Analytics Morphology and Transformation

Stephen Kaisler
SHK & Associates
skaisler1@comcast.net

J. Alberto Espinosa
American University
alberto@american.edu

William H. Money
The Citadel
wmoney@citadel.edu

Frank Armour
American University
fjarmour@gmail.com

Abstract
Earlier research in Big Data and Analytics led to the development of an analytic class taxonomy which organized analytics by application area. Further research motivated a need for better descriptions and a realization that analytics have many variations based on implementations for specific platforms. The authors realized that transformations of an analytic from one form to another could yield improvements in performance, space utilization, and other attributes. This paper examines analytic forms and the transformations between them with the goal of documenting them in a comprehensive manner. It proposes the development of an Analytics Catalog as a mechanism for documenting analytics classes, their membership, and the transformations among forms.

Keywords: Analytics, Taxonomy, Transformations.

1. Introduction

Kaisler and Cioffi-Revilla (2007) developed and published a taxonomy of analytic classes to organize analytics into categories along several dimensions. Their goal was to describe analytic methods and techniques, identify relevant features and limitations, provide guidance for applicability to scientific and social science problems, and provide some understanding of the morphology, diversity, breadth, and depth of the many analytics available to researchers. Different forms of the analytic would have different performance and scalability metrics. We found very few examples of morphological analysis or transformation of analytics to address these metrics. This paper examines the morphology of analytics and an approach to transforming one structure to another to enhance analytic performance characteristics in different domains.

2. Analytics, Algorithms and Heuristics

An *analytic* is a process of reasoning about a perception of elements of a situation – tangible or intangible - within a particular domain to achieve an understanding and explanation of what occurred; how and why it occurred; whether it might occur again; what decisions it supports; what is a preferred course of action, if any; and what action(s) to take a result. Implicit in this definition is coverage for the five types of analytics as described by Kaisler, Armour, Espinosa, and Money (2013, 2014).

An *algorithm* is a finite sequence of rigorous instructions, which can be used to solve a class of problems via computational methods (adapted from Cormen 2022) yielding an exact result. The "instructions" encompass both quantitative and qualitative means of solving problems. Some algorithms are purely mathematical – without reference to specific domains or applications, and some have textual descriptions with embedded domain knowledge.

A *heuristic* (adapted from Kahnemann and Tversky 1974) is an approach to problem solving or self-discovery that uses a practical method or logic that is not guaranteed to be optimal, perfect, or rational, but is nevertheless sufficient for reaching an immediate, short-term goal or approximation in reasonable, finite amount of time under specific constraints. Many heuristics are represented as sets of rules or logic statements that operate on symbolic information, but they can also operate on quantitative data as well.

The term 'analytic' is often equated with the term 'algorithm' because it is presumed to be a sequence of steps to yield a specific result. But, an 'analytic' is Armor also a set of heuristics in which the sequence of decisions is determined by the data or lack thereof.

In certain communities, an 'analytic' can be a relatively complex application such as for an analysis of call-data records or identifying trading opportunities in a commodities exchange. We use a narrower description in which the result is obtained by the computation of an algorithm or heuristic set given input data and, perhaps, configuration parameters.

Our examination of many analytics over the past years has indicated that an analytic may reside in several different classes. Thus, we do not require mutual exclusivity at his time. Moreover, different forms of analytics could/will result in different implementations being placed in different classes. As this is a Work in Progress, additional information through an extended literature search is being conducted to refine our placement of analytics into the taxonomy.

In Kaisler and Cioffi-Revilla (2007), a taxonomy of analytics was proposed, then revised in Kaisler, et al. (2014). Each class is composed of subclasses, and each subclass encompasses, possibly, many analytics. This plethora of analytics led us to begin to closely examine their morphology (structure) and properties to better understand their applicability to different domains, and how to make transformations between different structures.

2.1 Motivation

Catalogs of Algorithms have been developed and published over the past 60 years. Communications of the ACM (CACM) published one or more algorithms in its early monthly issues, such as an algorithm for matrix multiplication (Boothboyd 1963). Polya (1945) enumerated mathematical methods for solving simple computational problems. Teukolsky, Press and Vetterling (1986) published Numerical Recipes. Abdou Youssef of George Washington University and his students assisted the National Institutes of Science and Technology (NIST) in the creation of a Digital Library of Mathematical Functions (2010). Shoch (2012) enumerated over 100 analytics for computing centralities of graphs. Lengler and Eppler (2012) created a periodic table of more than 100 analytics for data visualization. Bordawekar, Blainey, Apte, and McRoberts (2011) surveyed over 50 different analytics across a variety of business domains. Analytics transformations abound in certain areas of mathematics which serves as evidence that transformations can provide different insights into problem solving. For example, substituting new variables, say y_1, y_2 and so on for powers of a variable x, e.g., x, x^2, x^3 and so on, in an equation, solving for the y_i's, and then resubstituting back into the original equation.

Program transformation is a process that takes a computer program and generates another computer program that is semantically equivalent to the original program. Research has been focused on automatically performing these transformations. Numerous publications have described program transformations, including Standish, Harrison and Kibler (1976) and Bacon, Graham, and Sharp (1993) are examples. Many of these transformations have been embedded in modern compilers at the source language or code generation stage.

We extend this idea to transformations that take an analytic expressed in one structural form and automatically transform it into another structural form while preserving its semantics.

In Kaisler et al. (2014), the term *Analytic Science* was proposed as the study and development of existing and new analytics. Little research has been done into analytics morphology or the transformations between representations to improve their utility. This paper extends our research into this "critical" aspect of analytic science.

2.2 Data Collection Methods

Numerous repositories for papers and technical reports, including IEEE Digital Library, ACM Digital Library, CEUR-WS, AISNet, AAAI, etc. have been systematically examined by the authors to extract information to revise the analytic taxonomy and begin a detailed study of analytics morphology.

A taxonomy classifies objects of interest according to dimensions by characterizing and discriminating between them. Our revision process adapts some method from Nickerson, Varshney, and Muntermann (2013).

Data on analytics collected from the literature challenged us to rethink the 2014 taxonomy and our approach to describing analytics.

2.3 Research Questions

The ideas contained within this paper have been smoldering or percolating over the past 20 years as we confronted solving different problems in different domains. These ideas motivated the following research questions:
RQ1: What is a comprehensive set of analytics classes that can inform researchers and problem solvers?
RQ2: What attributes describe analytics and what morphological forms can analytics take?
RQ3: What transformations can be applied to one structural form to yield another structural form that can enhance its utility given certain attributes?

3. Technical Approach

Our technical approach involved three activities: (1) developing a revised taxonomy of analytics classes; (2) identifying structural forms and descriptive attributes mapped to these classes; and (3) developing transformations between the forms that preserve the attributes and results while enhancing their utility according to certain functional attributes.

3.1 Revised Taxonomy of Analytics

The revised taxonomy in Table 1, reflects a deeper understanding of different types of analytics that have been developed as we reviewed the technical literature. During this review, we noted the different structural forms used to represent analytics in the different classes. A brief description of each class is presented in Appendix A.

Table 1. Analytic Structure Forms

Analytic Class	Structure
Dynamical Systems	Equations, Graphs (State Space Diagrams), Transfer Functions
Decision Theory Models	Equations
Game Theory Models	Matrices, Rules
Probabilistic Models	Equations
Evolutionary Computation	Strings, Vectors, Lists, Sets,
State Transition Systems	Strings, Graphs (State Space Diagrams), Grammars
Graph & Networks	Matrices, Vectors, Graphs, Equations
Agent-Based Simulation	Symbolic Rules, Equations, Integral and Difference Equations
Field Theory Models	Topological Maps, Graphs, Rules
Rule-based & Logic Systems	Logic Statements, Rules
Reduction Analytics: -Filtering Processes	Algorithms, Rules
Language Analytics	Text Structure, Strings, Vectors (bag-of-words), Lists, Sets
Geospatial Analytics	Graphs, Algorithms
Visualization Analytics	Algorithms, Graphs

Some Examples: Time series are an example of n-dimensional vectors. Graphs can be simple with one edge between two nodes, or complex and mesh like, with two or more edges with different attributes between a pair of nodes. A sequence of video frames can be a three-dimensional matrix with one axis as time and the other two axes as the xy representation of an image at an instant in time. Heuristics are often represented as symbolic rules.

4. Structure and Properties of Analytics

A *structural form*, such as depicted in Table 1, is a representation of an analytic. We propose an *analytic schema*, that describes an analytic and is manipulable by a computer program through transformations, which includes these elements:

- Characteristics: This element captures the attributes of the analytic.
- Structural Form: This element describes the analytic according to the schema for the representation style.
- Format: This element describes the format of the structural form.

4.1 Types of Analytics

There are two general types of analytics: *quantitative*, based on numerical data, and *qualitative*, based on symbolic data. Equations, matrices, vectors, algorithms are examples of numerical analytics. Graphs, symbolic rules, grammars, logic formula/statements, text structure, strings, vectors, and data frames are examples of symbolic analytics.

4.2 Attributes of Analytic Types

The *attributes* of an analytic describe their use and execution. We developed an initial set of attributes, depicted in Table 3,thorugh our literature review, and divided them into two groups: functional and non-functional. *Functional attributes* are objectively measurable that can be used to select analytics for particular domains. A *domain* is an area or subarea of an academic discipline, such as political science, that is used to organize knowledge and problem-solving techniques according to a specified set of dimensions. Domain examples include genomics, airline scheduling, and social network analysis. *Non-functional attributes* may be either objectively or subjectively evaluated. They convey important information about an analytics' usability

Table 3. Attributes of Analytics

Functional Attributes
Speed: the time required to execute the whole as well as subelements of the analytic.
Footprint: the amount of memory required for the analytic itself as well as used by it to process the input data; the analytic footprint versus the data footprint.
Accuracy/Precision: data dependent, but based on programming language variables and procedure representation.
Range: The set of values accepted and emitted by the analytic: continuous, categorical, or discrete.
Data Type: Number types (e.g., integer, real, etc.), String, Analytic Type, etc.
Complexity: a computability measure of the time and space to execute the analytic, such as Big 'O' notation..
Parallelizability: An attribute regarding the ability to parallelize all or part of the analytic.
Non-Functional Attributes

Clarity: the ease of understanding the analytic and its subelements functionality, e.g., a corollary to the Flesch reading scale.
Modifiability: the ease of transforming the analytics.
Reliability: the trustworthiness of the results provided by the analytic related to exact or approximate results.
Utility: the usefulness in solving problems within a domain.
Adaptability: the ease with which the analytic can be applied to different domains (with some modification).
Representation: the structural form(s) that the analytic has taken in the literature.
Text: A prose description (step by step of a standardized process) of the analytic and how it executes..
Static, Semidynamic, or Dynamic: is the representation of possible changes in the analytics structure, e.g., dynamic graphs or rule systems; but also how frequently the data used by the analytic changes.
Level of Effort: how difficult is it to use the analytic singly or in conjunction with other analytics.

Each structural form will have different values for the functional attributes when implemented in different programming languages and subsequently mapped to different hardware platforms. For example, certain business analytics are efficient when written in COBOL, but not so efficient if written in Common Lisp or Python. Another example would be to transform an analytic to have lower latency and more efficiency in local data storage in order to handle streaming data at a specific arrival speed.

4.2.1 Structural Forms

A simple graph can also be represented as a 2D matrix in which column and row headers are identifiers for vertices and the cells have entries for the edges representing a relationship between the vertices. A simple graph has a zero or one to represent the existence or not of an edge as shown below in Figure 2. Table 4 presents some graph variations.

Table 4. Some Graph Variation

Weighted Simple Graph (WSG)
Complete/Partial Genealogy Tree
Complex Graph (multiple edges between two nodes)
Weighted Complex Graph (WCG)
Dynamic Graphs – Nodes, Edges, & Attribute & Values

A simple graph with attributes on both nodes and edges could be represented by three matrices: the NxN matrix for depicting one or more edges between two vertices,. and two NxK matrices for edge and node attributes and their values. The K-dimension columns would be labeled with attribute names for the edge and node attributes, respectively, and the cell values would be their values of the attributes. This representation is not an ideal representation, but one possibility for representing the nodes, edges, and their attributes.

A complex graph might be represented as a 3D matrix with column, row and depth headers for vertices, edges, and attributes. A cell entry might be empty, or a list of key-value pairs for attributes. Considering meshes, e.g., graphs with multiple edges between nodes, might be transformed into multidimensional matrices.

For example, a large graph can be converted to a matrix form – either simple or complex – which allows substantial increases in performance, although the greater complexity may degrade performance under certain conditions such as sparsity and/or multi-index addressing. In his manner, a set of equations is converted to matrix form to use the *simplex method* to solve for variable values.

4.2.2 Functional Attributes

The functional attributes of an analytic are metrics that are often based on technology and have been used for the selection of an analytic. The first five are early and well-known metrics. Complexity is a measure of the difficulty of an analytic based on its characteristics.

Parallelizability affects the end-to-end speed of the analytic with a possible range from 0 to 100, with zero implying the analytic is essentially sequential. As noted in Kaisler (2005), it is an open conjecture whether there are algorithms which are inherently sequential and cannot be parallelized (or, at least, no method has been found to parallelize them as of yet).

4.2.3 Non-Functional Attributes

Non-functional attributes are associated with the usefulness of an analytic. These attributes explain what an analytic does. They address how it can be adapted transformed to a different structure with different functional attributes.

5. Transformers

The utility of an analytic can be evaluated by its usefulness, the availability and representation of data within the domain, and the desired thresholds for the functional attributes. The literature reviewed suggests little attention has been focused on transforming analytics from one structure to another to improve the values of the functional and non-functional attributes. As a result, we initiated an early investigation into the types of transformations from one structure to another was begun. In our literature review and research, we found both well-known and emerging types of transformers: structural transformers and performance transformers.

5.1 Structural Transformers

A *structural transformer* changes an analytic from one form to another form. The analytic has the same intent and functional result, but a different representation, which may be more efficient or facilitate some computational analysis. Table 5 presents some general transformations that we have identified.

Table 5. Selected General Transformations

Simple Graph to Simple Matrix
Complex Graph to Complex Matrix
Transition System to Grammar
Grammar to Symbolic Rule System or Logic
Equations/Difference Equations to Matrices
Equation to Algorithm(s) (procedures)
Symbolic Rules/Logic to Algorithms
String to List (Vector) and vice versa
Text Structure to Vector or Lists
Simple Graph to Difference Equation
Symbolic Rules to Logic; Logic to Symbolic Rules

An example of a structural transformer is to change a simple graph to a matrix, perhaps using the simple algorithm below:

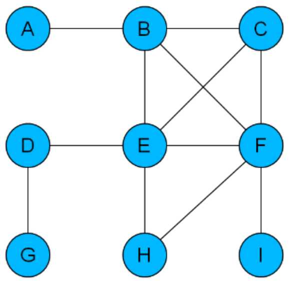
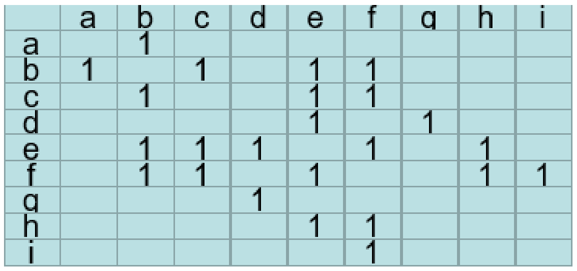


Figure 1. A simple Graph

```
Convert Simple Graph to Matrix:
    Initialize matrix to 0.
    For each node, label a row and a column in the matrix
    For each node, identify the edges to other nodes in the graph
    Place 1 in the row = source node and column = destination node
```

Figure 2. A Graph-to-Matrix Algorithm



	a	b	c	d	e	f	g	h	i
a		1							
b	1		1		1	1			
c		1			1	1			
d					1		1		
e		1	1	1		1		1	
f		1	1		1			1	1
g				1					
h					1	1			
i						1			

Figure 3. A Matrix Representation of G

We note that as the size of the graph grows in number of edges and nodes, the time to compute properties of the graph and traverse the graph increases, perhaps, non-linearly.

Variations of this transformer are defined for each of the types of graph variations mentioned in Table 4. For example, a WCG transformer require a different algorithm because the cells now hold the attribute-value pairs for edge attributes.

Transformers are being developed based on analogies between different disciplines, such as the paper by Dolovitsky, Beyer, Kolesnikov et al. (2011) for an image transformer suggested by a text transformer. Key guidance for adapting an analytic concept to different domains is provided in recent work.

5.2 Performance Transformers

A *performance transformer* modifies the structural form of an analytic to improve its performance (speed, memory usage, accuracy/precision, range, and complexity). As an example, transformers performing parallelization of an analytic are focused on improving the analytics' performance.

This idea is examined in the 1997 dissertation *Making Concurrency Explicit: Converting Object-oriented to Process-Oriented Programs* (Kaisler 1997). It specified constraints on object-oriented programs and proved a duality conjecture analogous to a conjecture by Lauer and Needham (1979). It also developed a methodology that enabled automated conversion of object-oriented programs to process-oriented programs.

6. The Idea of an Analytics Catalog

As noted in section 2, each of the analytics classes can have many different instances of analytics as representatives of the class. Examples of collections of analytics have been assembled, as mentioned in Section 2.1, but a broader effort is needed to organize the myriad of analytics according to the proposed taxonomy. This effort addresses the possibility of a standardized method for describing analytics that can assist researchers and practitioners in selecting analytics for problem solving.

An Analytics Catalog is proposed that is organized along the classes identified in the Taxonomy and briefly describe in Appendix A. For each analytic in each class or subclass, a brief description of the analytic will be provided, along with references, an assessment of the functional and non-function attributes, possible transformers, and a discussion of strengths and limitations. The Catalog will be arranged in volumes each of which corresponds to a major taxonomic class.

6.1 Analytics Description

There are two levels to the description of an analytic. The first is a description of its functional and non-functional attributes. Some of these such as speed and size require further experimentation to establish specific values under different implementation and execution regimes and for different platforms.

A second level will describe structural and performance transformers for selected analytics in each class. It is not clear that generic transformers can be developed for each analytic class given the variations of analytics within the class. The effort to describe transformers for every analytic will be extensive but initial examples and demonstrations will help to guide future contributors to the catalog.

6.2 Catalog Breadth and Depth

A primary purpose of the Analytics Catalog is to enumerate the individual analytics in each class or subclass. The contents of the catalog cannot be exhaustive as new analytics are being developed every year and will continue in to the future. The breadth and depth of content in each analytic class will vary. One purpose of the Analytics Catalog is to document and understand that variation across the different analytic classes.

6.3 Mining the Catalog

As the Catalog evolves, a second purpose will be to extract information about how the analytics are used in different domains to determine by analogy if they have the potential for applicability to other domains. The process will evolve as our understanding of the characteristics and role of the domain is better understood

6.4 Analytics Representation

Representation occurs in several forms, but three of these are: (1) a normal expression in technical papers, (2) a computational form that can be used in computing applications, and (3) a descriptive internal form that will allow transformer algorithms to convert from one internal from to another. Representation transformers convert from a computational form to an internal form and vice versa. The following sections discuss selected examples.

6.4.1 Graphs

A simple graph can be represented as a tuple of the form G = (V,E) where V is a set of vertices and E is a set of edges between vertices, An extended tuple might be G = (V, E, Av, Ae, ADv, ADe), where Av and Ae are sets of attributes for vertices and edges, respectively, and ADv and ADe are definitions of the attributes using fields such as value type, value range, etc.

Figure 4 presents one internal description of a complex graph.

```
(Characteristics
    (Type ‘simple)
    (Dimension ‘2D)
    (Attributes ‘(Scale, Performance, etc.)
    (Structure-Length)
)

(Structural Form
    (List of Nodes)
    (List of Node Attributes)
    (List of Edges)
    (List of Edge Attribute)
    (List of Node Attributes and Fields)
    (List of Edge Attributes and Fields)
)

(Format
    (Nodes List)
    (Edges PairsList)
    (NodeAttributes (Symbol PairsList))
    (EdgeAttributes (Symbol PairsList))
)
```
Figure 4. Internal Representation of a Complex Graph

Clearly, different algorithms would be needed to recognize the structure of a graph and convert it to variations on the internal form. Note that a simple graph representation is encompassed in the data structure above with some fields being null.

6.4.2 Equations

A second example considers an analytic schema for a simple non-linear equation, $Ax^2 + Bx + C = 0$, as presented in Figure 5.

```
(Characteristics
      (Type          SimpleNon-linear)
      (#Terms        <number>)
      (MaxPower      <number>)
      (MinPower      <number>)
)

(Structural Form
      (Exponents     (2 1 0 0))
      (Coefficients  (A B C NIL))
      (Variables     (x Mindpower NIL NIL))
      (Operators     (+ + =))
)

(Format
      (Exponents List)
      (Coefficients List)
      (Variables List)
      (Operators List)
)
```

Figure 5. Internal Representation of an Equation

As an example, the internal representation for an equation could be transformed into a list such as one used by Lisp with a Reverse Polish Notation format.

6.4.3 Handling Dynamics and Sets

The internal description will be complicated by the consideration of dynamic analytics, such as Petri Nets and System Dynamic graphs. Research and testing is required to understand effectively represent these and transform their dynamic aspects.

Transformation of a single equation to a list will make modifications to equations easier to perform. A set of difference equations can be represented by graphs in a System Dynamic format. The evolution and verification of such systems appears to be easier to modify in the graph form before being transformed back to the equation format.

6.4.4 Rules & Logic Statements

The symbolic form of some analytics can be represented in a list form. Conversion of a rule from an If..Then… statement format to a list format may make it easier to generate new rules and modify rules in a dynamic application. An example from a paper by Kaisler (1991) based on Lenat's RL1 representation format is:

English Version:
IfCostOfInterveningMilitarilyIncreasesAlotThenCha
ngesOccur

and the corresponding list representation:

```
IfDirection        ^^^
ThenIncrease       (GovtStability
                    DifficultyOfInterveningPolitically
                    RecentMilitarySuccessRate
                    MilitaryStrength
                    DomesticPublicImage)
ThenIncreaseAlot (DifficultyOfInterveningMilitarily)
ThenDecreaseSlightly (GovtControlOfCivilians)
```

Figure 6. English to List Form of a Rule

The list form allows a rule to be copied and edited by an algorithm to add or subtract clauses in the If and Then parts, or to modify the function of some clause to generate a new variant of the rule.

7. Conclusion and Future Work

This research paper proposes a new direction in the classifying analytics, their enumeration and description, their representation, and their transformation to other forms to attain benefits from different forms.

7.1 Addressing RQ1

Addressing RQ1, we have suggested the development of an Analytics Catalog based on our revision to a previously developed taxonomy of analytics. This Catalog contains sections which are based on the revised taxonomy of analytic classes. It will describe analytics according to their functional and non-functional attributes, and provides a brief description of the basic ideas underlying the analytic. The development of this Catalog will be a useful tool for researchers. It will aid in their understanding of the form and function of different analytics in different domains.

Our conclusion regarding this analytics catalog and the revised taxonomy is that the present work offers a strong foundation in several areas:
(a) Assesses the utility of the taxonomy and contributes to a greater understanding of the types and applications of analytic methods.
(b) Identifies key and optional descriptive attributes to assist researchers in using the catalog to find applicable analytic methods for their research;

(c) Aids researchers' understanding of and the features of each analytic, including input data, output results, procedural information, limitations, linkages to other analytics;
(d) Provides information needed to design transformers and develop programs to automatically perform transforms.

As described, the Analytic Catalog will provide a reference assisting researchers in understanding analytics and select analytics that may be useful and appropriate in their research.

7.2 Addressing RQ2

The literature search revealed that substantial effort is needed to understand and formalize the structural forms and their transformations. Table 1 captures many of the standard structural forms, but it is possible (likely) that other forms may exist, perhaps as modifications to the basic structural forms. Further research needs to be conducted to determine what other structural forms have been used to represent analytics, and how many transformations will be performed.

Selected transformations are presented in Table 5 from one form to another to initiate their study and begin to determine the efficacy of the different forms in problem solving. These transformations render an analytic in one form, which may be computationally intractable, into another form which can be computed with a reasonable finite use of resources because it reduces complexity, reduces barriers to use, and enhance their applicability to different domains. Further research is expected to yield different representation formats and additional transformations between formats.

7.3 Addressing RQ3

This paper includes a transformation from a graph to a matrix. The obverse transformation also exists, Additional research problems may show some transformations are one-way only, assess the performance of the transformations themselves, and demonstrate code how individual transformers enable measurement of some of their attributes for inclusion in the Analytics Catalog.
An extensive bibliography will be available from Kaisler upon request.

8. Future Work

The transformation of analytics from one structure to another is in its early stages of development although as demonstrated in Standish, Harrison, and Kibler (1976), considerable advances have been made in transforming programs to improve performance. But transformation of analytics goes beyond mere syntactic analysis to understand the semantics, e.g., what is the analytic trying to do in order to understand how to improve its functioning.

As we stated in Section 2, the ability to improve analytics is a critical aspect of the field of Analytic Science. We propose several future research opportunities within this subarea that need attention to develop new analytics:
(a) Develop a catalog of transformations from one analytic representation to another analytic representation. Some may be simple syntactic transformations, but all must preserve the semantics of the analytics. A transformation process may include multiple procedures with, perhaps, intermediate representations. As an example, transforming a static analytic to a dynamic analytic to handle streaming data as suggested by Chen and Hsu (2010).
(b) Develop metrics and measurement techniques to assess the semantic content of analytics for use in ensuring preservation of semantics across transformations.
(c) An open research problem is transforming an analytic to a different machine architecture to improve to improve its performance. An example is integrating two or more analytics into a single analytic that can operate on streaming data in a pipeline architecture. A simple solution might be to cascade the two analytics with appropriate syntactic and semantic couplings. An additional problem is resolving the names of variables and procedure calls in each analytic.
(d) The revised taxonomy contains fewer classes than the original taxonomy due to combining some of the analytics classes. Several open questions must be considered. (1) What criteria should be used to consider splitting a class into two classes? (2) Should an analytics class be partitioned into subclasses based on the attribute values of members in the class? (3) Are there additional analytics classes that have not yet been identified that should be added to the taxonomy?
(e) From the original taxonomy, several analytics classes were combined because of their apparent similarity in functionality. Complex analytics may be composites of simpler analytics as mentioned by Bordawekar, Blainey, Apte, and McRoberts (2011). Simpler analytics may reside in different classes from the complex analytic. Further research is required to determine what additional functional or non-functional attributes are need to fully describe a complex analytic.

References

Bacon, D.F., S.L. Graham, and O.J. Sharp. 1993. *Compiler Transformations for High-Performance Computing*, EECS Department, UCB/CSD-93-781, University of California at Berkeley, Berkeley, CA.

Boothboyd, J. 1963. "Algorithm 230: Matrix Multiplication", *Communications of the ACM*, 7(6):347.

Bordawekar, R., B. Blainey, C. Apte, and M. McRoberts. 2011. Analyzing Analytics: Part 1: A Survey of Business Analytics Models and Algorithms, IB Research Report RC2516, IBM, Yorktown Heights, NY.

Chen, Q. and M. Hsu. 2010. Experience in Extending A Query Engine for Continuous Analytics, HP Labs, HPL-2010-44, Palo Alto, CA.

Cormen, T.H. 2022. *Introduction to Algorithms*, 4th Edition, MIT Press, Cambridge, MA.

Dolovitsky, A., L. Beyer, A. Kolesnikov, et al. 2021. "an image is Worth 16x16 Words: Transformers for Image Recognition at Scale", retrieved from: https://arxiv.org/pdf/2010.11929.pdf on May1, 2023.

Kahnemann, D. and A. Tversky. 1974. "Judgment Under Uncertainty: Heuristics and Biases", *Science* 185: 1124–1131.

Kaisler, S. "Designing Geopolitical Simulations Using Knowledge-Based Systems", *World Congress on Expert Systems*, Orlando, FL, December 16-19, 1991.

Kaisler, S. 1997. *Making Concurrency Explicit: Converting Object-Oriented to Process-Oriented Programs*, D.Sc. Dissertation, Dept. of Computer science, George Washington University. [available on Researchgate.com].

Kaisler, S. 2005. *Software Paradigms*, John Wiley & sons, New York, NY.

Kaisler, S. and Cioffi-Revilla, C. 2007. Quantitative and Computational Social Science Methods, 40th Hawai'i Int'l Conference on System Sciences, Big Island, HI.

Kaisler, S., F. Armour, A. Espinosa, and W. Money. (2013) Big Data: Issues and Challenges Moving Forward, *46th Hawaii International Conference on System Sciences*, Maui, HI, IEEE Computer Society.

Kaisler, S., F. Armour, A. Espinosa, and W. Money. (2014) Advanced Analytics: Issues and Challenges in the Global Environment, *47th Hawaii International Conference on System Sciences*, Hilton Waikoloa, Big Island, HI.

Lauer, H.C. and R.M. Needham. 1979. "On the Duality of Operating System Structures". *Operating Systems Review*, 3(2):3-19.

Lengler, R. and M.J. Eppler. 2012. Visualization Analytics – A Periodic Table, www.visual-literacy.org.

Nickerson, R.C., U. Varshney, and J. Muntermann, 2013. "A Method for Taxonomy Development and its Application in Information Systems". *European Journal of Information Systems*, *22*(3), 336– 359.

National Institute of Science and Technology (NIST), Digital Library of Mathematical Functions, Version 1.1.9, https://dlmf.nist.gov/.

Oresky, C., A. Clarkson, D.B. Lenat and S. Kaisler. *Strategic Automated Discovery System (STRADS)*. 1990. published in Knowledge Based Simulation: Methodology and Application, ed. by P. Fishwick and D. Modjeski, Springer-Verlag.

Pick, J. S. Horan, and A. Sarkar. 2022. Spatial Business: Competing and Leading with Location Analytics, ESRI Press, Ann Arbor, MI.

Polya, G. 1945. *How To Solve It: A New Aspect of Mathematical Method*, Princeton University Press, Princeton, NJ.

Schoch, D. 2012. A Periodic Table of Centralities, http://schochastics.net/sna/periodic.html.

Standish, T.A., D.C. Harriman, D.F. Kibler et al. 1976. The Irvine Program Transformation Catalog, TR161, University of California at Irvine.

Teukolsky, S.A., W.H. Press, and W.T. Vetterling 1986. *Numerical Recipes: The Art of Scientific Computing.* Cambridge University Press, Cambridge, England.

Zhang, Y. 2008. "Constraint Solver Techniques for Implementing Precise and Scalable Static Program Analysis", Technical University of Denmark, IMM-PHD-2008-211.

Appendix A. Analytic Class Descriptions

Analytic Class	Brief Description
Dynamical Systems	Dynamical systems are comprised of a set of difference equations of low dimensionality representing known competing forces. Systems dynamics is a subclass that incorporates feedback mechanisms into the model. Control theory – linear (time-invariant) and non-linear (time-variant) – may incorporate feedback mechanisms to modify system behavior.
Decision Theory Models	Decision theory analytics focus on identifying the values, uncertainties, influences, and other factors associated with making a decision that result in a satisfactory or near-optimal decision. Subclasses include regression models, econometric and sociometric models, and Event Data/History Models.
Game Theory Models	Game theory studies strategic interaction between rational decision-making actors. It encompasses a wide variety of games from 2-person zero-sum games to n-person situations exhibiting local and strategic interdependence in cooperative or competitive situations. Subclasses include Zero Sum vs. Non-Zero Sum, Differential, Cooperative vs. Non-cooperative, and Discrete vs. Continuous games.
Probability Theory Models	These models are based on techniques for computing probabilities, based on statistical principles, such as computing distributions Subclasses include Survival, Hidden Markov, Reliability, Hazard, and Expected Utility models.
Evolutionary Computation	These analytics are based on the principles of biological systems for problem solving where the number of variables may be too large for traditional systems. Subclasses include genetic algorithms, particle swarm optimization, evolutionary algorithms, and ant colony optimization.
State Transition Systems	A State Transition System (STS) is comprised of labeled states that indicate domain-relevant entities and hold data. Transitions occur between states governed by rules which determine when transitions can occur. Transitions may occur in different forms. Subclasses include Petri Nets, Grammars, General Automata, Cellular Automata, and Learning Automata.
Graph & Networks	Graphs and networks link a set of entities together through edges (“relations”) which model connectivity within a domain. Semantically-enriched graphs represent deeper knowledge about the domain through attributes. We differentiate between static analytics where the graph has a fixed structure versus dynamic graphs where the graph is evolving over time along with the semantic information associated with nodes and edges.
Agent-Based Simulation	These models apply multi-agent system models to simulate human and social dynamics in complex environments. An early example is the STRADS geopolitical simulation systems (Oresky, Clarkson, Lenat, and Kaisler 1990). Agent-based simulations may be discrete, continuous, or a hybrid. Generally, they are rule-based – whether symbolic or hardcoded.
Field Theory Models	An analytic approach to understanding group/organization behavior by mapping the structure, elements and complexity of the interaction field and environment in which the behavior takes place.
Rule-based & Logic Systems	The use of symbolic rule-based and logical systems to represent and solve qualitative problems, using deductive, abductive, and inductive techniques. For example, the application of constraint solvers to support dynamic taint analysis in program understanding. (Zhang 2008)
Reduction Analytics: -Filtering Processes	These analytics transform data through reduction, enrichment, or structural methods to a new data set. Their purpose is to make the data set computationally tractable for the techniques used in the analytic classes above. Mathematical methods that reduce data dimensionality to identify essential variables, including Principal Component Analysis, Factor Analysis and Singular Value Decomposition.
Language Analytics	These analytics extract words, and phrases from text, process documents, paragraph, and sentences, perform resolution of terms, perform summarization of text, and support query - answer systems. Data may be used to update data and knowledge bases.
Geospatial Analytics	These analytics use data from different sensors to build data visualizations for understanding phenomena and finding trends in complex relationships between people and places (Pick, Horan, and Sarkar 2022).
Visualization Analytics	These analytics process text, imagery, and video to extract data about structure and content that can be transformed to different representations for use by the analytics above. Subclasses include text, image, and video processing analytics.