Analogies from Function, Flow and Performance Metrics

Cameron J. Turner[1], Julie Linsey[2]

[1] Design Innovation and Computational Engineering Laboratory, Clemson University
cturne9@clemson.edu
[2] Innovation, Design Reasoning, Engineering Education and Methods Laboratory, Georgia Institute of Technology
julie.linsey@me.gatech.edu

Abstract. The process of engineering design involves moving through different levels of abstraction of the design problem and solution. During this cycle, the use of analogies has been shown to be a powerful mechanism for the development of a design solution. These design analogies are often drawn from systems that embody a similar function-flow-performance metric combination. Yet, most existing design tools focus not on these abstract representations, but instead focus on functional, linguistic descriptions of the systems. This paper focuses on several significant concepts that are essential to the exploitation of function-flow-performance based comparisons of design analogies.

1 Introduction

Engineers often abstract complex ideas into the realm of mathematics and use these abstractions to predict the performance of designed artifacts. It is not uncommon for designers to proceed through multiple rounds of abstraction and de-abstraction in the development of a design solution. During this process, the use of analogies by experienced designers is well-known. Novice designers often lack the experience that allows expert designers to identify and employ these analogies. However, computational systems that may be able to assist novice and experienced designers explore new realms of analogies are of increasing interest to the engineering design community. Most efforts to date have focused on linguistic approaches, but in this paper, we discuss an approach based on the Functional Modeling as design problem abstraction technique. Use of this approach reveals novel insights into the complex world of analogies and how a formalized approach to abstraction may benefit the search for tools for computational analogy generation.

2 Analogies in Design

Studies of the activities of designers indicate that previous experience is used to identify solutions to many design problems [1-5]. This is achieved through a process of abstracting the design problem at hand to a level that allows other related solutions to be identified (as analogies) and then to de-abstract the analogies into solutions specif-

In Proceedings of the ICCBR 2016 Workshops. Atlanta, Georgia, United States of America

ic to the problem at hand. This process of abstraction and de-abstraction with analogies is used by the design engineers during development of a design [5-6]. Established analogy tools do exist, but many of these systems generate analogies via a verbal abstraction of the problem and perform matches through linguistic similarity and keyword searches. Yet, these are not the only abstractions employed by engineers. [7]

2.1 Established Tools

The established tools and approaches for analogy generation generally rely upon linguistic pattern matching to keywords. Linguistic resources can be explained in terms of patterns and contextual exploration based on syntactic and semantic constraints [8]. Prior research has been focused on the development of analogy database tools supported by linguistic similarity tools [9-12]. Linguistic pattern matching systems can incorporate concepts of adjacency, concatenation, containment, ordering and position of the textual units. Two significant tools that use linguistic pattern matching are the WordTree Method and AskNature.org website.

WordTree Method

The WordTree Method begins with the identification of key function(s) within the problem. Once identified, the user systematically represents the functions in a tree with the verbs associated with each individual function. The individual verbs are identified by specifying a broad-spectrum of verbs that are similar or analogous in meaning. These verbs can be identified either from the designer's knowledge base or through a linguistic pattern match repository, such as WordNet [13]. WordNet is large lexical database developed by Princeton University containing English nouns, verbs, adjectives, and adverbs that have been grouped as a set of cognitive synonyms [14]. For example, as shown in Figure 1, the key function for a laundry folding device is "fold". Using the designer's knowledge or WordNet, a more general (abstracted) functional description of cut is "change surface" (WordNet) or "prepare for storage" (designer knowledge). These descriptions lead to more domain-specific functional definitions such as "collapse" or "douse- as in collapse a sail". Repeating this process, additional analogous verbs can be identified and represented in the form of a tree (see [15] for a detailed explanation of the WordTree Method).

The WordTree Method is a tool that aids in the identification of additional analogies across analogous domains. These various domains allow for a connection to be made between the problem and externalities of the existing design domain. The WordTree database provides analogies that are maintained and continually grows past design solutions where novice engineers can be aided in their goal towards developing a unique solution to a design problem. An example using the WordTree Method is shown in Figure 1.

AskNature.org.

AskNature.org is a community generated online database of biomimetic entries [1] supported by the Biomimicry Institute 3.0 [16-17]. The Biomimicry Institute is a

non-profit organization dedicated to education about biomimicry in nature. Within the AskNature database, the biological and behavioral solutions to natural challenges faced by organisms are described. The AskNature biomimicry taxonomy library contains 8 groups, 30 sub-groups and 162 functions that are separated into the top level groups. These groups are further broken down into the subgroups, utilizing the verbs as the classification. Analogies are identified from matches in these classification groups. The AskNature database system is a unique tool that has been continually expanded since its introduction, limited by opportunity and available funding. When properly used, the AskNature database has the potential to yield numerous design analogies regardless of the user experience level.

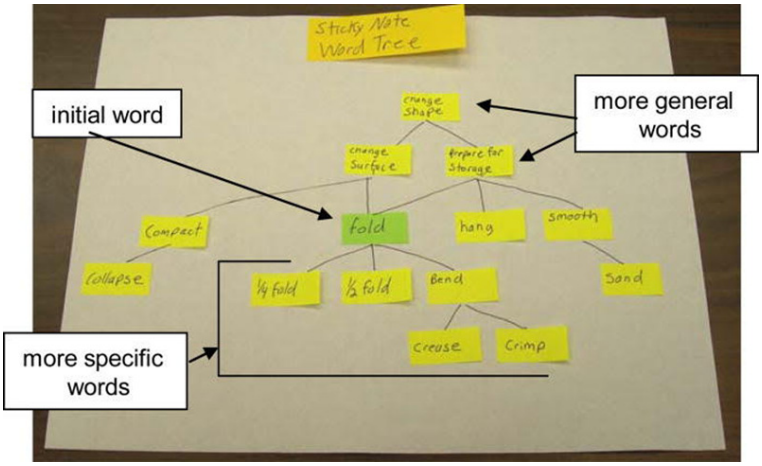

Fig. 1. A Completed WordTree, developed with WordNet. [13]

2.2 Function-Flow-Performance Abstractions & Tools

In a collaborative research effort, researchers at Clemson University, the Georgia Institute of Technology, and the Colorado School of Mines developed the Design Repository & Analogy Computation via Unit-Language Analysis (DRACULA) Design by Analogy tool [18-20]. The DRACULA tool aids Design by Analogy for both novice and expert design engineers alike. The program performs dimensional analysis matching using function, flow and performance descriptions with an evolving database repository of design analogies. The design repository database contains all the analogies available for DRACULA as well as additional analogy information.

The DRACULA tool has several aspects that are similar to other analogy search tools. First, DRACULA is a design analogy tool similar to the WordTree method and AskNature database. With an understanding of the revised functional basis to establish appropriate functions and flows, DRACULA can produce related results from its design repository. Since the foundation of this approach involves functional abstraction concepts, that is the next topic of discussion.

3 Function-Flow-Performance Concepts

A common abstraction tool used by design engineers is the creation of a functional model. A functional model consists of flows (energy, material, and signal) representing the inputs and outputs of the model, which are acted upon by functions (verb-noun descriptors), resulting in changes to the flows. Both the Functions and the Flows are described using a limited vocabulary of terms known as the Revised Functional Basis [5, 21]. The collection of functions and flows provides an abstract representation of the functionality of a design. The performance of a design is characterized by the changes to the flows across and within the functional model.

The key benefit of a functional modeling abstraction to the design engineer is the separation of *function* (what must be done) from *form* (how it is done). This abstraction process can reveal analogous forms that accomplish the same function. Through a de-abstraction process, these alternate forms can be integrated into a new design solution that becomes an analogy to the original design concept.

Generating analogies from a function-flow-performance representation does require the application of several concepts to reveal the foundational core of elements within a functional representation.

3.1 Critical Functionality

Not all functions within a functional model have the same level of significance to the performance of the design. The functions whose performance is crucial to the effective performance of the design are termed “Critical Functions”. Selecting an appropriate form solution for these functions significantly affects the performance of the overall design. Critical Functions are candidates for design analogies.

3.2 Critical Flows

Associated with the critical functions are certain flows (Material, Energy or Signal) whose management is significant to the performance of the device. Just as some functions are more important within the functional model than other functions, some flows are more significant to the performance of the design solution than other flows. These flows are designated as “Critical Flows” and are also candidates for design analogies.

3.3 Key Performance Parameters

The performance of different design solutions can be evaluated by examining how the critical flow(s) are modified by the critical function(s). Each of these measurements represents a Key Performance Parameter (KPP) that is often defined in the context of the design problem. For instance, a design problem may be to increase lift on an airfoil while reducing drag. The KPP could be expressed as the drag coefficient, the lift coefficient, or the lift to drag ratio. However, each of these expressions is a different way of measuring the effectiveness of the function “guide air” on the flows “air” and “kinetic energy” within a function structure, such as that shown in Figure 2.

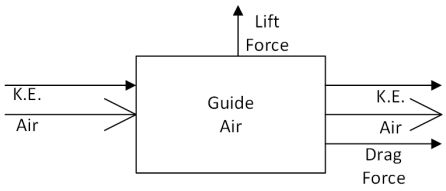

Fig. 2. Simple Functional Model of an airfoil.

3.4 Critical Chains

When taken together, the Critical Functions, Critical Flows, and KPPs represent a functional chain of critical design elements. This chain is defined as a "Critical Chain" within a functional model. One or more functional chains may exist within a functional model. These chains represent an opportunity to identify design analogies based on elements of function, flow, and performance. Furthermore, these chains also can be compared to identify analogies based on elements of Chain Similarity and Chain Architecture. To discuss these elements, we will use the critical chain in Figure 3 as the basis for comparison.

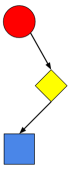

Fig. 3. Abstracted example of a Critical Chain where the shape and color of the block relate to a specific function from the revised functional basis. As a specific example, for a child's toy that has a battery and an electric motor, the red circle would be 'store electrical energy (EE)', yellow triangle 'convert EE to mechanical energy (ME)', and the blue block, 'transmit ME'.

3.5 Chain Similarity

Chain similarity is a measure of the similarity in function (and potentially also in flow) between two chains. Obviously, two chains with the exact same functions (and flows) would exhibit perfect similarity (as defined in Eq. 1) and would be analogies to each other. However, similarity does not need to be complete in order to be significant. For instance, the left example in Figure 4 exhibits partial similarity while the right example exhibits perfect similarity. The greater the similarity between two chains, the stronger the potential for a viable design analogy. Due to relationships between functions, perfect similarity is not required for a viable analogy to exist.

$$ChainA \cap ChainB = ChainA = ChainB \quad (1)$$

where:

ChainA = is the chain of a red circle, yellow diamond, and blue square
ChainB = is the chain of a yellow diamond, blue square, and a red circle.

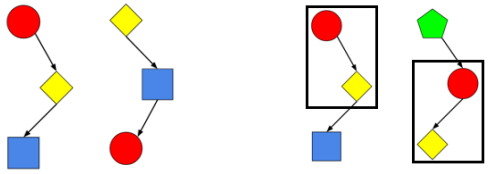

Fig. 4. The critical chains on the left exhibit perfect similarity (all functions exist in both chains) while the critical chains on the right exhibit only partial similarity (both chains share a common subchain).

Partial similarity can mean that only one function is shared between two chains. Conceptually, even a total lack of similarity can exist, due to conceptual relationships between descriptions in the revised functional basis.

3.6 Chain Architecture

The left example in Figure 4 exhibits perfect similarity but distinctly different chain architecture. Chain Architecture is also a factor in identifying analogy matches between critical chains. Even simple linear chains of three or four functions can exhibit a number of distinct architectures including: Identical, Mirror, Disordered, Mirrored Disordered and Unique. Taken as a whole the left example in Figure 4 exhibits a Disordered architecture where the yellow diamond precedes the blue square, but the red circle does not exhibit a common relationship to the other functions. The subchain of the yellow diamond preceding the blue square is an identical architecture, just as is the right example red circle followed by the yellow diamond in the right example. Figure 5 provides examples of the different architectures.

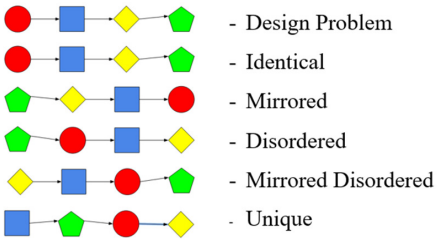

Fig. 5. Chain Architecture Examples.

The study of chain architecture is very interesting. Not only do different architectures exist even for fairly simple linear chains, but critical chains also exhibit additional more complex topologies including trees and potentially rings. Some of these architectural forms result in very close analogies as the order of functions in a functional model is not necessarily unique [22]. This property of functional models is rarely used and poorly exploited within functional models.

4 Matching with Similarity and Architecture

The effectiveness of similarity and architecture comparisons on critical chains can be evaluated using the aforementioned concepts to develop critical chains. Through studies of prior analogy implementations such as [18], and through the identification of previously identified analogies, a set of 26 critical chains [19], representing a total of 59 cases of implemented analogies (some chains led to more than one analogy implementation) was identified. An additional 1711 chain pairs were also available for criteria comparisons to these analogy chains [19]. Using criteria presented in the next section and derived from [20], an exhaustive study of these matches revealed that Similarity and Architecture metrics do produce positive responses for the identification of analogies.

4.1 Criteria

The first metric, *Similarity*, measures the similarity of two chains. Because chains may be of different lengths in the comparison, *Similarity* is defined in Eq. 2 as:

$$Similarity = \frac{2(\text{FcnShared})}{LC_1+LC_2} \tag{2}$$

where:

FcnShared = The number of functions two chains have in common

LC_1 = The total chain length of the input

LC_2 = The total chain length of the source required to cover all common functions.

The next metric, *Identical*, shown in Eq. 3, is nearly identical to the *Similarity* metric, with the exception that its numerator is based on whether or not the chains share the same function order. If the functions in the same location in the chain are the same, the FcnSharedOrder is 1, otherwise it is 0. Thus, if the functions do not share an identical order, the metric value is zero. This evaluation begins with the first shared function in the chains.

$$Identical = \frac{2(\prod \text{FcnSharedOrder})}{LC_1+LC_2} \tag{3}$$

Similarly, the calculation for the *Mirrored* metric, Eq. 4, is also nearly the same as that in Eq. 3. However, in this metric, the FcnSharedInverse term compares the i^{th} function to the m - i^{th} function in the chain where m is the length of the chain. If the terms are the same, the expression is equal to 1, otherwise its value is zero. Thus, the metric is 1 if and only if the chains have the same number of terms in opposite orders.

$$Mirrored = \frac{2(\prod \text{FcnSharedInverse})}{LC_1+LC_2} \tag{4}$$

The *Disordered* and *Deredrosid (i.e. Mirror Disordered)*, Eq. 5 and 6, metrics assign a value to the location of each shared function from the input chain (IFP) to the source chain (SFP), resulting in the average position differences two chains.

$$Disordered = \prod_{i}^{n} 1 - \frac{| IFP_i - SFP_i + 1 |}{n} \quad (5)$$

$$Deredrosid = \prod_{i}^{n} 1 - \frac{| n - IFP_i - SFP_i + 1 |}{n} \quad (6)$$

where:
n = the number of matched functions
IFP_i = the position of input function i
SFP_i = the position of source function i.

The last metric is the *Unique* metric, which is based upon the average of the Disordered and Deredrosid metrics as shown in Eq. 7.

$$Unique = 1 - \frac{Disordered + Deredrosid}{2} \quad (7)$$

All of the metrics range from 0 to 1 and represent an initial attempt to measure similarity and architecture between functions in critical chains. Similar efforts can be developed to also incorporate flows in the evaluations.

4.2 Criteria Evaluation

Our evaluation of these criteria consisted of an evaluation of known analogies versus simply random chain comparisons. If the metrics are detecting analogies, then their averages should deviate from the average of chain comparisons as shown in Table 1.

Table 1. Metric Performance versus a Random Set of Chains, a set of Known Analogies, and the Differences between the Metric Averages. [*Adapted from 20*].

	Criteria				
	Similarity	***Identical***	***Disordered***	***Deredrosid***	***Unique***
Random Avg. ± SD	0.535 ± 0.209	0.287 ± 0.300	0.716 ± 0.397	0.448 ± 0.257	0.202 ± 0.110
Analogy Avg. ± SD	0.704 ± 0.194	0.481 ± 0.362	0.906 ± 0.208	0.523 ± 0.116	0.247 ± 0.062
Difference of Averages	**+ 0.169**	**+ 0.194**	**+ 0.190**	**+ 0.075**	**+ 0.045**
95% Confidence Interval	**0.088 to 0.250**	**0.077 to 0.311**	**0.091 to 0.390**	**0.024 to 0.174**	**0.003 to 0.087**
2-Tail P-value	**0.0001**	**0.0011**	**0.0002**	**0.138**	**0.0376**

In the set of known analogies, we discovered (after the fact) that we did not have a mirrored analogy included in the study. Therefore, we do not have valid results to present concerning this criterion and thus it was omitted from Table 1. Based on the data in Table 1, the criteria appear to be measuring the presence of analogies. Further research into the analogies within the random sample that appear to be previous unidentified analogy matches is still needed to better understand and to further refine the proposed criteria.

5 Conclusions and Future Work

Based on this research, the use of Functional Models as an abstraction tool and the basis for identifying and matching analogies prior to de-abstraction appears to be a promising new approach. Further understanding and refinement of the criteria employed to date are necessary. In addition, the formulation of functional models exhibit varying use of grammars, syntax and levels of abstraction. Understanding and employing these stylistic differences will be important in the continued development of analogy matching tools based upon this abstraction approach.

Of course, functional models are not the only abstraction approach available and employed by engineers. Modeling forms such as SysML, Control Diagrams, and Bond Graphs offer alternative modes of abstraction that may also lead to attractive outcomes when used for analogy matching. Additionally, there a multiple approaches to functional modeling, such as Goel's Structure−Behavior−Function [22]; Gero's Function, Behavior, Structure [23]; amongst others. Each has advantages and disadvantages and much more research needs to explore these for various applications.

6 Acknowledgements

The authors would like to acknowledge the work of their graduate students, Peter Morgenthaler, Dr. Briana Lucero, and Peter Ngo in the development of this research, as well as the financial support of the National Science Foundation through Awards No. CMMI-1304383 and CMMI-1234859. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

7 References

1. L. L. Bucciarelli, Designing Engineers, Cambridge: MIT Press, 2000.
2. A. T. Purcell and J. S. Gero, "Design and other types of fixation," Design Studies, vol. 17, no. 4, pp. 363-383, 1996.
3. G. Pahl, W. Beitz, J. Feldhusen and K. H. Grote, Engineering Design: A Systematic Approach, Springer: 2007.
4. N. Cross, "Designerly ways of knowing: design discipline versus design science," Design Issues, vol. 17, no. 3, pp. 49-55, 2001.

5. Nagel, R. L., & Bohm, M. (2011). *On Teaching Functionality and Functional Model in an Engineering Curriculum.* Proceedings of the ASME International Design Engineering Technical Conferences (IDETC), Washington, DC.
6. B. T. Christensen and C. D. Schunn, "The relationship of analogical distance to analogical function an preinventive structure: The case of engineering design," Memory and Cognition, vol. 35, no. 1, pp. 29-38, 2007.
7. L. J. Ball and N. J. M. Thomas C Ormerod, "Spontaneous analogising in engineering design: a comparative analysis of experts and novices," Design Studies, vol. 25, no. 5, pp. 495-508, 2004.
8. S. B. Hazez, "Linguistic Pattern-Matching with Contextual Constraint Rules," Systems, Mand and Cybernetics, vol. 2, pp. 971-976, 2001.
9. M. R. Bohm, R. L. Nagel and M. L. Shirley, "A Tablet Based Application for Teaching Function and Exploring Concept Alternatives During Early Design," in ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, 2012.
10. J. Hey, J. Linsey, A. M. Agogino and K. L. Wood, "Analogies and Metaphors in Creative Design," International Journal of Engineering Education, vol. 24, no. 2, pp. 283-294, 2008.
11. I. Chiu and L. H. Shu, "Natural Language Analysis for Biomimetic Design," in ASME 2004 Design Engineering Technical Conference and Computers and Information in Engineering Conference, Salt Lake City, 2004.
12. P. Ngo, V. Viswanathan, C. J. Turner and J. Linsey, "Initial Steps Toward an Analogy Retrieval Tool Based on Performance Specification," in ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Portland, 2013.
13. J. S. Linsey, A. B. Markman and K. L. Wood, "Design by Analogy: a Study of the WordTree Method for Problem Re-Representation," Journal of Mechanical Design, 134(4), DOI: 041009-041001-041012, 2012.
14. "About WordNet," Princeton University, 2010. [Online]. Available: http://wordnet.princeton.edu.
15. "AskNature.org," Biomimicry Institute, 2015. [Online]. Available: www.asknature.org/article/view/why_asknature. [Accessed September 2015].
16. Biomimicry Group Inc., "Biomimicry 3.8," [Online]. Available: http://biomimicry.net. [Accessed September 2015].
17. "Sticky Proteins Serve as Glue: Blue Mussel," AskNature.org. [Online]. [Accessed 28 September 2015].
18. P. Ngo, “Surveying Trends in Analogy-Inspired Product Innovation,” Georgia Institute of Technology, Atlanta, GA, 2014.
19. B. M. Lucero, "Design-Analogy Performance Parameter System (D-APPS)," Colorado School of Mines, Golden, CO, 2014.
20. P. Morgenthaler, “Analogy Matching with Function, Flow and Performance,” Colorado School of Mines, Golden, CO, 2016.
21. Hirtz, J., Stone, R.B., McAdams, D.A. et al. Res Eng Design (2002) 13: 65. doi:10.1007/s00163-001-0008-3.
22. K. Otto and K. Wood, Product Design: Techniques in Reverse Engineering and New Product Development, Upper Saddle River: Prentice Hall, 2000.
23. Goel, A. K., & Bhatta, S. R. 2004. Use of Design Patterns in Analogy-Based Design. *Advanced Engineering Informatics, 18*(2), 85-94.
24. Qian, L., & Gero, J. S. 1996. Function-Behavior-Structure Paths and Their Role in Analogy-Based Design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, 10*(4), 289-312.