water or temperature. In this work, we refer to the term scientific assumption, as any decision or assumption made when generating a value for a scientific variable, whether it was observed, collected, or generated by a scientific model.

3.3.1 Variable Reconciliation: Approach

Supporting the automation of variable reconciliation relies on the annotations of data (i.e., formally described). Scientific variable annotations can be formally described in any computer-readable format. Our approach makes use of descriptions using the JavaScript Object Notation (JSON) format (Internet Engineering Task Force, 2017). An example of a variable annotation is presented in Table 4. A unique identifier, "id", is used to identify the variable and to relate this variable with other variables for which an equivalency is found. The "type" attribute allows categorization of the variable, in the example shown the variable is classified as water flow. Scientific assumptions are represented within the "assumptions" field, this field contains key-value pairs. In our example, a variable that represents the real flow of water (i.e., not natural flow) in Fort Quitman is depicted.

Table 4 Variable Annotation Example

```
{
        "id": "data5",
        "type": "WaterFlow",
        "_comment":"Fort Quitman",
        "assumptions": {
           "isNaturalFlow": "false",
           "hasLatitude": "31.06433",
           "hasLongitude": "-105.59508"
        }
     }
```

This description is then formally described by making use of a custom-made ontology. The Variable Reconciliation Ontology (VaR-O[2]) extends the OBOE (Madin et al., 2007) and ELSEWeb (Del Rio et al., 2013; Villanueva-Rosales et al., 2015) ontologies, details on the extended ontology are found in section 4.2 Variable Reconciliation. Our ontology is populated using the algorithm depicted in Table 5. This approach processes the JSON structured description of variables to create individuals for every variable, assumptions are described as data properties, and an automatic reasoner is used to identify equivalent variables by first identifying the superclass of the assumptions and finally creating a description logic query (DL Query (Baader et al., 2007)) using the superclasses to identify equivalent variables (i.e., individuals in the ontology context). The implementation of our approach is described in section 4.2 Variable Reconciliation.

Table 5 Pseudocode for Populating the VaR-O with Scientific Variables

1:	*targetVariable = desired variable provided in the request*
2:	*collectedVariable = available variable provided in the request*
3:	*componentCatalog = component catalog provided in the request*
4:	*equivalentVariables* = new Collection()
5:	createIndividualsAndDataProperties(*targetVariable)*
6:	createIndividualsAndDataProperties(*collectedVariable)*
7:	**for** each *component* ∈ *componentCatalog* **do**
8:	createIndividualsAndDataProperties(*component*.getInputsAndOutputs())
9:	**end for**
10:	*equivalentVariables* = *runReasoner*.getEquivalentIndividuals(*targetVariable*, *collectedVariable*, *componentCatalog*)

[2] https://purl.org/variablereconciliationontology

3.4 TARGET VARIABLES: A TIE-BREAKING STRATEGY

In our described approach for composing multivariable workflows, it might come to a case in which two or more computational components are selected to be analyzed given that there is enough information to execute them. In situations like this, a tie-breaking strategy is needed to decide which component to analyze first in the hopes of discovering the target variables. Common tie-breaking strategies consider intrinsic characteristics of the components, for example, running time, resources needed, availability, performance, and precision, among others. Our strategy consists of analyzing the number of target variables produced by a specific component, this tactic reduces the information needed for composing a multivariable workflow (e.g., running time, resources needed) and it is useful for case studies in which other information is not available.

In this subsection, we delve into our tie-breaking strategy by composing a heuristic function that considers target variables only.

3.4.1 Tie-Breaking Strategy: Approach

The tie-breaking strategy implemented in our approach (i.e. the heuristic adopted) requires the set of nodes to be analyzed, the set of target variables, and the set of available variables (i.e. variables provided by the user or outputs provided by pre-analyzed components). Our strategy improves our search algorithm, described in Table 6 in color blue, by selecting a component using an informed decision. Details of our strategy are described in Table 7 and Table 8, it analyzes every component and selects the one that produces more target variables. In other words, the number of variables is used as a heuristic to guide our search. In the case in which a tie is produced, we run our algorithm once more to expand our graph one more level with the nodes to break the tie (Table 8). If the tie still exists, the graph is expanded to a second level (Line 18 of the same table). If the tie is not broken with this second expansion, then the first component in the initial set

(without expansion) is selected as the best candidate to be analyzed (Line 3 of the same table).

Table 6 Pseudocode Modification for Multivariable Workflow Composition © 2022 IEEE

```
1:  targetVariable = desired variable provided in the request
2:  collectedVariable = available variable provided in the request
3:  iterationNumber = 1
4:  workflowMap = new Map(Integer, Process set)
5:  for each variable ∈ collectedVariable do
6:      variable.addOrigin("request")
7:      variable.addStep(0)
8:  end for
9:  while !collectedVariable.contains(targetVariable) do
10:     executableProcesses = ProcessCatalog.getProcessBy(collectedVariable)
11:     if executableProcesses.isEmpty() && !targetVariable.isEmpty() do
12:         return Exception("Target variable cannot be collected with available processes and initial variables")
13:     end if
14:     for each process ∈ getNextProcess(executableProcesses, targetVariable, collectedVariable) do
15:         for each variable ∈ process.getOutputs() do
16:             variable. addOrigin(process)
17:             variable. addStep(iterationNumber)
18:             collectedVariable.add(variable)
19:         end for
20:         executableProcesses.remove(process)
21:     end for
22:     iterationNumber++
23: end while
24: variableToBeCollected = desired variable provided in the request
25: for each variable ∈ variableToBeCollected do
26:     if workflowMap.contains(variable.getStep()) do
27:         processSet = workflowMap.get(variable. getStep())
28:     end if
29:     process = variable.getOrigin()
30:     for each variableInput ∈ process.getInputs() do
31:         variableToBeColleted.add(variableInput)
32:     end for
33:     processSet.add(process)
34:     workflowMap.put(variable.getStep(), processSet)
35: end for
```

Table 7 Pseudocode for Tie-Breaking Strategy Part I

```
1:  Method: getNextProcess(executableProcesses, targetVariable, collectedVariable)
2:  stepsToAnalyze = 2
3:  return getMaxTargetVariables(stepsToAnalyze, executableProcesses, targetVariable, collectedVariable)
```

Table 8 Pseudocode for Tie-Breaking Strategy Part II

```
1:  Method: getMaxTargetVariables(stepsToAnalyze, executableProcesses, targetVariable, collectedVariable)
2:  if stepsToAnalyze == 0
3:      return executableProcess.getFirstProcess()
4:  end if
5:  maxVariablesProduced = 0
6:  candidateProcess = null
7:  for each process ∈ executableProcesses do
8:      if process.getVariablesProduced(targetVariables) > maxVariablesProduced
```

9:	*maxVariablesProduced* = *process*.getVariablesProduced(*targetVariables*)
10:	*candidateProcess* = *process*
11:	**end if**
12:	**end for**
13:	**if** *maxVariablesProduced* != *0* and *candidateProcess* != *null*
14:	**return** *process*
15:	**end if**
16:	*collectedVariable*.add(*executableProcesses*.getOutputAllProcesses())
17:	*executableProcesses* = ProcessCatalog.getProcessBy(*collectedVariable*)
18:	return *executableProcesses*.getPreProcess(getMaxTargetVariables(*stepsToAnalyze* – 1, *executableProcesses*, *targetVariable*, *collectedVariable*)

Figure 4 illustrates our tie-breaking strategy. In this example, the variables "e11" and "e12" are to be generated (i.e., target variables). The algorithm shown in Table 8 gets executed when more than one component is to be analyzed (i.e., the components p2, p3, p4, and p5) and selects the component "p5" to be analyzed next as this component produces one target variable. After analyzing the "p5" component it gets removed from the components to be analyzed. Given that the target variable set is not empty another iteration is executed, this is indicated in line 9 of Table 6. Our tie-breaking strategy will analyze the next set of possible components, given that a tie still exists, it will run another iteration but now expanding the set of components. In this iteration, it is shown that "p6" produces one target variable and it will provide information to break the tie, therefore the components "p2" and "p3" are selected.

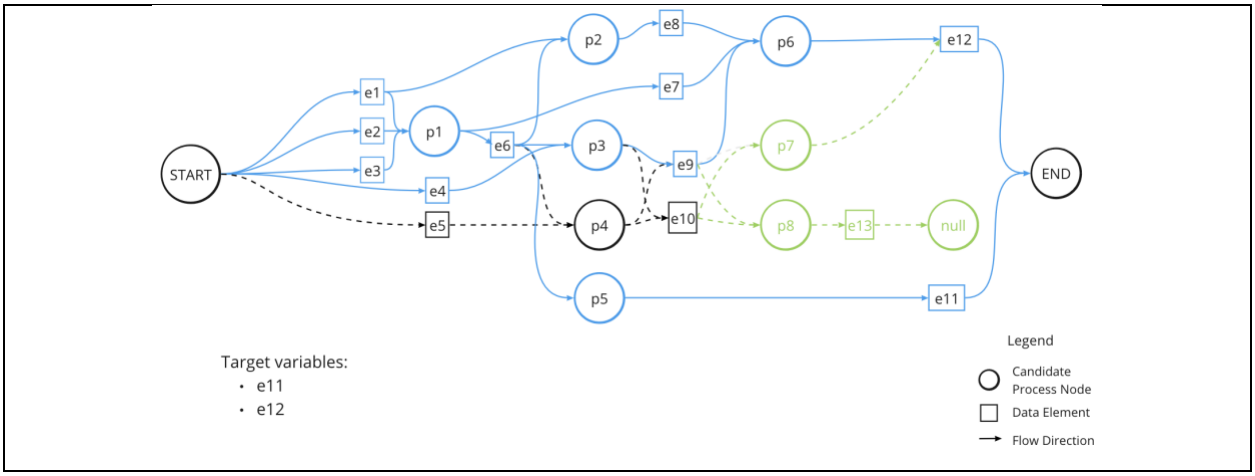

Figure 4 DFD of a Multivariable Workflow Example After Implementing our Tie-Breaking Strategy. Green Nodes are not Discovered Given the Heuristic Used.

Chapter 4: Results

In this chapter, we describe our implementations and results obtained with respect to our initial efforts to automatically compose a scientific workflow, our approach for variable reconciliation, and the tie-breaking strategy.

4.1 Automatic Workflow Composition

We first describe the case study used to evaluate our approach. This case study involves the creation of a scientific workflow that can be incorporated into the SWIM platform, a framework for enabling open access to scientific models to diverse stakeholders (Garnica Chavira et al., 2022). This section was originally published in (Vargas-Acosta et al., 2022) © 2022 IEEE.

4.1.1 The Workflow Composer Service

The Workflow Composer service receives an abstract model catalog and the workflow request as input. These two artifacts include platform-independent metadata that identifies models and data elements with general unique identifiers. The abstract model catalog contains the metadata of available modeling and transformation services (i.e. computational components). The abstract workflow request contains the user-defined payload.

The abstract model catalog contains metadata for the computational components, for example, URL if implemented as a service, required input, and output. An example of these components is scientific models or data transformation jobs. All the models and transformation jobs can be wrapped with a web service interface. Transformation services enable the serialization of data values in the format required by the following scientific model to be executed. Data changes might include units (e.g., Metric to English) or data structures (e.g., different schemas).

The workflow request contains desired output variables as well as available input variables. The available input variables will be used by the composer to search components to be executed with these available variables (i.e. available data). The request is serialized as JSON.

The resulting product of the workflow composer is a workflow plan serialized in JSON format. The serialized workflow plan contains metadata for the execution of every model as well as its prerequisites (i.e., models or transformation services that need to be executed beforehand). The workflow plan is sent as input to the Workflow CWL service.

The Workflow Composition implementation as a microservice is available online on GitHub[3] and as a docker image on the DockerHub registry[4].

4.1.2 The Workflow CWL Service

Our implementation for executing workflows is done as a web service and leverages the third-party CWL Python API for creating CWL workflows (Amstutz et al., 2016). A CWL serialization can be used to execute workflows in a WMS that uses this same standard (e.g., Pegasus). The CWL API is used to transform a workflow plan serialized as JSON into a CWL workflow and use the workflow management capabilities of the CWL tool. Implementing the workflow CWL as a web service and the components as web services enables the use of the *curl* command to send and receive messages with the HTTP protocol.

The Workflow CWL implementation as a microservice is available online on GitHub[5] and as a docker image on DockerHub[6].

This proposed algorithm and its implementation manage to explore a components catalog (e.g. a model catalog) to create a scientific workflow. However, two challenges are found: a

[3] https://github.com/iLink-CyberShARE/workflow-composer-public
[4] https://hub.docker.com/r/lagarnicachavira/workflow-composer-public
[5] https://github.com/iLink-CyberShARE/workflow-cwl-public
[6] https://hub.docker.com/r/lagarnicachavira/workflow-cwl-public

preprocessing step is required to identify equivalencies between scientific variables (i.e. to identify if a scientific variable can be used as input in another component), and a selection criterion is needed to decide which component to analyze first if two or more can be executed with available input.

4.1.3 Case Study

Overview

Our case study aims to support scientists and policy-makers in exploring different scenarios and the effects of short-term management strategies projected into the future. In particular, answering the question: *How does regional reservoir storage behave in an economically optimal water use scenario?*

This case study requires the integration of two heterogeneous models available in SWIM, namely the Water Balance Model (WBM) (R. Holmes, 2021) and the Hydroeconomic Model (HEM) (Ward et al., 2019b). The coverage area for both models is bounded to the Middle Rio Grande in the Paso del Norte region, which includes the south of New Mexico (NM), West Texas in the US, and the north of Chihuahua in Mexico. The WBM is a regional water supply simulation model driven by: upstream inputs to Elephant Butte Reservoir in NM, local climate, regional water demand, and existing reservoir operation rules. The HEM is an economic optimization model that maximizes profits from regional water use. Both models can take multiple inputs and generate output values for multiple variables.

Provided below is a description of the inputs used to generate this workflow and the outputs generated, which are also depicted in Figure 5.

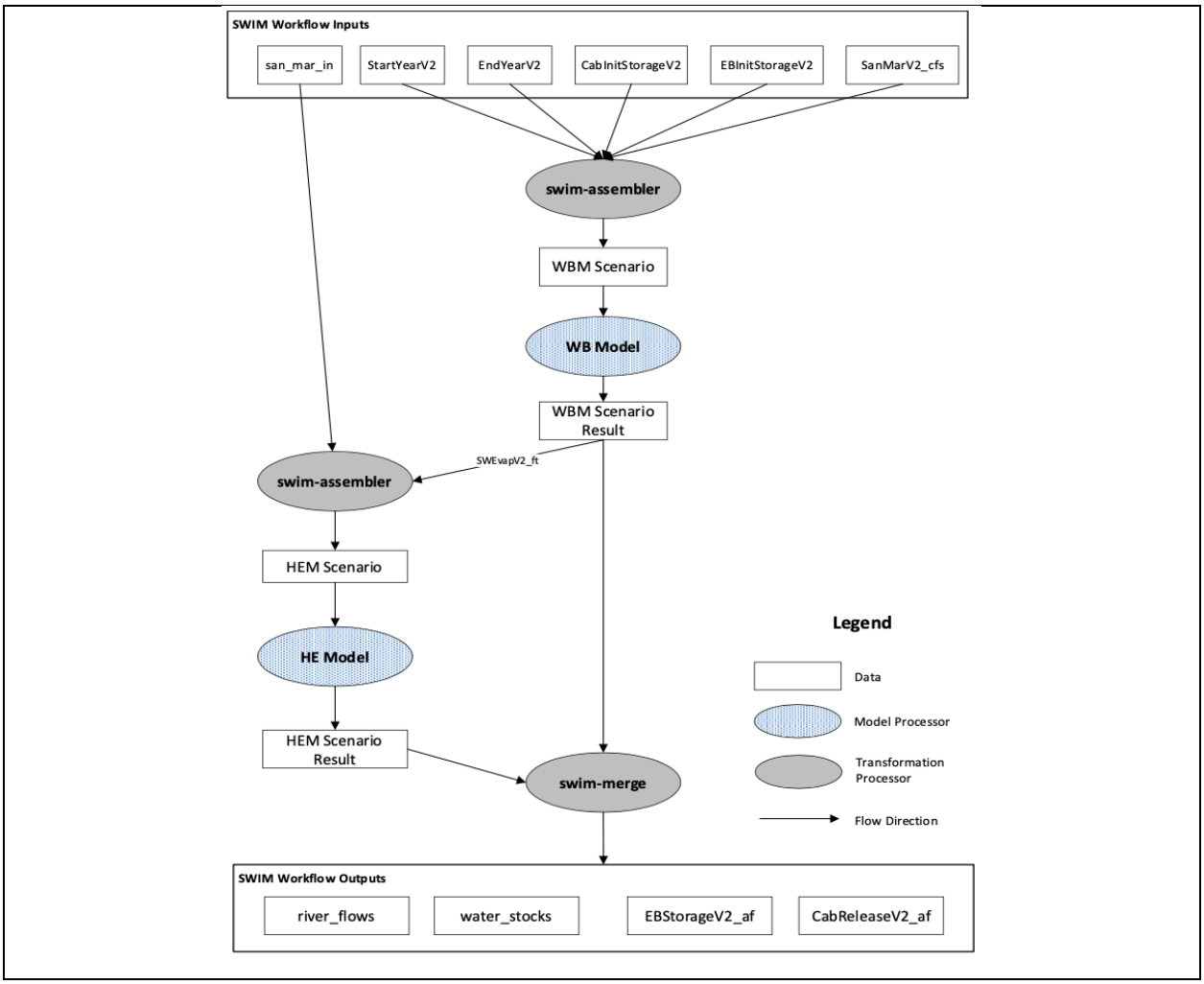

Figure 5 DAG of a SWIM Model-to-Model Integration Scenario Derived from the Case Study. © 2022 IEEE

4.1.4 SWIM Workflow Input

In this section, we describe the content of the workflow request payload used in our case study. The payload is divided into three blocks: inputs, outputs, and rules.

Table 9 SWIM Workflow Input Excerpt.

```
{ "inputs" : [
{"paramName" : "StartYearV2",
"paramValue" : 1994 }],
"outputs" : [
{"varName" : "EBStorageV2_af " },
{ "varName" : "water_stocks" } ],
"rules" : [
{ "excludeDefault" : ["evap_rat_p"] },
{ "equivalence" : ["evap_rat_p", "SWEvapV2_ft"]}]
}
```

Table 9 shows an excerpt of the SWIM workflow request payload. The first block of the input payload includes data inputs with custom values specified by the user. For each input entry, the “paramName” field carries a unique identifier for the data element. The “paramValue” field is a user-defined value for the parameter. The scenario provided applies a numeric value of 1994 to the input parameter with the identifier “StartYearV2”. The remaining inputs for this scenario are depicted in Figure 5. The workflow composer implementation supports numeric, table, and time-series data serialized in JSON. As the “paramValue” field is not bounded to a specific data type, we can potentially use the field to reference more complex data input types (e.g., Tiff, Geo-JSON, NetCDF). The outputs block specifies the target output data that will be obtained after the workflow execution. The “varName” field is a unique identifier for each output. Both input and output blocks will be extended with additional metadata that could enable data transformations such as unit conversions, resolutions, or time-series timesteps.

Finally, in the rules block (Table 9), the first rule disables the use of the default value for the variable parameter “evap_rat_p”. The second rule marks the variables “evap_rat_p” and “SWEvapV2_ft” as semantically equivalent.

4.1.5 SWIM Workflow Output

This section describes the content of the SWIM workflow output of the case study scenario. The workflow payload is divided into three blocks: metadata, provenance, and resource. Table 10 shows an excerpt of the SWIM workflow output.

Table 10 SWIM Workflow Output Excerpt.


```
{ "metadata": {
      "status": "success",
      "type": "Workflow Result" },
   "provenance": [ {
      "entity": "Model Output",
      "generatedAtTime": "2022.05.25.14.06.25",
      "id": "EBStorageV2_af",
      "wasGeneratedBy": "1fb918b3-bf35-40f3-9821-b4d907cd610f" }],
   "resource": [
            {
               "modelID": "7b7ac93638f711ec8d3d0242",
               "varName": " EBStorageV2_af",
               "varValue": "...",
               "varinfo": [
                  {
                   "lang": "en-us",
                   "varCategory": " Storage",
                   "varDescription": "Elephant Butte reservoir storage...",
                   "varLabel": "Elephant Butte Reservoir Storage",
                   "varUnit": "Acre-Feet"
                  },
                  {
                   "lang": "es-mx",
                   "varCategory": "Almacenamiento",
                   "varDescription": "Promedio anual en el volumen...",
                   "varLabel": " Almacenamiento en Presa del Elefante Butte",
                   "varUnit": “Acre-Pies"
                  }]
      }]
}
```


The first block of the output payload contains general metadata regarding the execution of the overall workflow; the excerpt in Table 10 includes the execution status and the type of artifact.

The provenance block shows a trace of where the workflow entities were generated. For example, the entity “Model Output” with id “CabReleaseV2_af” was generated by another entity with id “1fb918b3-bf35-40f3-9821-b4d907cd610f”. We anticipate being able to trace back to the

metadata of the modeling service corresponding to an identifier using an RDF graph for representing the provenance.

Finally, the resource block contains all the target data elements requested by the user (for simplicity, the excerpt contains only one output). The data elements include metadata in terms of the SWIM data model schema (Garnica Chavira et al., 2018), and generated data values on the "varValue" field.

4.1.6 Automatic Workflow Composition: Results

Testing the validity of integrating models is more dependent on the compatibility of the models, their inputs, and integration methods than on the usability of a workflow management tool. This task should consider how scientists and decision-makers can interpret the results from such complex data and model integrations. This subsection presents our initial efforts for evaluating model-to-model integration. In our case study, the reservoir evaporation rate, an output of the WBM, is an input to the HEM. Using SWIM's infrastructure, annual reservoir evaporation rates generated from the WBM are generated and used as input to the HEM. Results indicate that the output reservoir evaporation rate from the WBM was consumed as an input to the HEM. Figure 6 and Figure 7 show bar plots retrieved from the SWIM interface. In Figure 6, we can visualize the Surface Water Evaporation Depth values as an output of the WBM. These values were sent to the HEM as Reservoir Evaporation Rate values, as shown in Figure 7.

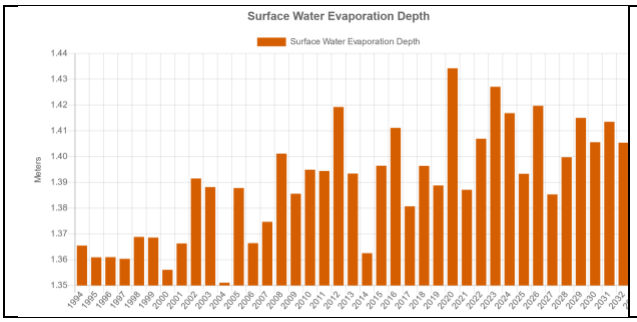

Figure 6 Surface Water Evaporation Depth Output of the WBM © 2022 IEEE.

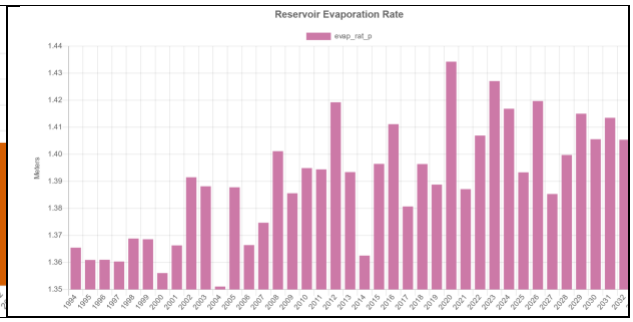

Figure 7 Reservoir Evaporation Rate Input to the HEM © 2022 IEEE.

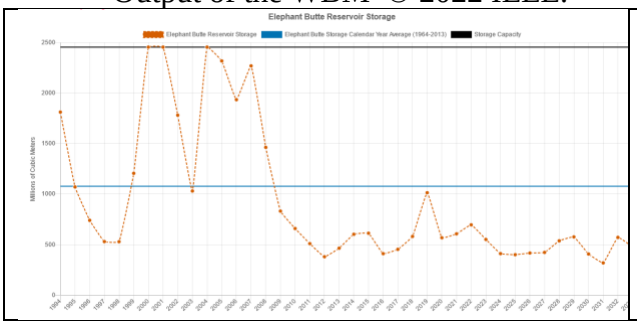

Figure 8 Elephant Butte Reservoir Storage Projected by the WBM © 2022 IEEE.

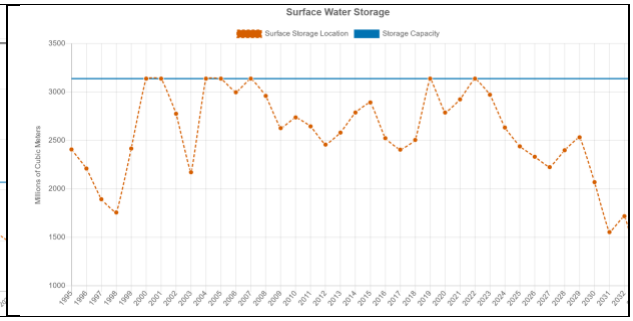

Figure 9 Surface Water Storage Projected by HEM. This Output is a Sum of the Two Regional Reservoirs, Elephant Butte and El Caballo © 2022 IEEE.

Inflows to the Elephant Butte Reservoir produced by the WBM and HEM are shown in Figure 8 and Figure 9 respectively obtained using SWIM in collaboration with M.S. Luis Garnica Chavira.

4.2 Variable Reconciliation

4.2.1 Ontology

To outline our approach for variable reconciliation, we describe an example with two different types of assumptions. We make use of a specific-valued assumption (e.g., a true/false value) to identify equivalent variables. In our example, we identify those variables that represent the natural flow of water (i.e., the flow of water without considering man-made modifications) by having the assumption *isNaturalFlow* set to true. This use-case allows us to depict how to describe

specific-valued assumptions allowing for their variables to be reconciliated based on a comparison. Our second assumption type makes use of continuous values, with this example, we aim to identify those variables whose assumption is within a specific range. In other words, two or more variables will be considered equivalent if their value for a specific assumption is within a specified range.

Scientific variables are represented within an ontology of our design as individuals. We extend the ontologies OBOE and ELSEWeb to describe variables. OBOE contains an extensive list of properties (i.e., the class Characteristic) an entity might have; in our context, a property is equivalent to a variable. ELSEWeb on the other hand is used to describe geographical regions either by specifying them by name or by latitude and longitude. Additional classes may be added to represent specific entities or processes in the domain. For example, in our real-world case study, we added *RealFlow*. Restrictions that apply to these classes according to the specific domain are represented as equivalent classes restrictions. This will allow us to classify scientific variables using reasoners. An example is illustrated in Table 11 using Manchester syntax (Horridge & Patel-Schneider, 2012), the class *RealFlow* is described as equivalent to the class that contains individuals (i.e., scientific variables) that contain a data property *isNaturalFlow* with the value set to *false* or the data property *isRealFlow* set to *true*. Classifying scientific variables using this methodology enables the retrieval of corresponding variables, even if they were described using different terminologies (i.e., *isNaturalFlow* as opposed to *isRealFlow*).

Table 11 Super Class Description Example

RealFlow isEquivalentTo (*isNaturalFlow* value "*false*") or (*isRealFlow* value "*true*")

Presented in Table 12 is an example of an abstract case for illustrating a numeric range, it encapsulates the concept of continuous data values through its data properties. Within this

example, the class *Range* is characterized by a relationship with the data property *abstractRange*, employing a restriction that is formally defined by an equivalence relation.

To illustrate, the OWL class expression describes *Range* as equivalent to *abstractRange* being set to a value in a specific segment, the closed interval from 73.45 to 123.23. Therefore, if an individual entity possessing an *abstractRange* attribute with a value of 73.46 it is systematically classified under the *Range* class, thereby satisfying the ontological constraint. This precision classification underlines the ontology's capacity to handle continuous data and illustrates the granularity of the data property-based class definitions.

Table 12 Super class description example

Range isEquivalentTo *abstractRange* some xsd:double[>= "*73.45*"^^xsd:double , <= "*123.23*"^^xsd:double]

4.2.2 Ontology Population

This process involves importing data from JSON files into the ontology with the help of the OWL Java API[7]. Within this framework, each variable is instantiated as an entity, classified according to its respective type, and its associated scientific assumptions are articulated through designated data properties. Table 13 describes the variable from Table 4 using Manchester Syntax. This is possible by using the OWL API (Horridge & Bechhofer, 2011) and parsing the JSON received in the package. Implementation of the ontology population is performed by a microservice using Java and Maven, the implementation is available online on GitHub[8].

[7] https://mvnrepository.com/artifact/net.sourceforge.owlapi/owlapi-distribution
[8] https://github.com/alex-vargas/workflow-composer-heuristic

Table 13 Example of Scientific Variable Described in our Ontology

```
Individual: data5
   Types: WaterFlow
   Facts:
      hasLatitude  = "31.06433"^^xsd:double
      hasLongitude = "-105.59508"^^xsd:double
      isNaturalFlow = "false"^^xsd:string
```

4.2.3 Automated Reasoning

In the subsequent stage of the ontology population task, we proceed to identify equivalencies among individuals, which, in our context, refer to scientific variables. We make use of Hermit (Shearer et al., 2008) as a reasoner engine to discover these equivalencies, leveraging data properties that represent the scientific assumptions, as the basis for this identification. All equivalent individuals are serialized using JSON. A representation is shown in Table 14. All the variables identified in the *equivalence* field are semantically equivalent given their scientific assumptions and types, therefore, the values of those variables can be used regardless of the model. The equivalent variables are found taking into consideration their unique scientific assumptions, however, if the stakeholder is aware of these constraints and decides to consider two or more variables equivalent regardless of their assumptions, they can specify those requirements in the *ignore* field.

Table 14 Equivalent Individuals Serialized as JSON

```
{
   "rules":[
      {
         "equivalence":[
            "Water-Balance-Model-StreamFlow-Input",
            "data4",
            "e1",
            "data6"
         ]
      }
   ],
   "ignore":[]
}
```

Table 15 shows equivalent individuals when the stakeholder indicates to ignore specific scientific assumptions. The assumption *hasNamedLocation* is ignored in that example and as a result, more variables are indicated as equivalent. We recognize that this action might affect the final workflow generated by our approach, however, this feature enables stakeholders to reuse a specific component in different conditions, for example, to use a component for a different geographic region for which it was initially designed.

Table 15 Equivalent Individuals Ignoring Scientific Assumptions

```
{
   "rules":[
      {
         "equivalence":[
            "Normalization-StreamFlow-Output",
            "Water-Balance-Model-StreamFlow-Input",
            "data4",
            "e1",
            "data6",
            "data5"
         ]
      }
   ],
   "ignore":[
      "hasNamedLocation"
   ]
}
```

This approach allows us to identify equivalent variables that are later digested by our workflow composer described in the previous section for computing a workflow. Stakeholders can decide if specific scientific assumptions should be ignored when computing the equivalency rules.

4.2.4 Variable Reconciliation: Results

The evaluation of our approach includes evaluating the results of the reconciliation process of variables given scientific assumptions, an abstract case study and a real-world case study were used for this purpose. The abstract case study involves the definition of edge cases, i.e. – validating the approach utilizing multiple operating parameters. The design and execution of edge cases provide information to evaluate the performance of our variable reconciliation approach under different executing scenarios. An evaluation of our approach in a real-world scenario is performed following the description of an environmental case study described in (R. Holmes, 2021;

Townsend & Gutzler, 2020). Holmes and Townsend describe scientific assumptions that can be used for outlining the importance of scientific tool description.

Given that this is an abstract case study, all the scientific assumptions, variables, and components are simulated and are not part of a specific real-world case. The definition of these edge cases involves clarifying what is the expected output and what is available data. The module is tested using a black-box approach in which the operations performed by the variable reconciliation component are unknown (Bunge, 1963). After executing all test cases an average running time of 8.121 seconds was obtained.

The real-world case study analyzed focused on further describing the analysis described in our previous work (Vargas-Acosta et al., 2022). Our work describes how the chaining of two scientific components is required to produce a hydro-economic analysis in the southwest region of the United States, more specifically the region bordering Mexico in El Paso, Texas. That analysis does not comprehend the scientific assumptions of how data was generated for the study, therefore, to further describe those assumptions we annotated the work reported in (R. Holmes, 2022; Townsend & Gutzler, 2020).

Data generated by the U.S. Bureau of Reclamation do not account for water diversions (i.e., human-made alterations). This characteristic needs to be accounted for and preprocessed before being used in a scientific component that requires measurements that represent the real flow of water, this can be visualized in Figure 10. Townsend & Gutzler (2020) detail the characteristics of a normalization process performed on this dataset in their work (Townsend & Gutzler, 2020).

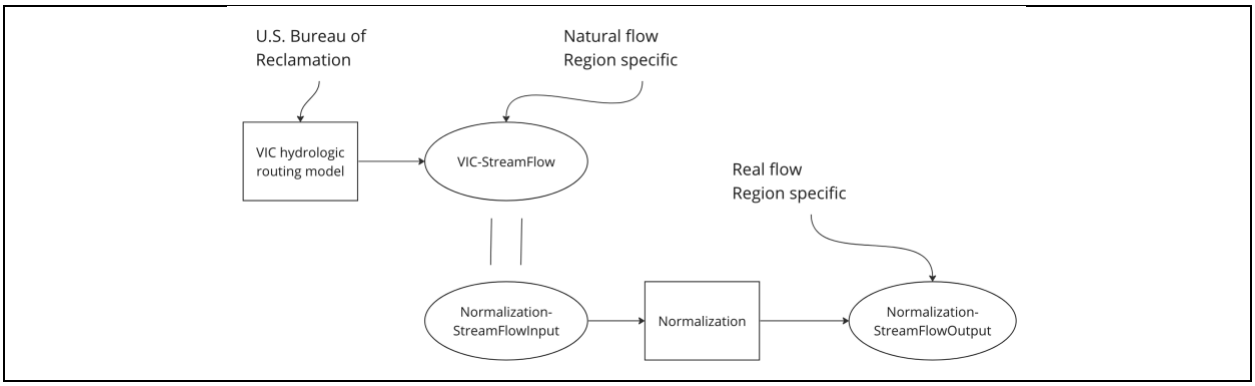

Figure 10 Data Flow Visual Representation

The annotation describing the input of the normalization process described in (Townsend & Gutzler, 2020) as natural flow is shown in Table 16. The *id* depicted is a unique identifier for that specific input enabling to mark two variables as equivalent. Consideration of artificial diversions is located in the attribute as *isNaturalFlow*. Classification of the variable is described in the *type* attribute.

Table 16 Annotation Example as JSON.

```
{
    "id": "a9612349a5c66f49ea20b1d33dafd9c1aa7bf32ad75e39aa8115600a720c2762",
    "type": "WaterStreamflow",
    "assumptions": {
          "isNaturalFlow": "true"
    }
}
```

A secondary abstract case study involves describing situations for continuous data. This ensures that our approach can be used for more complex situations than that of a string comparison (e.g., true/false). An example is shown in Table 17. This example shows a numeric value in which a superclass will use it as a range.

Table 17 Continuous Value Annotation Example as JSON.


```
{
    "id": "97ab7041165def1cb4d8d12f4db6d998852a9daa7e84273818bea178374663e1",
    "type":"abstract",
    "assumptions": {
        "abstractRange": "73.45"
    }
}
```


After annotating the components and their assumptions we tested our reconciliation approach with input shown in Table 18 and Table 19, the expected output is described in Table 20. Table 21 shows the output obtained after running our abstract case study. Our variable reconciliation process successfully generated equivalency rules

Table 18 Available Data for Abstract Case Study in JSON.


```
{
   "inputs": [
      {
         "id": "data1",
         "type": "WaterFlow",
         "assumptions": {
            "isNaturalFlow": "true"
         }
      },
      {
         "id": "data2",
         "type": "WaterFlow",
         "assumptions": {
            "isRealFlow": "false"
         }
      },
      {
         "id": "data3",
         "type": "WaterFlow",
         "assumptions": {
            "isRealFlow": "false"
         }
      },
      {
         "id": "data4",
         "type": "WaterFlow",
         "_comment":"San Marcial",
         "assumptions": {
            "isNaturalFlow": "false",
```

```
            "hasLatitude": "33.68511",
            "hasLongitude": "-106.98214"
         }
      },
      {
         "id": "data5",
         "type": "WaterFlow",
         "_comment":"Fort Quitman",
         "assumptions": {
            "isNaturalFlow": "false",
            "hasLatitude": "31.06433",
            "hasLongitude": "-105.59508"
         }
      },
      {
         "id": "data6",
         "type": "WaterFlow"
      },{
         "id": "data7",
         "type":"abstract",
         "_comment":"San Marcial",
         "assumptions": {
            "abstractRange": "73.45",
            "hasLatitude": "33.68521",
            "hasLongitude": "-106.98314"
         }
      }
   ],
   "outputs": [
      {
         "id": "e3",
         "type": "WaterPrice"
      },
      {
         "id": "e4",
         "type": "AvocadoPrice"
      }
   ]
}
```


Table 19 Component Catalog for Abstract Case Study in JSON.


```
{
   "context": "Context goes here, mainly for using generic vocab as prefix",
   "transformations": [],
   "models": [
      {
         "id": "p1",
         "inputs": [
            {
               "id": "e1p1",
               "type": "WaterFlow",
               "_comment":"San Marcial",
               "assumptions": {
                  "isNaturalFlow": "false",
```

```
                "hasLatitude": "33.68721",
                "hasLongitude": "-106.98254"
            }
        },
        {
            "id": "e2p1",
            "type": "Temperature"
        },
        {
            "id": "e3p1",
            "type": "AtmosphericPressure"
        }
    ],
    "outputs": [
        {
            "id": "e6p1",
            "type": "WaterPrice"
        },
        {
            "id": "e7p1",
            "type": "RunOff"
        }
    ],
    "computationInfo": {
        "method": "POST",
        "contentType": "Content-Type: application/json",
        "url": "http://p1-endpoint-url.com"
    }
},
{
    "id": "p2",
    "inputs": [
        {
            "id": "e1p2",
            "type": "WaterFlow",
            "_comment":"San Marcial",
            "assumptions": {
                "isRealFlow": "true",
                "hasLatitude": "33.68831",
                "hasLongitude": "-106.98644"
            }
        },
        {
            "id": "e2p2",
            "type": "RunOff"
        },{
            "id": "e3p2",
            "type":"abstract",
            "_comment":"San Marcial",
            "assumptions": {
                "abstractRange": "124.22",
                "hasLatitude": "33.68820",
                "hasLongitude": "-106.98013"
            }
        }
    ],
```

```
        "outputs": [
          {
            "id": "e4p2",
            "type": "AvocadoPrice"
          }
        ],
        "computationInfo": {
          "method": "POST",
          "contentType": "Content-Type: application/json",
          "url": "http://p2-endpoint-url.com"
        }
      }
   ]
}
```


Table 20 Expected Output for our Variable Reconciliation.


```
{"rules":[{"equivalence":["data4","e1p1","e1p2"]},{"equivalence":["e2p1"]},{"equivalence":["e3p1"]},{"equivalence":["e3","e6p1"]},{"equivalence":["e2p2","e7p1"]},{"equivalence":["data4","e1p1","e1p2"]},{"equivalence":["e2p2","e7p1"]},{"equivalence":["data7","e3p2"]},{"equivalence":["e4","e4p2"]}],"ignore":[]}
```


Table 21 Output Obtained for our Variable Reconciliation.


```
{"rules":[{"equivalence":["data4","e1p1","e1p2"]},{"equivalence":["e2p1"]},{"equivalence":["e3p1"]},{"equivalence":["e3","e6p1"]},{"equivalence":["e2p2","e7p1"]},{"equivalence":["data4","e1p1","e1p2"]},{"equivalence":["e2p2","e7p1"]},{"equivalence":["data7","e3p2"]},{"equivalence":["e4","e4p2"]}],"ignore":[]}
```


4.3 TARGET VARIABLES: A TIE-BREAKING STRATEGY

4.3.1 Tie-Breaking Strategy: Results

Our implementation of the tie-breaking strategy was integrated into the existing workflow composer. This existing microservice was developed using Java and Maven. Detailed inputs, obtained output, and expected output are described in Appendix A – Test cases for workflow composition using our tie-breaking strategy.

The tie-breaking strategy implementation as a microservice is available online on GitHub[9].

[9] https://github.com/alex-vargas/workflow-composer-heuristic

Chapter 5: Evaluation and Discussion

5.1 Automatic Workflow Composition

This section delineates the interpretation of data obtained by our workflow composition approach and lessons learned. Our approach for automated workflow composition was tested with the described case study in Figure 5, we successfully obtained a workflow serialized using the CWL. By serializing our workflow as CWL, it enables the execution of the workflow by a different set of workflow management systems, e.g., Pegasus (Jayawardana et al., 2022), thus fostering portability. The execution of our workflow was performed by the automated workflow enactment system provided by CWL[10] using scientific models and their descriptions (i.e., component catalog) available through the SWIM platform.

Results from the obtained workflow are shown in Figure 8 and Figure 9, corresponding to inflows to the Elephant Butte Reservoir by the WBM and HEM respectively, show that reservoir storage through time was clearly different, with much higher reservoir storage volumes shown in the HEM than in the WBM; as identified by M.S. Luis Garnica and later confirmed by Dr. Deana Pennington. This result is inconsistent from a scientific perspective. A closer analysis of the results indicated differences in the input parameters for starting reservoir storage that highly impacted storage through time. Further comparisons were manually made on the initial conditions of the models, in this case, the reservoir initial conditions. Additional analysis is required to verify water balance compatibility between both models.

[10] https://github.com/common-workflow-language/cwltool

A lesson learned from the initial evaluation is that the results of integrated models need to be validated from the technical and scientific perspectives. In addition, it is important to investigate how users understand and use the results of these complex systems.

5.2 VARIABLE RECONCILIATION

Results show that the algorithm can perform variable reconciliation for the following scientific assumptions: assumptions that rely on the fact that the variable's continuous value is within a range, and those assumptions that rely on a specific variable's value. We hypothesize that assumptions composed of variable's discrete values and time stamp values (i.e., data and time values) will be handled successfully by our approach, as shown by our continuous value case study shown in Table 21.

Data obtained from our case study matches the expected data. For example, in section 4.2.4 Variable Reconciliation: Results we showed that the obtained output from Table 21 matched the expected output, referenced in Table 20. Our implemented approach, discussed in 4.2 Variable Reconciliation, successfully identified equivalent variables based on scientific assumptions.

To validate the scientific assumptions needed to use the output of the VIC hydrologic routing model (Brekke et al., 2014) in the normalization process conducted by Townsend and Gutzler (2020) depicted in Figure 10, we consulted with Dr. Gutzler who is one of the modelers for this normalization process. Dr. Gutzler provided additional information about the normalization process to account for human diversions, which confirms our understanding that the natural flow provided by the VIC hydrologic routing model is not the same as the real flow (D. Gutzler, personal communication, March 29, 2024). This inquiry highlights the need for

multidisciplinary efforts where domain experts are key in the generation of metadata that can be used for automating processes.

We verified our understanding of the inputs of the Water Balance Model (R. Holmes, 2022), with another domain expert. Dr. Mayer also confirmed our understanding of the difference between natural and real flows used in our variable reconciliation test cases to represent scientific assumptions (A. Mayer, personal communication, April 29, 2024). In addition, Dr. Mayer provided additional information between inputs and outputs from the temperature-based model generated by (R. Holmes, 2022). A graphical description of the models used in the real-world case study in this work is presented in Figure 11. This figure provides additional context than Figure 5. Note that the last step of the workflow, i.e., the HydroEconomic model is described in (Ward et al., 2019a).

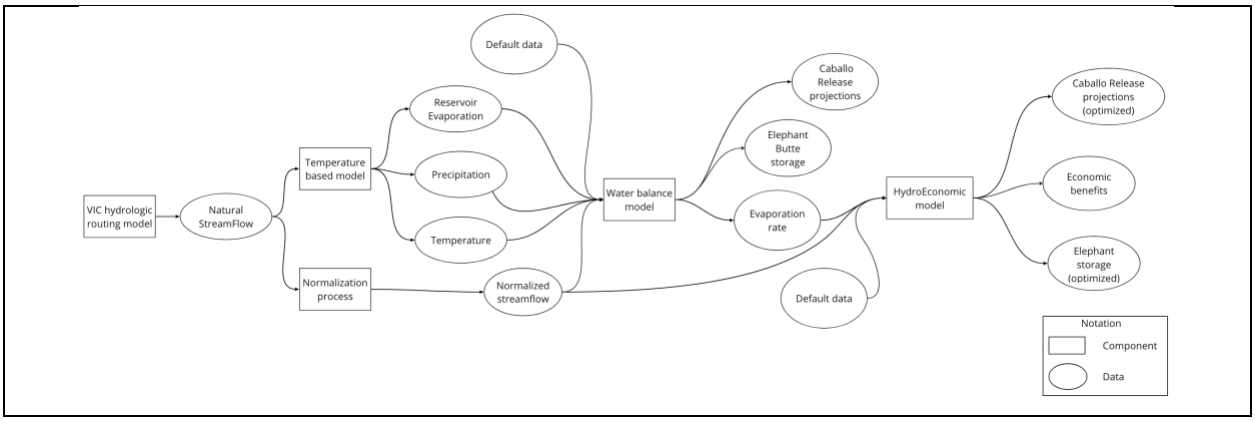

Figure 11 Graphical Depiction of Models Used as Part of the Case Study.

While scientific component annotation, e.g., Table 4, may be time-consuming, it is important to recognize that this effort can have significant implications for the future, particularly in terms of variable identification and reconciliation. Component annotation involves describing components and more importantly variables used as input and output generated by every component. Our validation of the scientific assumptions and workflow composition process used

in the real-world case study shows the importance of involving domain experts in this process. In addition, it is of high importance that every variable is identified using a unique identifier in the entire component catalog. The process of unique identifiers can be generated automatically using a hash function to avoid duplicates.

While our approach to model scientific assumptions uses the VaR-O, we rely on the capabilities of the reasoner to identify equivalent variables. More specifically, we utilize relational operators to identify superclasses. As a result of using this feature, an area of opportunity is to model assumptions that do not require relational operators.

5.3 TIE-BREAKING STRATEGY

In the composition of scientific workflows, particularly those involving the orchestration of components handling multiple scientific variables, the approach we adopt plays a crucial role in determining both efficiency and effectiveness. Our methodology employs a greedy approach for the selection of components, a decision primarily influenced by the inherent uncertainty associated with the availability of required data for the execution of a scientific component. This uncertainty arises from the dynamic nature of scientific data, where not all variables might be readily available for workflow composition. Consequently, the greedy approach that uses the number of target variables obtained, serves as a pragmatic strategy to navigate through this uncertainty, favoring immediate, locally optimal selections that cumulatively aim to converge on a workflow.

However, it is acknowledged that the reliance on a greedy algorithm is not without its potential for refinement. One such avenue for enhancement lies in the incorporation of more comprehensive metadata into our heuristic function. By expanding the metadata considered during component selection, we can potentially increase the accuracy and relevance of the chosen

components in relation to the target variables. This enrichment of metadata could encompass a wider range of attributes related to the scientific components, such as their historical performance metrics, data processing capabilities, or compatibility with other workflow elements. The integration of this additional metadata is visualized to refine our heuristic, allowing for more informed and precise component selection, thereby enhancing the overall robustness and efficiency of the workflow.

On the other hand, the introduction of an additional computational layer for component selection may appear as an overhead in the workflow composition process. This perception originates from the notion that any additional computation, particularly one involving a detailed examination of an extensive component catalog, could potentially increase the workflow composition process. However, we speculate that this is a necessary investment, especially in scenarios characterized by having a large number of components. In such contexts, the time and effort spent in the exploration and selection of components are likely to be less by the subsequent reduction in time required for component exploration. The rationale is that a thorough initial assessment leads to more accurate and relevant component selection, thereby diminishing the need for frequent re-evaluations or adjustments later in the workflow composition lifecycle. Essentially, the upfront computational investment in component selection is recuperated through enhanced workflow efficiency and a reduction in the cumulative time and resources required for workflow composition.

In summary, while our greedy approach to component selection in scientific workflow composition is shaped by the need to manage data uncertainty, it opens pathways for further refinement through enriched metadata integration. Simultaneously, we anticipate that the perceived overhead of this selection process is counterbalanced by the long-term efficiencies

gained in managing extensive component catalogs, underlining the strategic value of this approach in complex scientific workflow environments.

Chapter 6: Conclusions and Future Work

This section contains content originally published in (Vargas-Acosta et al., 2022) © 2022 IEEE.

This dissertation has focused on the significant role of computational workflows in modern scientific research, highlighting their advantages such as automated execution of computational experiments, sharing capabilities, and targeted step execution in scientific workflows. Despite their benefits, we identified the challenges in workflow design and reuse, particularly when manual composition and extensive domain expertise are required. In this dissertation, we review multiple approaches to automatically create workflows. These approaches include sketching out an initial workflow with the help of domain experts, our analysis shows that approaches like Wings (Gil et al., 2010) and SADI (Wilkinson et al., 2011) are typically used for this approach. Another approach delineated in APE (Kasalica & Lamprecht, 2020) considers the automatic creation of workflows by the use of selecting operations. While these considerable efforts are great contributions to this area, our work is complementary as we take a different point of view.

In this dissertation, we focus on scientific assumptions that affect the decision to use a specific computational component. Our research addresses these challenges through the development of an automated workflow planning microservice, the Workflow Composer, integrated into the SWIM infrastructure. This innovative approach facilitates the automatic composition of multivariable workflows and has been demonstrated through a real-world case study within the water sustainability domain.

The application of a breadth-first, uninformed search algorithm in our approach, is employed to explore and prune computational processes not contributing to target variables, thus improving the process of workflow composition. This methodology, coupled with a heuristic

function for an informed selection of computational components, showcases an integration of AI planning in workflow composition. Furthermore, the decoupled design and abstract microservices of our infrastructure present a versatile framework that can be extended to various application domains beyond the initial implementation in the SWIM infrastructure.

We also address the challenge of identifying equivalent scientific variables by considering scientific assumptions. Any design decision made during the creation of computational components might affect the decision to use data generated by them. To delve into this challenge, we explore the use of the Var-O ontology to describe scientific variables. Using an automatic reasoner, we identify variables that can be reused by other computational components, thus enabling the chaining of computational components (i.e., model-to-model integration).

We reflect on our research questions next, in regard to the first research question, *How can metadata and provenance be used to describe scientific assumptions of data consumed by scientific computations for the improvement of automated scientific workflow composition and repurposing of data?* In our approach explained in 3.3 Ontology-Driven Scientific Variable Reconciliation we leverage the potential hidden in metadata and provenance by documenting scientific components, data required to execute them, data generated by them, and scientific assumptions within all components. We identified that by using a formal description to document the mentioned components and assumptions, we were able to identify equivalent scientific variables with the variable reconciliation process. As shown in the results obtained in 4.2 Variable Reconciliation, this is a key step in the automatic workflow composition. However, we recognize that a manual interaction with domain experts is still necessary in order to describe components and variables. We also postulate that this unique interaction will enable the repurposing of components and data

by automatically discovering components if they are described using annotations as proposed in this work.

With respect to the second research question, *To what extent can current Artificial Intelligence (AI) planning techniques with a heuristic function be used to formulate a scientific workflow that considers scientific assumptions evaluated in a scientific domain?*, addressed in this work in section 3.4 Target Variables: A Tie-Breaking Strategy we proposed a strategy for breaking a tie when selecting components in the composition of workflows. Our strategy is useful when information about a component is unavailable, for example, running time, confidence, and resources used, to mention a few. Results obtained from our tests show that using a strategy could reduce the components analyzed, therefore improving the process of workflow composition.

One of the challenges in this work was identifying scientific assumptions in manuscripts or repositories without the guidance of domain experts. To address this challenge, we performed a deeper analysis of the real-world case study to find scientific assumptions suitable for our purpose and consulted domain experts. In Section 4.2 Variable Reconciliation we describe how scientific assumptions in water modeling can be represented to identify variables that are semantically equivalent given their scientific assumptions and generate equivalency rules. In Section 4.1 Automatic Workflow Composition we describe how equivalency rules guide the automated workflow composition. While we were unable to find a real-world case study to connect all the components proposed in this work, i.e., automate the process of generating equivalency rules and use them in the automated workflow composition, each step in the proposed methodology was evaluated separately using abstract case studies when needed.

Looking forward, we identify several key areas for future development. A notable limitation within the current SWIM model-orchestration service pool is the absence of a mechanism for automatic verification of technical and semantic data requirements essential for model-to-model integration. Addressing this gap, it is recommended to incorporate rich metadata annotations and explore the use of semantics and formal requirement descriptions, drawing inspiration from existing literature. This approach aims to overcome the case-specific limitations observed in current annotation processes.

Additionally, the importance of tracing data origins (i.e., provenance) in validating the origins of specific data elements generated by computational processes has been recognized. This need has been recognized by the W3C Provenance working group on the PROV ontology (PROV-O) (Lebo et al., 2013) to annotate provenance data. To enhance this aspect, we propose the creation of an RDF graph using PROV-O for capturing and enriching provenance information and incorporating our efforts into the SWIM workflow composer.

Collecting previously executed workflows can enhance our approach. We propose that having a cache of workflows based on components available, target variables, and available variables might help to reduce the overhead if a workflow is already composed.

An area of opportunity is also identified in the tie-breaking strategy of using the number of nodes. In section 3.4 Target Variables: A Tie-Breaking Strategy we described our approach for selecting a computational component to analyze, we acknowledge an area of opportunity by holding a cache of components already explored. This information can optimize our algorithm and make it more efficient with respect to time and resources.

In addition to cache exploration, we also propose the use of ranking scientific variables to improve our heuristic in our tie-breaking strategy. Ranking of variables can be performed based

on one or multiple attributes, for example, precision or resources used to execute the component that generates data. We anticipate that this approach will require the interaction of domain experts to collect their input in the area of precision and to analyze the running time of multiple computational components for identifying resources or time of execution.

Our efforts toward automating model-to-model integration hold the potential to improve scientific research and decision-making processes. By simplifying the usage of scientific components and reducing the reliance on manual curation of data and tool usage, we aim to enable a broader range of stakeholders to engage more effectively with scientific components. However, we recognize that a misuse or misinterpretation of data by stakeholders might be a dangerous scenario. Constraints can be implemented to minimize the misuse of data by stakeholders.

Computational components descriptions are an important element for automated workflow composition. The Open Modeling Interface (OpenMI) is an effort developed for this purpose. Created as a standard, it creates a wrapper among components (e.g., scientific models) by the use of its Software Development Kit. We propose the further analysis of OpenMI descriptions to be leveraged by our approach.

In conclusion, our work relies on the importance of using knowledge representation languages for modeling scientific assumptions in the composition of scientific workflows. When combined with sophisticated Artificial Intelligence techniques, this approach not only streamlines the workflow composition process but also empowers stakeholders including scientists; this process can assist them in defining and executing complex scientific experiments that require computational models. Policymakers can be assisted in the discovery of scientific information, and moreover in the understanding of scientific information with the use of scientific narratives (Vargas-Acosta et al., 2018). In addition to empowering stakeholders, scientific components and

data become findable, accessible, interoperable, and reusable (FAIR principles) (Wilkinson et al., 2016).

As technology becomes more interconnected, we recognize the value of making data and computational components available to everyone. Parashar recollects in his work the recognition of democratizing science, that is enabling fair access to data and software (Parashar, 2022), with this goal in mind, we aim to contribute to this goal by enabling non-experts with the discovery and generation of scientific information.

While we state that our work can improve the reuse of components by the automatic composition of workflows, we also recognize the need for further investigation in this area with a special interest in tie-breaking strategies, model-to-model verification, and the automatic generation of formal descriptions of computational components.

References

Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., Kartashov, A., Leehr, D., Ménager, H., Nedeljkovich, M., Scales, M., Soiland-Reyes, S., & Stojanovic, L. (2016). *Common Workflow Language, v1.0*. https://doi.org/10.6084/m9.figshare.3115156.v2

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., & Patel-Schneider, P. F. (Eds.). (2007). *The Description Logic Handbook: Theory, Implementation and Applications* (2nd ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511711787

Brekke, L., Wood, A., & Pruitt, T. (2014). *Downscaled CMIP3 and CMIP5 Hydrology Projections: Release of Hydrology Projections, Comparison with Preceding Information, and Summary of User Needs* (p. 110). https://gdo-dcp.ucllnl.org/downscaled_cmip_projections/techmemo/BCSD5HydrologyMemo.pdf

Brickley, D., Guha, R. V., & McBride, B. (2004). RDF vocabulary description language 1.0: RDF Schema. W3C Recommendation (2004). *Http://Www.W3.Org/Tr/2004/Rec-Rdf-Schema, 20040210*.

Buahin, C. A., & Horsburgh, J. S. (2018). Advancing the Open Modeling Interface (OpenMI) for integrated water resources modeling. *Environmental Modelling & Software*, *108*, 133–153. https://doi.org/10.1016/j.envsoft.2018.07.015

Bulatewicz, T., Yang, X., Peterson, J. M., Staggenborg, S., Welch, S. M., & Steward, D. R. (2010). Accessible integration of agriculture, groundwater, and economic models using the Open Modeling Interface (OpenMI): Methodology and initial results. *Hydrology and Earth System Sciences*, *14*(3), 521–534. https://doi.org/10.5194/hess-14-521-2010

Bunge, M. (1963). A General Black Box Theory. *Philosophy of Science*, *30*(4), 346–358.

Carrillo, J., Garijo, D., Crowley, M., Carrillo, R., Gil, Y., & Borda, K. (2019). Semantic Workflows and Machine Learning for the Assessment of Carbon Storage by Urban Trees. *Proceedings of the 3rd. International Workshop on Capturing Scientific Knowledge Co-Located with the 10th. International Conference on Knowledge Capture (K-CAP '19)*, *Vol-2526*, 1–6. http://ceur-ws.org/Vol-2526/

Carvalho, L. A. M. C., Wang, R., Gil, Y., & Garijo, D. (2017). NiW: Converting Notebooks into Workflows to Capture Dataflow and Provenance. *Proceedings of Workshops and Tutorials of the 9th International Conference on Knowledge Capture (K-CAP '17)*, *Vol-2065*, 12–16. http://ceur-ws.org/Vol-2065/paper04.pdf

Crusoe, M. R., Abeln, S., Iosup, A., Amstutz, P., Chilton, J., Tijanić, N., Ménager, H., Soiland-Reyes, S., & Goble, C. (2022). Methods Included: Standardizing Computational Reuse and Portability with the Common Workflow Language. *Communications of the ACM*, *65*(6), 54–63. https://doi.org/10.1145/3486897

Davies, G. S., Dent, T., Tápai, M., Harry, I., McIsaac, C., & Nitz, A. H. (2020). Extending the PyCBC search for gravitational waves from compact binary mergers to a global network. *Physical Review D*, *102*(2), 022004. https://doi.org/10.1103/PhysRevD.102.022004

Deelman, E., Peterka, T., Altintas, I., Carothers, C. D., van Dam, K. K., Moreland, K., Parashar, M., Ramakrishnan, L., Taufer, M., & Vetter, J. (2018). The future of scientific workflows. *The International Journal of High Performance Computing Applications*, *32*(1), 159–175. https://doi.org/10.1177/1094342017704893

Deelman, E., Vahi, K., Rynge, M., Mayani, R., Ferreira da Silva, R., Papadimitriou, G., & Livny, M. (2019a). The Evolution of the Pegasus Workflow Management Software. *Computing in Science & Engineering*, *21*(4), 22–36. https://doi.org/10.1109/MCSE.2019.2919690

Deelman, E., Vahi, K., Rynge, M., Mayani, R., Ferreira da Silva, R., Papadimitriou, G., & Livny, M. (2019b). The Evolution of the Pegasus Workflow Management Software. *Computing in Science & Engineering*, *PP*, 1–1. https://doi.org/10.1109/MCSE.2019.2919690

Del Rio, N., Villanueva-Rosales, N., Pennington, D., Benedict, K., Stewart, A., & Grady, C. J. (2013, November 15). Elseweb meets sadi: Supporting data-to-model integration for biodiversity forecasting. *2013 AAAI Fall Symposium Series*. 2013 Fall Symposium Series, Arlington, VA.

Ferreira da Silva, R., Garijo, D., Peckham, S., Gil, Y., Deelman, E., & Ratnakar, V. (2018). Towards Model Integration via Abductive Workflow Composition and Multi-Method Scalable Model Execution. *International Congress on Environmental Modelling and Software*. https://scholarsarchive.byu.edu/iemssconference/2018/Stream-A/14

Fotopoulos, F., Makropoulos, C., & Mimikou, M. A. (2010). Flood forecasting in transboundary catchments using the Open Modeling Interface. *Environmental Modelling & Software*, *25*(12), 1640–1649. https://doi.org/10.1016/j.envsoft.2010.06.013

Garijo, D., Khider, D., Gil, Y., Carvalho, L., Essawy, B., Pierce, S., Lewis, D., Ratnakar, V., Peckham, S., Duffy, C., & Goodall, J. (2018). A Semantic Model Catalog to Support Comparison and Reuse. *International Congress on Environmental Modelling and Software*. https://scholarsarchive.byu.edu/iemssconference/2018/Stream-A/11

Garnica Chavira, L., Caballero, J., Villanueva-Rosales, N., & Pennington, D. (2018). Semi-structured Knowledge Models and Web Service Driven Integration for Online Execution and Sharing of Water Sustainability Models. *International Congress on Environmental Modelling and Software*. https://scholarsarchive.byu.edu/iemssconference/2018/Stream-A/43

Garnica Chavira, L., Villanueva-Rosales, N., Heyman, J., Pennington, D. D., & Salas, K. (2022). Supporting Regional Water Sustainability Decision-Making through Integrated Modeling. *2022 IEEE International Smart Cities Conference (ISC2)*, 1–7. https://doi.org/10.1109/ISC255366.2022.9922004

Gil, Y., Cobourn, K., Deelman, E., Duffy, C., da Silva, R. F., Kemanian, A., Knoblock, C., Kumar, V., Peckham, S., & Carvalho, L. A. M. C. (2018). MINT: Model integration through knowledge-powered data and process composition. *9th International Congress on Environmental Modelling and Software*, *8*. https://scholarsarchive.byu.edu/iemssconference/2018/Stream-A/13/

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., & Myers, J. (2007a). Examining the challenges of scientific workflows. *Computer*, *40*(12), 24–32. https://doi.org/10.1109/MC.2007.421

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., & Myers, J. (2007b). Examining the challenges of scientific workflows. *Computer*, *40*(12), 24–32. https://doi.org/10.1109/MC.2007.421

Gil, Y., Garijo, D., Khider, D., Knoblock, C. A., Ratnakar, V., Osorio, M., Vargas, H., Pham, M., Pujara, J., Shbita, B., Vu, B., Chiang, Y.-Y., Feldman, D., Lin, Y., Song, H., Kumar, V., Khandelwal, A., Steinbach, M., Tayal, K., … Shu, L. (2021). Artificial Intelligence for Modeling Complex Systems: Taming the Complexity of Expert Models to Improve Decision Making. *ACM Transactions on Interactive Intelligent Systems*, *11*(2), 11:1-11:49. https://doi.org/10.1145/3453172

Gil, Y., Ratnakar, V., Kim, J., Gonzalez-Calero, P., Groth, P., Moody, J., & Deelman, E. (2010). Wings: Intelligent Workflow-Based Design of Computational Experiments. *IEEE Intelligent Systems*, *26*(1), 62–72. https://doi.org/10.1109/MIS.2010.9

Gil, Y., Ratnakar, V., Kim, J., Gonzalez-Calero, P., Groth, P., Moody, J., & Deelman, E. (2011). Wings: Intelligent Workflow-Based Design of Computational Experiments. *IEEE Intelligent Systems*, *26*(1), 62–72. https://doi.org/10.1109/MIS.2010.9

Gupta, S., Szekely, P., Knoblock, C. A., Goel, A., Taheriyan, M., & Muslea, M. (2015). Karma: A System for Mapping Structured Sources into the Semantic Web. In E. Simperl, B. Norton, D. Mladenic, E. Della Valle, I. Fundulaki, A. Passant, & R. Troncy (Eds.), *The Semantic Web: ESWC 2012 Satellite Events* (pp. 430–434). Springer. https://doi.org/10.1007/978-3-662-46641-4_40

Harpham, Q. K., Hughes, A., & Moore, R. V. (2019). Introductory overview: The OpenMI 2.0 standard for integrating numerical models. *Environmental Modelling & Software, 122,* 104549. https://doi.org/10.1016/j.envsoft.2019.104549

Hitzler, P., Krötzsch, M., & Rudolph, S. (2010). *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC.

Holmes, R. (2021). *Water Woes Worsen: Middle Rio Grande Reservoir Modeling Projects Declining Water Availability Under Climate Change Scenarios* [Master's thesis]. Michigan Technological University.

Holmes, R. (2022). *Water Woes Worsen: Middle Rio Grande Reservoir Modeling Projects Declining Water Availability Under Climate Change Scenarios* [Master's thesis]. Michigan Technological University.

Holmes, R. N., Mayer, A., Gutzler, D. S., & Chavira, L. G. (2022). Assessing the Effects of Climate Change on Middle Rio Grande Surface Water Supplies Using a Simple Water Balance Reservoir Model. *Earth Interactions*, *26*(1), 168–179. https://doi.org/10.1175/EI-D-21-0025.1

Horridge, M., & Bechhofer, S. (2011). The OWL API: A Java API for OWL ontologies. *Semantic Web*, *2*(1), 11–21. https://doi.org/10.3233/SW-2011-0025

Horridge, M., & Patel-Schneider, P. (2012). *OWL 2 Web Ontology Language. Manchester Syntax (Second Edition)* [W3C Note]. W3C. http://www.w3.org/TR/owl2-manchester-syntax/

Horrocks, I., Patel-Schneider, P. F., & van Harmelen, F. (2003). From SHIQ and RDF to OWL: The making of a Web Ontology Language. *Journal of Web Semantics*, *1*(1), 7–26. https://doi.org/10.1016/j.websem.2003.07.001

Internet Engineering Task Force. (2017). *The JavaScript Object Notation (JSON) Data Interchange Format* (RFC 8259). https://datatracker.ietf.org/doc/html/rfc8259

Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S. (2018). Microservices: The Journey So Far and Challenges Ahead. *IEEE Software*, *35*(3), 24–35. https://doi.org/10.1109/MS.2018.2141039

Jayawardana, Y., Ashok, V. G., & Jayarathna, S. (2022). StreamingHub: Interactive stream analysis workflows. *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, 1–10. https://doi.org/10.1145/3529372.3530936

Karpatne, A., Jiang, Z., Vatsavai, R. R., Shekhar, S., & Kumar, V. (2016). Monitoring Land-Cover Changes: A Machine-Learning Perspective. *IEEE Geoscience and Remote Sensing Magazine*, *4*(2), 8–21. https://doi.org/10.1109/MGRS.2016.2528038

Kasalica, V., & Lamprecht, A.-L. (2020). Workflow Discovery with Semantic Constraints: The SAT-Based Implementation of APE. *Electronic Communications of the EASST*, *78*(0), Article 0. https://doi.org/10.14279/tuj.eceasst.78.1092

Khattar, R., Hales, R., Ames, D. P., Nelson, E. J., Jones, N. L., & Williams, G. (2021). Tethys App Store: Simplifying deployment of web applications for the international GEOGloWS initiative. *Environmental Modelling & Software*, *146*, 105227. https://doi.org/10.1016/j.envsoft.2021.105227

Kim, J., Gil, Y., & Ratnakar, V. (2006). Semantic Metadata Generation for Large Scientific Workflows. In I. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, & L. M. Aroyo (Eds.), *Proceedings of the 5th International Semantic Web Conference (ISWC '06)* (Vol. 4273, pp. 357–370). Springer. https://doi.org/10.1007/11926078_26

Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., & Zhao, J. (2013). *PROV-O: The PROV Ontology*. World Wide Web Consortium.

Madin, J., Bowers, S., Schildhauer, M., Krivov, S., Pennington, D., & Villa, F. (2007). An ontology for describing and synthesizing ecological observation data. *Ecological Informatics*, *2*(3), 279–296. https://doi.org/10.1016/j.ecoinf.2007.05.004

Maechling, P., Chalupsky, H., Dougherty, M., Deelman, E., Gil, Y., Gullapalli, S., Gupta, V., Kesselman, C., Kim, J., & Mehta, G. (2005). Simplifying construction of complex workflows for non-expert users of the Southern California Earthquake Center Community Modeling Environment. *ACM SIGMOD Record*, *34*(3), 24–30. https://doi.org/10.1145/1084805.1084811

Musen, M. A. (2015). The protégé project: A look back and a look forward. *AI Matters*, *1*(4), 4–12. https://doi.org/10.1145/2757001.2757003

Noy, N., & McGuinness, D. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. *Knowledge Systems Laboratory*, *32*. http://protege.stanford.edu/publications/ontology_development/ontology101.pdf

Oberhauser, R., & Stigler, S. (2017). Microflows: Enabling Agile Business Process Modeling to Orchestrate Semantically-Annotated Microservices. *Proceedings of the Seventh International Symposium on Business Modeling and Software Design (BMSD '17)*, *1: BMSD*, 19–28. https://doi.org/10.5220/0006527100190028

Parashar, M. (2022). Democratizing Science Through Advanced Cyberinfrastructure. *Computer*, *55*(09), 79–84. https://doi.org/10.1109/MC.2022.3174928

Pavlovikj, N., Begcy, K., Behera, S., Campbell, M., Walia, H., & Deogun, J. S. (2014). A Comparison of a Campus Cluster and Open Science Grid Platforms for Protein-Guided Assembly Using Pegasus Workflow Management System. *Proceedings of the 2014 IEEE 28th International Parallel Distributed Processing Symposium Workshops (IPDPSW '14)*, 546–555. https://doi.org/10.1109/IPDPSW.2014.66

Peckham, S. (2014). The CSDMS Standard Names: Cross-Domain Naming Conventions for Describing Process Models, Data Sets and Their Associated Variables. *International Congress on Environmental Modelling and Software*. https://scholarsarchive.byu.edu/iemssconference/2014/Stream-A/12

Reed, S., Schaake, J., & Zhang, Z. (2007). A distributed hydrologic model and threshold frequency-based method for flash flood forecasting at ungauged locations. *Journal of Hydrology*, *337*(3–4), 402–420. https://doi.org/10.1016/j.jhydrol.2007.02.015

Riedel, B., Bauermeister, B., Bryant, L., Conrad, J., Perio, P. de, Gardner, R. W., Grandi, L., Lombardi, F., Rizzo, A., Sartorelli, G., Selvi, M., Shockley, E., Stephen, J., Thapa, S., & Tunnell, C. (2018). Distributed Data and Job Management for the XENON1T Experiment. *Proceedings of the Practice and Experience on Advanced Research Computing (PEARC '18)*, 1–8. https://doi.org/10.1145/3219104.3219155

Russell, S., & Norvig, P. (1995). *Artificial Intelligence: A Modern Approach* (1st Edition). Prentice-Hall, Inc.

Shearer, R. D., Motik, B., & Horrocks, I. (2008). Hermit: A highly-efficient OWL reasoner. *Owled*, *432*, 91. http://www.cs.ox.ac.uk/boris.motik/pubs/smh08HermiT.pdf

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, *5*(2), 51–53. https://doi.org/10.1016/j.websem.2007.03.004

Tarboton, D., Idaszak, R., Horsburgh, J., Heard, J., Ames, D., Goodall, J., Band, L., Merwade, V., Couch, A., Arrigo, J., Hooper, R., Valentine, D., & Maidment, D. (2014, June 16). HydroShare: Advancing Collaboration through Hydrologic Data and Model Sharing. *International Congress on Environmental Modelling and Software*. https://scholarsarchive.byu.edu/iemssconference/2014/Stream-A/7

Townsend, N. T., & Gutzler, D. S. (2020). Adaptation of Climate Model Projections of Streamflow to Account for Upstream Anthropogenic Impairments. *Journal of the American Water Resources Association (JAWRA)*, *56*(4), 586–598. https://doi.org/10.1111/1752-1688.12851

Vahi, K., Wang, M. H., Chang, C., Dodelson, S., Rynge, M., & Deelman, E. (2018). Workflows using Pegasus: Enabling Dark Energy Survey Pipelines. *Proceedings of the 28th Astronomical Data Analysis Software and Systems (ADASS '18), 523*, 689–692.

Vandervalk, B. P., McCarthy, E. L., & Wilkinson, M. D. (2009). SHARE: A Semantic Web Query Engine for Bioinformatics. In A. Gómez-Pérez, Y. Yu, & Y. Ding (Eds.), *Proceedings of the Asian Semantic Web Conference (ASWC '09)* (pp. 367–369). Springer. https://doi.org/10.1007/978-3-642-10871-6_27

Vargas-Acosta, R. A., Garnica Chavira, L., Villanueva-Rosales, N., & Pennington, D. (2018). Towards SWIM Narratives for Sustainable Water Management. In D. Garijo, N. Villanueva-Rosales, T. Kuhn, T. Kauppinen, & M. Dumontier (Eds.), *Proceedings of the Second Workshop on Enabling Open Semantic Science co-located with 17th International Semantic Web Conference, SemSci* (Vol. 2184, pp. 25–33). CEUR-WS.org. http://ceur-ws.org/Vol-2184/paper-03.pdf

Vargas-Acosta, R. A., Garnica Chavira, L., Villanueva-Rosales, N., & Pennington, D. D. (2022). Automating Multivariable Workflow Composition for Model-to-Model Integration. *2022 IEEE 18th International Conference on E-Science (e-Science)*, 159–170. https://doi.org/10.1109/eScience55777.2022.00030. © 2022 IEEE. Reprinted, with permission, from Raul Alejandro Vargas-Acosta, Automating Multivariable Workflow Composition for Model-to-Model Integration, 2022 IEEE 18th International Conference on E-Science (e-Science), October 2022

Villanueva-Rosales, N., del Rio, N., Pennington, D., & Garnica Chavira, L. (2015). Semantic Bridges for Biodiversity Sciences. In M. Arenas, O. Corcho, E. Simperl, M. Strohmaier, M. d'Aquin, K. Srinivas, P. Groth, M. Dumontier, J. Heflin, K. Thirunarayan, & S. Staab

(Eds.), *The Semantic Web—ISWC 2015* (pp. 310–317). Springer International Publishing. https://doi.org/10.1007/978-3-319-25010-6_20

Ward, F. A., Mayer, A. S., Garnica, L. A., Townsend, N. T., & Gutzler, D. S. (2019a). The economics of aquifer protection plans under climate water stress: New insights from hydroeconomic modeling. *Journal of Hydrology*, *576*, 667–684. https://doi.org/10.1016/j.jhydrol.2019.06.081

Ward, F. A., Mayer, A. S., Garnica, L. A., Townsend, N. T., & Gutzler, D. S. (2019b). The economics of aquifer protection plans under climate water stress: New insights from hydroeconomic modeling. *Journal of Hydrology*, *576*, 667–684. https://doi.org/10.1016/j.jhydrol.2019.06.081

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18

Wilkinson, M. D., Vandervalk, B., & McCarthy, L. (2011). The Semantic Automated Discovery and Integration (SADI) Web service Design-Pattern, API and Reference Implementation. *Journal of Biomedical Semantics*, *2*(1), 8. https://doi.org/10.1186/2041-1480-2-8

World Wide Web Consortium. (2012). *OWL 2 Web Ontology Language Document Overview (Second Edition)*. https://www.w3.org/TR/owl2-overview/

Zhang, L., & Malik, S. (2002). The Quest for Efficient Boolean Satisfiability Solvers. In E. Brinksma & K. G. Larsen (Eds.), *Computer Aided Verification* (pp. 17–36). Springer. https://doi.org/10.1007/3-540-45657-0_2

Zia, A., Bomblies, A., Schroth, A. W., Koliba, C., Isles, P. D. F., Tsai, Y., Mohammed, I. N., Bucini, G., Clemins, P. J., Turnbull, S., Rodgers, M., Hamed, A., Beckage, B., Winter, J., Adair, C., Galford, G. L., Rizzo, D., & Houten, J. V. (2016). Coupled impacts of climate and land use change across a river–lake continuum: Insights from an integrated assessment model of Lake Champlain's Missisquoi Basin, 2000–2040. *Environmental Research Letters*, *11*(11), 114026. https://doi.org/10.1088/1748-9326/11/11/114026

Appendix

APPENDIX A – TEST CASES FOR WORKFLOW COMPOSITION USING OUR TIE-BREAKING STRATEGY

Description: The component catalog is empty. Therefore, there are no components to generate the desired output. **Component catalog**: Empty **Request**: Input = e1, e2, e3, e4, e5 Target variables = e11, e12 **Expected output**: There are not enough scientific models to generate the desired output **Obtained output**: There are not enough scientific models to generate the desired output
Description: The target variable set is empty. Therefore, there is no workflow to generate. **Component catalog**: p1 = Inputs{e1, e2, e3}, Outputs{e6, e7} p2 = Inputs{e1, e6}, Outputs{e8} p3 = Inputs{e4, e6}, Outputs{e9, e10} p4 = Inputs{e5, e6}, Outputs{e9, e10} p5 = Inputs{e6}, Outputs{e11} p6 = Inputs{e7, e8, e9}, Outputs{e12} p7 = Inputs{e9, e10}, Outputs{e12} p8 = Inputs{e9, e10}, Outputs{e13} p9 = Inputs{e14}, Outputs{e15} **Request**: Input = e1, e2, e3, e4, e5 Target variables = Empty **Expected output**: Error **Obtained output**: Error
Description: The available variable set is empty. Therefore, as there is no available variable(s) to kick-start a component, the workflow can't be generated. **Component catalog**: p1 = Inputs{e1, e2, e3}, Outputs{e6, e7} p2 = Inputs{e1, e6}, Outputs{e8} p3 = Inputs{e4, e6}, Outputs{e9, e10} p4 = Inputs{e5, e6}, Outputs{e9, e10} p5 = Inputs{e6}, Outputs{e11}

p6 = Inputs{e7, e8, e9}, Outputs{e12}
p7 = Inputs{e9, e10}, Outputs{e12}
p8 = Inputs{e9, e10}, Outputs{e13}
p9 = Inputs{e14}, Outputs{e15}
Request:
Input = Empty
Target variables = e11, e12
Expected output:
Error
Obtained output:
Error

Description:
There are components to generate the target variables, however, input to those components is not provided and can't be generated. Therefore, a workflow can't be composed.
Component catalog:
p1 = Inputs{e1, e2, e3}, Outputs{e6, e7}
p2 = Inputs{e1, e6}, Outputs{e8}
p3 = Inputs{e4, e6}, Outputs{e9, e10}
p4 = Inputs{e5, e6}, Outputs{e9, e10}
p5 = Inputs{e6, e66}, Outputs{e11}
p6 = Inputs{e7, e8, e9, e99}, Outputs{e12}
p7 = Inputs{e9, e10}, Outputs{e12}
p8 = Inputs{e9, e10}, Outputs{e13}
p9 = Inputs{e14}, Outputs{e15}
Request:
Input = e1, e2, e3, e4, e5
Target variables = e11, e12
Expected output:
There are not enough scientific models to generate the desired output
Obtained output:
There are not enough scientific models to generate the desired output

Description:
There are no components to generate all target variables. Therefore, a workflow can't be generated.
Component catalog:
p1 = Inputs{e1, e2, e3}, Outputs{e6, e7}
p2 = Inputs{e1, e6}, Outputs{e8}
p3 = Inputs{e4, e6}, Outputs{e9, e10}
p4 = Inputs{e5, e6}, Outputs{e9, e10}
p5 = Inputs{e6}, Outputs{e11111}
p6 = Inputs{e7, e8, e9}, Outputs{e1222}
p7 = Inputs{e9, e10}, Outputs{e12}
p8 = Inputs{e9, e10}, Outputs{e13}
p9 = Inputs{e14}, Outputs{e15}
Request:
Input = e1, e2, e3, e4, e5

Target variables = e11, e12
Expected output:
There are not enough scientific models to generate the desired output
Obtained output:
There are not enough scientific models to generate the desired output

Description:
A workflow is generated. However, the chosen workflow is not guaranteed to be an optimal one.
Component catalog:
p1 = Inputs{f}, Outputs{a, c}
p2 = Inputs{f}, Outputs{e}
p3 = Inputs{c}, Outputs{d}
p4 = Inputs{e}, Outputs{a, b}
p5 = Inputs{d}, Outputs{b}
p6 = Inputs{e7, e8, e9}, Outputs{e12}
p7 = Inputs{e9, e10}, Outputs{e12}
p8 = Inputs{e9, e10}, Outputs{e13}
p9 = Inputs{e14}, Outputs{e15}
Request:
Input = f
Target variables = a, b
Expected output:
A workflow composed of components that generate the target variables. However, the workflow does not contain the shortest possible path.
Obtained output:
{"2":[{"inputs":["f"],"outputs":["a","c"],"id":"p1","computationInfo":{"method":"POST","contentType":"Content-Type: application/json","url":"http://p1-endpoint-url.com"},"prerequisites":[]},{"inputs":["f"],"outputs":["e"],"id":"p2","computationInfo":{"method":"POST","contentType":"Content-Type: application/json","url":"http://p2-endpoint-url.com"},"prerequisites":[]}],"4":[{"inputs":["e"],"outputs":["a","b"],"id":"p4","computationInfo":{"method":"POST","contentType":"Content-Type: application/json","url":"http://p4-endpoint-url.com"},"prerequisites":["p2"]}]}
Explanation:
The proposed implementation uses a heuristic function to select “p1” as the most promising component to explore first as it produces a target variable “a”. However, that path is a dead end as the other variable “b” is generated two components after it.

Curriculum Vitae

Raul Alejandro Vargas Acosta is a Ph.D. candidate in Computer Science at the University of Texas at El Paso. He received a master's degree in Computer Science from UTEP in 2018. In 2016 he was awarded a master's degree in Open Source Software from the Universidad Autonoma de Chihuahua, Juarez, Mexico. His bachelor's degree was awarded in 2007 from the Universidad Autonoma de Ciudad Juarez, Juarez, Mexico. While as a student at UTEP, he received multiple awards including the 2021 Upsilon Pi Epsilon (UPE) Dan Drew Award; Google's 2020 Generation Scholarship; and 2019 Computing Alliance of Hispanic Serving Institutions scholar; among others.

Alex's research interest lies in leveraging the power of computer science to aid other disciplines in better understanding our environment. Alex's first exposure to interdisciplinary research was conducted in water sustainability, where he aimed to improve the composition of scientific workflows by means of Artificial Intelligence approaches. He enjoys reading discoveries in different sciences, particularly in astrophysics.

His professional experience includes software engineering, server administration, and teaching. He currently teaches Database Systems as a Ph.D. Assistant Instructor at UTEP. He loves to spend his free time with his family and enjoys swimming as a form of continuous exercise.

Contact Information: avargas.rava@gmail.com, alejandrovargas123@hotmail.com