

AUTOMATED INCLUSIVE DESIGN HEURISTICS GENERATION
WITH GRAPH MINING

A Dissertation

by

SHRADDHA CHANDRAKANT SANGELKAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Daniel A. McAdams
Committee Members,	Julie Linsey
	Richard Malak
	Dylan Shell
Head of Department,	Andres A. Polycarpou

August 2013

Major Subject: Mechanical Engineering

Copyright 2013 Shraddha Chandrakant Sangelkar

ABSTRACT

Inclusive design is a concept intended to promote the development of products and environments equally usable by all users, irrespective of their age or ability. This research focuses on developing a method to derive heuristics for inclusive design. The research applies the actionfunction diagram to model the interaction between a user and a product, design difference classification to compare a typical product with its inclusive counterpart, graph theory to mathematically represent the comparison relations, and graph data mining to extract the design heuristics. The goal of this research is to formalize and automate the inclusive-design heuristics generation process.

The rule generation allows statistical mining of the design guidelines from existing inclusive products. Formalization results show that, the rate of rule generation decreases as more products are added to the dataset. The automated method is particularly helpful in the developmental stages of graph mining applications for product design. The graph mining technique has capability for graph grammar induction, which is extended here to automate the generation of engineering grammars. In general, graph mining can be applied to extract design heuristics from any discrete and relational design data that can be represented as graphs.

Concept generation studies are conducted to validate the heuristics derived in this research for inclusive product design. In addition, an inclusivity rating is created and verified to evaluate the inclusiveness of the conceptual ideas. Finally, appreciation and

awareness about inclusive design is important in an engineering design course, hence, a module is compiled to teach inclusive design methods in a capstone design course.

The results of the exploratory study and validation show that there is problem dependency in the application of the representation scheme. It cannot be stated with certainty at this point if the representation scheme is helpful for designing consumer products, where only the activities related to the upper body are involved. However, self reported feedback indicates that the teaching module is effective in increasing the awareness and confidence about inclusive design.

ACKNOWLEDGEMENTS

I would like to thank all my family, friends and peers for their support and encouragement. I would also like to thank my committee members, Dr. Julie Linsey, Dr. Richard Malak, and Dr. Dylan Shell for their valuable input for my research. I also wish to thank Dr. Schneider for being a mentor for the senior design projects.

Dr. Daniel A. McAdams has been the best advisor. I cannot express my gratitude in words. He ensured that I successfully complete my doctoral work and secure a future in academia. For the future students who may read this – Dr. McAdams is a very patient teacher who will let you grow and become independent.

I want to acknowledge the efforts of my lovely undergrads, Wesley Jones and Aerial Corey, without whom I wouldn't have finished on time. Most importantly, they did the work right with very little guidance so that I need not repeat any of it. I was amazed by the productivity of our undergrads. I wish to thank Michael Glier for proofreading and for countless stimulating discussions on research ideas. Michael has been a great friend through 3 years of the PhD. I also wish to thank Wei Li for rating the concept generation studies.

My Mum and Papa have made so many sacrifices to get me this far. I thank them for believing in me. My lovely dog Sasha comforted me through the toughest final phase of my dissertation. She is the sweetest dog – gentle, smart and loving – and my family here in the United States.

Finally, I wish to thank the National Science Foundation for funding a part of this research and the Department of Mechanical Engineering at Texas A&M University for supporting me as an instructor for the senior capstone design.

TABLE OF CONTENTS

	Page
ABSTRACT.....	ii
ACKNOWLEDGEMENTS	iv
TABLE OF CONTENTS.....	vi
LIST OF FIGURES	x
LIST OF TABLES	xv
CHAPTER I INTRODUCTION AND LITERATURE REVIEW.....	1
Scope of the Work	2
Background and Related Work.....	5
Inclusive Design.....	5
Measurement of Inclusive Design	7
Data Mining	8
Association Rule Mining	8
Graph Mining.....	9
Graphs in Engineering Design.....	9
Graph Grammar Induction.....	10
Computational Design Synthesis	10
Product Family Design	12
CHAPTER II CREATING ACTIONFUNCTION DIAGRAM	14
Functional Modeling.....	15
Activity Diagram	16
ICF Lexicon	17
Actionfunction Diagram	20
Step 1: Problem definition	22
Step 2: Activity Modeling.....	23
Step 3: Functional Modeling.....	24
Step 4: Build the Actionfunction Diagram	26
Step 5: Checklist for Actionfunction Diagram	27
Caveats of Actionfunction Diagram	29
Summary	32

	Page
CHAPTER III PRODUCT PAIR COMPARISON	33
Design Difference Classification	34
Product Pair Comparison Data.....	38
Summary	43
CHAPTER IV ASSOCIATION RULE MINING	45
Mining Association Rules for Inclusive Design	46
Results of the Association Rule Mining	49
Summary	58
CHAPTER V FORMALIZING AND EXPLORING THE TRANSFERABILITY OF THE RULES	60
Rule Generation	61
Pre-Processing.....	62
Association Rule Mining	63
Post-Processing	64
Comparison with Initial Exploration	70
Rules Set Size and Product Set Size	74
Clustering of Product Pairs and Transferability of the Rules	77
Summary	89
CHAPTER VI GRAPH DATA MINING	91
Limitations of Association Rule Generation.....	91
Modified Data Analysis	92
Graphical Representation.....	93
Frequent Pattern Search	97
Graph Visualization	100
Results of the Graph Data Mining	101
Comparison of the Frequent Subgraphs with Association Rules.....	105
Summary	108
CHAPTER VII AUTOMATED HEURISTICS GENERATION.....	110
Background: Computation-Based Design Synthesis	111
Research Approach	113
Creation of the Input	116
Graph Data Mining	120
Interpretation of the Output	122

	Page
Graph Grammar Generation	124
Graph Grammar Results	128
Summary	130
CHAPTER VIII INCLUSIVITY RATING	133
Quality Rating	134
Inclusivity Rating	137
Reliability of the Ratings	140
Summary	141
CHAPTER IX EXPLORATORY STUDY	142
Exploratory Study Setup	143
Study Procedure	143
Study Population	149
Results of the Exploratory Study	150
Participant Feedback	155
Summary	160
CHAPTER X VALIDATION STUDY	162
Validation Study Setup	163
Study Procedure	163
Study Population	168
Teaching Module for Inclusive Design	168
Results of the Validation Study	169
Participant Feedback	173
Summary	176
CHAPTER XI CONCLUSIONS AND FUTURE WORK	178
Contributions	178
Overall Results	179
Limitations	181
Future Work	182
REFERENCES	183
APPENDIX A: INCLUSIVE DESIGN REPRESENTATION SCHEME	196
APPENDIX B: DESIGN TASKS FOR THE VALIDATION STUDY	199

	Page
APPENDIX C: EXACT MATCHING GRAPH GRAMMARS	207
APPENDIX D: INCLUSIVE DESIGN REPOSITORY OF 65 PRODUCT PAIRS	210
APPENDIX E: GRAPHS OF 65 PRODUCT PAIRS	220

LIST OF FIGURES

	Page
Figure 1. The scope of work and major parts of this dissertation.....	3
Figure 2. An illustration of an actionfunction diagram for a bathtub. Dashed boxes represent user activities with related product functions contained within each activity.....	21
Figure 3. Method to create and actionfunction diagram of a product.....	22
Figure 4. Manually operated can opener	23
Figure 5. Activity diagram of a can opener	24
Figure 6. Functional model of a can opener	25
Figure 7. Actionfunction diagram of a can opener	27
Figure 8. A product pair of a Fiskars Rotary Cutter (top) and standard box cutter (bottom).....	33
Figure 9. A steep ramp with slope greater than 1:12 illustrating an inaccessible entrance (left) and a parametric difference in the ramp slope to create a gradual ramp as an accessible entrance (right) [84].....	35
Figure 10. An inaccessible building entrance based on a stair morphology (left) and an accessible building entrance (right) based on a ramp morphology illustrating a morphological difference from the stair [84].....	35
Figure 11. An inaccessible stair step (left) and an accessible building entrance with a wheel chair lift (right) illustrating a functional difference [84]	36
Figure 12. Legend used in actionfunction diagram comparisons	37
Figure 13. The typical can opener is shown on the left and the inclusive can opener is shown on the right	38
Figure 14. Activity diagram and functional model for a typical can opener	39
Figure 15. Activity diagram and functional model for an inclusive can opener.....	40

	Page
Figure 16. The actionfunction diagram for a typical can opener (top) and an inclusive can opener (bottom)	42
Figure 17. A typical phone on left and an inclusive phone with voice activated dialing on right [92].....	52
Figure 18. Actionfunction diagram of a typical phone (top) and inclusive phone (bottom). The internal functions that require ‘no change’ are encircled	53
Figure 19. Automating a wine opener eliminates the need for twisting the hand [92]. Typical wine opener is shown on the left and the automatic wine opener is shown on right	55
Figure 20. Actionfunction diagram of a typical wine opener (top) and inclusive wine opener (bottom). The addition of internal functions is circled for emphasis	56
Figure 21. Rule generation process to study the inclusive design characteristics	62
Figure 22. Total number of association rules generated by the Apriori algorithm against the number of product pairs in the dataset	75
Figure 23. Number of filtered association rules after post-processing against the number of product pairs in the dataset	76
Figure 24. Clusters of inclusive products sharing Rule 1 that recommends a parametric change to the function position hand for better “grasping”	79
Figure 25. Product pairs in cluster #1 sharing Rule 1 and Rule 24 that suggests a morphological change to function of transferring human energy for easier “manipulating”	80
Figure 26. Product pairs in cluster #2 sharing Rule 1 and Rule 12 that suggests addition of a functionality to secure hand while the user “grasps”	81
Figure 27. Product pairs in cluster #3 sharing Rule 1 and Rule 19 that suggests a parametric change to guide solid function to aid “pushing with fingers”	82
Figure 28. Product pairs in cluster #4 sharing Rule 1 and Rule 11 that suggests a morphological change to the cutter	83

	Page
Figure 29. Clusters of architectural product pairs sharing Rule 5, Rule 6, Rule 7, and Rule 9 that deal with a parametric change to allow gross user access.....	84
Figure 30. Clusters of product pairs sharing Rule 8, Rule 10, Rule 16, and Rule 17	85
Figure 31. Clusters of product pairs consisting of a variety of products	87
Figure 32. Other hand held products that exhibit Rule 1 and Rule 12 in form of an ergonomic handle design.....	88
Figure 33. Transferability of Rule 1 between an inclusively designed spatula (left) and a trowel (right) in form of an inclusive add-on module	89
Figure 34. Modified data analysis with graphical representation and frequent-pattern search to study the inclusive design characteristics	93
Figure 35. The bipartite graph representation of a can opener product pair comparison	96
Figure 36. Node labeling and adjacency matrix for the can opener example, where M, F, and N denote morphological change, functional change, and no change, respectively.	98
Figure 37. Visualization of the input data for can opener example along with node numbering.....	101
Figure 38. Sample output from the FSG algorithm.....	102
Figure 39. Visualization for the sample output t # 2-21	103
Figure 40. Application of a frequent subgraph as design heuristic for inclusive design	105
Figure 41. Example of a rule found both by graph data mining and association rule mining.....	106
Figure 42. Example of a subgraph t # 8-0 of size 8 along with its visualization	107
Figure 43. Preliminary phase of engineering design.....	112
Figure 44. Automated heuristics extraction process	114

	Page
Figure 45. Graphical user interface (GUI) for interpreting the graph mining output.....	115
Figure 46. Functional model and image of the Bissell Hand Vacuum from the design repository [62, 97].....	117
Figure 47. The mathematical (txt) format of a graph for the Bissell Hand Vacuum	118
Figure 48. GraphSynth (gxml) format of graph for the Bissell Hand Vacuum	119
Figure 49. A part of xml format (extensible markup language) for the Bissell Hand Vacuum	120
Figure 50. A sample of the output format generated by the FSG (frequent subgraph-mining) algorithm.....	123
Figure 51. Visualization for the output with JUNG (Java Universal Network/Graph Framework) where green color of edge symbolizes energy flow.	124
Figure 52. Rule 5 from Campbell's graph grammar rule set illustrating a propagation rule [98]	126
Figure 53. A frequent subgraph obtained with graph data mining corresponding to Rule 5 from Campbell's set having a frequency of 12.....	128
Figure 54. Rule 24 from Campbell's graph grammar rule and the corresponding frequent subgraph along with a graph with more details	131
Figure 55. Rating to measure quality of ideas.....	135
Figure 56. Rating to measure inclusivity of ideas.....	137
Figure 57. Explanation for the inclusivity rating with a Venn diagram to show the exclusion of people.....	138
Figure 58. Problem statement for the food processor design along with an image of the product.....	146
Figure 59. Problem statement for the refrigerator design	147
Figure 60. The average quality of ideas generated by participants in the control and the experimental condition for the food processor problem.....	151

	Page
Figure 61. The average quality of ideas generated by participants in the control and the experimental condition for the refrigerator problem (p-value 0.11) ...	151
Figure 62. The average inclusivity of ideas generated by participants in the control and the experimental condition for the food processor problem (p-value 0.37).....	152
Figure 63. The average inclusivity of ideas generated by participants in the control and the experimental condition for the refrigerator problem (p-value 0.10).....	153
Figure 64. The percentage of ideas generated by participants that have zero inclusivity in the control and the experimental condition for the refrigerator problem (p-value 0.02).....	154
Figure 65. Results of two tailed t-test to compare the quality and inclusivity for both design problems based on self-reported understanding of the actionfunction diagram (AFD)	158
Figure 66. Plot showing the trends of average quality and inclusivity based on self-reported usefulness of actionfunction diagram (AFD) for both design problems	160
Figure 67. Problem statement for the toaster design.....	165
Figure 68. Problem statement for the blender design	165
Figure 69. General instructions given to the participants for both the design problems	165
Figure 70. Task 1 for the control condition of the toaster design	166
Figure 71. Task 1 for the experimental condition of the toaster design.....	167
Figure 72. Participant feedback on effectiveness of the lecture for inclusive design representation scheme	174
Figure 73. Participant feedback on the perception of inclusive design and effectiveness of the method.....	175
Figure 74. Participant feedback collected to check if the experiment’s objective is met	176

LIST OF TABLES

	Page
Table 1. Alphanumeric code, definition and interpretation for some ICF terminologies	19
Table 2. Comparison of the actionfunction diagram of a typical can opener with the actionfunction diagram of an inclusive can opener	43
Table 3. The association rules generated by the data-mining algorithm, Apriori, for inclusive design	51
Table 4. List of rules that suggest ‘no change’ to the user - product interaction.....	54
Table 5. List of the internal functions of products that requires ‘no change’ for inclusive design	57
Table 6. Rules suggesting addition of internal functions to improve product accessibility	58
Table 7. Comparison of association rules obtained from the dataset of 15 product pairs to the dataset of 65 product pairs.....	71
Table 8. Association rules generated with minimum support of two instances suggesting changes in the product functions for the dataset of 15 product pairs	72
Table 9. Association rules generated with minimum support of two instances suggesting changes in the product functions for the dataset of 65 product pairs	73
Table 10. Sample of input format for the can opener example.....	100
Table 11. List of products studied for the graph grammar generation from the design repository	116
Table 12. Campbell’s rule set categories	127
Table 13. Transaction identifier list for subgraph t # 3-36 corresponding to Rule 5	129
Table 14. Examples of quality scale	136

	Page
Table 15. Examples of inclusivity scale	139
Table 16. Outline for the exploratory study	144
Table 17. Features in a food processor and refrigerator	148
Table 18. Weighted Cohen’s Kappa (linear-weighted) for rating reliability for 50% of the data in the exploratory study	150
Table 19. Participants’ feedback on inclusive design representation scheme	156
Table 20. Outline for the validation study	164
Table 21. Number of participants in the validation study in the control and experimental condition	169
Table 22. Weighted Cohen’s Kappa (linear-weighted) for the validation study	170
Table 23. Chi-Square tests of the observed and the expected average percentage of inclusivity ratings by the control and experimental conditions for the toaster problem (Chi-square = 22.18)	171
Table 24. Chi-Square tests of the observed and the expected average percentage of inclusivity ratings by the control and experimental conditions for the blender problem.....	171
Table 25. Chi-Square tests of the observed and the expected average percentage of quality ratings by the control and experimental conditions for the toaster problem	172
Table 26. Chi-Square tests of the observed and the expected average percentage of quality ratings by the control and experimental conditions for the blender problem.....	173
Table 27. Participant’ perception on understanding the inclusive design rules across the red and blue group	175

CHAPTER I

INTRODUCTION AND LITERATURE REVIEW

Forty-nine million individuals over the age of fifteen in America report having a disability [1]. The number of individuals with a disability constitutes almost 16 percent of the total population. This number represents approximately one in every seven Americans. Based on current demographic trends, particularly the aging population, the number of people with disabilities is expected to increase in the foreseeable future.

Nonetheless, people with a disability are often an underserved segment of the population and an underused resource [2]. Disability is seen as a result of an interaction between a person and that person's environment and contextual factors [3]. Given that disability is viewed in the context of the built environment, a better design of this environment helps to reduce the disability faced by an individual. Designers demand more specific examples of, and methods for, good inclusive design [4].

Within the overarching goal of inclusive design, the specific goal of this research is to establish a product representation framework for modeling the existing inclusive design information, and to develop a method to create inclusive design heuristics. In addition, derive inclusive design heuristics that are applicable in the early stages of product development, and validate those heuristics. The research applies the actionfunction diagram to model the interaction between a user and a product, design difference classification to compare a typical product with its inclusive counterpart, graph theory to mathematically represent the comparison relations, and graph data

mining to extract the design heuristics. The goal of this research is to formalize and automate the inclusive-design heuristics generation process. In addition, the potential of clustering for creating inclusive design product families is explored.

Engineering graph grammars is a very powerful tool for computational design synthesis. However, one of the major difficulties lies in deriving the graph grammar rules. Currently, the engineering grammars, or design heuristics in general, are compiled by an expert based on the empirical knowledge. The graph mining technique has capability for graph grammar induction, which is extended here to automate the generation of engineering graph grammars. In general, graph mining can be applied to extract design heuristics from any discrete and relational design data that can be represented as graphs.

In addition, an inclusivity rating is created and verified to evaluate the inclusiveness of the conceptual ideas. A module is compiled to teach inclusive design methods in a capstone design course to develop appreciation and awareness about inclusive design in an engineering design course. The subsequent part explains the work proposed for this research.

Scope of the Work

Figure 1 explains the scope and the major parts of this dissertation. The scope of study includes generation, and validation of heuristics for inclusive design. The generation of heuristics involves establishing a product representation framework and developing a heuristics extraction procedure.

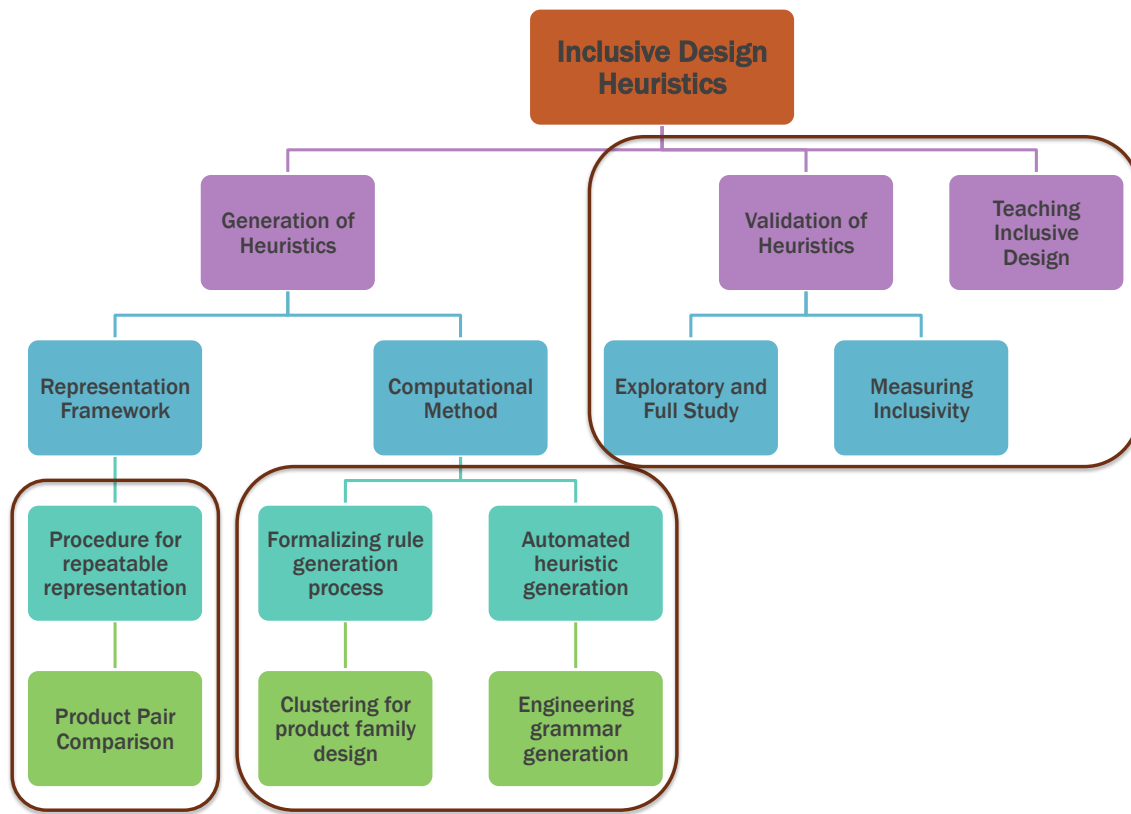


Figure 1. The scope of work and major parts of this dissertation

Chapter II and Chapter III consists of establishing a product representation framework to facilitate repeatable modeling of the design data. The representation framework allows expansion of the inclusive design repository, which is created for building a knowledge base of existing inclusive solutions. In addition, a product pair comparison method is established to compare existing inclusive products with its typical counterparts.

The heuristic extraction procedure includes a computational method to find the trends from the inclusive design repository, as explained in Chapter IV, Chapter V,

Chapter VI, and Chapter VII. The objective is to replace the need for human interpretation with machine learning to obtain statistically significant heuristics. The procedure is formalized and automated for quick extraction of heuristics from the updated inclusive design repository. In addition, the trends in the generation of rules are analyzed to explore that a finite set of inclusive design heuristics can be captured from an arbitrarily large and varied set of products. Further, the derived heuristics are analyzed in detail to evaluate their potential for transferability and reuse from one product to another. Specifically, the products from inclusive design repository are clustered based on the applicable heuristics to develop inclusive product families. Finally, the automated heuristics generation method is tested for automated graph grammar induction for other engineering design research.

The heuristics obtained in this research needs validation to test its applicability for inclusive design, which is explained in Chapter VIII, Chapter IX, and Chapter X. Concept generation study is conducted to test the hypothesis *“application of actionfunction diagrams and inclusive design heuristics helps to generate better ideas for inclusive design.”* Participants are asked to generate ideas for inclusive products with and without inclusive design heuristics. The difference in the inclusiveness of ideas generated in the two conditions is compared.

Ideation metrics have been developed for measuring quantity, quality, novelty, and variety of the generated concepts [5-7]. Currently, we do not have a pre-defined rating to measure the inclusiveness of a design. An inclusivity rating is created to

evaluate the inclusiveness of a conceptual idea and the rating is tested for the inter-rater reliability before application for this research.

The capstone design curriculum does not emphasize on design for disability or inclusive design. It is important to develop an appreciation and awareness about user-centric design in engineering students. A module is compiled to teach mechanical engineering undergraduate students the essence of and the methods for inclusive design.

Background and Related Work

This part explains the background of this research and some related work. The sub-parts describe the relevance of inclusive design, data mining, graph mining, and computational design synthesis as applicable to this research.

Inclusive Design

Inclusive design is a concept intended to promote the development of products and environments to be used effectively by all users without adaptation or stigmatization [8]. Inclusive design is also referred as universal design, design for disability, design for all, or accessible design by other researchers. The landscape of inclusive design literature is vast and contains significant coverage of historical and social context [9].

A team of researchers organized through the Center for Universal Design at North Carolina State University has compiled seven principles of universal design [8]. These principles have been well received by designers in a range of disciplines. Vanderheiden has developed a set of guidelines for the design of consumer products [10]. These guidelines primarily focus on the accessibility of standard computer hardware and software for people with disabilities. Housed in the Center for Inclusive

Design and Environmental Access at the University of Buffalo is an active group of researchers with focus on inclusive design [11, 12]. Though this group focuses on architectural design and comes from an architectural background, it has performed research on appliances and other applications that extend to product design.

A team of researchers at the University of Cambridge has produced implementable results for inclusive product design [13, 14]. The focus of the Cambridge research group has been in modeling user groups, creating product assessment methods, and extending the needs of inclusive design to modern product design processes. Recently, the researchers at Cambridge has compiled an inclusive design toolkit which consists of design process checklist, integrated design log, business case materials, exclusion calculator, simulation gloves and glasses, impairment simulator software [15].

Several conferences have been organized the past decade on inclusive design or universal design. The Universal Design Summit is a national conference organized every two years to foster community integration and participation [16]. The International Conference on Aging, Disability, and Independence (ICADI) promotes approaches that support people as they age to maintain independence in daily living at home, at work, and in the community [17]. The International Association for Universal Design (IUAD) in Japan has arranged three international conferences on inclusive design to support designs that incorporate facility for fair inclusion of the diverse population [18]. An international conference on inclusive design, namely Design for 21st Century, aims for development towards inclusive future and offers a platform for a growing number of practitioners of inclusive design [19]. Such conferences provide valuable information

and sources for inclusive design of products and environments. Of note, the primary focus of these conferences is on the accessibility of the built-in environment and public spaces.

In summary, significant effort has been put into improving design for those with a disability. Nevertheless, inclusive design remains an open challenge with more knowledge and discovery needed to better enable the design of products and services for all potential users. There is still opportunity for significant contributions to be made in areas focused on early design issues and decisions.

Measurement of Inclusive Design

The ideation metrics to measure the creativity in terms of quantity, quality, novelty, and variety have been researched thoroughly and implemented widely [6, 7, 20-25]. However, there are no studies reporting any rating for inclusive design at the conceptual level. Researchers at University of Cambridge have developed an exclusion calculator which gives a detailed percentage of population excluded [15, 26]. While the exclusion calculator is a great tool for evaluating a preliminary design or a prototype, evaluating over a thousand conceptual designs at feature-level with the exclusion calculator is extremely impractical.

Mullick implemented a usability rating scale and an environmental functional independence measure to quantify the universal design of a bathroom [27]. Both the measures implemented by Mullick work well for fully functional prototypes and involve observation of the user interacting with the product. Talley et al. tested the application of Universal Design Performance Measure for Products (UDPMP), which was originally

developed by the Center for Universal Design [8, 28]. However, UDPMP is applicable to the existing products or physical prototypes but not applicable to the ideas at conceptual stages. Thus, prior research efforts in measuring inclusive design have been focused on existing products or preliminary designs. A rating to measure inclusiveness of conceptual designs is required for benchmarking and comparing concept generation among the various methods developed for inclusive design.

Data Mining

Manual analysis of valuable information for decision-making becomes difficult as the database of available information grows [29]. Data mining, a method for extracting patterns of interest from large databases, has application in various fields such as market basket analysis, decision support systems, time series analysis, customer relation management, earth geophysics, atmospheric science, and sky survey cataloging [30, 31]. Product design data is discrete and irregular. Data mining helps to extract useful information from such databases that cannot be studied analytically. This research builds on the usefulness of data mining in the context of research in engineering design.

Association Rule Mining

Association rule learning is a data mining technique that finds associations between variables in large data sets. Association rule learning mines data organized as transactions of items. The data contains $I = \{i_1, i_2, \dots, i_n\}$ which is a set of categorical attributes called *items* and $T = \{t_1, t_2, \dots, t_n\}$ which is a set of *transactions*. Each transaction in T contains a subset of the items in I . An association rule is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. The itemsets X and Y

are called the antecedent and consequent of the rule, respectively. Association rule mining is a developing research area with very few applications developed for research in engineering design [30, 32, 33]. The Apriori algorithm is a well-known and generally accepted algorithm for association rule mining; hence, the Apriori algorithm is employed in this research to mine causal relations [34].

Graph Mining

Data mining aims to find interesting trends from the relations between the entities. In certain cases, the relations between the entities are complicated structural patterns. To extract frequent structural patterns, the data must be represented in a form that not only captures relational information but also outputs comprehensible and meaningful results [35]. Graph mining focuses on mining the data that is represented as graphs. The general form of graph data representation consists of the entities given by the nodes, their attributes specified by the node labels and the relationships between the entities stated by the edges and edge labels. Graph mining has been extensively developed for structured data like chemical compounds, financial networks, and web browsing history. Applying graph mining to product design research has not been explored yet.

Graphs in Engineering Design

Using graph theory as a representation framework for functional models, researchers have developed a set of graph grammar rules for creating, combining, and transforming function structures [36-40]. Zhang recognizes that a functional model stored as a directed graph can be efficiently created, stored, and accessed [41]. Function-

based graph grammars and combinatorics have been applied for product family design [42, 43]. Shai proposes transforming the engineering problem into graphical representation and solving the problem using the tools from graph theory [44]. Other applications for graph-based representation include software for design by analogy, rapid prototyping process selection, material selection model, and optimization of disassembly processes [45-48]. Researchers have applied graph theory as a mathematical representation of functional models. However, the application of function-based graph representation for product design is still an emerging research.

Graph Grammar Induction

The artificial intelligence community has explored graph grammar induction. Most of the induction systems are developed for the text-based application. Recently, the induction techniques are developed for deriving graph grammars [49]. The algorithms developed for graph grammar induction are Apriori based Graph Mining (AGM), frequent subgraph discovery (FSG), and gSpan [50-52]. Note that, the researchers from artificial intelligence community perceive different meaning for the term heuristics [53]. Here, the term design heuristics mean experience based techniques for solving design related problems. In this research, the application of graph grammar induction is explored for mining design heuristics.

Computational Design Synthesis

Functional modeling is a design synthesis method that helps abstract the intended functionality of a product. For computational design synthesis, two main aspects of the functional modeling have scope for improvement: the creation of functional models, and

the application of functional models to generate ideas. Research efforts for creation of functional model have been more substantial as compared to application of functional model for concept generation [54]. This research focuses on creating heuristics that aid generating concepts for inclusive design from a function-based product representation. In the context of this research, the importance of inducing engineering grammars and the importance of being able to mine an updated design repository is discussed.

Engineering graph grammars have tremendous scope for improving overall design research by formalizing the design process to achieve consistent results, and by creating reliable guidelines for product design. Stiny introduced the concept of using shape grammars for generating shapes with shape specific rules [55]. Currently, engineering grammars are derived manually based on the experience to address various engineering design challenges [43, 56-59]. Researchers acknowledge that engineering grammars would be easily applied if the rules could be learned and adapted on their own [54]. Orsborn identifies that statistical analysis of product characteristics is required to determine consistent rules; hence, principal component analysis is applied to automate identification of shape grammar rules [60, 61]. Principal component analysis is mostly suitable for continuous numerical data rather than discrete categorical data. However, functional models are built on a case-by-case basis, hence functional models are discrete rather than continuous constructs [42, 43]. Hence, a computational method to induce engineering grammars is required.

A design repository is developed for collection, storage, and retrieval of the functional models for over 184 products [62]. Research efforts in automated concept

generation suggest addition of a search mechanism to the set of graph grammar rules for exploring various potential solutions from the design repository [56]. Other efforts for automated concept generation include an automated morphological matrix, relational matrices, and unsupervised learning, all in combination with a mechanism to search the design repository [63-65]. The methods for automated concept generation rely on access to a design repository or similar formal knowledge base. As a repository is expanded to contain targeted content, for domains such as inclusive design, mining the repository manually for heuristics or grammar becomes difficult or intractable. An automated heuristic extraction process that has the capability to efficiently mine the updated repositories and find new heuristics for design practice is required.

Product Family Design

Product family design is a way to achieve cost-effective mass customization by allowing highly differentiated products to be developed from a common platform while targeting products to distinct market segments [66, 67]. Here, mass customization offers an avenue to provide persons with disabilities a wide range of products [68, 69]. Related previous efforts extend methods from mass customization and product family design to create specific methods for inclusive product family design [69, 70]. Moon developed several methods to create and make decisions about inclusive product families [68]. The basic framework for the product family is a function-based modular architecture [71]. Tucker applied data mining for product family design [72, 73]. This work explores if rule clustering along with data mining can lead to useful strategies for inclusive product

family design. Further, Chapter II and Chapter III explains product representation framework for modeling the existing inclusive design information.

CHAPTER II

CREATING ACTIONFUNCTION DIAGRAM ¹

Users are perhaps the most important aspect of consumer product design. There is a significant body of research about the ways to measure and achieve user-friendliness in a product. A customer-driven approach to product design aims for greater user comfort and better product experience. Design tools that keep the user in mind during early stages of design can greatly improve the usability of the end product.

Systematically breaking down a product into its constituent functions as is done in functional modeling allows its designer to focus on one function of the product at a time. Functional modeling allows designers to abstract the functionality of a product irrespective of the product's form or shape and is well received as a product representation framework within the design research community. This chapter introduces the concept of an actionfunction diagram, which builds on the framework of functional modeling. Actionfunction diagram incorporates the associated user activities in addition to the functions and the flows in a functional model. Thus, the actionfunction diagram allows designers to effectively model the user-product interaction.

In an actionfunction diagram, an activity diagram and a functional model are combined into a single graphical representation. Product functions are modeled using the Functional Basis and user activities are modeled using the International Classification of

¹ Reprinted with permission from “Creating Actionfunction Diagrams for User Centric Design” by Sangelkar, Shraddha, and McAdams, Daniel A., 2012, Proceedings of the ASEE 2012 Annual Conference and Exposition, San Antonio, TX, Copyright 2012 by Shraddha Sangelkar.

Functioning, Disability and Health (ICF) lexicon. The ICF was established by the World Health Organization and provides a standardized lexicon and taxonomy for the description of health and health-related states. This chapter also explains the application of ICF lexicon for creating actionfunction diagrams.

Modeling the user interaction with a product is a crucial step in product design. The activity diagram is widely recognized as a tool to model user activities in relation to the product. However, the activity diagram stands by itself and does not reflect how a change in user activity affects product function; the user activities in an activity diagram are independent of the product functions. A product representation framework like the actionfunction diagram overcomes the limitations of the activity diagram, enhances the effectiveness of functional modeling, and provides a means to incorporate user-product interactions in the early stages of design. This chapter details the procedure for creating actionfunction diagrams with a case study on a can opener. The chapter also provides a checklist for building actionfunction diagrams and lists the caveats for creating consistent actionfunction diagrams.

Functional Modeling

A functional model is a representation of the process by which the product being designed will be functionally implemented [74]. Functional modeling abstracts the overall functionality of the product and decomposes that functionality into its constituent elements, allowing the designer to focus on partial solutions for enhancing the creativity. Functional modeling allows designers to balance between alternative choices thus reducing designer biases towards specific solutions.

To represent product function, this work uses the Functional Basis and its associated flow based functional modeling methodology [75]. The Functional Basis is a thoroughly evaluated and accepted method for representing product function [76-79]. Nagel and Otto offer good summaries of discussions on how to use the function based modeling method [74, 80].

Activity Diagram

An activity diagram is a network of high-level user activities encompassing the life cycle of a product from purchasing to recycling or disposal. Activity diagram is also known as a network of parallel and sequential tasks carried out by the user, where parallel activities lead to parallel product functions. An activity diagram gathers customer needs in the initial phases of product design and helps establish the boundaries of the product or system under consideration.

The main objective of an activity diagram is to ensure that the designer is aware of the entire set of customer needs during the life cycle of a product. Generally, the list of activities does not include design or manufacturing related activities. Although improving purchase and disposal of a product can significantly impact the design of product, those user activities are not considered in this discussion of the activity diagram for brevity.

To easily understand the actionfunction diagram, a good understanding of the activity diagram is useful. Otto and Wood provide a detailed explanation of the activity diagram creation process [74]. A formal representation for describing varied user

activities improves consistency and communication among different designers. For the user centric design, user activities can be modeled using ICF that is explained next.

ICF Lexicon

The International Classification of Functioning, Disability and Health (ICF), established by the World Health Organization provides a standardized common language for the description of health and health-related states [3]. The ICF provides alphanumeric codes and definitions for a terminology that allows a uniform description of human functional ability and limitation. The ICF is originally intended to serve as a statistical, research, clinical, social policy and education tool [3]. The following discussion explains the ICF in further detail as it applies to product design.

The ICF is classified in two major parts, namely *i)* functioning and disability and *ii)* contextual factors. This research mainly focuses on the functioning and disability part, which further divides into two components: *a)* body functions and structures and *b)* activities and participations. Body functions and structures are useful from a biomechanical perspective of product design. However, for the description of user activities related to a consumer product, the component activities and participation seems appropriate.

The ICF uses an alphanumeric system of classification. The letters b, s, d, and e are used to express the body functions, body structures, activities and participation, and environmental factors, respectively. The letter is followed by a numeric code; first digit of which is the component number followed by two digits signifying second level of detailed classification. In certain cases, there might be a third and fourth level of

classification specified by one following digit each. For example, *d4* is mobility related activities, where *d450* to *d469* cover walking and moving; *d450* is the activity of walking, *d455* is the activity of moving around, and *d4552* is the activity of running.

Qualifiers are placed to the right of a decimal point following the alphanumeric code [3]. The qualifier for activities and participation component is a capacity or performance qualifier indicating the level of ability or limitation. The qualifiers are specific to the product being designed; hence the qualifiers are not included in the discussion to maintain generality.

Formally modeling user activity with the ICF is still a developing method and requires some interpretation. Table 1 lists the ICF terminologies with its alphanumeric code and the formal ICF definition. An interpretation of the ICF terminologies as user activities for product design is also listed in the last column of Table 1.

Table 1. Alphanumeric code, definition and interpretation for some ICF terminologies

ICF term	ICF code	ICF Definition	Interpretation for activity modeling
Picking up	d4400	Lifting or taking up a small object with hands and fingers, such as when picking up a pencil.	Pick up hand held products
Grasping	d4401	Using one or both hands to seize and hold something, such as when grasping a tool or a door knob.	Hold an object firmly in hand for required operation
Manipulating	d4402	Using fingers and hands to exert control over, direct or guide something, such as when handling coins or other small objects.	Complex hand activities that requires manipulation with fingers
Pulling	d4450	Using fingers, hands and arms to bring an object towards oneself, or to move it from place to place, such as when pulling a door closed.	Pulling with finger, arm, hand (grasping of the object is included)
Pushing	d4451	Using fingers, hands and arms to move something from oneself, or to move it from place to place, such as when pushing an animal away.	Pushing with finger, arm, hand (can be performed with a closed fist)
Reaching	d4452	Using the hands and arms to extend outwards and touch and grasp something, such as when reaching across a table or desk for a book.	Reach out or extend outwards to position an object using hands
Turning	d4453	Using fingers, hands and arms to rotate, turn or bend an object, such as is required to use tools or utensils.	Turning knob or tap. Rotate something with hand
Carrying, moving and handling objects	d449	Carrying, moving and handling objects, other specified and unspecified.	For importing and positioning an objects
Communicating with-receiving- written messages	d325	Comprehending the literal and implied meanings of messages that are conveyed through written language (including Braille).	Reading signs and symbols for directions
Pushing with lower extremities	d4350	Using the legs and feet to exert a force on an object to move it away, such as pushing a chair away with a foot.	Pushing with leg force
Sitting	d4103	Getting into and out of a seated position and changing body position from sitting down to any other position, such as standing up or lying down.	Getting into and out of a seated position
Standing	d4104	Getting into and out of a standing position or changing body position from standing to any other position, such as lying down or sitting down.	Getting into and out of a standing position
Moving around within the home	d4600	Walking and moving around in one's home, within a room, between rooms, and around the whole residence or living area.	Approaching household appliances and positioning oneself with respect to it
Transferring oneself	d420	Moving from one surface to another, such as sliding along a bench or moving from a bed to a chair, without changing body position.	Moving from wheelchair to seat or vice versa
Speaking	d330	Producing words, phrases and longer passages in spoken messages with literal and implied meaning, such as expressing a fact or telling a story in oral language.	Give voice commands to operate a device

For example, the activity of grasping (*d4401*) is defined as using one or both hands to seize and hold something, such as when grasping a tool or a doorknob. In the context of product design, grasping represents the activity to hold an object or the handle for operation of the device. The activity of speaking (*d330*) has the implied meaning of giving voice commands for activation of devices. Other general activities like carrying, moving and handling objects (*d449*) and moving around within the home (*d4600*) are used while defining the peripheral activities.

Note that during the activity modeling effort, the designer can move back and forth between different levels of the ICF. For example, *d415* and *d4452* are at different levels of specification, or fidelity, within the ICF taxonomical structure. The goal is to model the user activity as precisely as the ICF lexicon allows rather than to use a consistent level within the ICF.

Actionfunction Diagram

An actionfunction diagram is a product representation framework for modeling the user-product interaction. Actionfunction diagram is a formal representation used to analyze the interplay between user and product in the early stages of design [81-83]. In an actionfunction diagram, an activity diagram and a functional model are combined into a single graphical representation.

Figure 2 shows an actionfunction diagram of a bathtub where the dashed boxes represent user activities and solid boxes represent the related product functions contained within each activity. The arrows represent the flow of energy or material similar to a

functional model. Figure 2 is for illustration purposes only and the detailed procedure to build an actionfunction diagram follows.

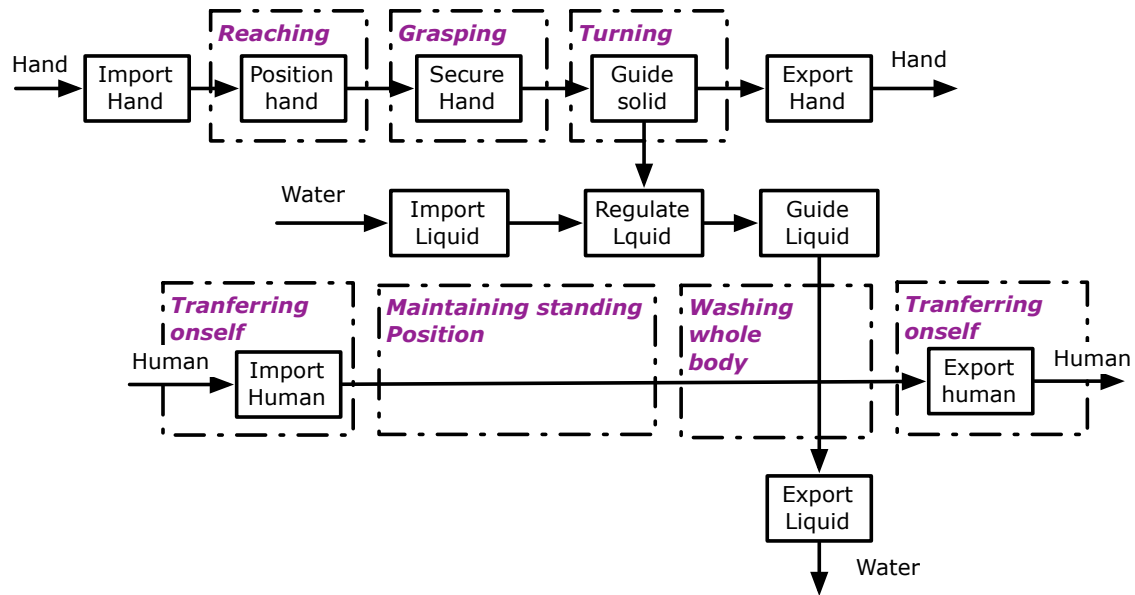


Figure 2. An illustration of an actionfunction diagram for a bathtub. Dashed boxes represent user activities with related product functions contained within each activity

Figure 3 shows the steps in creating an actionfunction diagram: problem definition, activity modeling, functional modeling, building actionfunction diagram, and checklist for actionfunction diagram.

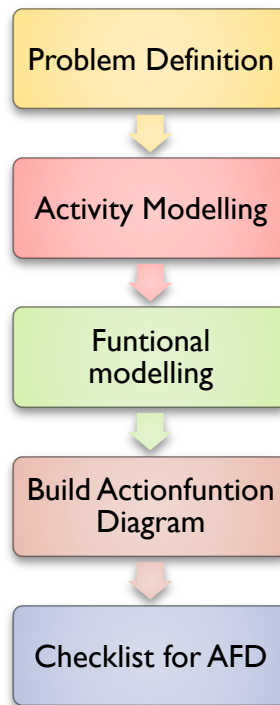


Figure 3. Method to create and actionfunction diagram of a product

Step 1: Problem definition

The problem definition step signifies “knowing what you are designing” before starting with the process. This step is similar to the preliminary design task found in any of the textbooks on product design. Otto and Wood describes this step as formulating a mission statement [74]. The elements of the problem definition those are crucial for actionfunction diagram are emphasized here.

- Gather and list all the customer needs
- Decide the scope of design and set the system boundaries
- Define the overall functionality of the product

As an example we will proceed to create an actionfunction diagram of a pre-existing product, a manually operated can opener, as shown in Figure 4. The example will help to illustrate step 2, step 3, and step 4.



Figure 4. Manually operated can opener

Step 2: Activity Modeling

When creating an activity diagram, list all the activities that are performed or can be potentially performed while using the product. List only the high level activities and not the detailed activities at this stage. Posing a question like, “What are the basic steps the user needs to perform to complete the task using the product?” can help with the activity modeling. Ensure that all customer needs are taken into consideration while listing the activities. Also, clarify the scope of design at this stage.

Once all the activities are listed, arrange all the activities in parallel and sequential manner and connect the activities with arrows to form a network. The direction of arrows indicates the precedence of one activity with respect to the other activities. The example of an activity diagram for the can opener follows. Figure 5 shows the activity diagram of a can opener. Note that in the activity diagram, the overall

activities, represented using natural language, are denoted in blue boxes with formal ICF activities and the associated alphanumeric code contained within black dashed boxes.

While opening a can with an opener, the user carries the unopened can and places it on the work surface. Next, the user picks up the opener and engages it with the can. After engaging the opener with the can, the user twists the handle with one hand while holding the opener with the other. Finally, the user removes the lid.

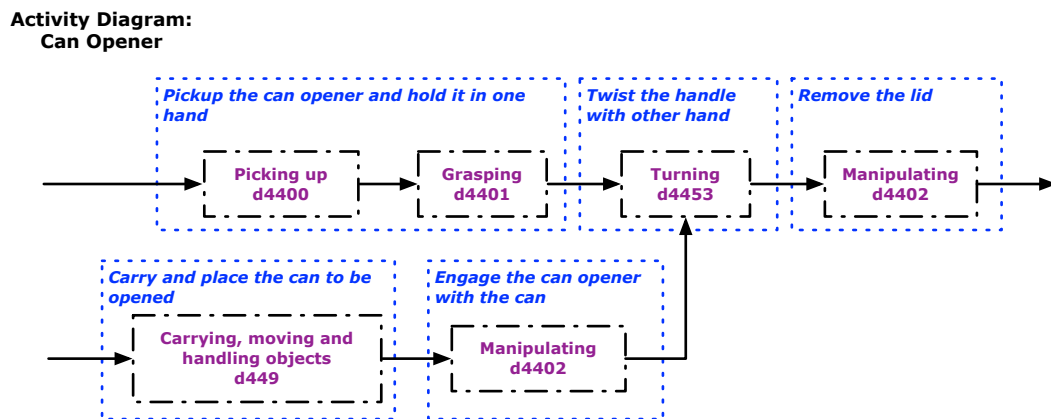


Figure 5. Activity diagram of a can opener

Step 3: Functional Modeling

To develop a functional model, first list all the product functions that are required to achieve the overall functionality of the product. Otto and Wood recommend developing an activity diagram as a prior phase of function structure development, specifically for formulating sub-functions through task listing [74]. Also, mapping the product functions to the customer needs is helpful for functional modeling. Next, describe all products functions using functional basis. Refer to Otto and Wood for the

guidelines on function structures and the Functional Basis [74]. Place the functions in parallel and sequential manner based on the flows of material, energy and signal. A functional model acts as the basic layout for building actionfunction diagram.

Figure 6 shows the functional model of a can opener. The can opener imports and positions a user hand. The handle of the opener provides the function of positioning hand. The product also has a provision for coupling the can to the opener. Further, the handle transfers human energy to rotate the gear. The set of gears convert the human energy into rotational energy, which is expended to guide the can relative to the opener and rotate the cutter. The rotary cutter performs the cutting action, and separates the lid from the can.

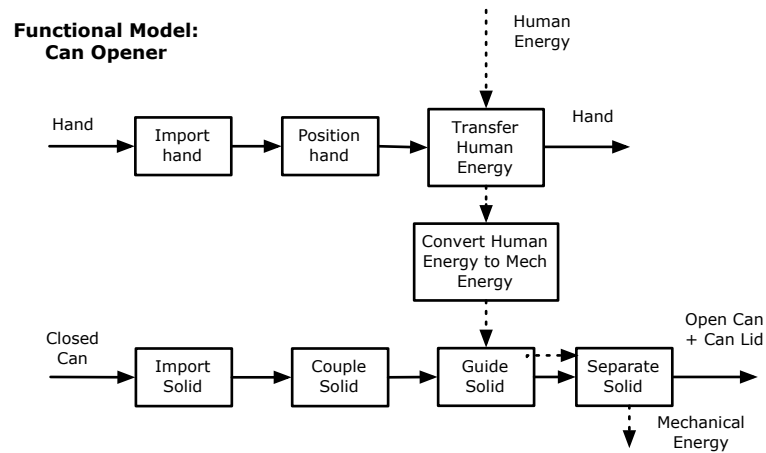


Figure 6. Functional model of a can opener

Step 4: Build the Actionfunction Diagram

An actionfunction diagram is created by merging the activity diagram with the corresponding functional model. The relevant user activities are superimposed on corresponding product functions within the functional model. For instance, when the user is turning the handle of the opener, the product is transferring human energy and converting it to mechanical energy. The mechanical energy is used to rotate the cutter and cut through the lid. The lid is not exported by the system but removed by the user manually. Figure 7 shows the actionfunction diagram of a can opener.

Some activities may not have functions associated with them. For example, the activity of manipulation, to remove the lid away from the can, does not have any product function associated with it. Similarly, some functions may not have any activities associated with them. For example, the guide solid function performed by the set of gears is independent of the user. Such functions are referred as the internal functions of the device.

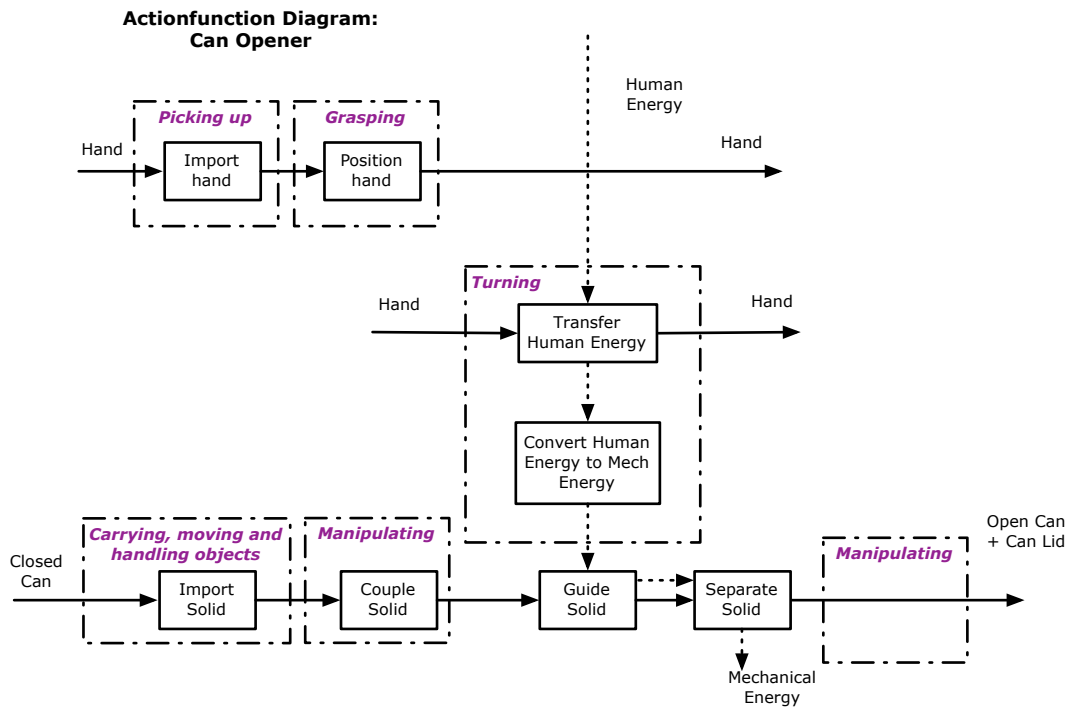


Figure 7. Actionfunction diagram of a can opener

Step 5: Checklist for Actionfunction Diagram

It is recommended to go through the checklist after creating an actionfunction diagram as a check for completeness and correctness. The checklist also acts as a guide for creating an actionfunction diagram. However, it is advisable to first create the actionfunction diagram with the procedure described above and then use the checklist.

- Is the overall functionality of the product achieved?
 - Check the problem definition of the product
- Are all the customer needs considered?
 - Check the elementary list of user activities, created in step 1, to see if all activities are included

- Are the laws of conservation maintained?
 - Energy and material that goes into the system must also go out of the system
- Are the system boundaries clearly defined?
 - Verify the scope of design established before starting with the design
- Are the sequential and parallel functions and activities correctly placed?
 - If the activities and functions can be rearranged in more than one technically correct way, then choose the arrangement that is most appropriate
- Are all the energy flows considered?
 - Check if all forms of energies are included, specifically human energy
- Are any signals going into and out of the system?
 - Include all necessary signals, especially signal input from the user and signal output to the user
- Are the arrows specifying energy, material and signal distinct and clearly marked?
 - Specify the legend if its different form the conventions used for functional modeling
- Are all functions described using the Functional Basis?
 - Refer to Otto and Wood for Functional Basis definitions and classification
- Are all the activities expressed using the ICF?
 - Refer to ICF browser <<http://apps.who.int/classifications/icfbrowser/>>
- Are the activities and functions abstract?

- Use non-specific terms for describing activities and functions to maintain generality of the design and its application
- Can any of the activities and functions be removed without affecting the overall functional requirements?
 - Remove all the redundant functions and activities
- Are the activities and functions atomic?
 - If the function or activities can be divided into two or more, then divide it to the lowest level possible

Caveats of Actionfunction Diagram

This part describes some caveats of actionfunction diagram modeling. They provide helpful guidelines for creating consistent user-product interaction models.

- Include “export” function only if the product intentionally sends material or energy out of the system boundary.
 - *Example:* Popcorn is exported out of a popcorn maker but the heat generated due to friction is not exported by the system.
- Use the function “secure hand” only when a specific component of the product get attached to user hand.
 - *Example:* A circular loop around the handle that prevents the product or a part of the product from slipping out of the hand is modeled as a function “secure hand”.

- When the user's hand is repositioned to perform another set of tasks, the activity of “reaching” is repeated.
 - *Example:* The user has to reach out to open the door of a washer and again reach out to press the start button.

- The ICF user activity of “maintaining body position” is used to model a sitting or standing posture of the user. This activity is important for modeling, especially, when the system boundary includes the space around the product.
 - *Example:* While designing the large household appliances the location and position of the appliance in the house is an important consideration for user-centric design.

- “Grasping” is not required when the activity can be performed with a closed fist.
 - *Example:* A round doorknob needs “grasping” and “turning”, while lever type knob allows “pushing” with a closed fist.

- The activity of “picking up” signifies that the user picks up the product itself. Weight of the product that does not require “picking up” activity is not critical for the design.
 - *Example:* Plier type stapler needs to be lifted up before using as compared to the desk stapler that need not be picked up.

- The activity of “carrying, moving and handling objects” represents that the object to be operated upon is imported within the system boundary.

- *Example:* The user carries the can to be opened or the paper to be punched into the system boundary.
- The touch type or press type buttons should be modeled differently.
 - *Example:* The touch type buttons are modeled as a function “actuate signal” and the related user activity is “reaching”. On the contrary, for push buttons are modeled as the activity “pushing with fingers” while the product “guides solid” to actuates the signal. Also note that, the human energy is not required for touch type buttons but is required for push type buttons.
- Residual energy though important for conservation of energy, does not affect the user-centric design.
 - *Example:* The vibration and noise generated by the electric knife can be neglected in the actionfunction diagram.
- Importing human energy function is not required; it is assumed that human energy is used to perform the activities. Explicitly state conversion of human energy into some other form of energy when significant amount of human energy is required.
 - *Example:* Human energy is not required to operate touch buttons but is required to crank an engine.
- Tracking of human or human parts is significantly important for user centric design. To be precise, ‘Human’ and ‘Hand’ are used specifically. ‘Human’ is used if the

entire human is involved in the activity. When the user performs a task with upper limb or hand, it is represented as ‘hand’.

- *Example:* Activities like entering or exiting a bathtub are modeled as “human” for and the activities like operating a switch are modeled as “hand”.

Summary

The chapter explains the procedure to create an actionfunction diagrams for user centric design of products. Some of the advantages of the actionfunction diagram are: focusing closely on user-product interactions, highlighting those functions of a product in which the user is involved, and allowing for analysis of user-product interaction in the early design stages. This method can also be introduced in an engineering design curriculum for user-centric design.

CHAPTER III

PRODUCT PAIR COMPARISON²

Two products that provide the same high-level function but differ in their level of accessibility are termed a product pair [82]. The inclusive product better accommodates the user with a disability by introducing one or more design features that are not observed in the typical version. Figure 8 illustrates a utility cutter product pair. The Fiskars Rotary Cutter and the typical box cutter provide the similar function of paper and cardboard cutting, but the Fiskars Rotary Cutter has features that make it preferable for users with reduced hand functioning. The inclusive design features of the Fiskars Rotary cutter are an ergonomic handle for better grip, a circular blade that can cut in both a push and pull direction, and a simple push button to retract the blade.



Figure 8. A product pair of a Fiskars Rotary Cutter (top) and standard box cutter (bottom)

² Reprinted with permission from "Adapting ADA Architectural Design Knowledge to Product Design Using Association Rule Mining: A Function Based Approach" by Sangelkar, Shraddha, and McAdams, Daniel A., 2012. Journal of Mechanical Design, 134 (7), 071003, Copyright 2012 by Shraddha Sangelkar.

Specific characteristics of a product pair make it preferable for this research. Inclusive products are selected based on the products selection criteria explained next, and their typical counterparts are identified to form product pairs for this research. Commercially successful inclusive products are preferable product pair candidates. Products consistent with a design specification mandated by law also indicate a good candidate product pair. In addition, examples of inclusive design provided by researchers in this field are used as a benchmark while selecting good product pairs. The inclusive products are selected from a variety of sources to maintain the generality of the dataset studied in this research. Though it is a finite set of products selected for this research, the product set reflects a range of scales, uses, and manufactures in non-competing industries.

Design Difference Classification

Previous inclusive design research indicates that in many cases the difference between inclusive and typical products in a product pair is minimal; differences between the products are often subtle and a significant portion of the components are essentially identical [82]. The differences that do exist can be classified as parametric, morphological, or functional. These concepts are clarified here.

A parametric difference between a typical and inclusive product refers to two products that could be described with the same parameterization, but have a differing value for some parameter. Parametrically different products exhibit common detailed functionality, solution principle, and form. A sloped ramp entrance can be used to illustrate a parametrically different product pair in Figure 9.

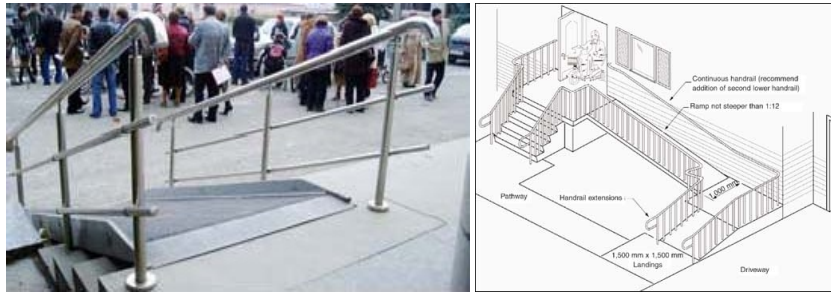


Figure 9. A steep ramp with slope greater than 1:12 illustrating an inaccessible entrance (left) and a parametric difference in the ramp slope to create a gradual ramp as an accessible entrance (right) [84]

A morphological difference refers to two products that retain the same detailed functionality but exhibit a different physical solution principal, form, or geometric topology. Again, using a building entrance as an example, a ramp and a stairway can be used to illustrate a morphologically different product pair. Figure 10 shows an inaccessible entrance based on a step morphology (left) and an accessible entrance using a ramp morphology (right).

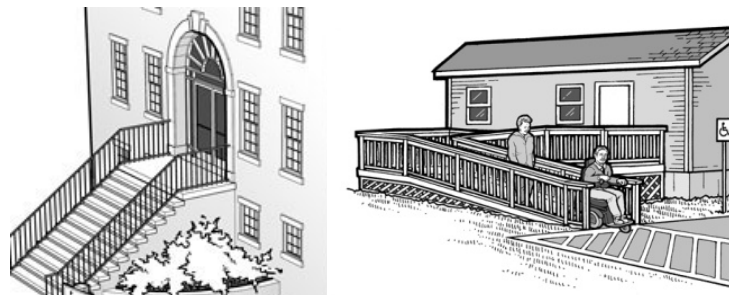


Figure 10. An inaccessible building entrance based on a stair morphology (left) and an accessible building entrance (right) based on a ramp morphology illustrating a morphological difference from the stair [84]

A functionally different product pair indicates the addition or deletion of a product sub-function, or the change of some specific product sub-function, to improve its accessibility. Figure 11 shows a wheel chair lift in addition to, or in place of, stairways at a building entrance is an example of a functional difference. The lift adds a new set of functions to make the building entrance accessible.

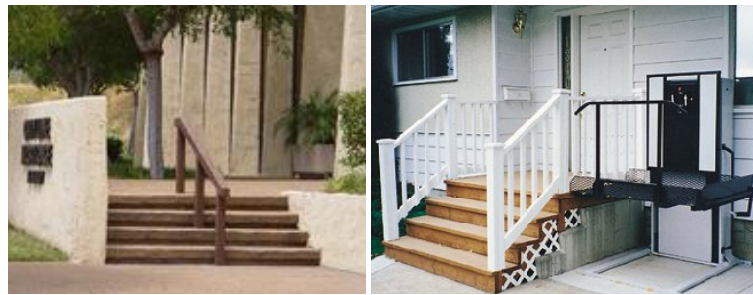


Figure 11. An inaccessible stair step (left) and an accessible building entrance with a wheel chair lift (right) illustrating a functional difference [84]

The representation scheme to reflect parametric, morphological, or functional difference as a product changes from typical to inclusive is shown in Figure 12. A solid line, a dotted line, and a dashed line represent the material, energy, and signal flows respectively. The dashed boxes represent user activities and solid boxes represent the related product functions contained within each activity.

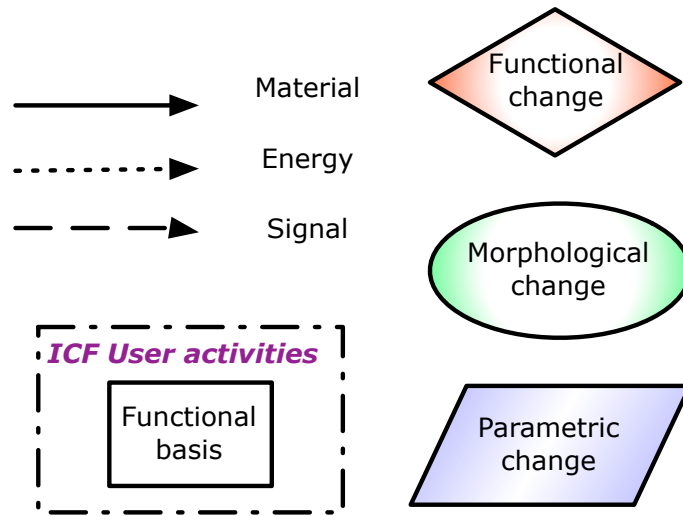


Figure 12. Legend used in actionfunction diagram comparisons

Our classification of functional, morphological, and parametric design differences is new in the context of formally comparing design differences in product pairs. However, it is important to note that similar design classifications of design abstraction have been developed and accepted. As an important example, our classification system is analogous to Gero's well-established function, behavior, and structure (FBS) framework. The functional, morphological, and parametric difference can be viewed as a different level of design problem abstraction for which the design is tasked with seeking a solution.

Gero classified the formal representation of a system as function, behavior, and structure; each specifying different levels of abstraction [85, 86]. Gero's function seeks the function to achieve the desired objective; our notion of function is largely equivalent, though we pull from the formal definition and style of function from the Function Basis [75]. Our notion of morphological refers to the gross structure and solution principle

used to achieve the needed functionality. Similarly, Gero's behavior defines how the structure of an artifact achieves the desired function. Our notion of parametric refers to the specific part geometry and instantiation of materials and etc. used to embody the design morphology. Gero's structure details the physical components to achieve the desired behavior and overall functionality of the system.

Product Pair Comparison Data

This part explains the process of comparing a product pair. The process is illustrated through a product pair of can openers as shown in Figure 13. The typical can opener shown on left needs grasping with one hand and twisting with the other hand. In contrast, simply grasping with one hand operates the electric can opener on the right.



Figure 13. The typical can opener is shown on the left and the inclusive can opener is shown on the right

To begin the analysis, functional models and activity diagrams are created for both the typical and the inclusive can opener. Figure 14 shows the activity diagram and functional model for a typical can opener. Procedure for creating an activity diagram and a functional model for a typical can opener is detailed in Chapter II. Similarly, Figure 15 shows the activity diagram and functional model for an inclusive can opener.

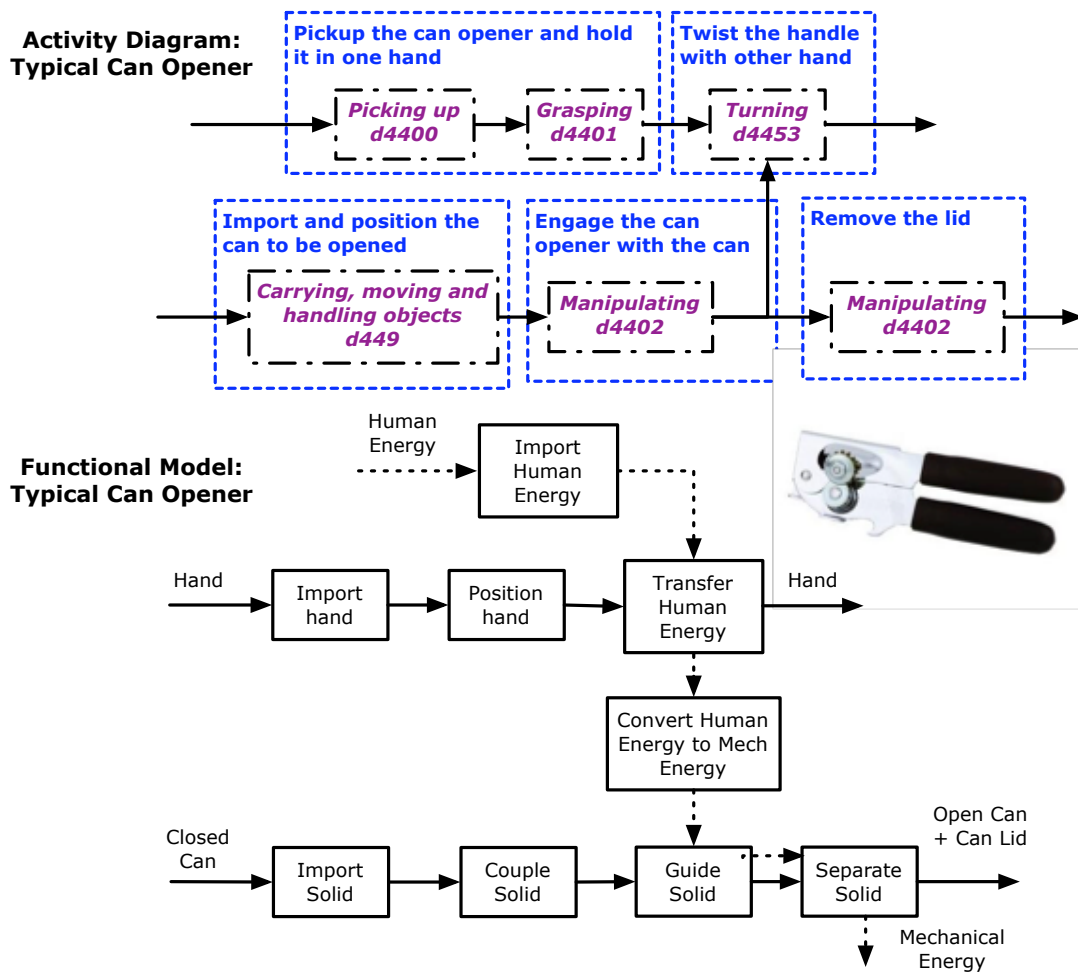


Figure 14. Activity diagram and functional model for a typical can opener

In an inclusive can opener, the user grabs the electric opener and places it on the lip of the can. The activity of grabbing, formally “picking up” in the ICF, is similar in both designs. To activate the electric opener, the user has to press the switch with the same hand in which they are holding the opener. Thus, the complicated user activity of twisting the handle with a second hand and removing the lid, which was observed in typical can opener, is eliminated in the inclusive version. The electric opener *imports* and *stores electrical energy* as well as *supplies* it when the switch is activated. A motor

converts electric energy to rotational energy, which moves the can relative to the opener. As the can rotates, a stationary point cutter cuts through the lid. A magnet lifts up the lid after cutting.

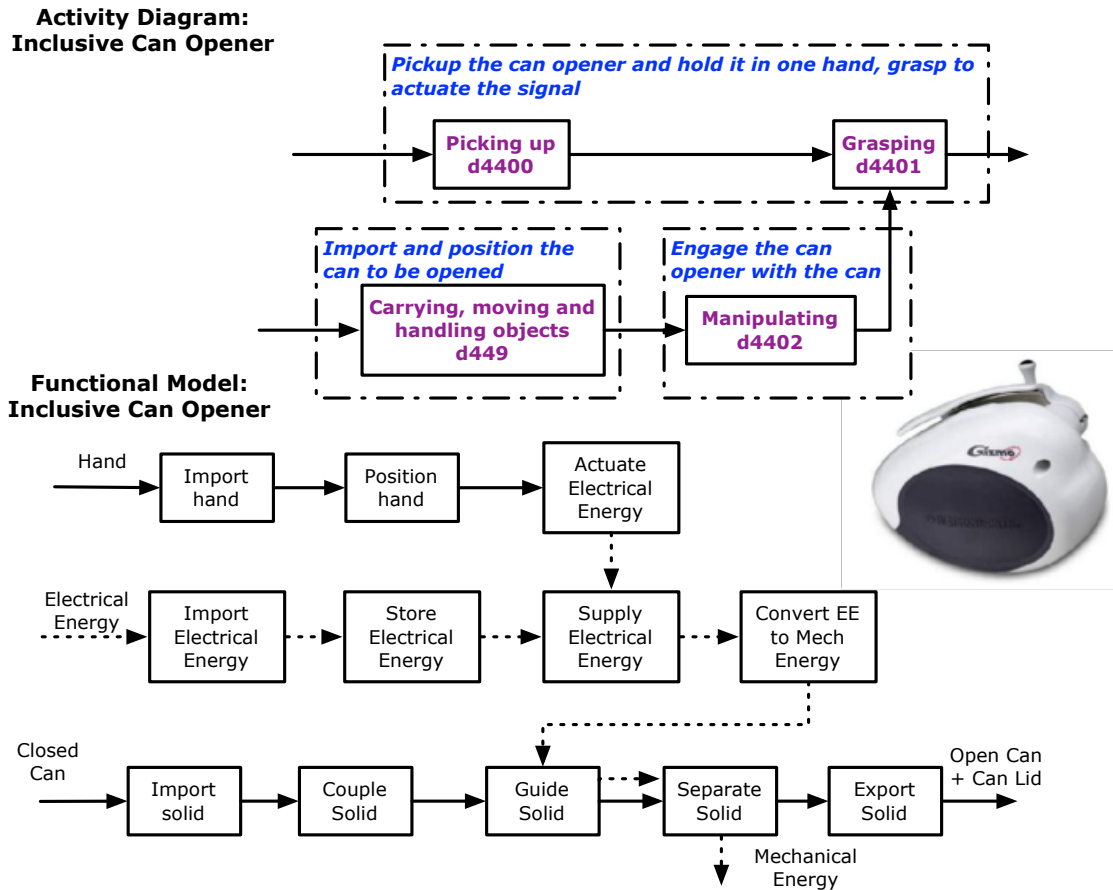


Figure 15. Activity diagram and functional model for an inclusive can opener

The activity diagram is merged with the corresponding functional model to form an actionfunction diagram, as explained in chapter II. The actionfunction diagrams of a typical can opener and an inclusive can opener are compared to study the similarities and

the differences in Figure 16. The typical can opener relies on human energy. The inclusive can opener uses electrical energy for several functions. Thus, the functions like *import human energy*, *transfer human energy*, and *convert human energy to mechanical energy* are deleted in the inclusive design. The functions added in the inclusive design are *import electrical energy*, *store electrical energy*, *supply electrical energy*, and *actuate electrical energy* as well as *convert electrical energy to mechanical energy*. The magnet adds the functionality of *exporting* the lid away from can. The rotary cutter is morphologically different from the stationary point cutter. Both openers offer morphologically different provisions for “grasping” with hand.

After comparing the actionfunction diagrams for the can openers, the next step is to tabulate the user activities and product functions, as shown in Table 2. For example, the user activity of “picking up” the can opener corresponds to the product function of *import hand* into the system, which remains the same for both the designs. The *export solid* function performed by the magnet to lift the lid, shown on the last row of Table 2, is an addition of functionality. The *export solid* function has no associated activity in the inclusive design. However, the typical design requires the user to remove the lid by “manipulating”.

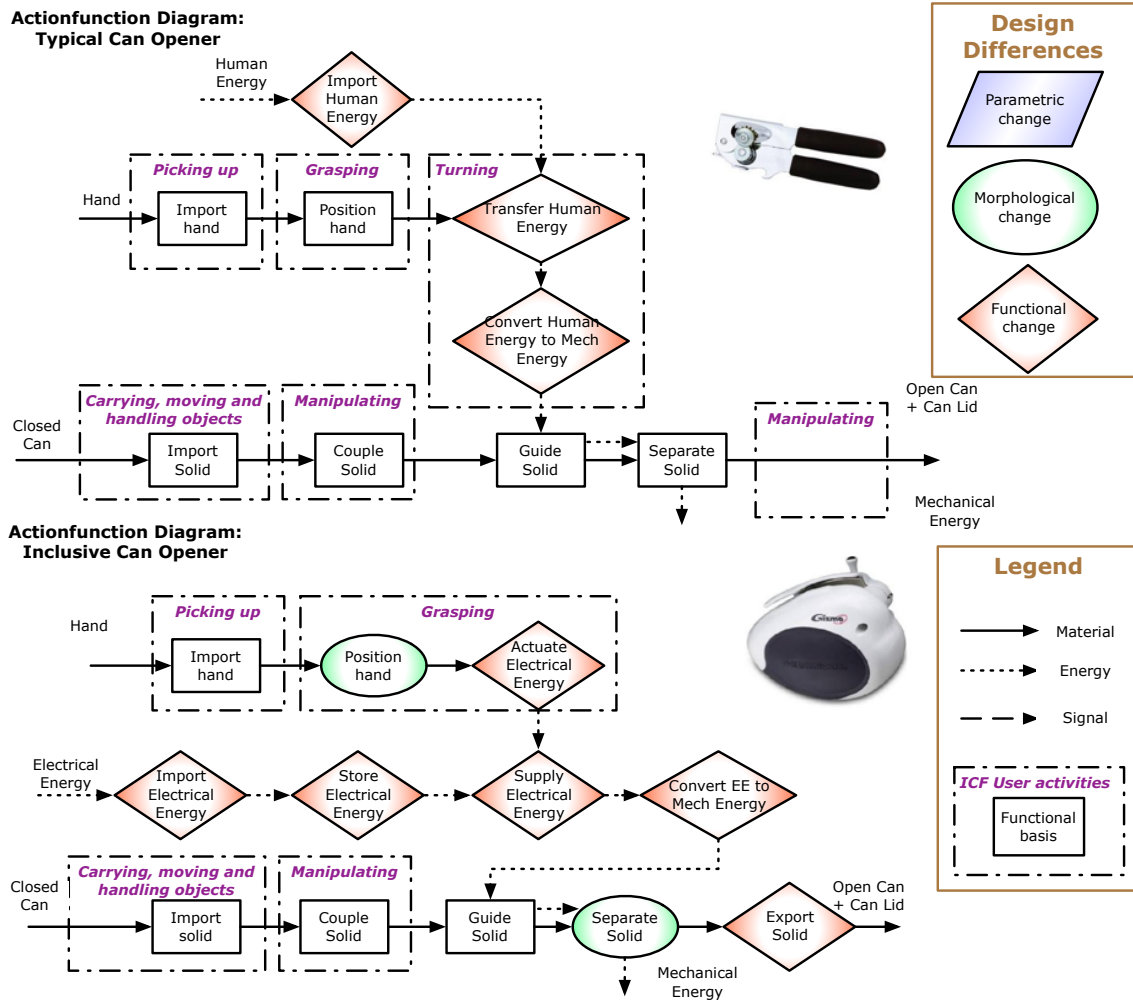


Figure 16. The actionfunction diagram for a typical can opener (top) and an inclusive can opener (bottom)

Table 2. Comparison of the actionfunction diagram of a typical can opener with the actionfunction diagram of an inclusive can opener

Product function Typical	User activity Typical	Product function Inclusive	User activity Inclusive	Change
Import Hand	Picking up	Import Hand	Picking up	None
Position Hand	Grasping	Position Hand	Grasping	Morphological
Import Human Energy	N/A	Import Human Energy	N/A	None
Transfer Human Energy	Turning	N/A	N/A	Function Deletion
Convert Human Energy to Mechanical Energy	Turning	N/A	N/A	Function Deletion
Import Solid	Carrying, moving and handling objects	Import Solid	Carrying, moving and handling objects	None
Couple Solid	Manipulating	Couple Solid	Manipulating	None
Guide Solid	N/A	Guide Solid	N/A	None
Separate Solid	N/A	Separate Solid	N/A	Morphological
N/A	N/A	Actuate Electrical Energy	Grasping	Function Addition
N/A	N/A	Import Electrical Energy	N/A	Function Addition
N/A	N/A	Store Electrical Energy	N/A	Function Addition
N/A	N/A	Supply Electrical Energy	N/A	Function Addition
N/A	N/A	Convert Electrical Energy to Mechanical Energy	N/A	Function Addition
N/A	Manipulating	Export Solid	N/A	Function Addition

The aim of this exercise is to track the changes in the product function as the product becomes more accessible. Changes in the user activity, if any, are interesting as well. Sixty-five product pairs are selected and compared for this study; the database of all the product pairs is available online at www.prosedesign.org [87]. In a similar manner, the similarities and differences in all product pairs are tabulated for the study.

Summary

This chapter presents a function-based framework for comparison of typical and inclusive products through an actionfunction diagram. The method serves as elemental building blocks to capture the characteristics of inclusive design. The classification of design differences as parametric, morphological, or functional in an actionfunction

diagram is a structured way to compare typical and inclusive products in the context of product pairs. The 65 product pairs listed in the Appendix D provides inclusive design repository for researchers and designers. Further, Chapter IV, Chapter V, Chapter VI, and Chapter VII describes the computational methods used to derive heuristics for inclusive design from the existing products.

CHAPTER IV

ASSOCIATION RULE MINING ³

As more product pairs are studied, a formal method to seek patterns in the data is needed. Association rule learning is a data mining technique to find associations in large datasets. The data contains $I = \{i_1, i_2, \dots, i_n\}$ which is a set of categorical attributes called *items* and $T = \{t_1, t_2, \dots, t_n\}$ which is a set of *transactions*. Each transaction in T contains a subset of the items in I . An association rule is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. The itemsets X and Y are called the antecedent and consequent of the rule, respectively.

Market basket analysis serves as a simple example of association rule learning. The items are goods available at a supermarket and a transaction is what a single shopper puts into their shopping basket for purchase. Mining through a large number of transactions, an association rule might determine that there is a strong association between an itemset of peanut butter and jelly and an itemset of bread: peanut butter and jelly is an antecedent for bread as a consequent.

In association rule learning, support (*supp*) of an itemset X is the percentage of transactions in T that have a specific itemset X , and confidence (*conf*) is the percentage of transactions in T such that, given they contain X , also contain Y . Formally,

³ Reprinted with permission from "User Activity – Product Function Association Based Design Rules for Universal Products " by Sangelkar, Shraddha, Cowen, Nicholas and McAdams, Daniel A., 2012. Design Studies, 33 (1), 85-110, Copyright 2012 by Shraddha Sangelkar.

$$\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y) / \text{supp}(X) \quad (1)$$

Generating association rules from a database requires an algorithm to search and extract rules typically based on a user supplied minimum level of support and confidence [34]. Several algorithms have been developed and are available for mining association rules from datasets, for example, Apriori, Eclat, and FP-Growth [88, 89]. The Apriori algorithm, as implemented in TANAGRA software, is employed to extract association rules in the initial exploration stages [34, 90].

Mining Association Rules for Inclusive Design

The product pair comparison is explained in the Chapter III. Thoroughly analyzing product pairs with actionfunction diagrams and recording design changes generates the dataset of transactions. This chapter explains the generation of association rules from this data. Association rule generation consists of three steps: pre-processing input, running the Apriori algorithm, and post-processing the output. Fifteen inclusive product pairs are selected for the initial exploration. The overall functionality of the fifteen product pairs is diverse in applications. The products selected for the initial exploration include an arm chair, a box cutter, a can opener, a car ingress and egress system, a pruner, voice activated dialing, a chopping bowl, vehicle drive controls, a recliner, a pair of scissors, signage, an auto lifting toilet-seat, a remote control, an adjustable height sink, and food storage containers.

Pre-processing is an essential step to obtain accurate association rules [32]. Essentially, the data is cleaned. For example, the TANAGRA software is case sensitive, so typographical errors must be corrected. The pre-processed data is converted into an

input file compatible with TANAGRA software. Running the algorithm consists of importing data, selecting input items, and choosing the appropriate algorithm for rule generation. Input items are product functions, user activities in the typical products, changes in the product from typical to inclusive, and the user activities in the inclusive products.

The user activity and product function items of “grasping” and *position hand* serve as an example of candidate rules and their associated measures of support and confidence. There are 140 transactions in the dataset. The items “grasping” and *position hand* occur together thirteen times in the typical products studied. Further, comparison of the typical product to the inclusive product in each pair and mining of the data shows that 6 out of 13 transactions suggest a parametric change. The other 7 transactions suggest no change, functional change, or a morphological change. Thus, support of the rule – $(Position\ Hand, Grasping) \rightarrow (Parametric, Grasping)$ is $6/140$, or 4.3%. Confidence of the same rule is $(6/140)/(13/140)$, or 46%.

Generally, values of minimal support and confidence are used to focus the search on strong or interesting rules. In the study here, product pairs selected are diverse and contain a varied set of transactions. Thus, establishing criteria to set specific values for support and confidence that produce an interesting or a broadly applicable result is problematic. For instance, the dataset contains only a single transaction in which a morphological change to a *guide solid* function is suggested to change a user activity from “pushing with lower extremity” to “pushing”. The specific case is a design change in automobile controls that allow the user to operate either with the lower or with the

upper limb. For this transaction, the support is $1/140$, or 0.7% , and the confidence is 100% .

Though a single transaction occurrence is not generally of interest in the sense that it does not suggest a strong rule, it is chosen to over explore the potential rule space in this research to find a greater number of strategies for inclusive design. Thus, the minimal support is set to low value in this search. As a result, the minimum support level was set at 0.5% . A low value of support is admissible as it implies statistical significance [91]. In order to extract a large number of rules, the minimum confidence level was set at 40% . Thus, the Apriori algorithm is set to generate rules with a minimum confidence level of 40% , and minimum support level of 0.5% .

The last step in association rule generation is post-processing of the output. The algorithm outputs all possible combinations of the same items in a transaction. To avoid duplicate rules, repetitive rules are ignored. For the data searched here, the algorithm outputs 104 rules that are reduced to 64 rules. The method for elimination and reduction to the 64 rules is discussed next. The post-processing step is similar to data transformation in Shahbaz [32].

The specific interest is in the interplay between user activity, product function, and any observable design trends that make a product more accessible in that context. Thus, rules with the format of $(product\ function, user\ activity\ typical) \rightarrow (change, user\ activity\ inclusive)$ or $(PF, UA_T) \rightarrow (Change, UA_I)$ are of interest. In terms of data mining terminology, the product function (PF) and user activity of the typical product (UA_T) are the antecedent, and change in product function ($Change$) and user activity of

the inclusive product (UA_I) are the consequent of a rule. The identification and selection of rules based on the format $(PF_T, UA_T) \rightarrow (Change, UA_I)$ is done manually in the post-processing step as the algorithm treats all input items equivalently and doesn't recognize that we are only interested in the (PF, UA_T) itemsets as a antecedent. Post-processing also removes some specific details such as computational time needed to find the rule that are provided by the software. These details are not of interest at this point. The post-processed results are easily readable association rules, which are discussed below.

Results of the Association Rule Mining

The 64 association rules generated in the initial phase are listed here. The input to the Apriori algorithm consists of 140 transactions. The 64 rules are exhaustive in the sense that they are those generated by the algorithm according to the data mining criteria specified. The rules have not been subsequently evaluated, culled, and modified to improve their specific applicability and impact for inclusive design. A few rules are presented here in detail that includes presentation of the associated actionfunction diagram. These detailed presentations highlight both the research approach and the way in which the rule is applied to a typical product to make it more inclusive.

Table 3 lists the rules generated by the Apriori algorithm that follow the $(product\ function, user\ activity\ typical) \rightarrow (change, user\ activity\ inclusive)$ format. In some sense, these are the rules that lead to a clearly applicable design guideline: given the user activity in the typical product and product function what type of change made, at the functional level, improves product accessibility and what is the user activity in the

resultant inclusive design? For the rules listed in Table 3, the value of confidence is above 40 %. A column is added that includes an assessment of the impact of the change in user activity compared to the typical design. The rules are arranged based on values of support and confidence, with stronger rules first.

The ‘Activity Change’ column in Table 3 represents the change in user activity in the inclusive product. In several cases, a user activity is deleted in concert with a function deletion. For example the activity of “turning” is removed along with the function *convert human energy to mechanical energy* in the inclusive can opener: the addition of electrical power to the inclusive can opener eliminates this usage for human energy. In several cases, a user activity is added in inclusive design in concert with a function addition. For example that activity of “pushing” is added with the function *actuate signal* in the PT Cruiser to allow the user to actuate the entry system with the push of a finger.

The first nine rules in Table 3 represents the case where there was a functional addition to the inclusive product. The function addition cases are typically related to automating the product. Thus, the typical product function is listed as no function in the table. In these cases, the addition of some product function improved accessibility. For example Rule 4 of Table 3 states that typical products add a *sense status signal* function to enable a user activity of “speaking”. To illustrate this rule, consider the example of an inclusive phone shown in Figure 17.

Table 3. The association rules generated by the data-mining algorithm, Apriori, for inclusive design

	Antecedent		Consequent		Activity Change	Measures	
	User Activity (Typical)	Product Function	Change	User Activity (Universal)		Confidence (%)	Support (%)
1	No Activity	No Function	Function Addition <i>Actuate Signal</i>	Pushing (fingers)	Add	100	3.6
2	Standing	No Function	Function Addition <i>Guide Human</i>	Standing	Easier	100	2.9
3	No Activity	No Function	Function Addition <i>Import Hand</i>	Reaching	Added	100	2.1
4	No Activity	No Function	Function Addition <i>Sense Status Signal</i>	Speaking	Added	100	0.7
5	Grasping	No Function	Function Addition <i>Store Solid</i>	No Activity	Deleted	100	0.7
6	Maintain body position	No Function	Function Addition <i>Position Solid</i>	Maintain sitting position	Easier	100	0.7
7	Sitting	No Function	Function Addition <i>Guide Human</i>	Sitting	Easier	100	0.7
8	No Activity	No Function	Function Addition <i>Actuate Electrical Energy</i>	Grasping	Added	100	0.7
9	Manipulating	No Function	Function Addition <i>Export Solid</i>	No Activity	Deleted	100	0.7
10	Turning	Convert Human Energy to Mechanical Energy	Function Deletion	No Activity	Deleted	100	0.7
11	Communicating - receiving (signs)	Indicate Status	Morphological	Communicating - receiving (signs and Braille)	Easier	100	0.7
12	Pushing with lower extremity	Guide Solid	Morphological	Pushing (hand)	Easier	100	0.7
13	Turning	Guide Solid	Morphological	Pushing (hand)	Easier	100	0.7
14	Pulling (hand)	Separate Solid	Morphological	Pushing/Pulling (hand)	Easier	100	0.7
15	No Activity	Separate Solid	Morphological	No Activity	No	100	0.7
16	Manipulating	Transfer Human Energy	Morphological	Manipulating	No	67	1.4
17	Transferring oneself while sitting	Import Human	Morphological	Moving Around	Easier	50	0.7
18	Transferring oneself while sitting	Position Human	Morphological	Moving Around	Easier	50	0.7
19	Transferring oneself while sitting	Position Human	Morphological	Transferring oneself while sitting	Easier	50	0.7
20	Turning	Transfer Human Energy	Function Deletion	No Activity	Deleted	50	0.7
21	Manipulating	Couple Solid	Parametric	Manipulating	Better	50	0.7
22	Grasping	Position Hand	Parametric	Grasping	Better	46	4.3
23	Manipulating	Guide Solid	Morphological	Pushing (fingers)	Easier	40	1.4

Actionfunction diagrams for a typical and inclusive phone are illustrated in Figure 18. The voice activated dialing feature is captured by the functions *sense signal status* and *convert status to control* in the inclusive phone. The user activity of

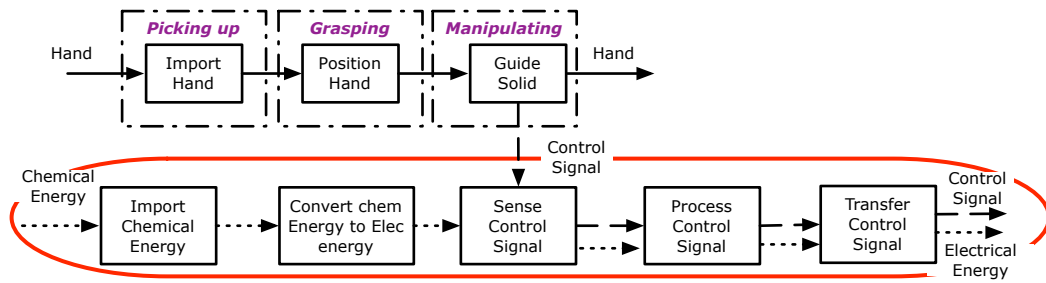
“speaking” is added. Thus, a user who has difficulty with the activity of “manipulating” need not do so. Note that the manual dialing functions are not removed as the user can place a call either by pressing keys or by speaking.



Figure 17. A typical phone on left and an inclusive phone with voice activated dialing on right [92]

In cases of parametric function change, such as Rule 21 and 22 in Table 3, the user activity remains the same but the activity is made comparatively easier in inclusive design. For instance, a parametric change to the function *couple solid* makes “manipulating” easier or a parametric change to function *position hand* makes “grasping” easier.

**Actionfunction
Diagram: Typical**



**Actionfunction
Diagram: Universal**

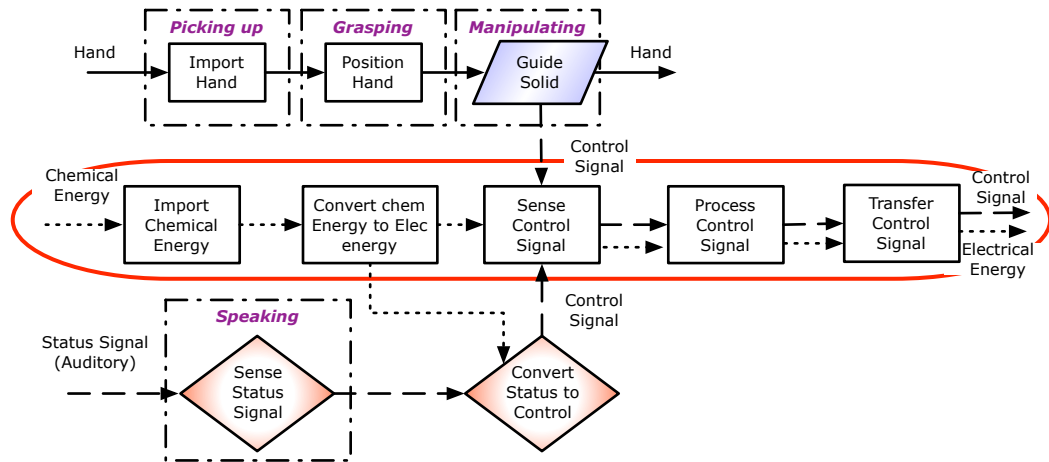


Figure 18. Actionfunction diagram of a typical phone (top) and inclusive phone (bottom). The internal functions that require ‘no change’ are encircled

Also found through data mining are rules suggesting no change, no change to internal functions, and the addition of internal functions. These rules are presented here in distinct categories for discussion. Table 4 states the rules that require no change for inclusive design. These rules have confidence above 65 %. These no change rules are primarily related to user-product interactions that do not represent the most opportune avenues to improve accessibility.

Table 4. List of rules that suggest ‘no change’ to the user - product interaction

	Antecedent		Consequent		Measures	
	User Activity (Typical)	Product Function	User Activity (Universal)	Activity change	Confidence (%)	Support (%)
1	Carrying, moving and handling objects	Import Solid	Carrying, moving and handling objects	No	100	2.9
2	Reaching	No function	Reaching	No	100	2.9
3	Sitting	Import Human	Sitting	No	100	2.1
4	Sitting	Position Human	Sitting	No	100	2.1
5	Manipulating	Store Solid	Manipulating	No	100	0.7
6	Moving around building other than home	Import Human	Moving around buildings other than home	No	100	0.7
7	Moving around home	Import Human	Moving around home	No	100	0.7
8	Picking up	No function	Picking up	No	100	0.7
9	Manipulating	No function	Pulling	Yes	100	0.7
10	Pulling (hand)	Transfer Human Energy	Pushing/Pulling (hand)	Yes	100	0.7
11	Picking up	Import Hand	Picking up	No	88	5.0
12	Manipulating	Separate Solid	Manipulating	No	67	1.4
13	Reaching	Import Hand	Reaching	No	67	1.4

Rule 1 in Table 4 shows that even as products are designed differently to be inclusive, no changes were made to address the user activity “carrying, moving, and handling objects” and product function *import solid*. Figure 19 shows a typical wine opener on the left and an automatic wine opener marketed as inclusive on the right. Figure 20 illustrates the actionfunction diagram of a typical wine opener (top) and an inclusive wine opener (bottom). The wine opener needs to import the closed wine bottle. The user imports the bottle into the opener. In both the typical and the inclusive designs the user imports the bottle, hence the function *import solid* remains the same.

Rule 10 in Table 4 implies that the housing of both the typical and inclusive product transfers human force. The function of *transferring human energy* results in no change to achieve the inclusive design. However, the related user activities are not the same. When the user activity is shared with two or more product functions the activity

might change due to a morphological or parametric change to one function even though the other function remains the same.



Figure 19. Automating a wine opener eliminates the need for twisting the hand [92]. Typical wine opener is shown on the left and the automatic wine opener is shown on right

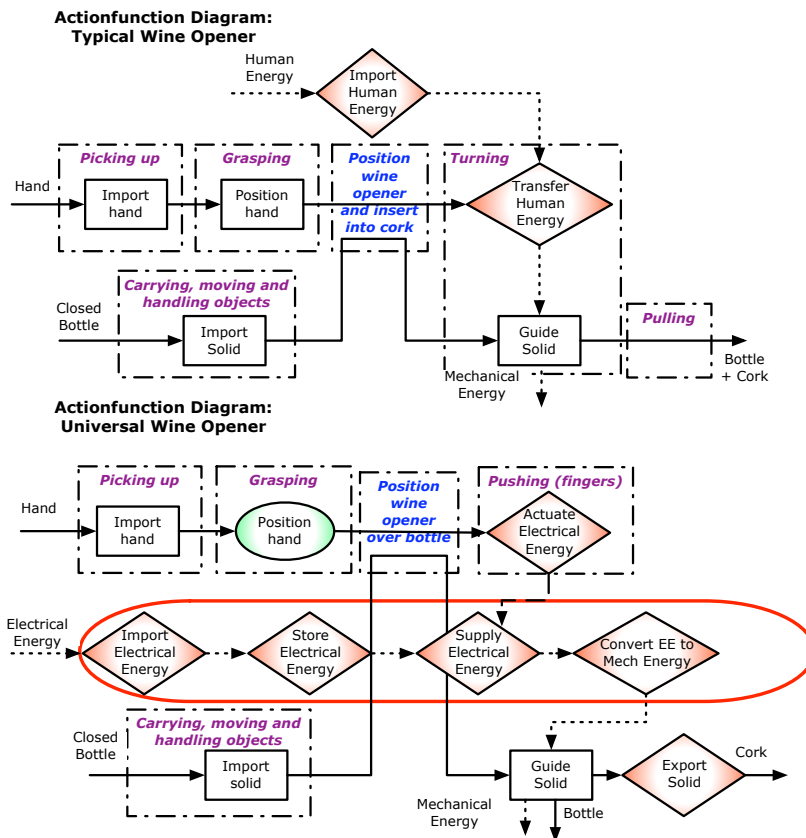


Figure 20. Actionfunction diagram of a typical wine opener (top) and inclusive wine opener (bottom). The addition of internal functions is circled for emphasis

Table 5 lists the internal functions for which no change is suggested as the product design is refocused from typical to inclusive. These rules have confidence above 65 %. Here, the notion of internal is used to denote functions that do not have a specific user activity interaction. Based on the product data set sampled, the functions required to *import* and *convert energy*, or *sense*, *process* and *export signal* require no change. Additionally, functions that *import*, *store*, *guide*, *regulate*, or *transfer material* to meet the main functionality of the product do not appear crucial in designing an inclusive product.

Table 5. List of the internal functions of products that requires ‘no change’ for inclusive design

	Antecedent	Consequent	Measures	
	Product Function	Change	Confidence (%)	Support (%)
1	Import Chemical Energy	None	100	2.1
2	Convert Chemical Energy to Electrical Energy	None	100	1.4
3	Export Control Sign	None	100	0.7
4	Export Liquid	None	100	0.7
5	Export Signal	None	100	0.7
6	Guide Liquid	None	100	0.7
7	Import Liquid	None	100	0.7
8	Import Solid	None	100	0.7
9	Process Control Signal	None	100	0.7
10	Process Signal	None	100	0.7
11	Regulate Liquid	None	100	0.7
12	Sense Control Signal	None	100	0.7
13	Sense Signal	None	100	0.7
14	Store Liquid	None	100	0.7
15	Transfer Liquid	None	100	0.7
16	Import Human Energy	None	75	4.3
17	Guide Solid	None	67	1.4

Table 6 lists the rules that recommend the addition of an internal function for improving the accessibility of a product. The listed rules have confidence greater than 65 %. Table 6 suggests that most of the inclusive products require the addition of internal functions like *importing, transferring, storing, or supplying energy*. Furthermore, the external energy imported, either electrical or chemical in the form of fuel or a battery, needs to be transformed into another form of energy.

Table 6. Rules suggesting addition of internal functions to improve product accessibility

	Antecedent	Consequent	Measures	
	Product Function	Change	Confidence (%)	Support (%)
1	Convert Electrical Energy to Mechanical Energy	Function Addition	100	2.9
2	Import Electrical Energy	Function Addition	100	2.1
3	Transfer Electrical Energy	Function Addition	100	1.4
4	Convert Human Energy to Hydraulic Energy	Function Addition	100	0.7
5	Convert Status Signal to Control Signal	Function Addition	100	0.7
6	Store Electrical Energy	Function Addition	100	0.7
7	Store Hydraulic Energy	Function Addition	100	0.7
8	Supply Electrical Energy	Function Addition	100	0.7
9	Supply Hydraulic Energy	Function Addition	100	0.7
10	Supply Chemical Energy	Function Addition	100	0.7
11	Convert Chemical Energy to Mechanical Energy	Function Addition	67	1.4

The algorithm outputs 12 rules that suggest the addition of internal functions. Evaluation of these suggested function additions show them to be consistent with high-level strategies for inclusive design. For example, rule 5 in Table 6 states that a user status signal in visual, tactile, or auditory form should be converted to control signal compatible with the device. The addition of such a conversion feature to a product allows the user to input a signal in more than one form, thus making it inclusive.

Summary

This chapter explores the applicability of data mining techniques to improve inclusive design of products. Particularly, association rule mining is applied to extract guidelines for inclusive design based on product function, user activity, and the changes to the product function to improve the accessibility. The Apriori algorithm generates association rules quickly and efficiently.

Generally, association rule mining is applied to dataset with large number of items in each transaction. The rules are mined based on the minimum value of support and confidence. The rules may or may not have all the items from a transaction. Also, there is no restriction on the items that constitutes the antecedent or items that constitutes the consequent. Thus, the rules generated in this research may appear as weak association rules. Here, we dictate the item that needs to be part of either the antecedent or the consequent as we are searching for a specific causality. Also, rules with low values of confidence and support are considered. The reason for setting low values of minimum support and confidence is the size and diversity of the data set studied. In addition, the rules with low support are interesting cases for inclusive design of products.

Nevertheless, association rule mining allows efficient analysis of data. Importantly, association rule mining is able to produce rules that suggest design changes in the context of user activity and product function interplay and thus are applicable during the design process. Subsequent chapters explore other data mining techniques and more robust algorithms for extraction of inclusive design guidelines. Better software allows flexibility in the input format and reduces the amount of manual post-processing required.

CHAPTER V

FORMALIZING AND EXPLORING THE TRANSFERABILITY OF THE RULES⁴

The initial exploration demonstrates the potential of association rule mining for generating inclusive design rules based on user-product interactions. The interaction between the user and a product as embodied in the actionfunction diagrams gives valuable information for serving the needs of individuals with a disability. This chapter extends the association rule mining to study a larger dataset consisting of 65 product pairs. The set of rules obtained from the initial exploration of 15 product pairs are compared to the rules obtained from current dataset of 65 product pairs.

The main purpose of this chapter is to extend and formalize the application of state-of-the-art Apriori algorithms for inclusive design rule generation. The data analysis procedure to obtain a set of inclusive design rules is formalized. A formal approach allows repeatable mining of rules such that each time more product pairs are added to the product dataset, the set of design rules can be updated. Particularly, post-process filtering of the rules generated by the Apriori algorithm is formalized to capture the information

⁴ Reprinted with permission from "Formalizing and Exploring the Transferability of Inclusive Design Rules," by Sangelkar, Shraddha, and McAdams, Daniel A., 2013. *Journal of Mechanical Design*, in print, Copyright 2012 by Shraddha Sangelkar. And "Formalizing User Activity- Product Function Association Based Design Rules for Universal Products" by Sangelkar, Shraddha, and McAdams, Daniel A., 2011, *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Washington, DC, Copyright 2012 by Shraddha Sangelkar.

pertinent to inclusive design. The post-process filtering facilitates the application of the widely implemented Apriori algorithm for mining rules from inclusive design data.

Additionally, this chapter aims to determine if a tractable, or perhaps asymptotically finite, set of inclusive design rules can be captured by studying a larger and varied dataset of product pairs. Further in this work, the rules are analyzed in detail to evaluate their potential for transferability and reuse from one product to another. Of particular interest is the transferability of the rules across apparently disparate product domains. The conceptual and physical similarity of the rules is discussed in the results. As a compliment to the broad applicability or transferability of the discovered rules, some investigation of clustering products are presented. Products are clustered based on the association rules that are applicable to them. These product clusters represent an opportunity to form an inclusive product family. The commonality in the clustered product pairs has the potential to serve as a platform of inclusive elements. The next part explains the data generation and data analysis.

Rule Generation

An overview of the research activity is depicted in Figure 21. In the first step, the inclusive design information is generated from the existing inclusive products with the process explained in the Chapter III. In the second step, the data generated in the first step is mined to find rules applicable for inclusive design. The actionfunction diagrams of 65 product pairs are the input for the data analysis step. Data analysis consists of three steps: pre-processing, association rule mining, and post-processing.

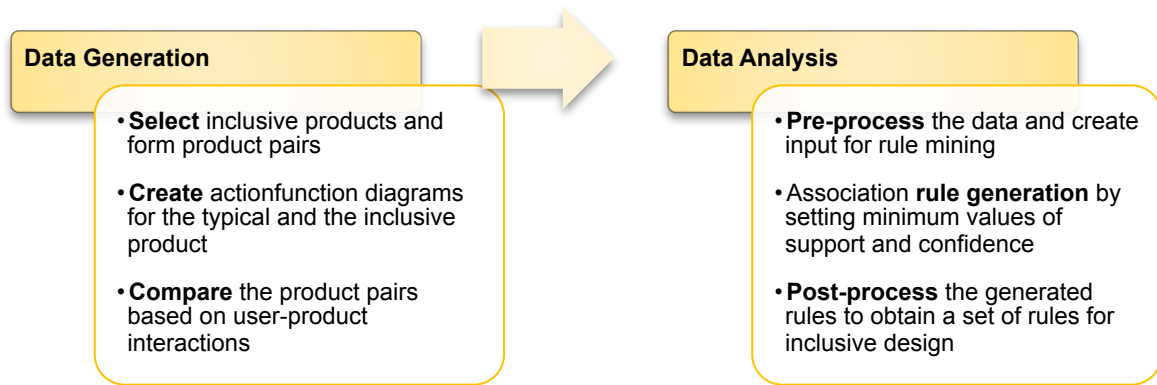


Figure 21. Rule generation process to study the inclusive design characteristics

Pre-Processing

The comparison data obtained from the actionfunction diagrams of all the product pairs is tabulated in the pre-processing step. A candidate function set is a product function in a typical product along with the corresponding function in its inclusive counterpart, which is a possible candidate for a design change that leads to an inclusive product. A user activity may or may not be associated with a candidate function set. Each row of the table corresponds to a candidate function set. Each candidate function set acts as an input transaction to the Apriori algorithm. The items recorded in a transaction are product functions and user activities in the typical and the inclusive products in addition to the type of change in the product function.

The product pairs studied in this chapter cover a broad range of complexity. For example, a product pair such as a typical and inclusive washer with comprehensive actionfunction diagrams result in a large number of input transactions to the data set. On the contrary, a simpler product pair like a shovel has fewer input transactions. The

average number of transactions per product pair for the data studied here is 8. The number of transactions in a product pair ranges from 3 to 16 in this dataset.

Association Rule Mining

Pre-processed product function and user activity data is input to the association rule mining algorithm. The Apriori algorithm requires the user to select minimum threshold values for the support and confidence. If the minimum value of support and confidence is set to a high threshold, say 5% support and 90% confidence, the rules obtained contain only the most frequent itemsets and all permutations of those itemsets. For instance, the importing functions for energy, material, and human are most frequent in the dataset. Hence, only the association rules pertaining to importing functions are mined by setting high threshold values. For this study, the minimum values of support and confidence are set to a low threshold to discover as wide a range of rules and results as possible. Low values of support and confidence allow mining of rules over a wide range of activities and functions, thus giving better insight into inclusive design as established in Chapter IV.

Consider that an antecedent A leads to two different consequents, B and C . Two possible rules are $A \rightarrow B$ and $A \rightarrow C$, and the confidence of each rule is 50%. Due to a variety of product pairs in the dataset, there are instances of the same antecedent leading to two different consequents. Such rules with 50% confidence gives the product designer a choice between two consequents, given the antecedent. For example, either a parametric or morphological change to the product function of *indicate status* can make the activity of “seeing” more inclusive. A designer can choose between the two rules for

the preferred solution to address all customer requirements during the conceptual stages of design. Therefore, the minimum value of confidence is set to 50% for this study. Though a single transaction occurrence is not generally of interest in the sense that it does not suggest a strong rule, it is chosen to over explore the potential rule space to find a greater number of strategies for inclusive design. The minimal support level is set at 0.15% ($<1/523$), such that the algorithm mines itemsets that occur one or more times in the dataset.

Post-Processing

The Apriori algorithm is set to generate rules with a minimum confidence level of 50% and minimum support level of 0.15%. A negative of setting low values of confidence and support is that the algorithm mines 13,300 rules from 523 transactions. Hence, the rules require further post-processing to obtain a set of rules with the desired order of itemsets in the antecedent and consequent which is explained further.

In this research, we are interested in the interplay between user activity, product function, and any observable design feature that makes a product more inclusive. In the initial stages of design, the only information available to a product designer is the product function, the user activity, and the associated user-product interaction from the actionfunction diagram of a typical product. The designer is confronted with the question, what change to one or more product functions would make the product more inclusive? In other words, given the user activity and product function in typical design (antecedent), what type of design change (consequent) in the product function would make it inclusive? This method aims to formalize design rules that would be based on

the user-product interactions of the typical design. The design rules would suggest a change in attribute or characteristics of the function in typical product that would possibly yield an inclusive product.

Apriori algorithm cannot be programmed to mine a particular format of rules that are needed for inclusive design. What the Apriori algorithm mines instead is explained here. The Apriori algorithm generates different permutations of the same itemsets. The algorithm treats all types of itemsets, even those with different headers, equally for rule mining. A header is the general term describing the itemset; for example, ‘product function in typical’ is the header for the itemset ‘position hand’. The Apriori algorithm, as it stands, cannot be programmed to generate rules with specified headers in the consequent and the remaining headers in the antecedent. This limitation tends to produce invalid design rules and will be explained in the subsequent paragraphs.

Apriori cannot be customized to find rules such that the antecedent contains the information only from the typical product while the consequent contains the information about the inclusive version of product or the type of change. An example of rules mined by the Apriori algorithm containing the same itemset but with different permutations is given below.

1. (“Product function in typical” = ***Position Hand***, “User activity in typical” = ***Grasping***)
→ (“Change” = ***Parametric***, “User activity in inclusive” = ***Grasping***, “Product function in inclusive” = ***Position Hand***)
2. (“Change” = ***Parametric***, “User activity in typical” = ***Grasping***)
→ (“Product function in typical” = ***Position Hand***, “Product function in inclusive” = ***Position Hand***)
3. (“Product function in typical” = ***Position Hand***, “User activity in typical” = ***Grasping***)
→ (“Change” = ***Parametric***)

4. (“Change” = *Parametric*)
→ (“User activity in inclusive” = *Grasping*)
5. (“Product function in inclusive” = *Position Hand*)
→ (“Product function in typical” = *Position Hand*)

The first rule contains the type of change in the consequent, which is useful, but it also contains additional information stating the function and activity of the inclusive design in the consequent. A parametric type of design change, by definition, must not change the activity or function of the inclusive design. Hence, the activity and the function of the inclusive design in the consequent of first rule are both redundant. The second rule and the fourth rule contain the change as the antecedent, which is not particularly useful for inclusive design. The fifth rule presents incomplete information. The rules shown in the example above contain essentially the same information but in a different sequence. However, the third rule contains all the information that is sought for inclusive design.

The rules given by the Apriori algorithm require further post-processing to eliminate the repetitive information given by different permutations of the same itemsets. In addition, the rules are filtered such that they contain the information about a typical design in the antecedent and the type of change in the consequent. If the change made to a function in a typical design leads to an activity change in the inclusive product, then activity change must be indicated in the consequent. To produce rules in the desired format for designers, filters are created to post-process the rule set as generated by the Apriori. All the rules are discussed in the results part; here specific example are provided to explain the filters.

Filter 1

To be a rule with a parametric change, no change, or a function deletion change, the antecedent must contain the product function and the user activity from the typical design. Furthermore, the consequent is the change advocated by the rule. The user activity and the product function remains unchanged for the inclusive design in the case of a parametric change or no change. For the rules stating deletion of a function, both the user activity and the product function are deleted in the inclusive design.

The generic form of the rule is (*Product function in typical, User activity in typical*) → (*Change*), where *Change* = *Parametric or No change or Function deletion*. For instance, the user-product interaction of “grasping” and *position hand* is made inclusive by parametric improvements to the handle design; this rule is stated as *position hand* + “grasping” → Parametric. The ergonomic handle design in the OXO good grips products, such as peelers, mashers, and garlic press, makes the tool convenient for people with limited hand dexterity. Such an ergonomic handle can be added in the design of other hand-held devices.

An example of a no change type of rule would be, *guide liquid* + “no activity” → no change; which means that no change is required to product function *guide liquid* when no user activity is associated with it for a product like a kitchen sink. The shape of the sink bowl guides the water out into drainage. The rules suggest that no change is required to the shape of sink bowl to make it inclusive.

An example of a function deletion type change would be *transfer human energy* + “turning” → function deletion; which suggests that the function of *transferring human*

energy is deleted in the inclusive design along with the *turning* activity. In a manual can opener the user turns the handle to cut open the can; this turning activity is deleted in an electric can opener.

Filter 2

To be a rule with a morphological change, the antecedent must contain the user activity and product function of the typical design. The type of change and user activity in the inclusive design is sought as the consequent of the rule. Specifying the user activity in the consequent helps to keep track of the changes in user activity, if any. In most morphological changes, the user activity remains the same but performing that activity is now easier for the user. However, we are particularly interested in the activity change for the morphological design change as the activity in the antecedent may be an activity that a user with a disability is unable to perform.

The generic form of the rule is (*Product function in typical, User activity in typical*) → (*Change + User activity in inclusive*), where *Change = Morphological*. Consider the rule, *Indicate status* + “communication using written” → morphological + “communication using Braille.” This rule suggests augmenting the written text with braille script to assist a blind user. In this rule, the user activity changes from typical to inclusive design by incorporating morphological change to the function *indicate status*.

Filter 3

To be a rule with the change of a function addition, the antecedent must contain the user activity from the typical product and product function from the inclusive product. The consequent must be the type of change and the user activity in inclusive

product. The difference in the format for the rules with function addition is because; the function addition change does not have a product function in the typical design. The functions are added in the inclusive design; hence, the product function from the typical design cannot serve as the antecedent. Rules with function addition indicate what functions, if added, make the product inclusive.

The generic form of the rule is (*Product function in inclusive, User activity in typical*) \rightarrow (*Change +User activity in inclusive*), where *Change = Function addition*. The rule, *import electrical energy* + “no activity” \rightarrow function addition, indicates that the addition of the function to *import electrical energy* makes a product inclusive.

In the same context, rules providing incomplete information are filtered out. For instance, the rule (“Change” = Parametric) \rightarrow (“User activity in inclusive” = Grasping) does not give sufficient information for inclusive design. The antecedent of the rule does not specify the function being addressed for a parametric change, nor is the typical design expressed in the antecedent.

A code is executed in MATLAB to execute these three filters on the association rules generated by the algorithm. The code outputs a set of function-based association rules for inclusive design in a format that is easy to comprehend. The post-processing step also eliminates the unnecessary information given by the Apriori algorithm such as number of cycles performed or the computation time. The code can also be tuned to select only those rules with high values of confidence and support, thus allowing better statistical control on the association rules filtered.

Comparison with Initial Exploration

In Chapter IV, the Apriori algorithm is applied to mine association rules for inclusive design from 15 product pairs. Here, the association rules obtained from 15 product pairs are compared to the rules obtained from 65 product pairs. Also in the previous chapter, the data mining software TANAGRA was applied for mining association rules and the post-processing of rules was done manually [90]. In this chapter association rules are mined with WEKA software and the post-processing step is automated. The three steps of data analysis as described in the data analysis part are performed on the prior dataset of 15 product pairs to eliminate any inconsistencies and to allow comparison between the two datasets.

Table 7 compares the results obtained from the previous dataset of 15 product pairs to the results obtained from the current dataset of 65 product pairs. The number of input transactions for the dataset of fifteen products is 135 with an average of nine transactions per product pair. Similarly, the dataset of sixty-five product pairs studied here consist of 523 transactions altogether, with an average of eight transactions per product pair. The number of transactions per product pair provides an estimate of the complexity of product pairs studied. The overall complexity of products in both datasets is similar; thus it is reasonable to compare the two datasets.

The total number of association rules counts all the rules mined by the Apriori algorithm for a confidence of 50% and support that reflects a minimum of one transaction in the dataset. Filtered association rules are those rules obtained after filtering in the post-processing step. In addition, Table 7 lists the association rules with support of

more than one instance both with and without a change. Rules with a change are those recommending either a parametric, a morphological, or a functional change to the user-product interaction.

Table 7. Comparison of association rules obtained from the dataset of 15 product pairs to the dataset of 65 product pairs

		15 product pairs	65 product pairs
Number of input transactions		135	523
Total number of association rules		6969	13300
Filtered association rules		65	124
Association rules with support of more than one instance	With change	11	24
	No change	7	37

Table 8 and Table 9 list the filtered association rules for the dataset of 15 product pairs and dataset of 65 product pairs, respectively. Support of these rules is such that the consequent occurs more than once for a given antecedent. In other words, the rule is observed in at least two product pairs. The confidence of association rules listed is greater than 50%. The antecedent of a rule is the product function and the user activity while the consequent is the change in product function and change in the user activity in the inclusive design. Rules listed are only those reflecting a change in the product function; rules resulting in no change are not presented here for brevity.

Table 8. Association rules generated with minimum support of two instances suggesting changes in the product functions for the dataset of 15 product pairs

ANTECEDENT		CONSEQUENT		Confidence	Support	
Product Function	User Activity	Functional Change	User Activity Change			
1	Import EE	No Activity	Functional Addition	No Activity	100	2.96
2	Convert EE to ME	No Activity	Functional Addition	No Activity	100	2.96
3	Guide Human	Standing	Functional Addition	Same Activity but Easier	100	2.22
4	Guide Human	Sitting	Functional Addition	Same Activity but Easier	100	1.48
5	Actuate Signal	No Activity	Functional Addition	Pushing with fingers	80	3.70
6	Import Hand	No Activity	Functional Addition	Reaching	75	2.22
7	Transfer HE	Manipulating	Morphological	Same Activity but Easier	67	1.48
8	Convert CE to ME	No Activity	Functional Addition	No Activity	67	1.48
9	Guide Solid	Manipulating	Morphological	Pushing with fingers	67	1.48
10	Separate Solid	No Activity	Morphological	No Activity	60	2.22
11	Guide Solid	Pulling	Morphological	No Activity	50	1.48

Table 8 and Table 9 sort the rules in order of decreasing level of support. Eight rules are common between the two datasets as shown in Table 8 and Table 9. Sixteen new association rules are added due to the addition of 50 product pairs to the dataset. Three rules from Table 8, namely Rule 6, Rule 9 and Rule 11, are not repeated in Table 9. The reason for not finding the rules in the larger dataset is that the confidence of a rule depends on the possibilities for a consequent. With the addition of more products to the dataset, there are more possibilities for a consequent of a given antecedent. Rules that are not mined in the larger dataset do not have the same level of confidence as that of the smaller dataset.

Table 9. Association rules generated with minimum support of two instances suggesting changes in the product functions for the dataset of 65 product pairs

ANTECEDENT			CONSEQUENT			
	Product Function	User Activity	Functional Change	User Activity Change	Confidence	Support
1	Position Hand	Grasping	Parametric	<i>Same Activity but Easier</i>	69	4.21
2	Convert EE to ME	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	82	1.72
3	Import EE	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	56	1.72
4	Actuate Signal	<i>No Activity</i>	Functional Addition	Pushing with fingers	80	1.53
5	Position Solid	Carrying in Hands	Parametric	<i>Same Activity but Easier</i>	100	1.15
6	Position Hand	Reaching	Parametric	<i>Same Activity but Easier</i>	75	1.15
7	Position Human	Maintain Body Position	Parametric	<i>Same Activity but Easier</i>	75	1.15
8	Import CE	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	50	0.96
9	Position Hand	Carrying in Hands	Parametric	<i>Same Activity but Easier</i>	80	0.76
10	Indicate Status	Seeing	Morphological	<i>Same Activity but Easier</i>	67	0.76
11	Separate Solid	<i>No Activity</i>	Morphological	<i>Same Activity but Easier</i>	50	0.76
12	Secure Hand	Grasping	Functional Addition	<i>Same Activity but Easier</i>	100	1.15
13	Store EE	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	100	0.57
14	Supply EE	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	100	0.57
15	Guide Human	Standing	Functional Addition	<i>Same Activity but Easier</i>	100	0.57
16	Sense Status	<i>No Activity</i>	Morphological	<i>Same Activity but Easier</i>	100	0.57
17	Convert CE to ME	<i>No Activity</i>	Functional Addition	<i>Same Activity but Easier</i>	75	0.57
18	Transfer HE	Turning	Functional Deletion	<i>Same Activity but Easier</i>	60	0.57
19	Guide Solid	Pushing with fingers	Parametric	<i>Same Activity but Easier</i>	50	0.57
20	Position Hand	<i>No Activity</i>	Functional Addition	Reaching	100	0.38
21	Guide Human	Sitting	Functional Addition	<i>Same Activity but Easier</i>	100	0.38
22	Indicate Status	Communication Written	Morphological	Communication Braille	100	0.38
23	Position Hand	Manipulating	Parametric	<i>Same Activity but Easier</i>	100	0.38
24	Transfer HE	Manipulating	Morphological	<i>Same Activity but Easier</i>	67	0.38

Rules that are repeated in both datasets, highlighted in bold type, do not necessarily have the same values of confidence and support, since values of support and confidence depend on the total number of transactions in the dataset. The minimum value of support for the rules listed in Table 8 is $2/135$ or 1.48% . Similarly, minimum value of support for the association rules listed in Table 9 is $2/523$ or 0.38% .

Rules Set Size and Product Set Size

Of interest is the trend in rules generated as more products are added to the dataset. Specifically, does the set of association rules for inclusive design trend towards some finite number as the product set grows arbitrarily large. In some sense, a small number of rules applicable to a large set of products would indicate some core, fundamental, design needs for inclusive products. If the number of rules generated correlates directly with the growth of products analyzed, it indicates that each product must be designed uniquely to be inclusive.

To explore the relationship between rule set size and product set size, the following procedure is performed. The rule mining algorithm is set to generate rules with support of at least one instance and a minimum value of confidence greater than 50%. Data analysis is repeated on the dataset while incrementally adding product pairs to the dataset. Here, we have added any two random product pairs, without replacement, till all 65 product pairs are included. The rule generation process is repeated 5 times, each for different random order of the product pairs, to calculate the error in the number of the rules generated.

Figure 22 and Figure 23 show plots of the number of association rules mined verses the number of products in the dataset. Figure 22 shows a plot of the total number of association rules generated by the Apriori algorithm against the number of product pairs in the dataset. Figure 23 shows the number of filtered association rules, after post-processing, against the number of product pairs studied. In Figure 22 and Figure 23, the cross represents the mean and the error bar represents the standard deviation.

The trends in Figure 22 and Figure 23 show that the number of association rules generated per product pair reduces as more products are included in the dataset; the slope of the graph decreases gradually and then almost reaches a constant value. After having a considerable quantity and variety of inclusive products in the dataset, the number of association rules mined does not increase proportionately with the addition of every new product pair to the dataset. Based on this analysis, one would expect that new product pairs studied to add new rules to the rule set. But the number of rules added per product pair will be a smaller number when one product pair is added to dataset of 100 product pairs as compared to rules obtained per product pair when one product pair is added to dataset of 10 product pairs.

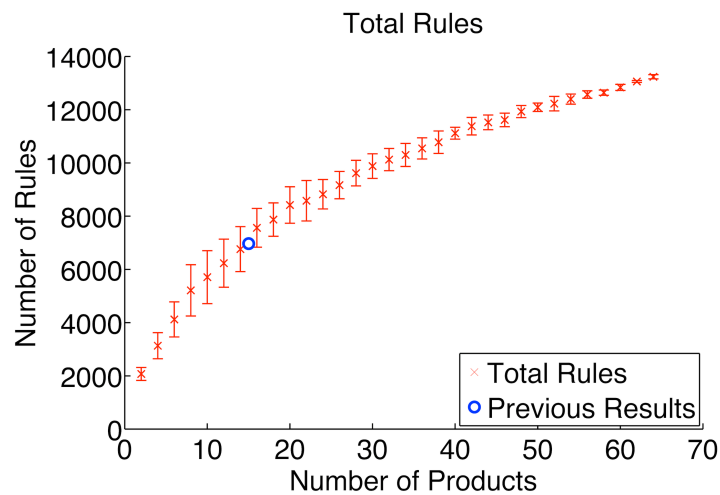


Figure 22. Total number of association rules generated by the Apriori algorithm against the number of product pairs in the dataset

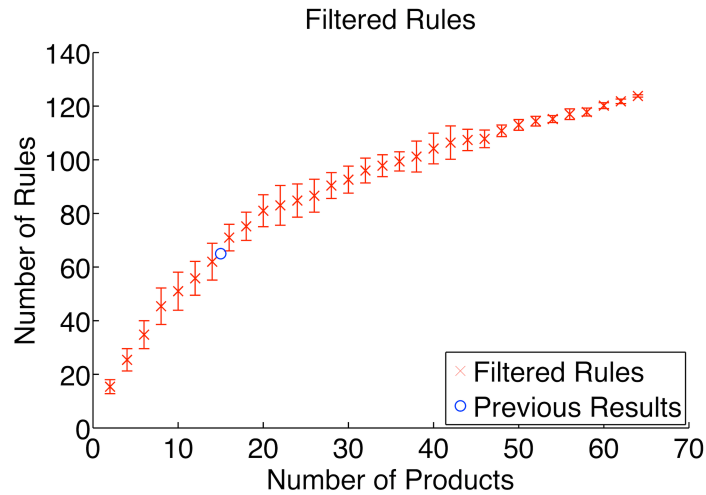


Figure 23. Number of filtered association rules after post-processing against the number of product pairs in the dataset

In this analysis, the first 15 products are not necessarily the dataset of the initial 15 product pairs. A point called “previous results” is added in Figure 22 and Figure 23 to depict the data point for the initial dataset of 15 product pairs. Though the specific rules resulting from the analysis of the two datasets may be different, the number of rules is similar indicating a common rate of rule generation for different set of products.

The decreasing rate of association rules mined as depicted in Figure 22 and Figure 23 indicates the potential to capture some tractable set of inclusive design rules. Thus, the inclusive design knowledge can be captured and formalized by a set of association rules based on functional representation. As products are added and analyzed, rules that are repeated more often increase in the values of support and confidence.

Clustering of Product Pairs and Transferability of the Rules

The general template for the rules generated is that given a product function and user activity, how do a typical and inclusive product differ. Specifically, it is important to express and understand this difference in a manner that allows a designer to better design, or redesign, a product for accessibility. The rules are to be applied at the concept generation stage of design and are rooted in a functional abstraction of the design problem. In practice, a designer could query for a rule based on the actionfunction diagram of the product being designed. The rule could suggest functional, morphological, parametric, or no change - with associated support and confidence numbers – to the designer. The designer is then tasked with applying the suggested change to their design: the designer is transferring design knowledge in the form of rules from prior designs to a new design. To support such design activity, a knowledge base of existing designs could be provided to the designer for easier understanding and implementation. In this context, the transferability of rules from one product to the next is explored.

The transferability of rules in the context of clusters of products is explored based on sharing common rules. The clustering of products is important for several reasons. Products that contain multiple common rules are likely to be more similar, thus facilitating a tighter analogy between products and simpler rule transfer. Clusters are also important as they present the potential to create inclusive product families based on the products that share common rules and in turn common embodiments of those rules.

To cluster products, a straightforward rule search is employed. It is intended to search for products that share two or more rules in common. The electric versions of the wine opener and the can opener are almost identical in terms of a function-based comparison. Also, the guiding mechanism in the recliner and toilet seat are functionally the same. Thus, there are only two instances of products – the wine opener and the can opener as well as the recliner and the toilet seat – that share five or more rules in common. However, there are multiple instances of products sharing 2, 3, or 4 rules in common, hence three iterative loops search through the products. The first loop searches for clusters of products with any four rules in common. The second loop searches for clusters of products with any three rules in common. The third loop searches for clusters of products with two common rules.

Clusters of products and the sharing of the rules are represented using Venn diagrams. One such Venn diagram is shown in Figure 24. The products contained within each circle represent a cluster. The rules contained within every circle are applicable to all the products within the same circle. Two or more clusters might share a rule. Furthermore, two or more clusters can share a product. Figure 24 consists of clusters of the handheld product pairs that share Rule 1. Rule 1 recommends a parametric improvement in the design of a handle for better “grasping” while the product *positions hand*. Figure 25 thru Figure 28 shows product pairs that adopt an ergonomically designed handle for the inclusive product.

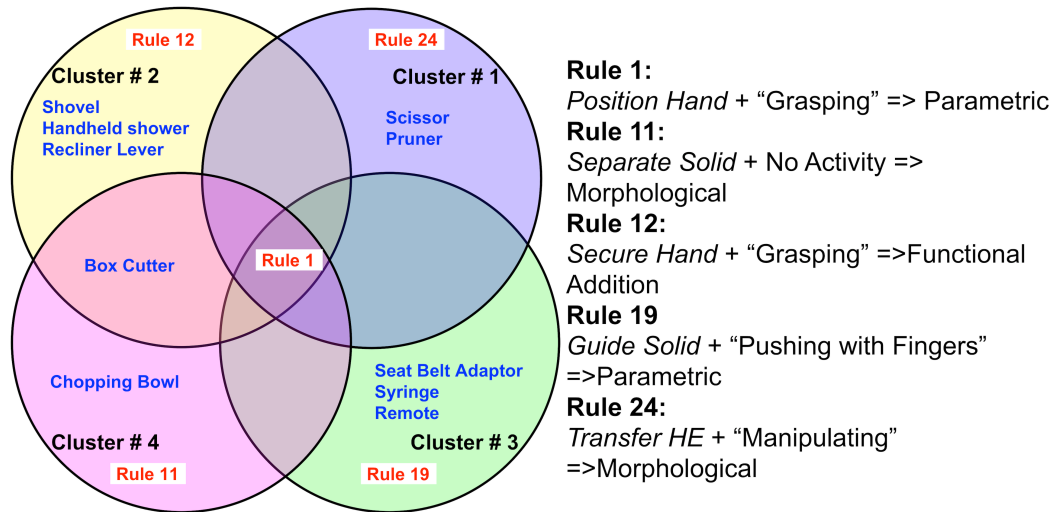


Figure 24. Clusters of inclusive products sharing Rule 1 that recommends a parametric change to the function *position hand* for better “grasping”

Figure 25 shows product pairs from cluster #1 that share Rule 1 and Rule 24. In addition to ergonomic handle design, these product pairs exhibit morphological change in the way the human energy is transferred for cutting action. Inclusive scissors incorporate a spring to aid the cutting action. The inclusive pruners have a four-link mechanism to transfer the human force while cutting. The scissors and pruners provide a close domain design analogy for transferability of rules between office supplies and garden tools.





	Typical	Inclusive
Scissors		
Pruner		

Figure 25. Product pairs in cluster #1 sharing Rule 1 and Rule 24 that suggests a morphological change to function of *transferring human energy* for easier “manipulating”

The product pairs used to generate the rules are shown side by side in Figure 25. The transfer of rules from one product to the next moves up and down in the figure. The scissors and pruners example in Figure 25 illustrate the transfer of rules from similar products. Scissors and pruners are functionally and morphologically similar to each other with specific parametric changes implemented to improve their intended applications. The fact that the function-based rule clustering method groups the scissors and pruners indicates that the method finds close domain products with intuitively transferable rules.

Figure 26 shows product pairs from cluster #2 that share Rule 12 and Rule 1. Rule 12 is the addition of a *secure hand* function to a product that involves the user activity of grasping. As embodied in a handheld shower nozzle, a shovel, a recliner lever, and a box cutter, the result is the addition of a closed loop handle that prevents the product from slipping out of user’s hand. As a contrast to the scissors and pruners case, the products shown in Figure 26 are dissimilar. The product domains represent bathroom fixtures, garden equipment, and furniture. Nevertheless, inspection of the design changes

in the different products in Figure 26 shows similar implementation of the rule. Thus, design knowledge in one domain can be applied to a different domain with function-based association rules for designing inclusive products.

	Typical	Inclusive
Hand Held Shower		
Shovel		
Recliner Lever		

Figure 26. Product pairs in cluster #2 sharing Rule 1 and Rule 12 that suggests addition of a functionality to *secure hand* while the user “grasps”

Figure 27 shows product pairs from cluster #3 that share Rule 19 and Rule 1. Rule 19 recommends parametrically large tabs or buttons on products that requires a user to “push with fingers” for operating the device. Rule 19 can be applied for inclusive design of an electronic device, automobile interiors, or a medical device. For cluster #3, the physical embodiment of Rule 19 is quite distinct, thus, providing distant analogies for inclusive design. Products shown in Figure 27 are from diverse domains. Though they represent a common rule, the physical embodiment of the rule is less similar than in the case of cluster #2 illustrated in Figure 25.







	Typical	Inclusive
Remote		
Seat Belt Adaptor		
Syringe		

Figure 27. Product pairs in cluster #3 sharing Rule 1 and Rule 19 that suggests a parametric change to *guide solid* function to aid “pushing with fingers”

Figure 28 shows the product pairs in cluster #4 that share Rule 11 and Rule 1. Rule 11 suggests a morphological change to the function *separate solid*. A chopping bowl and a box cutter exhibit a morphologically different cutter for the inclusive design. Though product pairs from cluster #4 are from a similar domain, both are cutting tools; the physical embodiment of Rule 11 is distinct.





	Typical	Inclusive
Chopping Bowl		
Box cutter		

Figure 28. Product pairs in cluster #4 sharing Rule 1 and Rule 11 that suggests a morphological change to the cutter

The discussion on transferability of rules continues with more examples demonstrating the physical embodiment of the rules. The Venn diagrams shown in Figure 29, Figure 30, and Figure 31 are similar to Figure 24. A designer can look up such examples of rules as analogies to generate ideas for inclusive design.

Figure 29 shows clusters of product pairs, sharing Rule 5, Rule 6, Rule 7, and Rule 9. Cluster #7 comprises the washer, the dishwasher, and the oven and they share Rule 5, Rule 7, and Rule 9. Rule 7, which is also shared by cook top, suggests the provision of a knee space for allowing access to a user on a wheel chair. Rule 5 and Rule 9, which are also shared by a refrigerator, recommend some intermediate counter space next to the product for placing hot or cold objects. The cook top and the refrigerator share Rule 6 that specify that controls must be located within the reach range of any user. Product pairs stated in cluster #5, cluster #6, and cluster #7 are closely related, as all of them are household appliances. The space around the product is modified to accommodate gross user access to the device. The rules embodied in these products can be conveniently transferred to inclusive design of other household appliances. For

instance, Rule 7 can be applied for the design of a bathroom sink where knee space could be provided beneath the sink to accommodate a wheelchair user.

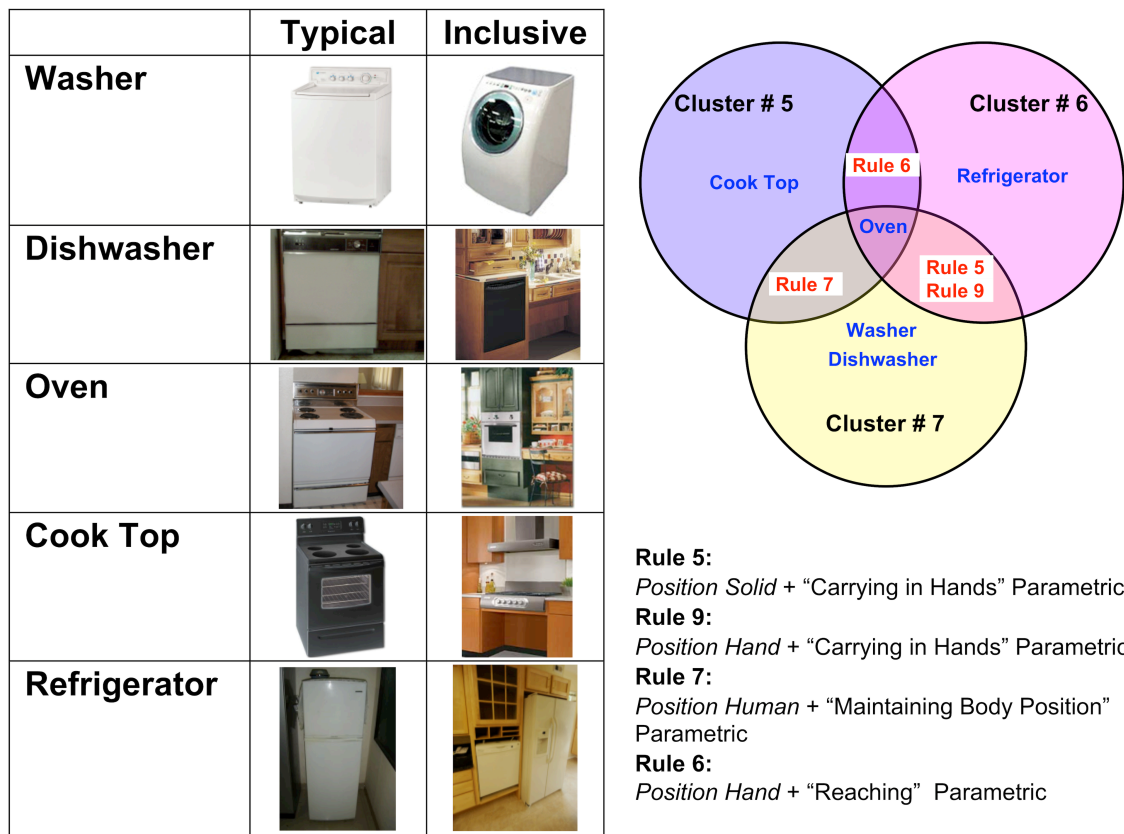


Figure 29. Clusters of architectural product pairs sharing Rule 5, Rule 6, Rule 7, and Rule 9 that deal with a parametric change to allow gross user access

Figure 30 shows clusters of product pairs sharing Rule 8, Rule 10, Rule 16, and Rule 17. Cluster #8 contains a kitchen scale, a blood pressure monitor, and a thermometer. Rule 10 and Rule 16 recommend a morphological change to the functions *sense status* and *indicate status* such that “seeing” is easier. The morphological change

recommended by Rule 10 to the function *indicate status* is implemented by a digital display. The morphological change suggested by Rule 16 to the function *sense status* is achieved with a sensor that operates on a different physical principle. Rule 8 and Rule 17 suggest the addition of functionalities to *import chemical energy* in the form of fuel or batteries and *converting chemical energy into mechanical energy*. It is worth noting that product pairs from cluster #8 are from the distinct product domains of kitchen equipment and medical devices, but all of them sense, measure, and display some physical parameter. In contrast, cluster #9 presents a distant domain analogy between an automobile and a medical device for *importing external chemical energy*.

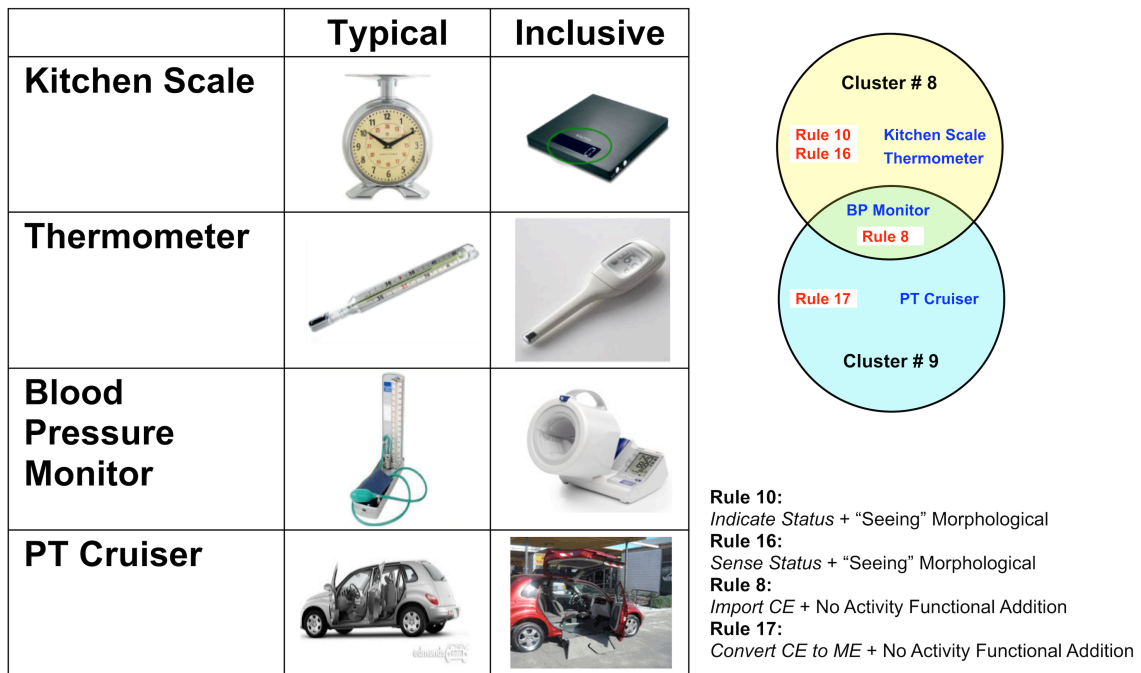

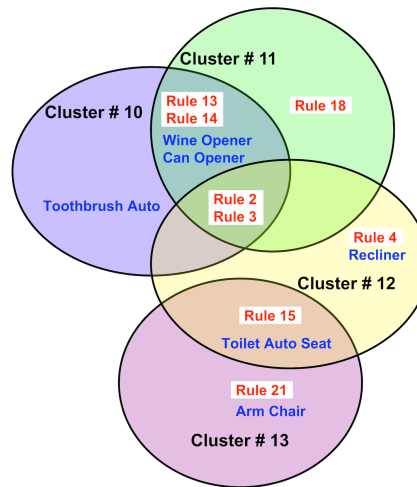


Figure 30. Clusters of product pairs sharing Rule 8, Rule 10, Rule 16, and Rule 17

In Figure 31, cluster #10 comprises a wine opener, a can opener, and an automatic toothbrush and the applicable rules are Rule 2, Rule 3, Rule 13, and Rule 14. These rules suggest the addition of functionalities like *import, store and supply electrical energy* and *convert electrical energy to mechanical energy* to automate the devices. Cluster #11 consists of a wine opener and a can opener similar to cluster #10, except for the addition of Rule 18. Rule 18 recommends deletion of the “turning” activity. A auto assist toilet seat, a recliner, and an armchair share Rule 15 that states a functional addition to *guide human* while “standing.”

Of note, cluster #11 contains product that are closely related and represent near domain analogies. Cluster #10 contains dental care products and kitchen tools that are fairly distinct. Product pairs from very distinct product domains like residential furniture, bathroom fixtures, dental care products, and kitchen tools share Rule 2 and Rule 3 which are external energy related functions. Product pairs from cluster # 12 and cluster #13 are quite similar.

	Typical	Inclusive
Wine Opener		
Can Opener		
Toothbrush		
Recliner		
Auto Assist Toilet Seat		
Arm Chair		



- Rule 2:**
Convert EE to ME + No Activity Functional Addition
- Rule 3:**
Import EE + No Activity Functional Addition
- Rule 4:**
Actuate Signal + No Activity Functional Addition + "Pushing with fingers"
- Rule 13:**
Store EE + No Activity Functional Addition
- Rule 14:**
Supply EE + No Activity Functional Addition
- Rule 15:**
Guide Human + "Standing" Functional Addition
- Rule 18:**
Transfer HE + "Turning" Functional Deletion
- Rule 21:**
Guide Human + "Sitting" Functional Addition

Figure 31. Clusters of product pairs consisting of a variety of products

One of the goals is to observe the conceptual similarity of a rule, as it is applicable to products from disparate product domains. The desire is to have sufficient conceptual similarity such that the rules provide insight for inclusive design across a wide range of products. Such similarity is seen in the product cluster shown in Figure 26. Further, I wish to explore the physical similarity in the context of creating inclusive product families based on a platform of that common inclusive element. In other words, could the actual embodiment of the rule result in a common component that can be shared across multiple products serving as a product platform? The garden tools in

Figure 32 illustrate such a component sharing case. An ergonomic, non-slip type of handle can act as inclusive product platform that can be shared by other handheld devices. In this case, the product family is built from similar products from a common domain. Products that can be derived from the platform of inclusive handle are shown in Figure 32.



Figure 32. Other hand held products that exhibit Rule 1 and Rule 12 in form of an ergonomic handle design

Figure 33 illustrates transferability of Rule 1 from a spatula to a trowel in form of an inclusive add-on module. In this case, the rule is observed in a kitchen tool and applied for inclusive design of a garden tool. The tool can be initially designed to be inclusive as shown in Figure 32 or an inclusive module can be retrofitted on the product to make it inclusive as shown in Figure 33.

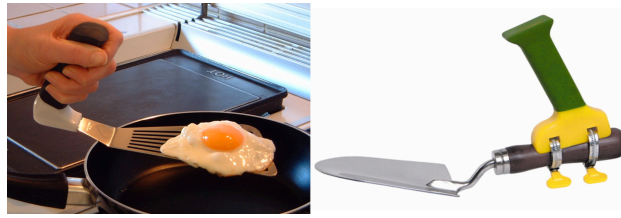


Figure 33. Transferability of Rule 1 between an inclusively designed spatula (left) and a trowel (right) in form of an inclusive add-on module

Clustering of product pairs that share the same rule can indicate transferability of rules. The physical embodiment of a rule can be quite distinct from one product to another. Presenting a product designer with examples of the rules can stimulate concept generation for inclusive design. Examples of physical embodiment along with the set of function-based association rules would help inclusive design.

Summary

This chapter aims to capture and formalize the inclusive design knowledge from functional representations of an arbitrarily large set of products. Particularly, a set of association rules is mined from a dataset of 65 product pairs that are formally compared based on user-product interactions. The rules are post-processed to filter out unnecessary information and the post-processing step is automated to allow the addition of more product pairs to the dataset.

Moreover, the set of rules obtained based on the 65 product pairs are compared to the initial results obtained with 15 product pairs. Comparison with the initial exploration shows that adding more product pairs to the dataset improves the statistical significance of the rule in terms of superior values of confidence and support. This chapter also

investigates the rate of rule generation per product pair as more products are added to the dataset. The results indicate that the rate of rule generation decreases and then approaches a limiting value after having a considerable quantity and variety of inclusive products in the dataset.

The transferability of rules from one product pair to another is explored. In this context, it is interesting how the function-based rules provide insight to the specific physical embodiment of a product. To perform this exploration, products are clustered based on rule commonality. Then within these clusters, the products are evaluated for conceptual and physical similarity shown by the specific embodiment of the rule. In the case of close domain products such as pruners and scissors, the rule implementation is highly analogous. Such a result indicates that the rule mining methodology presented in this chapter, including both representation and mining scheme, is producing expected results.

Additionally, the rules are found to be transferable across products in diverse domains such as plumbing fixtures and furniture. The transferability of the rules across diverse domains indicates that the inclusive design knowledge is broadly applicable in product design. The results indicate that the rules, and product clusters based on the rules, may provide opportunities to create product families based on sharing common components that make the product inclusive. An opportunity to create a product family of diverse kitchen and gardening products is presented.

CHAPTER VI

GRAPH DATA MINING ⁵

This chapter explores the feasibility and advantage of graph data mining for generating inclusive design heuristics. Another important contribution is the application of bipartite graphs for modeling the product pair comparison data. The application of graph theory for the mathematical representation of actionfunction diagrams and graph visualization for comprehending graphs is explained. The limitations of association rule mining in the context of generating rules for inclusive design is explained below.

Limitations of Association Rule Generation

As explained in Chapter IV and V association rule mining improves the efficiency in extracting inclusive design heuristics. However, association rule mining has some limitations as applied to the product-user data produced by actionfunction diagrams. There are multiple transactions corresponding to a single product pair, but there is no provision to group these transactions together on a per product basis in association rule mining. For instance, the relationship between the deletion of the *import human energy* function in a typical opener, as a change based on the addition of the *import electrical energy* in the inclusive opener, cannot be modeled. To preserve the relation between the candidate function sets in a product pair, the entire product pair

⁵ This chapter has been submitted for publication in the Journal of Computing and Information Science in Engineering, refer to: Sangelkar, Shraddha, and McAdams, Daniel A., "Mining Functional Model Graphs to Find Product Design Heuristics with Inclusive Design Illustration," Journal of Computing and Information Science in Engineering, in review.

could be modeled as a single transaction. For example, all the items in Table 2 could be modeled as a single transaction.

But there are two major limitations if all candidate function sets in a product pair are modeled as one transaction. Firstly, functions, activities, and design differences are all treated as items in the same category. For example, *picking up* is of the category ‘user activity’ and *import hand* is of the category ‘product function,’ but here they will be treated as one general category ‘attributes of a can opener’. Secondly, the effect of a particular candidate function set on the inclusive nature of design cannot be captured; for example of a grasping activity and position hand function that is related to ‘no change’ but cannot be modeled together.

Table 2 shows the flat-attribute type input for association rule generation obtained on comparison of the actionfunction diagrams of the can openers. Each candidate function set, a product function in a typical product along with the corresponding function in the inclusive counterpart, is an input transaction to the Apriori algorithm. Each row of Table 2, which is also a candidate function set, is a flat-attribute type transaction input to the Apriori algorithm. The graph mining captures complex relational information as compared to the flat-attribute type representation.

Modified Data Analysis

The data generation process for this research remains the same as that explained in previous chapter. However, the data analysis process is modified to overcome the limitation of the flat-attribute type representation. Figure 34 shows the modified rule generation process, with specific emphasis on the data analysis.

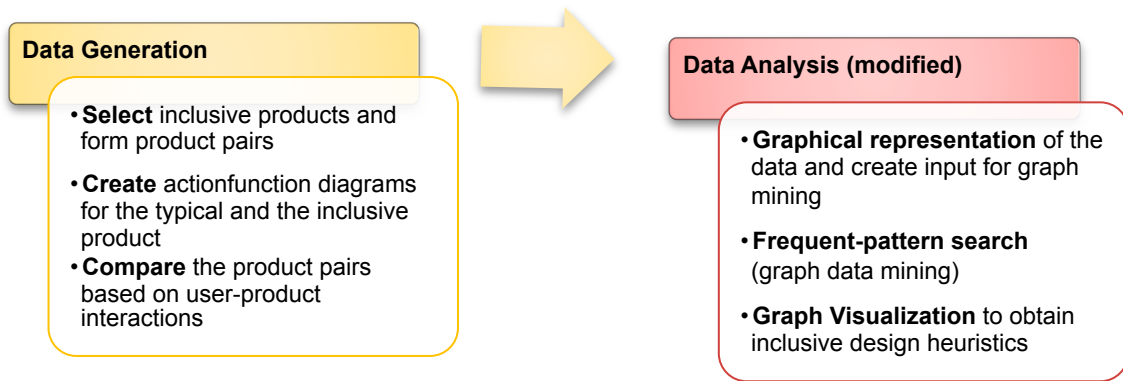


Figure 34. Modified data analysis with graphical representation and frequent-pattern search to study the inclusive design characteristics

The pre-processing step converts the comparison data into a flat-attribute type representation. The modified data analysis replaces the pre-processing step with graphical representation, which is explained in the next part. A frequent-pattern search algorithm, a graphical data mining technique, replaces the association rule generation. The FSG can be configured to find ‘maximal-patterns’ only. The maximal-pattern constraint finds all the rules satisfying other conditions and excludes the repetitive information. Thus, with the maximal-pattern constraint the post-processing step explained in Chapter V can be eliminated. The graph visualization step is added for efficient interpretation of the mathematical form of frequent subgraphs.

Graphical Representation

The general form of graph data representation consists of the entities given by the nodes, their attributes specified by the node labels, and the relationships between the entities stated by the edges and edge labels. The comparison of actionfunction diagrams yields complex relational information between user-product interactions. The traditional

graph representation with nodes referring to the functions and edges referring to the flows cannot capture the comparison information. The comparison between a product pair is best modeled with a bipartite graph. A bipartite graph is a graph whose nodes can be divided into two disjoint sets such that every edge connects a node in one set to a node in the other set [93]. This part explains the creation of input for graph data mining and the advantages of this graph representation.

The two products in a product pair are the two sets of a bipartite graph, one is the typical product and one is the inclusive product. The nodes of a graph are the user-product interactions, and the node labels are the corresponding product functions and user activities. A user-product interaction is the activity that a user performs when the product performs the function. In a few cases, a function is added in the inclusive product to aid an activity associated with the typical device. Such an activity in the typical product by itself constitutes as a user-product interaction. Certain functions of the product are essential for overall functioning of the device but do not necessarily have an activity associated with it. Such functions also count as user-product interaction with null activity reported. A user-product interaction may not contain more than one function or one activity.

The edges represent the changes as one moves from a typical design to inclusive design. An edge in a bipartite graph connects the design differences between the user-product interactions of the typical set and the inclusive set. The edge labels specify the type of design difference, namely parametric (P), morphological (M), functional (F), or no change (N). Interactions are connected by an edge if the design comparison impacts

the accessibility of the inclusive product. Notably, the edges are not physical connections within the same product. For example, the window controls are not related to the window, but the push button of an automatic window control in the inclusive automobile is related to the manual window lever in a typical automobile.

Figure 35 explains the conversion of the comparison data into graphical format. The user-product interactions of the typical can opener acts as set X of the bipartite graph while the ones of inclusive opener act as set Y. Nodes 0 to 9 are in set X, and nodes 10 to 21 are in set Y. The edges are colored purple, green, orange, and grey for the parametric, morphological, functional, and no change type of design differences, respectively. For instance, node 1 in set X is connected to node 11 in set Y with a green edge. This means that the user-product interaction ‘position hand + grasping’ changes morphologically from typical to inclusive design.

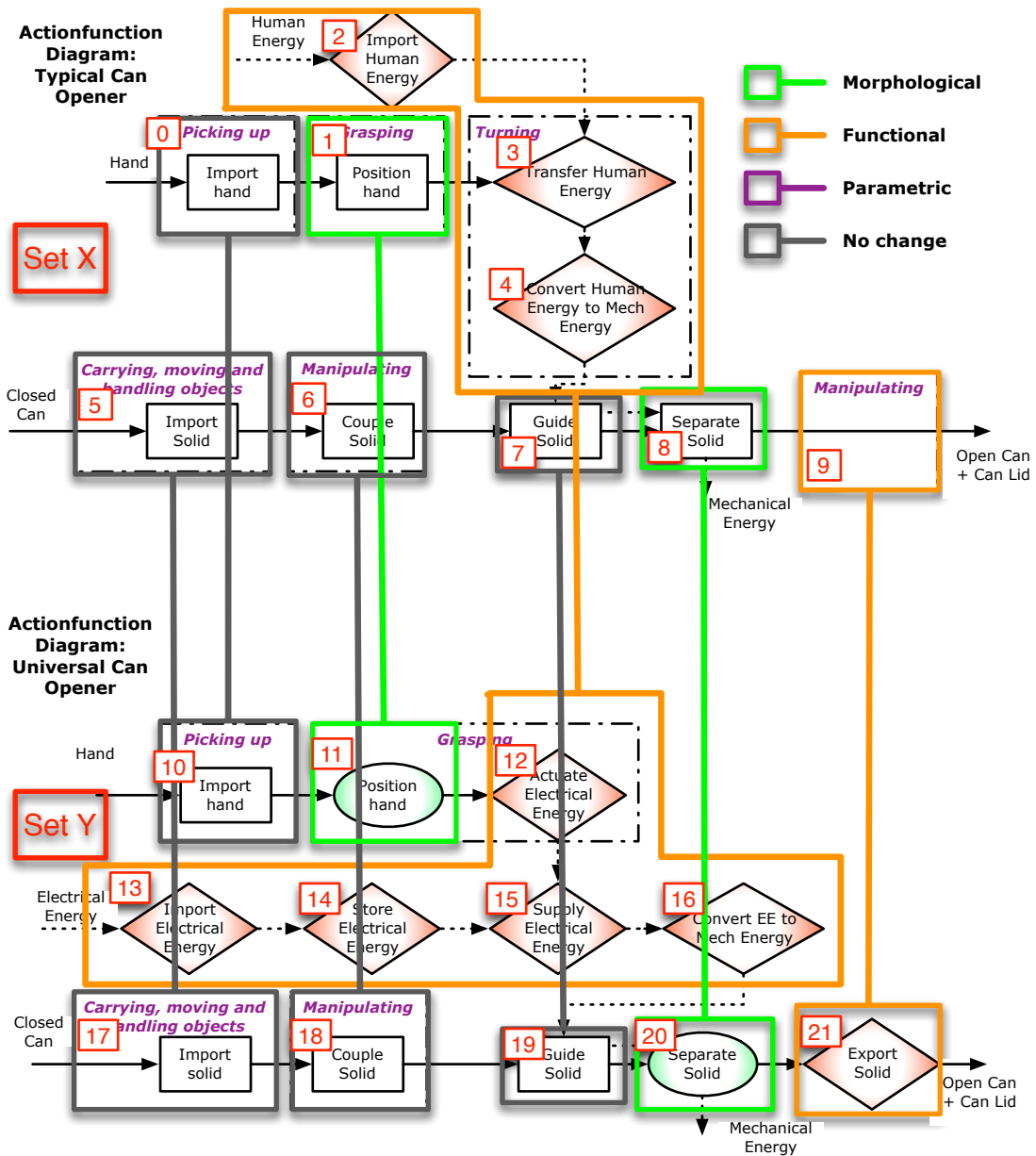


Figure 35. The bipartite graph representation of a can opener product pair comparison

With the graphical representation each product pair is modeled as one graph. The data is mined from a database of 65 product pairs modeled as graphs. The database of sixty-five product pairs along with the corresponding bipartite graphs is available in the

Appendix D and E and uploaded online at www.prosedesign.org [87]. In addition, the graphical representation allows modeling of the relations between multiple user-product interactions of a single product pair. For example, addition of electrical energy to a manual can opener eliminates the activity of *turning* in the case of the electric can opener. The addition of *electrical-energy-related* interactions is related to the deletion of the *human-energy-related* interactions. Thus, each of the nodes, namely node 2, node 3, and node 4 from set X are connected to each node 12, node 13, node 14, node 15, and node 16 in set Y. Such relations between apparently unrelated components can be modeled with a graphical representation. Multiple relationships between interactions are mostly observed for functional design differences.

It is worth noting that the function of *exporting lid*, performed by the magnet in the electric opener, eliminates the *manipulating* activity in the typical design. However, the concept of using a magnet to export the lid is independent of the concept of including electrical energy. Hence, the functional change associated with *export solid* and *manipulating* is not related to the *electrical-energy-related* interactions. Thus, node 9 in set X is connected to node 21 in set Y by an orange edge, but node 9 is not connected to node 12, node 13, node 14, node 15, or node 16. The next part explains the graph data mining technique called frequent pattern search.

Frequent Pattern Search

PAFI is a software package developed by Karypis at the University of Minnesota to find interesting patterns from large and diverse databases [94]. The PAFI software package includes the LPMiner for databases with itemsets in transactions, the SPMiner

for sequential databases, and the FSG for databases with undirected graphs. The FSG algorithm is used here to find the undirected frequent subgraphs satisfying the minimum threshold support constraint, where support is the percentage of frequent subgraphs found in the database.

Figure 36 shows node labeling and an adjacency matrix for the can opener example. A node is denoted with ‘v’ followed by node number and node label. An adjacency matrix of a graph shows the edges connecting the nodes along with an edge label. For instance, nodes v 1 and v 11 are connected by an edge of morphological type, which denotes a morphological change to the function position hand when the user is grasping. Recall that nodes 0 to 9 belong to the typical product, and nodes 10 to 21 belong to the inclusive product.

	Node			Adjacency Matrix																					
	X	Product Function	User Activity	v0	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13	v14	v15	v16	v17	v18	v19	v20	v21
Typical	v0	Import Hand	Picking up	0	0	0	0	0	0	0	0	0	0	0	N	0	0	0	0	0	0	0	0	0	0
	v1	Position Hand	Grasping	0	0	0	0	0	0	0	0	0	0	0	M	0	0	0	0	0	0	0	0	0	0
	v2	Import Human Energy	None	0	0	0	0	0	0	0	0	0	0	0	0	0	F	F	F	F	F	0	0	0	0
	v3	Transfer Human Energy	Turning	0	0	0	0	0	0	0	0	0	0	0	0	0	F	F	F	F	F	0	0	0	0
	v4	Convert Human Energy to Mech Energy	Turning	0	0	0	0	0	0	0	0	0	0	0	0	0	F	F	F	F	F	0	0	0	0
	v5	Import Solid	Carrying	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	N	0	0	0
	v6	Couple Solid	Manipulating	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	N	0	0
	v7	Guide Solid	None	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	N	0
	v8	Separate Solid	None	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	M
	v9	None	Manipulating	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Inclusive	v10	Import Hand	Picking up	N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v11	Position Hand	Grasping	0	M	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v12	Actuate Signal	Grasping	0	0	F	F	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v13	Import Elec Energy	None	0	0	F	F	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v14	Store Elec Energy	None	0	0	F	F	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v15	Supply Elec Energy	None	0	0	F	F	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v16	Convert Elec Energy to Mech Energy	None	0	0	F	F	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v17	Import Solid	Carrying	0	0	0	0	0	N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v18	Couple Solid	Manipulating	0	0	0	0	0	0	N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v19	Guide Solid	None	0	0	0	0	0	0	0	N	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	v20	Separate Solid	None	0	0	0	0	0	0	0	0	0	M	0	0	0	0	0	0	0	0	0	0	0	0
	v21	Export Solid	None	0	0	0	0	0	0	0	0	0	0	0	F	0	0	0	0	0	0	0	0	0	0

Figure 36. Node labeling and adjacency matrix for the can opener example, where M, F, and N denote morphological change, functional change, and no change, respectively.

For the complete description of a graph, a list of nodes with node labels and the adjacency matrix is sufficient. However, the input to the FSG algorithm requires the list of nodes followed by the list of edges. Each node line consists of a node identifier and a node label. Each edge is denoted with the nodes that it connects and an edge label. Table 10 shows the input format to the FSG algorithm for the can opener example. The node label headers PF_Typ, PF_Inc, UA_Typ, and UA_Inc mean product function in typical product, product function in inclusive product, user activity in typical product, and user activity in inclusive product, respectively. Here, ‘v’ and ‘u’ denote the node and the edge, respectively. For example, the edge $u\ 0\ 10\ N$ connects the nodes $v\ 0$ and $v\ 10$ with a ‘no change’ type of edge, where both the nodes represent a user-product interaction of the function *import hand* and activity *picking up*.

The input file contains the graph transactions for the 65 product pairs studied here. Frequent patterns are discovered based on the user specified threshold value of support. The FSG can be programmed to generate only the maximal-patterns. In addition, the user can obtain the transaction identifier list corresponding to each frequent pattern found. Below, the results discuss the output from the FSG algorithm and its comparison with the previous results of association rule mining. First, the next part explains the graph visualization tool.

Table 10. Sample of input format for the can opener example

```

t # p0 Can Opener
v 0 PF_Typ=Import_Hand,UA_Typ=Picking_up
v 1 PF_Typ=Position_Hand,UA_Typ=Grasping
v 2 PF_Typ=Import_Human_Energy,UA_Typ=None
v 3 PF_Typ=Transfer_Human_Energy,UA_Typ=Turning
v 4 PF_Typ=Convert_Human_Energy_to_Mech_Energy,UA_Typ=Turning
v 5 PF_Typ=Import_Solid,UA_Typ=Carrying
v 6 PF_Typ=Couple_Solid,UA_Typ=Manipulating
v 7 PF_Typ=Guide_Solid,UA_Typ=None
v 8 PF_Typ=Separate_Solid,UA_Typ=None
v 9 PF_Typ=None,UA_Typ=Manipulating
v 10 PF_Inc=Import_Hand,UA_Inc=Picking_up
v 11 PF_Inc=Position_Hand,UA_Inc=Grasping
v 12 PF_Inc=Actuate_Signal,UA_Inc=Grasping
v 13 PF_Inc=Import_Elec_Energy,UA_Inc=None
v 14 PF_Inc=Store_Elec_Energy,UA_Inc=None
v 15 PF_Inc=Supply_Elec_Energy,UA_Inc=None
v 16 PF_Inc=Convert_Elec_Energy_to_Mech_Energy,UA_Inc=None
v 17 PF_Inc=Import_Solid,UA_Inc=Carrying
v 18 PF_Inc=Couple_Solid,UA_Inc=Manipulating
v 19 PF_Inc=Guide_Solid,UA_Inc=None
v 20 PF_Inc=Separate_Solid,UA_Inc=None
v 21 PF_Inc=Export_Solid,UA_Inc=None
u 0 10 No_change
u 1 11 Morphological
u 2 12 Functional
u 2 13 Functional
u 2 14 Functional
u 2 15 Functional
u 2 16 Functional
u 3 12 Functional
u 3 13 Functional
u 3 14 Functional
u 3 15 Functional
u 3 16 Functional
u 4 12 Functional
u 4 13 Functional
u 4 14 Functional
u 4 15 Functional
u 4 16 Functional
u 5 17 No_change
u 6 18 No_change
u 7 19 No_change
u 8 20 Morphological
u 9 21 Functional

```

Graph Visualization

Figure 37 shows the visualization of the graph constructed with the sample input shown in Figure 36. The edges are colored as purple, green, orange, and grey for the parametric, morphological, functional, and no change type of design differences, respectively. The red circles represent the nodes of the graphs. All the nodes on the left

hand side belong to the set X or the typical product, while all the nodes on the right hand side belong to the set Y or the inclusive product. The visualization tool promotes a better understanding of the mathematical format of graph representation.

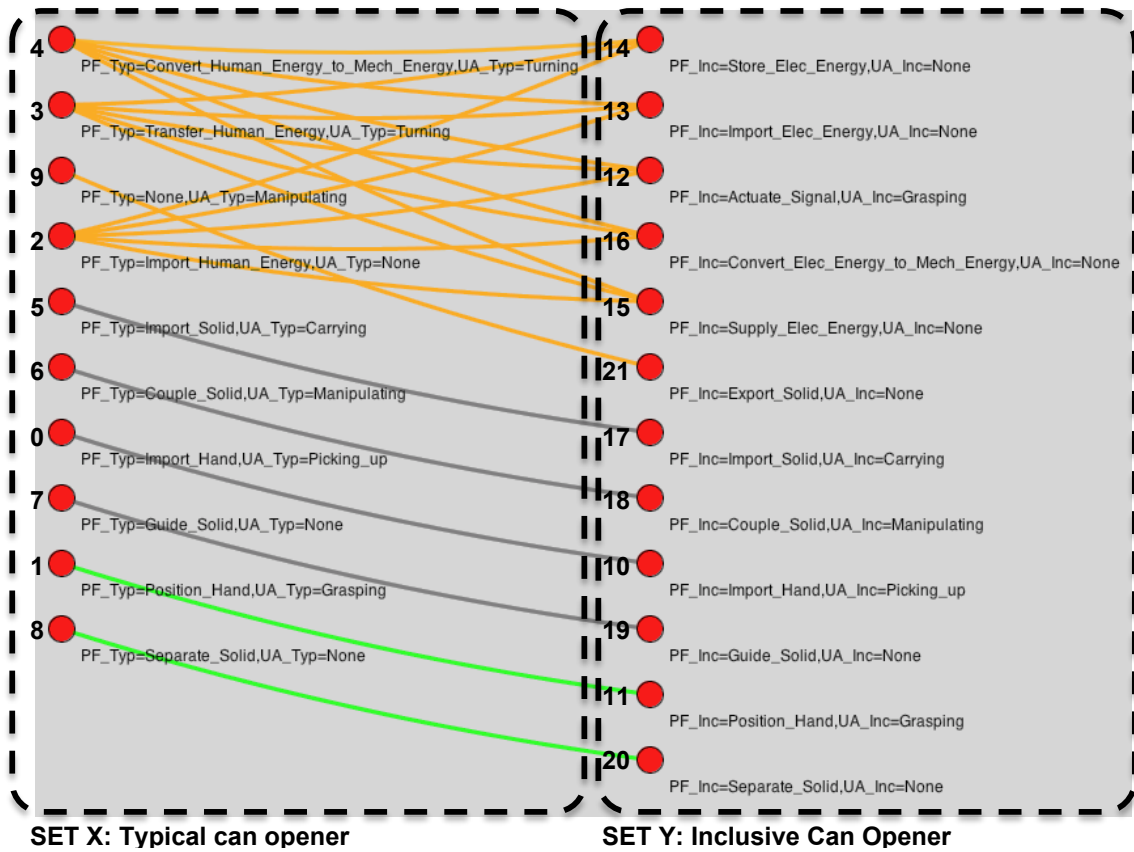


Figure 37. Visualization of the input data for can opener example along with node numbering

Results of the Graph Data Mining

This part discusses the results of frequent pattern search and the comparison of frequent subgraphs with the association rules. The FSG algorithm outputs the frequent

patterns discovered in the format, $t \# \langle \text{pattern ID} \rangle, \langle \text{count} \rangle$ followed by the nodes and edges of the frequent pattern. Figure 38 shows a sample output from the FSG algorithm.

Figure 38 also explains how to interpret the sample output.

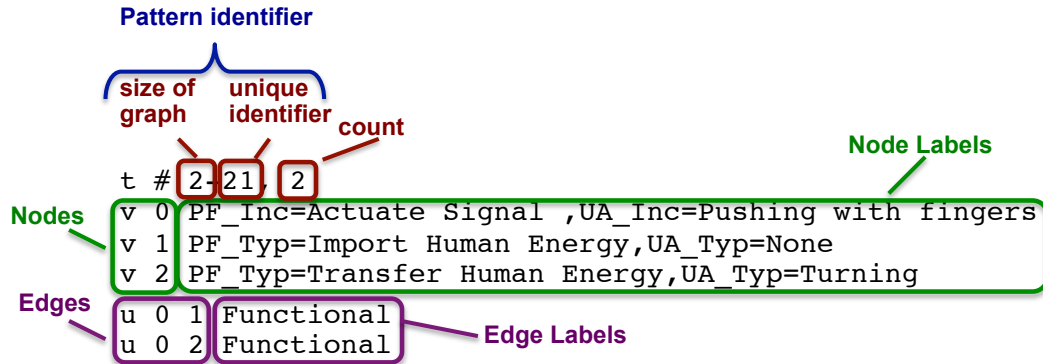


Figure 38. Sample output from the FSG algorithm

The pattern identifier of a subgraph denotes the size of the graph and a unique identifier. The pattern size $t\#2-21$ is two, meaning that the subgraph consists of two edges. The unique identifier is the identifier assigned by the algorithm when each subgraph is created. The count of a subgraph stated after the pattern identifier specifies the value of support or the frequency of occurrence. The subgraph $t\#2-21$ occurs twice in the database. The three nodes of the frequent pattern are given by $v 0$, $v 1$, and $v 2$ and the two edges are given by $u 0 1$ and $u 0 2$. The nodes and edges are followed by node labels and edge labels. The ‘PF_Inc = Actuate Signal, UA_Inc = Pushing with fingers’ label of the node indicates the user-product interaction of actuating signal while the user is pushing with fingers in the inclusive product. ‘PF_Typ = Import Human Energy,

‘UA_Typ = None’ represents the *import-human-energy* function in the typical design. ‘PF_Typ = Transfer Human Energy, UA_Typ = Turning’ represents *transfer-human-energy* function when the user is turning in the typical design. Both the edges *u 0 1* and *u 0 2* represent a functional type of change.

The transaction identifier list of the FSG records the product in which the frequent pattern is observed. The frequent pattern *t#2-21* is found in the wine opener and automobile window control. The pattern *t#2-21* can be interpreted as adding functionality for *actuating signal* by *pushing with fingers* to avoid *importing* and *transferring of human energy* while turning. The subgraph can be applied for design of inclusive products. When the typical product requires complicated activities like *turning*, for which the product *transfers human energy*, replacing it with simple activities like *pushing with fingers to actuate signal* can make it inclusive. Figure 39 shows the visualization of the frequent pattern *t # 2-21*, which consists of three nodes and two edges. The orange color of the edge indicates functional type of change shown for both the edges.

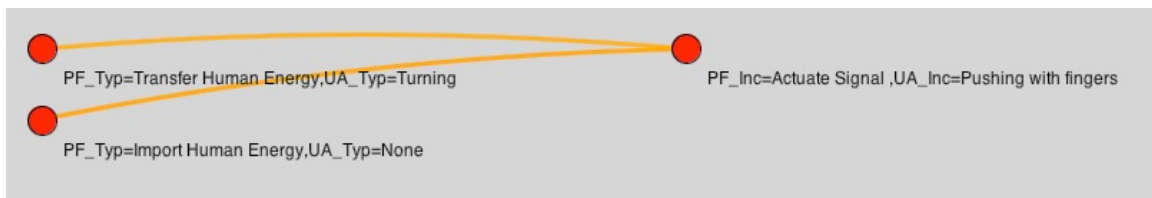


Figure 39. Visualization for the sample output *t # 2-21*

The association rule corresponding to the subgraph *t#2-21* is given as (“Product function in inclusive” = *Actuate Signal*, “User activity in typical” = *None*) → (“Change” = *Functional*, “User activity in inclusive” = *Pushing with fingers*). However, the format of association rule does not capture the targeted functions and activities in the actionfunction diagram of a typical product. The graph data mining suggests a change relative to the meaningful patterns observed in the typical design and is thus preferable. Such heuristics have greater possibilities for product design application.

Figure 40 shows the application of the frequent subgraph as a design heuristic for inclusive design. The nodes in the left hand side or the typical set of design heuristic acts as the antecedent of the rule, and the nodes on the right hand side or inclusive set, along with the edges, acts as the consequent. While designing an inclusive product, the designer creates an actionfunction diagram for typical design. When a part of an actionfunction diagram of typical design resembles the antecedent of the heuristic, the consequent of the heuristics has potential application in the inclusive product.

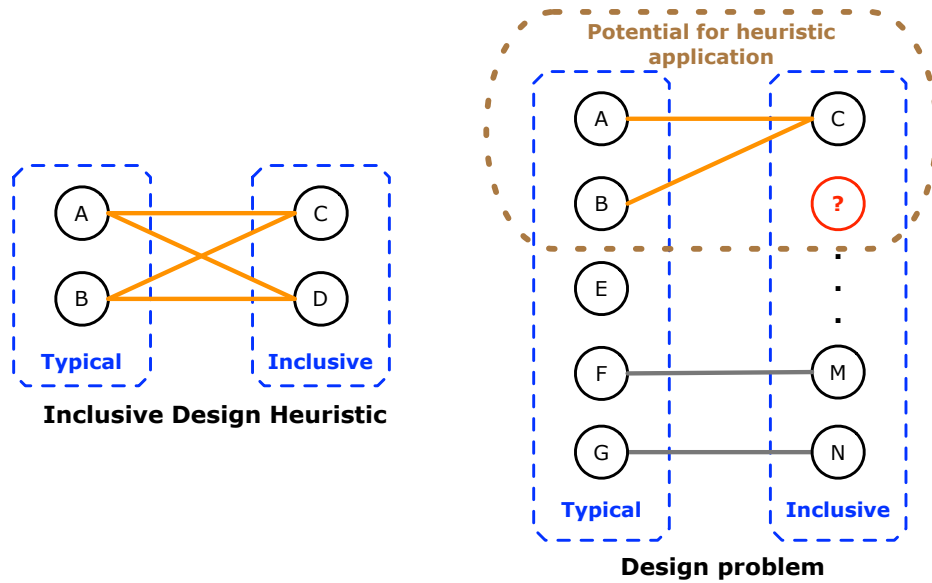


Figure 40. Application of a frequent subgraph as design heuristic for inclusive design

Comparison of the Frequent Subgraphs with Association Rules

The total number of frequent subgraphs discovered from 65 product pairs with support of 2 or more transactions is 67. The algorithm is programmed to generate the maximal-patterns only. The maximum pattern size found is 8 with one frequent subgraph. In addition, fifty-nine subgraphs of size 1, three subgraphs of size 2, one subgraph of size 3, two subgraphs of size 4, and one subgraph of size 5 are found.

In Chapter V, association rule mining is performed on the same database of 65 products pairs as analyzed here. The results obtained from association rule mining are used for comparison and validation of the frequent-pattern search algorithm. Sixty-two filtered association rules are mined with the support of two or more transactions. The

frequent subgraphs of size 1 are used for comparison of frequent-pattern search with the association rule mining.

Out of 59 frequent subgraphs of size one, 46 subgraphs are identical to the association rules in terms of the information contained. An example of an association rule that is also found as frequent subgraph is stated here. The association rule (“Product function in typical” = *Position Hand*, “User activity in typical” = *Reaching*) → (“Change” = *Parametric*) has a support of 6 instances. The graph data mining discovers the pattern *t#1-22* with a support of 6 instances that is in agreement with the association rule. The association rule and the corresponding frequent pattern are shown in Figure 41, which suggest making a parametric design change when the user is *reaching* for a part of the product.

Frequent Subgraph by PAFI's FSG

```
t # 1-22, 6
v 0 PF_Typ=Position_Hand,UA_Typ=Reaching
v 1 PF_Inc=Position_Hand,UA_Inc=Reaching
u 0 1 Parametric
```



Association rule by Apriori Algorithm

(“Product function in typical” = *Position Hand*, “User activity in typical” = *Reaching*)
 → (“Change” = *Parametric*)

Figure 41. Example of a rule found both by graph data mining and association rule mining

The graph data mining identifies 13 subgraphs of size 1 that are not mined as association rules. The primary reason for the difference in the outputs of the two mining methods is the change in the input format from the flat-attribute-type transaction input to the graphical-type input. Another reason for the difference is that association rule mining gets 523 input transactions, whereas the graph data mining gets only 65 input transactions.

The subgraphs of size 2 or more cannot be directly compared with the association rules. Some of the association rules are incorporated into more complicated subgraphs of size more than one. Figure 42 shows an example subgraph *t # 8-0* of size 8 along with its visualization. The frequency of occurrence of the subgraph *t # 8-0* is two.

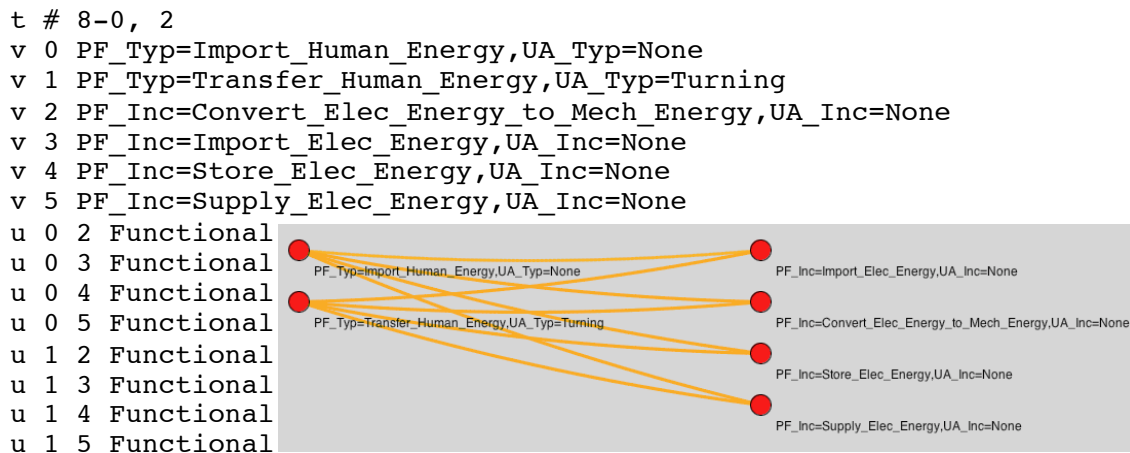


Figure 42. Example of a subgraph *t # 8-0* of size 8 along with its visualization

A subgraph of size 8, shown in Figure 42, encompasses 5 association rules discussed below. The support of each of the association rule might be different from the overall support of the frequent subgraph *t* # 16-0.

- (“Product function in inclusive” = ***Import Elec. Energy***, “User activity in typical” = ***None***)
→ (“Change” = ***Functional Addition***)
- (“Product function in inclusive” = ***Store Elec. Energy***, “User activity in typical” = ***None***)
→ (“Change” = ***Functional Addition***)
- (“Product function in inclusive” = ***Supply Elec. Energy***, “User activity in typical” = ***None***)
→ (“Change” = ***Functional Addition***)
- (“Product function in inclusive” = ***Convert Elec. Energy to Mech. Energy***, “User activity in typical” = ***None***) → (“Change” = ***Functional Addition***)
- (“Product function in typical” = ***Transfer Human Energy***, “User activity in typical” = ***Turning***) → (“Change” = ***Functional Deletion***)

Thus, graph data mining extracts the information that is found by association rule mining and also finds additional information. The frequent-pattern search algorithm groups the related information into meaningful chunks. In addition, the visualization tool greatly improves readability of the information.

Summary

This chapter presents a powerful tool of graphical data mining to develop heuristics for inclusive design. The results of graph data mining are compared with association rule mining, which shows that the frequent-pattern search not only extracts statistically significant design heuristics but also mines additional information with better details. Graph data mining has the capability to efficiently search for new design heuristics from the updated repository of inclusive products.

The advantages of graphical representations over flat-attribute type representations are discussed in the context of function-based representation framework. The function-based comparison data contains complex relational information; the graphical representation effectively captures all the pertinent information and preserves the relations. The graphical representation explained in this chapter can be extended for many other functional representations and will be discussed in next chapter. Particularly, graph representation and graph data mining can be applied for the expansion and mining of the design repository with some alterations in the graphical format.

This research applies a graph visualization tool that helps the reader to better understand the graphical information. Graph visualization increases the comprehensibility of the information by converting the relational information into pictorial form. In addition, the graph visualization helps to verify the creation of graphs; thus allowing quick updates to the database.

CHAPTER VII

AUTOMATED HEURISTICS GENERATION

Graph grammars, a technique for creating new graphs based on a set of rules, is a very powerful tool for computational design synthesis and has been applied in many contexts. Graph grammars have been used to facilitate automated generation of functional models, automated concept generation, product platform design, feature representation, and machine design [36, 43, 57, 95, 96]. Graph grammar is particularly useful because it enables the designer to synthesize new design knowledge with a set of guiding principles. However, one of the major difficulties lies in deriving the grammar rules. Typically, graph grammars are compiled by an expert based on empirical knowledge. Principal component analysis has been applied for automated shape grammar generation [60, 61]. However, principal component analysis works well for continuous numerical data but not for discrete categorical data; hence, it is not generally applicable for automated graph grammar generation.

This chapter proposes to extend the application of graph data mining beyond inclusive design heuristics to derive engineering grammars. Function-based models can be graphically represented using nodes and edges. A frequent-pattern search algorithm mines information from a graphically represented design dataset. The frequent sub-graphs found by the mining algorithm can be formulated into graph grammars. This application of graph data mining is new to the engineering design research. Particularly, graph data mining has a potential for automated graph grammar generation. An expert is

still required to check the grammar rules obtained, but the rule generation process is made less labor intensive. Graph data mining can be used to either assist an expert to create grammars or to check the grammar rules that are already obtained. In general, graph data mining can be applied to extract design heuristics from any discrete relational data that can be represented as graphs.

The process of extracting design heuristics is explained in the research approach. Results discuss the comparison of the machine generated grammar rules with expert derived grammar rules to validate the method. Next part explains some background of computation based design synthesis as related to engineering graph grammars.

Background: Computation-Based Design Synthesis

Engineering design is the process of formulating a plan that assists the engineer in creating an engineered system satisfying a set of customer needs. The preliminary phase of engineering design, which establishes the roadmap of the desired functionality, affects the final design but it can be very difficult for the designers. Functional modeling is one such design synthesis method that helps abstract the intended functionality of a product from customer needs.

The preliminary design phase starts with gathering customer needs as shown in Figure 43. The customer needs are then processed to establish the product function and create a functional model. In function modeling, the overall functionality of a product is decomposed into constituent functional elements. The function model is used as a roadmap to generate ideas for the conceptual design. Two main aspects of the functional

modeling, that have scope for improvement, are the creation of functional models and the application of functional models to generate ideas.

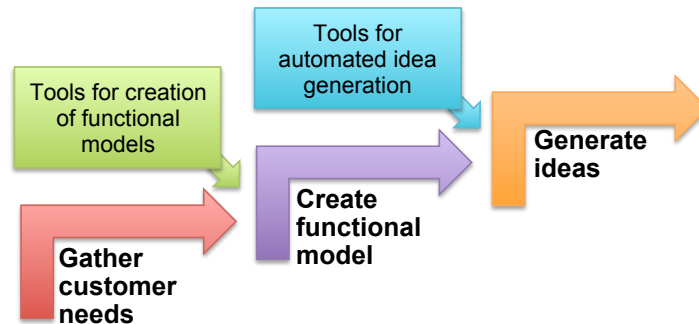


Figure 43. Preliminary phase of engineering design

To support functional modeling, a design repository is developed for collection, storage, and retrieval of the functional models for over 184 products [62]. Research efforts focus on the representation in terms of developing both the framework and the language for functional modeling [54]. Automated Design Lab at University of Texas, Austin developed methods to guide the creation of functional models [40]. A rule set of graph grammars are formulated for assisting the untrained engineers and for automated creation of functional models.

Once the functional model is created, the next phase is concept generation. There is considerable ongoing research to facilitate application of functional models for concept generation [54]. Research efforts in the automated concept generation suggest addition of a search mechanism to the set of graph grammar rules for exploring various potential solutions from the design repository [39, 56]. Other efforts for automated concept generation include an automated morphological matrix, relational matrices, and

unsupervised learning, all in combination with a mechanism to search the design repository [63-65]. In general, the methods for automated concept generation rely heavily on the design repository and the repository must be appended to accommodate broader categories of design problems. Once the repository is expanded, a method to extract new patterns from it is required.

Research efforts for creation of the functional model have been more substantial as compared to application of the functional model. Avenues for further exploration include developing design heuristics for concept generation from functional models, developing algorithms for finding engineering design heuristics, and developing unified language for graph grammars. Here, the algorithms for automatically creating design heuristics are explored.

The goal here is to automate the process of extracting heuristics from a design repository. With automated heuristic extraction process the design repository can be continually expanded and new heuristics can be found. This process will reduce the need for human interpretation and increase machine learning to obtain statistically significant heuristics. The extracted heuristics can serve as suggestion or guidelines for designers during concept generation from functional models. This chapter demonstrates the extraction of engineering graph grammars from a repository of functional models.

Research Approach

Figure 44 shows the automated heuristics extraction process to obtain engineering graph grammars from the design repository. This process implements three different software: GraphSynth to create the input for graph mining, FSG to extract the

frequent patterns with graph mining, and JUNG to visualize the output or the frequent from graph mining. These three software are developed independently for different applications. A graphical user interface (GUI) is built in JAVA to facilitate the information flow between the three software and provide a single interface. The GUI code not only creates a user interface but also helps to exchange information amongst the software. The JAVA GUI, shown in Figure 45, has the following features that are explained subsequently: convert of xml format to txt format, compile of a single input file, flip edges and nodes in a graph, separate the subgraphs with specified size and frequency, and display the subgraphs.

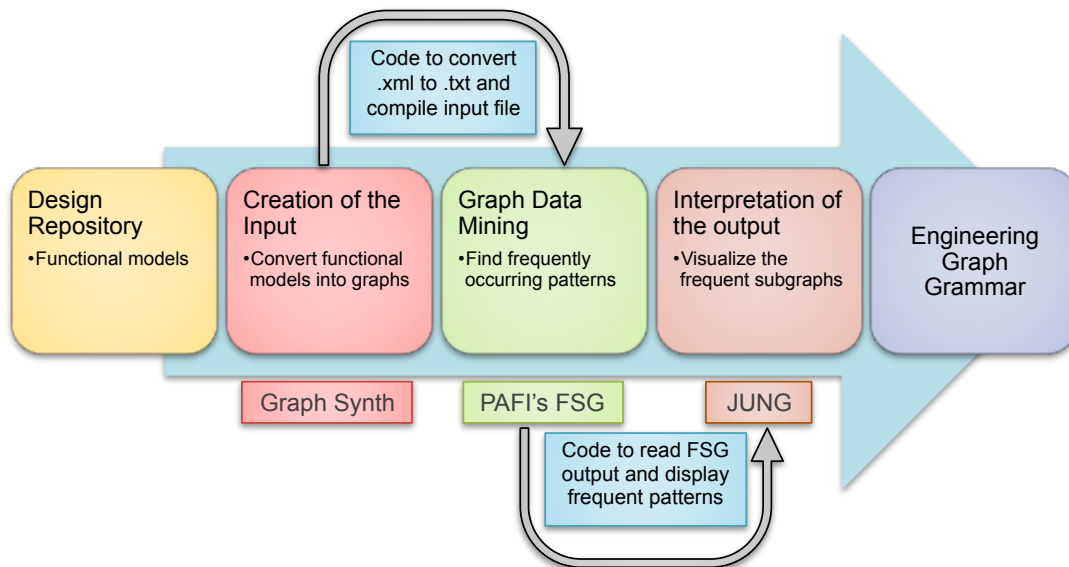


Figure 44. Automated heuristics extraction process

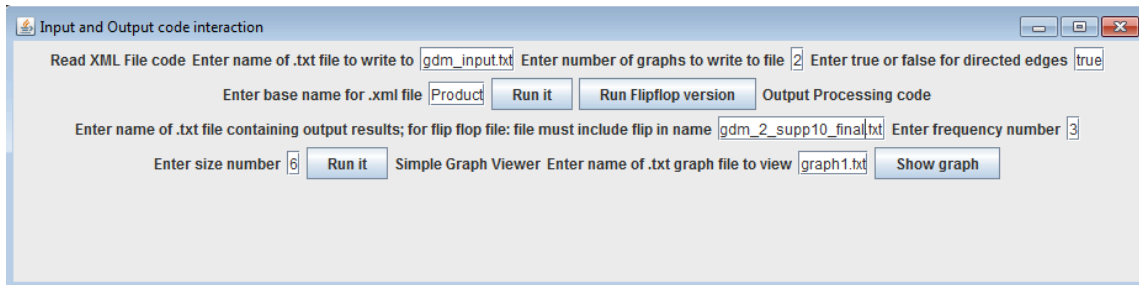


Figure 45. Graphical user interface (GUI) for interpreting the graph mining output

The process starts with the design repository which consists of over 184 products and 6906 artifacts ranging from electro-mechanical products to biological systems [62]. Fifty electro-mechanical devices are selected for this study and listed in Table 11. The product numbers listed in Table 11 are identifiers that helps to trace back the origin of frequent patterns. Functional models of the selected products are available online in the design repository. For this study only electro-mechanical devices are considered. The functions observed in a biological artifact such as an armadillo armor may not be observed in regular household products. Products from a closely related domain are chosen for the study to demonstrate the extraction of statistically significant set of grammars from a relatively small dataset.

Table 11. List of products studied for the graph grammar generation from the design repository

No.▼	Name of the product	No.▼	Name of the product... continued
0	Bissell Hand Vacuum	25	All in One Printer
1	Black and Decker Can Opener	26	Blower-VAC
2	Black and Decker Dustbuster	27	Induction Cooktop
3	Black and Decker Jigsaw	28	Tape Player
4	Black and Decker Palm Sander	29	CD Player
5	Black and Decker Rice Cooker	30	Coolit Drink Cooler
6	Braun Coffee Maker	31	Nextec Multitool
7	Brother Sewing Machine	32	Delta Circular Saw
8	Colgate Motion toothbrush	33	Delta Drill
9	Cordless Kettle	34	Delta Jigsaw
10	Delta Nail Gun	35	Delta Sander
11	DeWalt Sander	36	Dremel MultiMax
12	Dyson Air Multiplier	37	Dryer
13	Firestorm Circular Saw	38	RC Helicopter
14	Firestorm Drill	39	iPhone
15	First Alert Basic Smoke Alarm	40	Kid Smart Smoke Detector
16	Game Controller	41	Milwaukee Palm Nailer
17	GE Microwave	42	Oliso Frisper
18	Hot Air Popcorn Popper	43	Oliso Smart Iron
19	Milwaukee Copper Tubing Cutter	44	Oral B toothbrush
20	Neato Robotics Vacuum	45	Polaroid Pogo
21	Presto Salad Shooter	46	Random Orbital Sander
22	Shop-VAC	47	Ridgid Job Max
23	Black and Decker Electric Screwdriver	48	Gillete M3 Razor
24	Black and Decker Slice Right	49	Versapak Circular Saw

Creation of the Input

This part explains the creation of input for graph mining algorithm and the essence of implementing Graphsynth. The functional models of the products, listed in Table 11, are converted into a graph format. The conversion is explained with an example of Bissel hand vacuum. The functional model and image of the Bissel hand vacuum is shown in Figure 46.

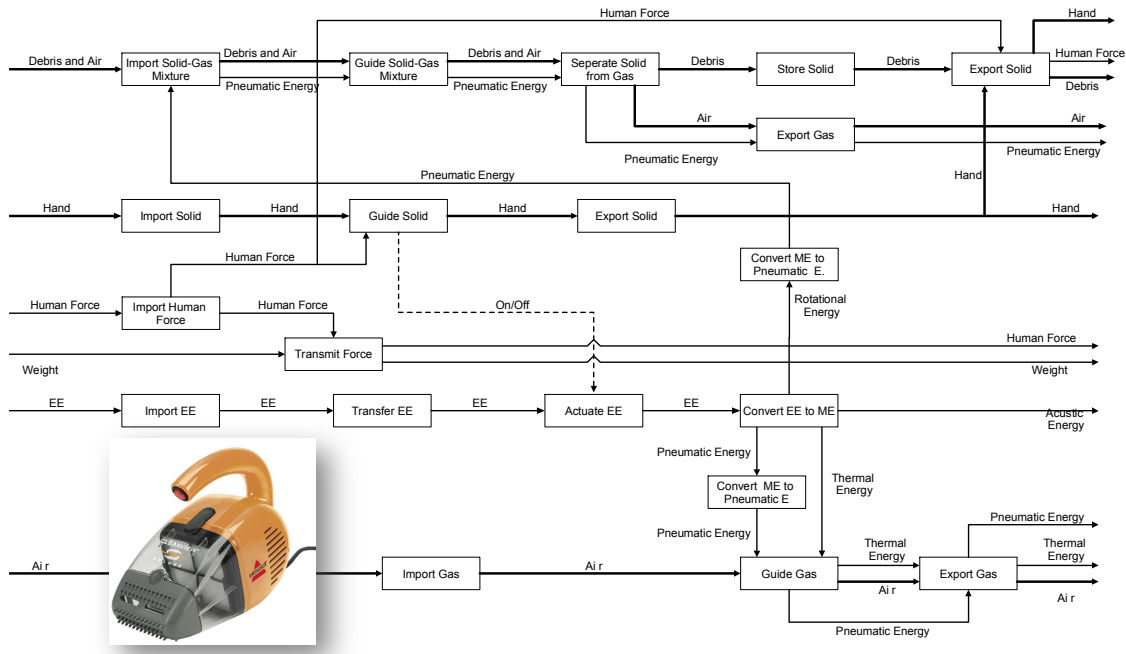


Figure 46. Functional model and image of the Bissell Hand Vacuum from the design repository [62, 97]

Functional model is a visual format that allows a reader to see all the flows connecting the functions. The information contained in a functional model can be captured in a directed graph. However, the mathematical format of a graph, as shown in Figure 47, is not very easy to visualize and this is the format that the graph mining algorithm understands. The primary motivation behind using graphical representation is to model the complex relations, but it is also essential that the format is human readable. Besides, the conversion of functional models into graph format should be simple and quick to justify the implementation of graph mining. The conversion and visualization of 50 graphs, or even more in future, can become time intensive process.

```

t # 0 Bissell Hand Vacuum
v 0 Import_Solid
v 1 Guide_Solid
v 2 Seperate_Solid
v 3 Store_Solid
v 4 Export_Solid
v 5 Export_Gas
v 6 Export_Hand
v 7 Guide_Hand
v 8 Import_Hand
v 9 Import_HE
v 10 Transmit_HE
v 11 Convert_ME_PE
v 12 Import_EE
v 13 Transmit_EE
v 14 Actuate_EE
v 15 Convert_EE_ME
v 16 Convert_ME_PE
v 17 Guide_Gas
v 18 Export_Gas
v 19 Import_Gas
u 2 3 Material
u 3 4 Material
u 4 6 Material
u 4 9 Energy
u 7 9 Energy
u 9 10 Energy
u 12 13 Energy
u 13 14 Energy
u 14 15 Energy
u 17 19 Material
u 16 17 Energy
u 15 17 Energy
u 15 16 Energy
u 11 15 Energy
u 0 11 Energy
u 7 8 Material
u 6 7 Material
u 7 14 Signal
u 17 18 Energy & Material
u 2 5 Energy & Material
u 1 2 Energy & Material
u 0 1 Energy & Material

```

Figure 47. The mathematical (txt) format of a graph for the Bissell Hand Vacuum

To overcome these issues, GraphSynth is implemented for converting functional models into graphs. The nodes are the functions in a product and edges are the flows between the functions; namely, material, energy, or signal. GraphSynth is a well-established package originally developed for interactive creation of graphs with the help of graph grammar rules [98]. GraphSynth enables interactive creation of graphs in editable gxml (guideline extensible markup language) format.

The gxml format of the Bissell Hand Vacuum is shown in Figure 48. There is also a provision to save the xml (extensible markup language) format of the same graph, refer Figure 49. The xml version of the graph is a machine-readable format while gxml is human-readable. Both gxml and xml formats of the graph can save all the pertinent

information of a graph such as node numbers, node label, edge connections, edge labels, node locations, display type of nodes, display type of edges. The information required for graph mining is node numbers, node labels, edge connections, and edge labels. The capability of GraphSynth to display all the relations in a graph at once is valuable for graph mining implementation.

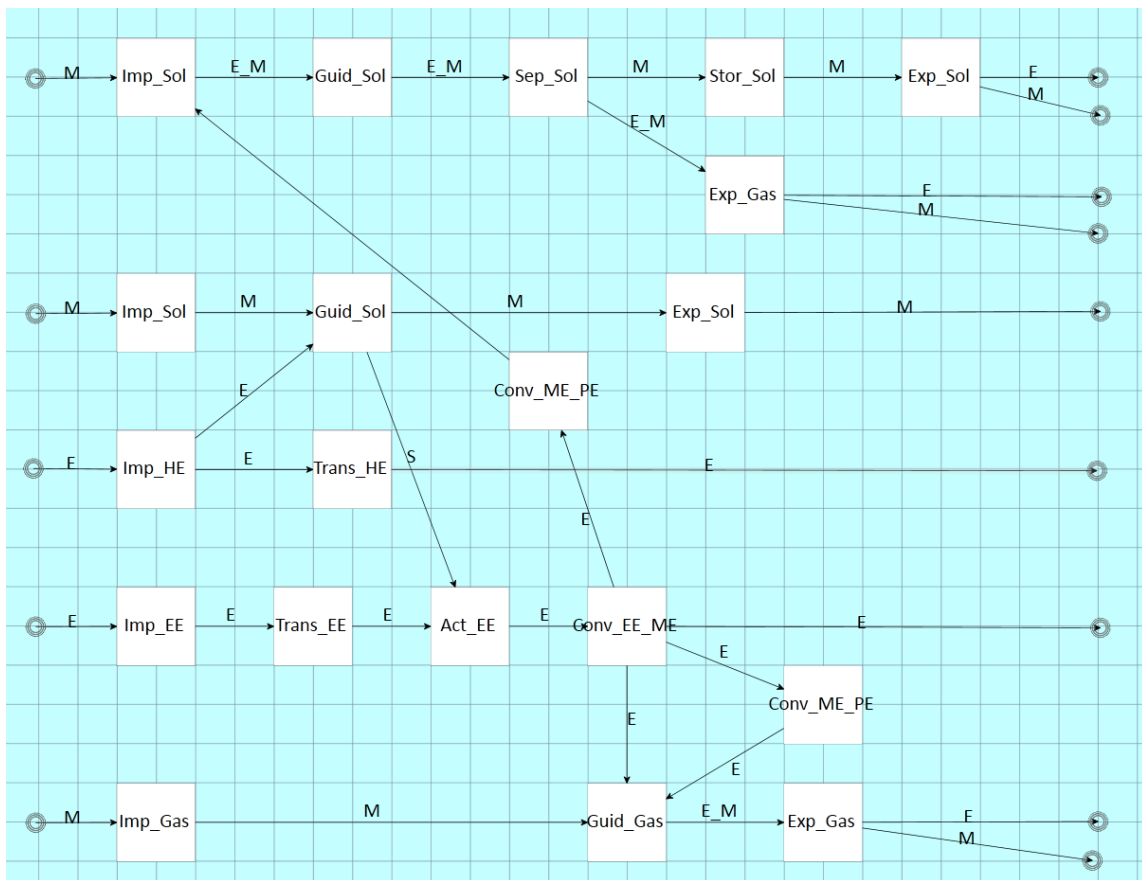


Figure 48. GraphSynth (gxml) format of graph for the Bissell Hand Vacuum

```

- <node>
  <name>n16</name>
  - <localLabels>
    <string>Conv_ME_PE</string>
  </localLabels>
  <localVariables/>
  <X>928</X>
  <Y>263.71428571428578</Y>
  <Z>0</Z>
</node>
- <node>
  <name>n17</name>
  - <localLabels>
    <string>Guid_Gas</string>
  </localLabels>
  <localVariables/>
  <X>1008.952380952381</X>
  <Y>101.80952380952385</Y>
  <Z>0</Z>
</node>
- <node>
  <name>n18</name>
  - <localLabels>
    <string>Exp_Gas</string>
  </localLabels>
  <localVariables/>
  <X>1218.4761904761904</X>
  <Y>120.85714285714289</Y>
  <Z>0</Z>
</node>
- <node>
  <name>n19</name>
  - <localLabels>
    <string>Imp_Gas</string>
  </localLabels>
  <localVariables/>
  <X>489.90476190476193</X>
  <Y>111.33333333333348</Y>
  <Z>0</Z>
</node>
</nodes>
<hyperarcs/>
</GraphSynth:designGraph>
</Page>

- <arc>
  <name>a44</name>
  - <localLabels>
    <string>S</string>
  </localLabels>
  <localVariables/>
  <From>n7</From>
  <To>n14</To>
  <directed>true</directed>
  <doublyDirected>>false</doublyDirected>
</arc>
- <arc>
  <name>a0</name>
  - <localLabels>
    <string>E_M</string>
  </localLabels>
  <localVariables/>
  <From>n17</From>
  <To>n18</To>
  <directed>true</directed>
  <doublyDirected>>false</doublyDirected>
</arc>
- <arc>
  <name>a6</name>
  - <localLabels>
    <string>E_M</string>
  </localLabels>
  <localVariables/>
  <From>n2</From>
  <To>n5</To>
  <directed>true</directed>
  <doublyDirected>>false</doublyDirected>
</arc>
- <arc>
  <name>a3</name>
  - <localLabels>
    <string>E_M</string>
  </localLabels>
  <localVariables/>
  <From>n1</From>
  <To>n2</To>
  <directed>true</directed>
  <doublyDirected>>false</doublyDirected>
</arc>

```

Figure 49. A part of xml format (extensible markup language) for the Bissell Hand Vacuum

Graph Data Mining

A frequent subgraph-mining algorithm (FSG), a software package developed by Karypis at the University of Minnesota, is used in this research to mine the undirected frequent subgraphs [94]. Frequent patterns are discovered based on the user specified threshold value of support, where support is the percentage of frequent subgraphs found

in the database. Other graph mining algorithms that provide similar capabilities are gSpan (Graph-Based Substructure Pattern Mining), FFSM (Fast Frequent Subgraph Mining), PEGASUS (Peta-scale graph mining system), and Apriori based Graph Mining (AGM) [50, 52, 99-101]. The FSG is validated by other researchers and available in a stable executable format, hence the FSG is chosen for this study [102]. Moreover, the FSG incorporates a provision to generate a transaction identifier list (TID list), which helps to trace a frequent pattern back to the parent graph. The actual embodiment of the frequent pattern from the graph of the parent product can serve as an example of the design heuristic.

Fifty individual graphs of the functional models are created using Graphsynth and saved in xml format. The JAVA GUI executes a code to convert each of the graphs in xml format into txt format. Essentially, the code extracts the required information from xml file, namely node numbers, node labels, edge connections, and edge labels to represent it in the format shown in Figure 47. Additionally, the code combines 50 graphs into one single text file, which serves as the input for the graph mining algorithm.

The FSG algorithm is executed to extract frequently occurring subgraphs with a support of 6 % or more. A support of 6 % implies that the frequent patterns are observed in at least 3 out of 50 graphs. The current implementation of the FSG mines only undirected graphs. The FSG is set to generate the TID list and parent-child list. The TID list is an important feature of the FSG algorithm.

Interpretation of the Output

The Java Universal Network/Graph Framework (JUNG) is a software that provides a common and extendible language for modeling, analysis, and visualization of data represented as graphs [103, 104]. JUNG provides a visualization framework in JAVA, which is implemented here to visualize the frequent subgraphs mined by the FSG algorithm. Similar to the difficulty in comprehending the txt format of the input graphs, the output format is also difficult to read and analyze.

Figure 50 shows the sample output from the FSG. Of note, txt format cannot be converted into gxml format since the gxml format requires the additional layout information to display the graph. Furthermore, the editable graphs are not necessarily required for analyzing the FSG output. The JAVA GUI incorporates the features of JUNG to create a visual of the graph for analyzing the output. The JAVA GUI separates the subgraphs with a specified size and frequency and displays the separated subgraphs using JUNG routines. Figure 51 displays visualization of the frequent pattern shown in Figure 50. With the help of JAVA GUI, the output from the FSG can be analyzed more efficiently to establish the design heuristics.

The implementation is demonstrated here with the graph grammar application but the process can be applied to any other relational database to extract frequently occurring trends. Essentially, the automated process where three softwares GraphSynth, FSG, and JUNG are integrated with the JAVA GUI can mine patterns from relational databases. The graph mining is particularly suitable for discrete or categorical variables

that occur commonly in product design space. The automated implementation is useful exploratory data analysis for the relational databases.

```
# Options -----
#   Min Output Pattern Size:           1
#   Max Output Pattern Size:         2147483647(INT_MAX)
#   Min Support Threshold:           10.0% (5 transactions)
#   Generate Only Maximal Patterns:   No
#   Generate PC-List:                 Yes
#   Generate TID-List:                Yes
#
t # 2-1, 10
v 0 Trans_EE
v 1 Act_EE
v 2 Sup_EE
u 0 1 E
u 0 2 E
t # 2-5, 10
v 0 Chang_ME
v 1 Conv_EE_ME
v 2 Trans_ME
u 0 1 E
u 0 2 E
```

Figure 50. A sample of the output format generated by the FSG (frequent subgraph-mining) algorithm

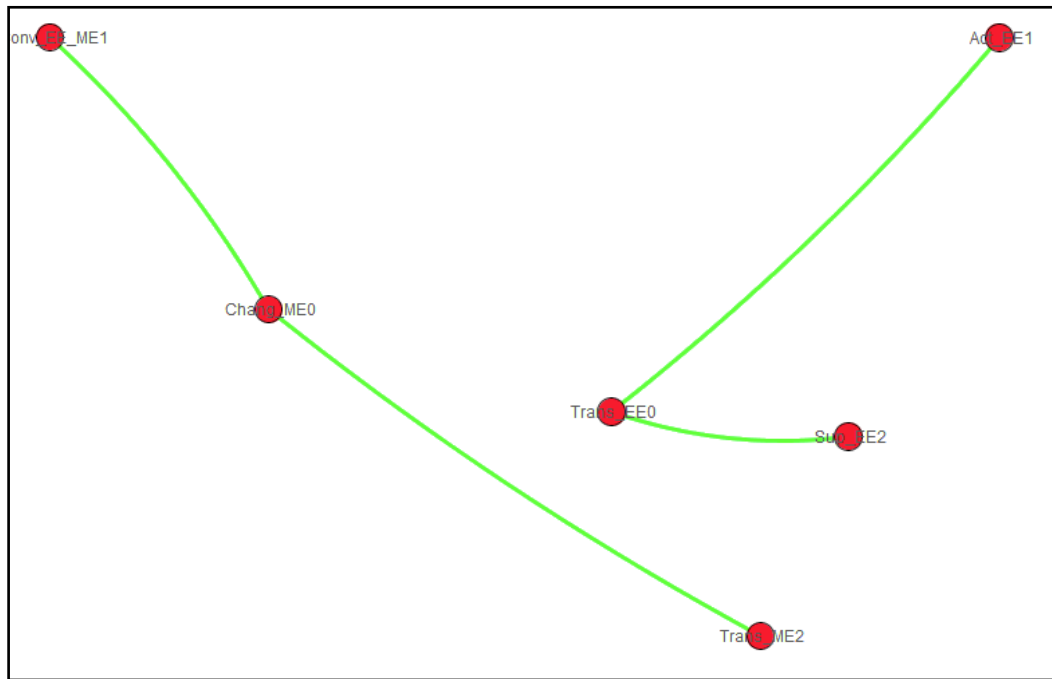


Figure 51. Visualization for the output with JUNG (Java Universal Network/Graph Framework) and green color of edge symbolizes the energy flow.

Graph Grammar Generation

As mentioned earlier in the research approach, the JAVA GUI has capability to flip the nodes and edges in a graph. The code reads the gxml file where functions are nodes and flows are edges and converts it to flipped txt format where the nodes are flows and the edges and functions. This feature is added since the current implementation of graph mining can mine subgraphs consisting of nodes and edges connecting those nodes. In other words, the edge with only one node connection is not included in the frequent subgraphs. Hence, the flipping feature is added to provide this flexibility. Even with flipping the relations are preserved and there is no loss of information from original gxml format. The GraphSynth, the graph mining, and the JUNG implementation remain

unaltered when flipping nodes and edges. Only the code to convert .xml to .txt is modified to flip the nodes and edges.

The reason for using flipping feature for graph grammar extraction is the nature of graph grammars as given in the rule set derived by Campbell [98, 105]. The graph grammars to be derived have flows coming into the nodes and it contains edges with only one node connection, refer Figure 52. The researchers at University of Texas at Austin derived a set of 69 grammar rules for construction of functional models. These grammar rules will be referred as Campbell's Grammar Rules in the discussion below. Figure 52 illustrates Campbell's Grammar Rule 5, which is a propagation rule. Campbell describes Rule 5 as "*Rule 5, a typical propagation rule, recognizes Electrical Energy from the function 'Import'. It then adds the functions 'Transmit' and 'Actuate' to the head of the open flow and creates another open flow in the front. This is a pretty common module in most products as it represents the familiar electric plug point used to import electricity, the wire to transmit electricity and some kind of switch to actuate electricity.*" [98, 105].

Rule 5

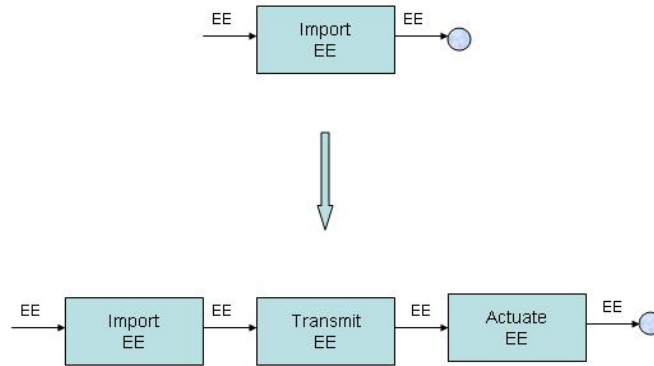


Figure 52. Rule 5 from Campbell's graph grammar rule set illustrating a propagation rule [98]

The Campbell's Grammar Rule set is divided into three categories for comparison with the FSG output results: (A) the rules that can be mined with the current setup, (B) the rules that cannot be mined with the current setup, and (C) the rules cannot be mined. The current setup has two restrictions, namely the database and the current implementation of the FSG. If certain functions or flows are not included in the database then the rules related to those will not be mined; these rules are classified in the B1 category which contains 11 rules. The current implementation of the FSG cannot handle nodes with multiple edges connected to them. The rules with multiple input and output edges are categorized as B2 and contain 29 rules. The rules from the category B1 can be mined in future by adding more products to the database. Similarly, rules from the category B2 can be mined in future by implementing an improved version of graph mining algorithm that can handle multiple edge connections. Thus, the category B

consisting of 58% of Campbell’s Grammar Rules cannot be mined with the current setup.

Table 12. Campbell’s rule set categories

Categories	Description	#		Note
A: Rules that can be mined with current setup	Found	5	28%	26% found
	Found (not exact match)	10		53% not exact
	Not found	4		21% not found
B: Rules that cannot be mined with current setup	With multiple input and output	29	58%	Future work – improve graph mining algorithm
	Words not in current database	11		Future work – add more products to the database
C: Rules that cannot be found with graph mining	Initiation or termination	4	14%	Cannot handle
	Pre-primary initiation	6		Cannot handle
	Total rules	69		

Only the expert can derive rules from the category C1, which consists of initiation and termination rules. The initiation and termination rules cannot be derived with machine learning. Similarly, the pre-primary initiation rules, categorized here as C2, from the Campbell’s Rule set cannot be derived with the FSG. Thus, only the expert can derive the rules from category C which constitutes 14% of the Campbell’s Grammar Rule set.

Rules from the category A can be derived with the current setup, which constitutes 28% of Campbell’s Grammar Rule set. Out of the 19 rules that can be found, 5 exactly matching subgraphs are actually found. Ten rules from Campbell’s grammar rule set are mined by the FSG but they do not match exactly with the expert-derived

grammar rule. Remaining rules from category A are not found. Thus, only 21% of the rules that should have been derived by the FSG are not found.

Graph Grammar Results

Figure 53 show a frequent subgraph mined that matches with Campbell’s Grammar Rule 5. The interpreted graph grammar is also shown in Figure 53 which states that transmit and actuate functions are added after the import electrical energy function. This module is commonly observed in devices with plug point to import electricity, the wire to transmit electricity, and a switch to actuate electricity.

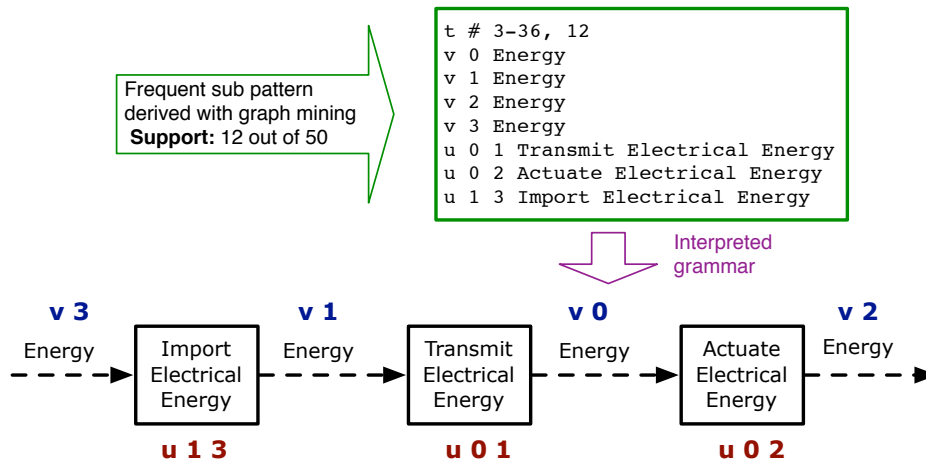


Figure 53. A frequent subgraph obtained with graph data mining corresponding to Rule 5 from Campbell’s set having a frequency of 12

The machine-derived grammar has a support of 12 and the TID lists records the products in which this module occurs, refer Table 13. Thus, the subgraph $t \# 3-36$ occurs in First Alert Basic Smoke Alarm, CD Player, Game Controller, Craftsman Nextec Multi-tool, Neato Robotics – Vacuum, RC Helicopter, All in One Printer, KidSmart

Smoke Detector, Induction Cooktop, Milwaukee Palm Nailer, Tape Player, and Ridgid JobMax.

Table 13. Transaction identifier list for subgraph t # 3-36 corresponding to Rule 5

TID No.	Product name	Product Image	TID No.	Product name	Product Image
15	First Alert Basic Smoke Alarm		29	CD Player	
16	Game Controller		31	Craftsman Nextec Multi-tool	
20	Neato Robotics - Vacuum		38	RC Helicopter	
25	All in One Printer		40	KidSmart Smoke Detector	
27	Induction Cooktop		41	Milwaukee Palm Nailer	
28	Tape Player		47	Ridgid JobMax	

An important result of automated graph grammar extraction is that it not only predicts manually derived grammars but also adds additional information to it. Figure 54 shows the Campbell's Grammar Rule 24, the corresponding frequent subgraph, and subgraph with added information. Campbell's Grammar Rule 24 recommends the addition of a position solid function after the import solid function. The subgraph t # 2-155 states the same. Interestingly, the subgraph t # 3-1 derives the same grammar rule with addition of a secure solid function after the position solid function. Subgraphs like t # 3-1 can help to improve manually derived graph grammar rules. Three other exactly matching grammars are listed in Appendix C along with the original Campbell's Grammar Rules.

Summary

This study exploits the application of three established software that were originally developed for different applications into a single automated method to extract patterns from a database consisting of graphs. The automated method consisting of the GraphSynth, the FSG and the JUNG along with the JAVA GUI is particularly helpful in the developmental stages of graph mining applications for product design. The automated graph grammar method can derive the grammar rules that are originally derived by the experts. In addition, the process can be used to improve existing grammar rules.

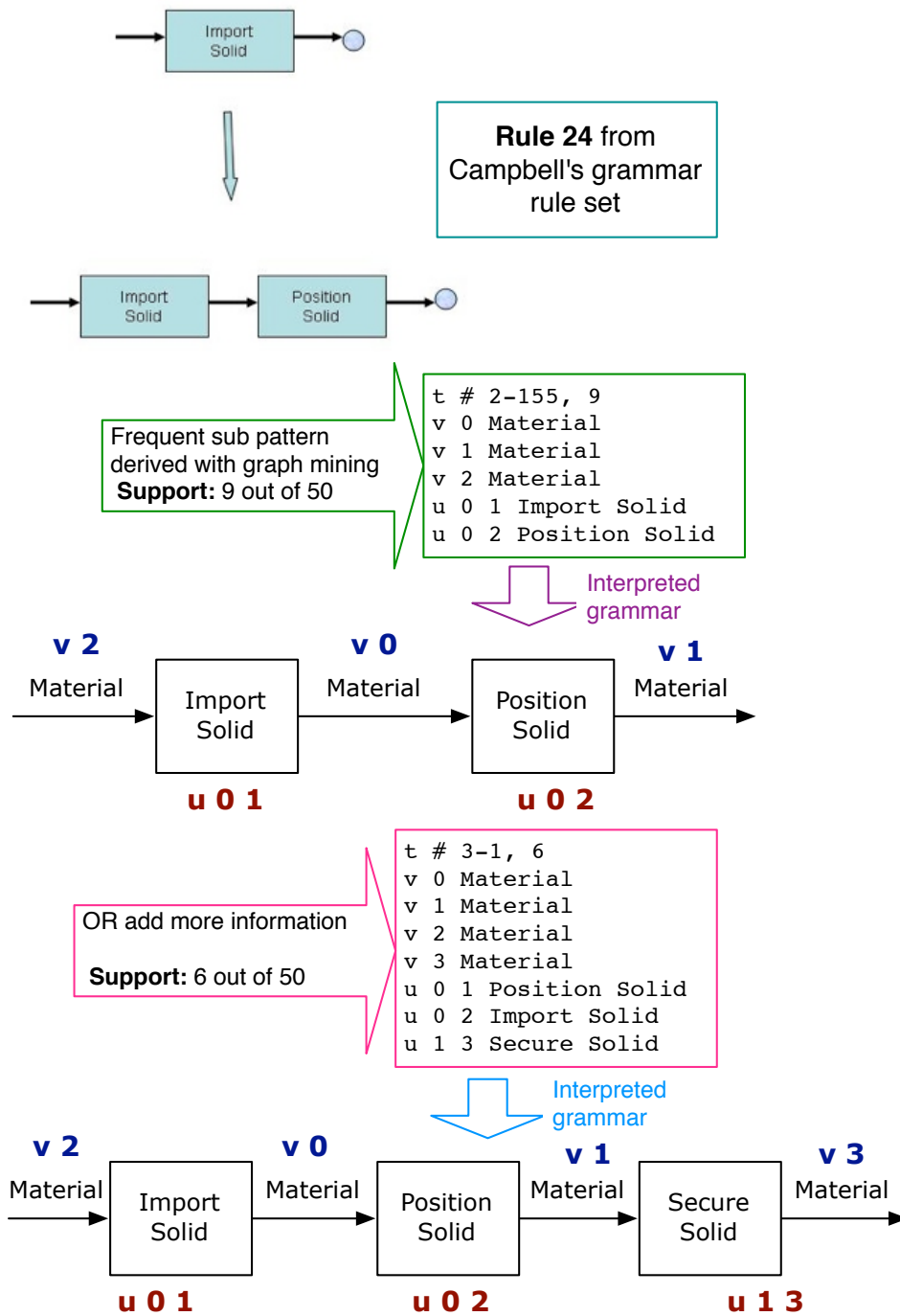


Figure 54. Rule 24 from Campbell's graph grammar rule and the corresponding frequent subgraph along with a graph with more details

In layman's term, we can teach the machine to guide the designers and researchers. Machine learning is quick and efficient compared to empirical heuristics developed by experts. Besides, the graphical representation of function structures has potential for generalization and automation. The potential outcome of the graphical data mining for product design is the capability to continually update the design repository and search for new design heuristics. This process can also be applied to product family design for finding commonalities to identify a product platform. In general, this chapter explores the feasibility and advantage of graphical data mining for product design.

Further, Chapter VIII, Chapter IX, and Chapter X describes the validation of inclusive design heuristics, rating to measure inclusivity, and dissemination of the knowledge with teaching module.

CHAPTER VIII

INCLUSIVITY RATING ⁶

Ideation ratings have been developed for measuring quantity, quality, novelty, and variety of the generated concepts [6, 7, 22, 24]. Currently, there is no pre-defined rating to measure the inclusiveness of a design. This chapter introduces an inclusivity rating to measure the inclusiveness of conceptual designs. The inter-rater reliability of the inclusivity rating is checked before applying it to this research.

Srivasthsavis et al. suggests that coarser scales when applied at a feature level yields higher inter-rater reliability [24]. Hence, the ratings used in this study have a coarse scale with 4-points. The ideas from the study are evaluated at a feature level rather than at general concept level. The experimental data is divided into features prior to rating. Features are used to count the number of ideas generated by a participant. If an idea addresses two features, then the idea is rated twice for each feature. All of the features are weighted equally in this research.

The researcher who divided the data is not involved in the ratings. All of the identifying information about the participant or the condition is removed before rating the ideas. The raters are blind to the condition of the ideas they are rating, whether control group or experimental group. The raters are graduate students in mechanical

⁶ Reprinted with permission from “An Exploratory Study on the Effectiveness of an Inclusive Design Tool with a Metric to Evaluate Inclusivity of Conceptual Designs” by Sangelkar, Shraddha, and McAdams, Daniel A., 2013, Proceedings of the ASME 2013 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Portland, OR, Copyright 2012 by Shraddha Sangelkar.

engineering at Texas A&M University with an engineering design research background. The main rater rates all of the ideas for quality and inclusivity separately. The second rater rates only a part of the data set for both design problems to ensure inter-rater reliability. The quality and inclusivity rating are explained further.

Quality Rating

Figure 55 shows the quality rating applied to rate the quality of the ideas in this study. The rating is a 4-point scale based on the quality rating from Linsey [6, 7]. Participants are instructed to generate commercially viable solutions. The idea of a commercially viable solution is the one that can be sold in a competitive market; either the price should be comparable with other regular products or the benefits should be sufficient to justify the added cost. This condition ensures that the ideas suggesting complete automation are not necessarily rated high on the quality scale. The feasible and commercially viable ideas are further divided based on the complexity of the change.

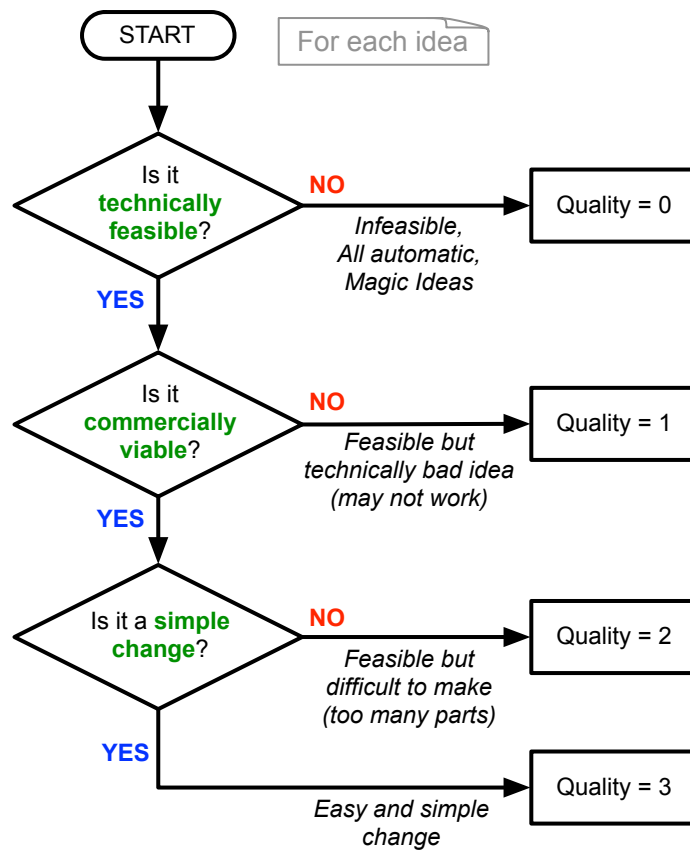


Figure 55. Rating to measure quality of ideas

Table 14 explains the quality rating with examples. The ideas with quality of 0 are the ideas that are infeasible with current state-of-the-art technology for a household purpose. For instance, ideas such as *buy a ninja* or *drink water intravenously by injecting a needle* instead of using an ice-water dispenser count as quality of 0. Note that *drinking water intravenously* or *cooling food with liquid nitrogen* is technically feasible but not feasible for application in home appliances. The ideas that are not completely infeasible but may not be commercially viable are rated as quality of 1. Examples of the ideas with

quality of 1 include *food processor blades made of rubber or wire*, or *implementing an outdoor box-like refrigerator* in places with extremely cold weather.

Table 14. Examples of quality scale

Quality	Meaning	Examples
Q = 0	Infeasible with state-of-the-art technology	1. Buy a ninja 2. Use of liquid nitrogen to cool food 3. Drink water intravenously
Q = 1	Feasible change which may not work	1. Blades made of rubber or wire 2. Outdoor refrigerator in cold places 3. Hydraulically actuated mesh cutter
Q = 2	Feasible but complicated change	1. Mechanism with electromagnetic force 2. Refrigerator like vending machine 3. Automatic blade changer in food processor
Q = 3	Simple change	1. Larger buttons or touch screen 2. Spout to pour liquid 3. Spring actuated mechanisms

The ideas that are feasible and commercially viable are categorized as quality of 2 or 3 based on the complexity of change. The changes that are complicated and significantly add to the product's cost are considered to be complex changes with a quality of 2. For the purpose of analysis the cost factor is assumed to be based on engineering complexity of the idea. For instance, adding *a spout to drain liquid* from a food processor is a simpler change as compared to an *automatic mechanism for changing the blades*.

Inclusivity Rating

Figure 56 shows the rating used to measure inclusivity of ideas. Similar to the quality rating, the inclusivity rating is also a 4-point scale with 3 indicating high inclusivity and 0 indicating low inclusivity. Figure 57 shows a Venn diagram to provide further the explanation for exclusion of people with or without disability as categorized in the inclusivity rating.

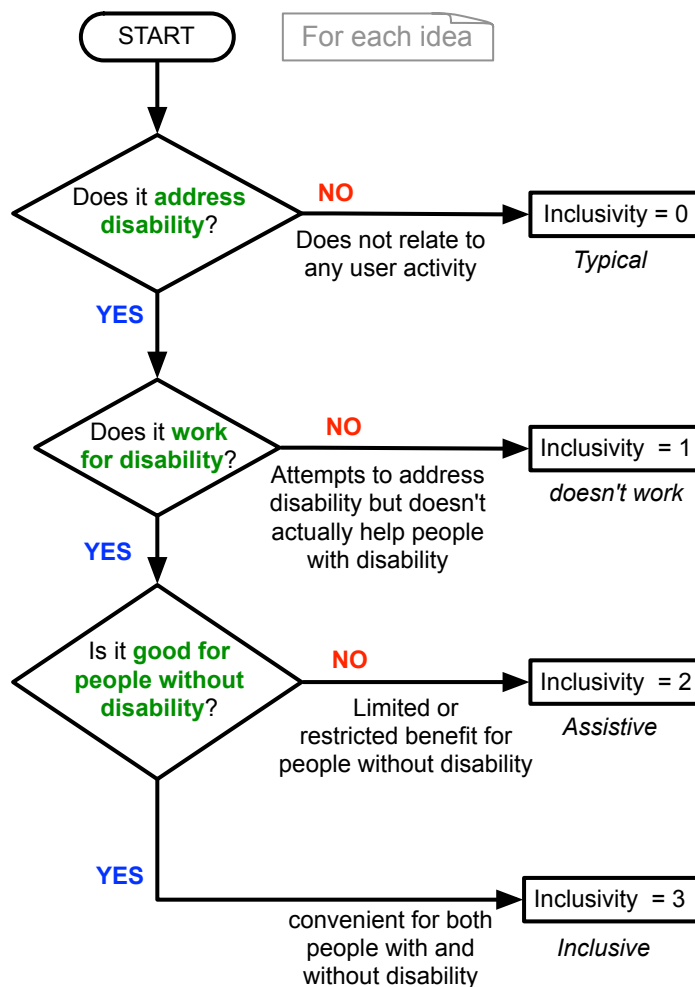


Figure 56. Rating to measure inclusivity of ideas

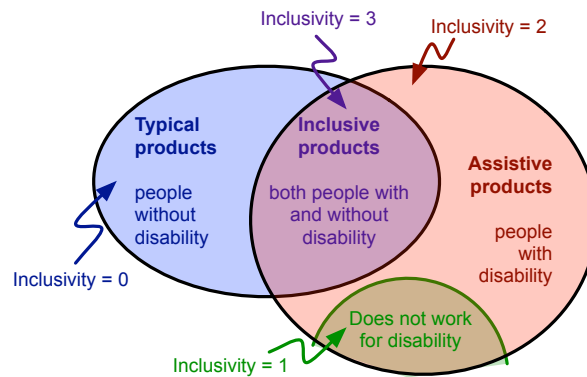


Figure 57. Explanation for the inclusivity rating with a Venn diagram to show the exclusion of people

Table 15 provides the examples of the inclusivity scale. The ideas dealing with internal functions of the device, the functions with which the user does not interact, do not address any disability. The ideas such as using *a metallic grey color* for the device may address other customer needs but do not address any aspects of disability. Since safety is required for all, whether or not designing for disability, safety-related ideas are also considered to have an inclusivity of 0.

The ideas that attempt to address disability issues, but do not serve as intended are assigned an inclusivity of 1. These ideas are related to some user activity, but they do not alleviate any disability. The ideas such as *soundproofing* are related to the user activity of hearing but it does not serve any specific disability for the food processor design. While it might be nice to have a quietly operating food processor, soundproofing is more of a desired feature than a design for disability. Other ideas such as *putting a ramp in front of the refrigerator* attempt to address the limitation of people on a

wheelchair, but it is not practically possible to open a refrigerator door while sitting in a wheelchair on a ramp.

Table 15. Examples of inclusivity scale

Inclusivity	Meaning	Examples
I = 0	Ideas that do not relate to any user activity	1. Anything related to aesthetics 2. Improved cutter designs 3. Any electrical energy related ideas
I = 1	Ideas related to some user activity but does not work for disability	1. Soundproofing 2. Ramp for refrigerator 3. Use knife and cutting board
I = 2	Ideas that address disability but not good for people without disability	1. Provide a magnifying glass 2. Complicated assistive devices 3. Shorter and wider refrigerator
I = 3	Ideas that are good for both people with and without disability	1. Automatic garage door opener 2. Motion sensing automatic faucet 3. Retractable electric cord

The ideas that help people with a disability but have limited or restricted benefit for those without a disability are assigned an inclusivity of 2. These are features of the product that add substantial efforts or time requirements for a person without any disability. In other words, these features interfere with the normal activities of an able-bodied individual. Finally, the ideas that are good for both people with and without disability are assigned an inclusivity of 3. For instance, a wider population generally accepts ideas such as an *automatic garage door* or a *motion sensing automatic faucet*. Of note, personal preference should not be considered while categorizing ideas on the

inclusivity scale of 2 or 3. For instance, *a shorter and wider refrigerator* is better for people in a wheelchair but cumbersome for taller individual or someone who has difficulty in bending over to a lower height. Thus, *a shorter and wider refrigerator* idea is given an inclusivity value of 2, while *a refrigerator with adjustable height* is given an inclusivity value of 3.

Improving, or measuring, the general creativity of the concepts is not the central focus of this research. Hence, other ideation metrics like quantity and novelty are not considered for this study. The purpose of the inclusive design representation scheme is to improve inclusiveness of a conceptual solution by shifting the designer's focus from the internal functions of the device to the user-product interactions. For example, to design an inclusive automobile the designer must redesign the steering wheel or the gas pedal but not the engine or wheels of the automobile. From an inclusive design perspective, the engine and wheels are internal functions of the automobile and do not influence the inclusive nature of the design. While the creativity and inclusiveness of a design cannot be totally separated, the analysis in this chapter is limited to the measurement of quality and inclusivity of the ideas generated in the study.

Reliability of the Ratings

This part reports the weighted Cohen's Kappa values for the inter-rater reliability of the quality and inclusivity rating. Thirty-two ideas from the excluded participants are randomly selected as a training set to develop the inclusivity rating. Two raters rated the training set three times each. With each rating iteration, the rating is revised to reduce

any ambiguity in description. For the training set, the linear weighted kappa values from the final iteration for the quality and inclusivity are 0.46 and 0.68, respectively.

Summary

This chapter introduces an inclusivity rating to evaluate the inclusiveness of conceptual solutions. The inclusivity rating is a 4-point scale similar to the quality rating developed for measuring ideation [6]. The inclusivity rating will act as a benchmarking tool for various methods for inclusive design. The rating is easy to apply for rating ideas at a feature level in the preliminary stages of design. I acknowledge that a more thorough analysis and validation of the rating would be helpful, but given the fact that currently no rating exists for inclusive design at the conceptual stage, the inclusivity rating developed is an important step forward.

CHAPTER IX

EXPLORATORY STUDY⁷

The previous chapters detailed a specific representation scheme to support inclusive design. The representation scheme aims to guide the designer in the early stages of the inclusive design process, in which product functions are established and solution concepts are generated. This chapter explores the applicability of the representation scheme for inclusive product design.

The inclusive design representation scheme enables designers to apply an empirically derived set of inclusive design rules to the actionfunction diagram of a product. In this chapter, an exploratory study is conducted to test the hypothesis “*the application of inclusive design representation scheme helps to generate better ideas for inclusive design.*” Participants are asked to generate ideas for inclusive product design with and without the representation scheme. The difference in the quality and inclusiveness of ideas generated in the two conditions is compared to determine the effectiveness of the representation scheme.

Ramachandran conducted a user study to evaluate difference between the quality and quantity of ideas generated with functional model versus function interaction model (FIM user study) [25]. The passive functions in a FIM or the functions that are

⁷ Reprinted with permission from “An Exploratory Study on the Effectiveness of an Inclusive Design Tool with a Metric to Evaluate Inclusivity of Conceptual Designs” by Sangelkar, Shraddha, and McAdams, Daniel A., 2013, Proceedings of the ASME 2013 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Portland, OR, Copyright 2012 by Shraddha Sangelkar.

performed on the product are similar to the user-product interactions in an actionfunction diagram. For instance, the passive function ‘insert tortilla filling’ is modeled as ‘product function = import solid’ and ‘user activity = carrying, moving, and handling objects’ in an actionfunction diagram [25]. The FIM user study is very similar to this exploratory study except for two major differences. FIM user study aims to capture the creativity of designers while this study aims to capture the inclusivity of the ideas generated. Secondly, a functional model is provided in the control condition of the FIM user study while no design tool is provided in the control condition of the exploratory study.

The results of the exploratory indicate that the inclusive design representation scheme helps in the design of products in the architectural domain where the space around the product is an important consideration. Based on the experiment presented here, the impact of the representation scheme for consumer products cannot be determined with certainty. The participant self-reported feedback about the usefulness of the representation scheme is positive. The representation scheme has potential to improve inclusive design but further investigation is needed to verify its specific impact on various product domains.

Exploratory Study Setup

This part describes the study procedure, characteristics of the study population, and the data analysis procedure.

Study Procedure

Table 16 explains the outline of the exploratory study procedure. Participants are randomly divided into two groups, A and B. Two product design problems are

considered for the exploratory study: design of an inclusive food processor and design of an inclusive refrigerator. The design problem from the control condition of group A is used for the experimental condition of group B and vice versa. Each design problem has a control group and an experimental group. With this experimental setup, each participant spends an equal amount of time on the study. In addition, each participant experiences designing with and without the representation scheme, and all participants are trained to apply the inclusive design representation scheme.

Table 16. Outline for the exploratory study

Task	Group A	Group B	Time (min)	Description
Introduction	-----	-----	5	Consent form, General instructions Introduce inclusive design
Control Condition	Food processor	Refrigerator	45	Introduction to the problem Concept Generation
Break	-----	-----	5	-----
Training	-----	-----	20	Inclusive design representation scheme
Experimental Condition	Refrigerator	Food processor	45	<i>(With inclusive design representation scheme)</i> Introduction to the problem Concept Generation
Participant Feedback	-----	-----		Participant demographics Feedback on the study
Total time	-----	-----	120	Up to 2 hours

The experiment begins with an introduction to the study, consent procedure, and an introduction to inclusive design. In the control condition, the participants are introduced to the problem statement followed by a concept generation activity. The total time of the control run is 45 minutes. Participants are given a 10-minute break after the

control run. After the break, participants are introduced to the concept of an actionfunction diagram and inclusive design rules using a pre-recorded training session. A training questionnaire is administered after the training to test whether participants understand the inclusive design representation scheme.

The experimental run follows the training session. Participants receive the actionfunction diagram of the product along with the problem statement for the experimental condition and they have access to the inclusive design rules provided in the training session. Total time for the experimental run is 45 minutes including the time allowed for studying the problem statement and the actionfunction diagram. Participant feedback and demographics are collected at the end of the study.

The study is conducted with groups of two to six students in engineering classrooms on weekends and weekday evenings. To maintain uniformity in the experimental procedure, the instructions and the training are pre-recorded and played at the time of study. Also, study packets are created with labeled sections: control condition labeled task 1, training session with inclusive design rules and the training questionnaire labeled training, and experimental condition with actionfunction diagram labeled task 2. Participants are instructed to open the appropriate section at the beginning of each activity. The packets for group A and group B are color coded to ensure even distribution.

The problem statements given to the participants state: *“Assume you are a product designer in a large consumer products manufacturing firm. The company is planning to launch a universally designed [product name],”* where the *[product name]*

is *food processor* or *refrigerator*. Of note, the problem statement suggests redesign of existing products. Using a redesign problem such as this is consistent with companies redesigning common products to be inclusive. Performing an experiment with a similar setup for an abstract design problem, such as ‘design a device to cool food and drink’, remains as future work.

For the food processor design, both conditions receive a gray-scale image of the food processor along with the problem statement, as shown in Figure 58. The customer needs specify that the food processor must perform grating, slicing, chopping, mixing, pureeing, and kneading dough.

<p>Problem Description Assume you are a product designer in a large consumer products manufacturing firm. The company is planning to launch a “universally designed food processor.”</p> <p>Definition: Food processor is an electric kitchen appliance used for chopping, mixing, or pureeing foods.</p> <p>Customer needs: Must perform following functions</p> <ul style="list-style-type: none">• Grating, slicing• Chopping• Mixing, pureeing• Kneading dough



Figure 58. Problem statement for the food processor design along with an image of the product

Participants do not receive any image for the design of the refrigerator. The problem description defines the refrigerator as an appliance or compartment that is artificially kept cool and used to store food, drink and make ice, as shown in Figure 59. For all conditions, the participants are instructed to generate as many ideas as possible,

record all the concepts that occur to them, include partial solutions, preferably sketch the idea, and write any notes they feel necessary. This study is investigating the inclusive design representation scheme on functionally dissimilar products.

Problem Description

Assume you are a product designer in a large consumer products manufacturing firm. The company is planning to launch a “**universally designed refrigerator.**”

Definition: Refrigerator is an appliance or compartment that is artificially kept cool and used to store food, drink and make ice.

Figure 59. Problem statement for the refrigerator design

The food processor and refrigerator have little common functionality. The products designed in the control condition should not influence the design of products in the experimental condition. The activities performed by the user in the two design problems and the complexity of the two products is different. The ideas from the study are evaluated at a feature level and the features of the food processor and the refrigerator are listed in Table 17.

Table 17. Features in a food processor and refrigerator

	No.	Features		Examples / Explanation
		Product Function	User Activities	
Food Processor	1	Import signals from user	Turning/ Manipulating	On/off, power level, types of food
	2	Export signals to user	Seeing/ Hearing	Clear view of mixing, sound, light
	3	Import food ingredients	Add food	Includes feeding food with a plunger
	4	Couple solid	Add/ remove attachments	Selecting the blade type
	5	Couple solid	Open/ close lid	Includes small opening for adding liquid
	6	Couple solid	Attach/ detach from motor	Fixing the container in place before operation, safety latch
	7	-	Lift container	Handles or feature to prevent slipping
	8	-	Remove food	Includes scrapping sides for better mixing or removing all food
	9	-	Wash container	Includes washing the attachments and blades
	10	Import/ transmit electrical energy	Plug/ unplug cord and Wind/ unwind wire	Includes activities related to the cord
	11	-	Store the food processor	Moving it to convenient location for operation or storage
	12	Electrical energy related	-	Transmitting and converting energy (or source of energy)
	13	Mixing and cutting related	-	Design of blades / cutters
	14	Safety related	-	Do not operate until closed, blade cutting hazard
Refrigerator	1	Guide solid	Open / close refrigerator	Includes door and handle design
	2	Position human	Maintain body position	Includes space around the product and the supporting features
	3	Regulate EE	Turning	Refrigerator controls: Includes temperature settings
	4	Export solid/ liquid	Drinking	All functions and activities related to ice/ water dispenser
	5	-	Store or search food items	Organization of compartments, including labels
	6	Electrical energy related	-	Includes import/ transmit/ convert energy
	7	Refrigeration cycle related	-	Includes function supporting the main cooling system

Study Population

The participants recruited for this study are both graduate and undergraduate mechanical engineering students who had prior training in basic design methods and functional modeling. The study population includes 26 males and 10 females. The participants received extra course credit for participation in the study. Group A consists of 9 undergraduate students and 8 graduate students and group B consists of 11 undergraduate students and 8 graduate students. It is essential that the participants completed both tasks for this study. One undergraduate student who completed only the first half of the experiment is excluded from the analysis. Another graduate student, who had prior exposure to the experimental procedure and the problem statement, is also removed from the analysis. Four participants received the same problem in both conditions due to a procedural error; hence, those participants are also excluded from the analysis. However, the ideas generated by the excluded participants are used as an initial training set for developing the inclusivity rating.

Participants are assigned randomly to the 2 groups. However, the self-reported GPA of the undergraduate students in the Group A is significantly higher than Group B (p -value 0.10). This may be a confounding factor in the study. No significant difference is observed between the GPA's of the graduate students divided into two groups. Also, there is no significant difference in the GPA values between the control and experimental groups when both graduate and undergraduate students are combined. For the purpose of this study it is assumed that both graduate students and seniors have similar abilities to generate ideas, and they all are treated as one sample.

Results of the Exploratory Study

Inter-rater reliability is tested for 50% of the data for each design problem. The data tested includes ideas from both the control and the experimental condition. The linear-weighted Cohen's Kappa values are reported in Table 18. The weighted Cohen's Kappa values show moderate agreement between the raters. Thus, the quality and inclusivity ratings used in this exploratory study are reasonably reliable.

Table 18. Weighted Cohen's Kappa (linear-weighted) for rating reliability for 50% of the data in the exploratory study

	<i>Quality</i>	<i>Inclusivity</i>
Food processor	0.53	0.51
Refrigerator	0.51	0.59

The following part discusses the results comparing the control and experimental condition for the average quality and inclusivity of the ideas generated by participants. The criteria for statistical significance is set at $\alpha = 0.10$ for the exploratory study. The average number of ideas generated per participants is 6 for the food processor problem and 8 for the refrigerator problem.

Figure 60 indicates that the average quality of ideas generated by participants in the control and experimental condition for the food processor problem is independent of the condition. The inclusive design representation scheme has no effect on the quality of ideas generated for the food processor design. On the other hand, Figure 61 shows that the average quality of ideas generated by participants for the refrigerator problem is

higher in the experimental condition and the difference is close to being statistically significant (p -value = 0.11). Thus, improvement in quality of ideas with the inclusive design representation scheme is likely problem dependent.

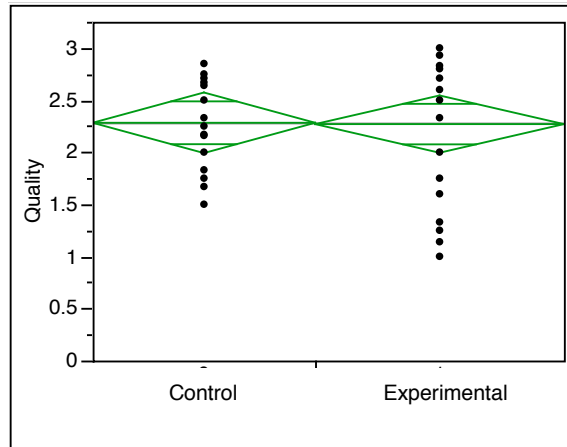


Figure 60. The average quality of ideas generated by participants in the control and the experimental condition for the food processor problem

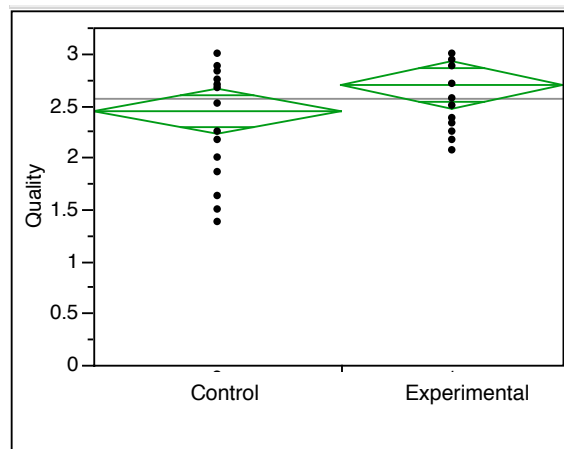


Figure 61. The average quality of ideas generated by participants in the control and the experimental condition for the refrigerator problem (p -value 0.11)

Figure 62 shows a decreasing trend in the average inclusivity of ideas generated by participants in the experimental condition as compared to the control condition for the food processor problem, but the difference is not significant (p-value 0.37). However, the average inclusivity of ideas generated by participants in the experimental condition of the refrigerator problem is significantly higher than the control condition (p-value 0.10), refer to Figure 63.

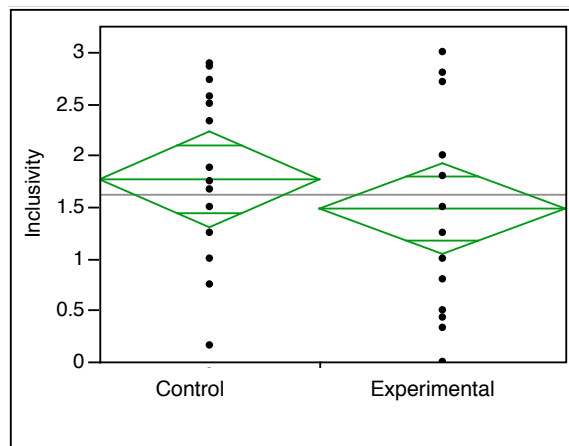


Figure 62. The average inclusivity of ideas generated by participants in the control and the experimental condition for the food processor problem (p-value 0.37)

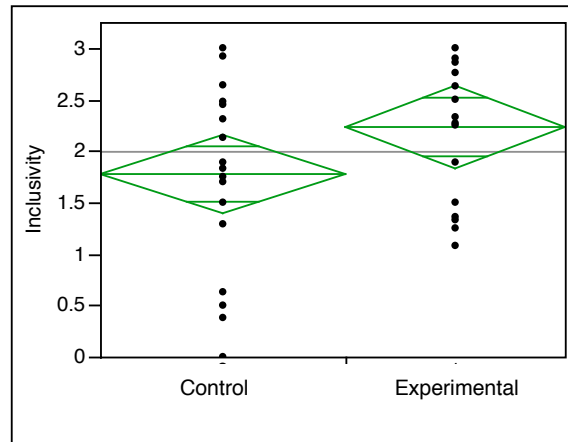


Figure 63. The average inclusivity of ideas generated by participants in the control and the experimental condition for the refrigerator problem (p-value 0.10)

There is no significant difference between the average quality and inclusivity of ideas generated by the participants in the two conditions for the food processor design. However, the average quality and inclusivity of ideas generated by the participants in the experimental condition is higher as compared to the control condition for the refrigerator design. The results of the study indicate that the efficacy of the inclusive design representation scheme is problem dependent.

To further confirm the results from the refrigerator problem; a two-tailed t test is performed between percentages of ideas with zero inclusivity generated by participants in the two conditions. Recall that the ideas with zero inclusivity are those related to internal functions of the device. A higher percentage of ideas with zero inclusivity indicates that the participant is generating ideas that are not even related to user activity. Whether or not the idea related user activity is helpful for any disability is a separate

concern. The designer should first consider the disability issues to be able to design effectively for a wider audience.

The percentage of ideas generated with zero inclusivity is significantly higher in the control condition as compared to the experimental condition (p-value 0.02) as shown in Figure 64. This result indicates that the representation scheme helps to shift the focus from internal functions of the device to user-activity related functions for the refrigerator problem. Similar analysis is performed for the food processor problem, but no significant difference is observed between the control and the experimental conditions.

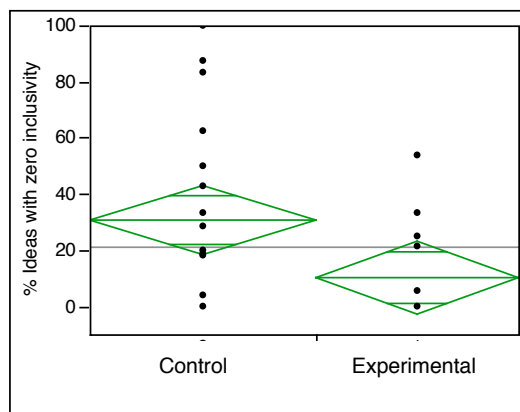


Figure 64. The percentage of ideas generated by participants that have zero inclusivity in the control and the experimental condition for the refrigerator problem (p-value 0.02)

Research has previously reported that the applicability of the ADA guidelines (*Americans with Disabilities Act*) is limited to the architectural products as compared to the consumer products [83, 106]. It is important to note that products are classified as

architectural products when the space around the product is an important consideration for inclusive design. In this context, a refrigerator is an architectural product while a food processor is a consumer product. The results of the study suggest that the inclusive design representation scheme helps designing products from the architectural domain. With the exploratory study there is not enough evidence to support the usefulness of the representation scheme for the products from the consumer domain.

As shown in Appendix A, the inclusive design rules are categorized into three sets: hand activities, communication, and gross body movements. The rules categorized as gross body movements are applicable to the refrigerator design but not to the food processor design. The food processor problem is mostly related to hand activities. Probably for this reason, the participants find more avenues to apply inclusive design rules in the refrigerator problem than the food processor problem. The next part discusses the feedback collected from the participants at the end of the study.

Participant Feedback

Feedback is collected from the participants to understand their perspective on the inclusive design representation scheme. Ninety-five percent of the participants reported that they clearly understood the problem statements. Eighty percent of the participants are familiar with food processors and ninety five percent are familiar with a refrigerator equipped with an icemaker. At the end of training session, the participants are asked if they could create an actionfunction diagram. Eighty percent of the participants reported that they need more practice for creating an actionfunction diagram. Recall that the participants are not asked to generate actionfunction diagram in this study. The feedback

question aims to assess how confident the participants feel about the concept of an actionfunction diagram.

Table 19 shows other feedback collected from the participants at the end of the experimental condition. When asked if the actionfunction diagram is helpful, seventy-eight percent of the participants reported ‘yes’ or ‘somewhat’. Similarly, seventy-two percent of the participants think that the actionfunction diagram gives them ideas more quickly for inclusive design. Most of the participants did understand the concept of an actionfunction diagram and the inclusive design rules. Only fourteen percent of the participants did not understand the design difference classification scheme i.e., parametric, morphological or functional changes.

Table 19. Participants’ feedback on inclusive design representation scheme

Feedback Questions	No	Somewhat	Yes
Does actionfunction diagram help you with universal design?	8 (22%)	18 (50%)	10 (28%)
Does actionfunction diagram gives you ideas for universal design more quickly?	10 (28%)	11 (30%)	15 (42%)
Did you understand the concept of actionfunction diagrams?	1 (3%)	19 (53%)	16 (44%)
Did you understand the universal design rules (guidelines) provided in the training packet?	0 (0%)	19 (53%)	17 (47%)
Did you understand the three types of design differences <i>or</i> design changes?	5 (14%)	16 (44%)	15 (42%)

The training questionnaire is collected after the training session to test the participants’ understanding of the inclusive design representation scheme. Of particular

interest, participants who reported that they have better understanding of the actionfunction diagram also scored significantly higher on the training questionnaire (p-value 0.07). Hence, the participants' self-reported feedback is a valuable measure in this exploratory study.

An interesting result of the study is that the participants who reported that they have a strong understanding of the representation scheme also have significantly higher average quality and average inclusivity on both the design problems. Figure 65 shows the results of two-tailed t tests and p-values for average quality and average inclusivity of both design problems when compared to the understanding of the representation scheme. Only one student answered 'no' to the question 'Did you understand the concept of actionfunction diagrams?' but that student scored high on the training questionnaire; so that data point is treated as outlier for the analysis reported in Figure 65.

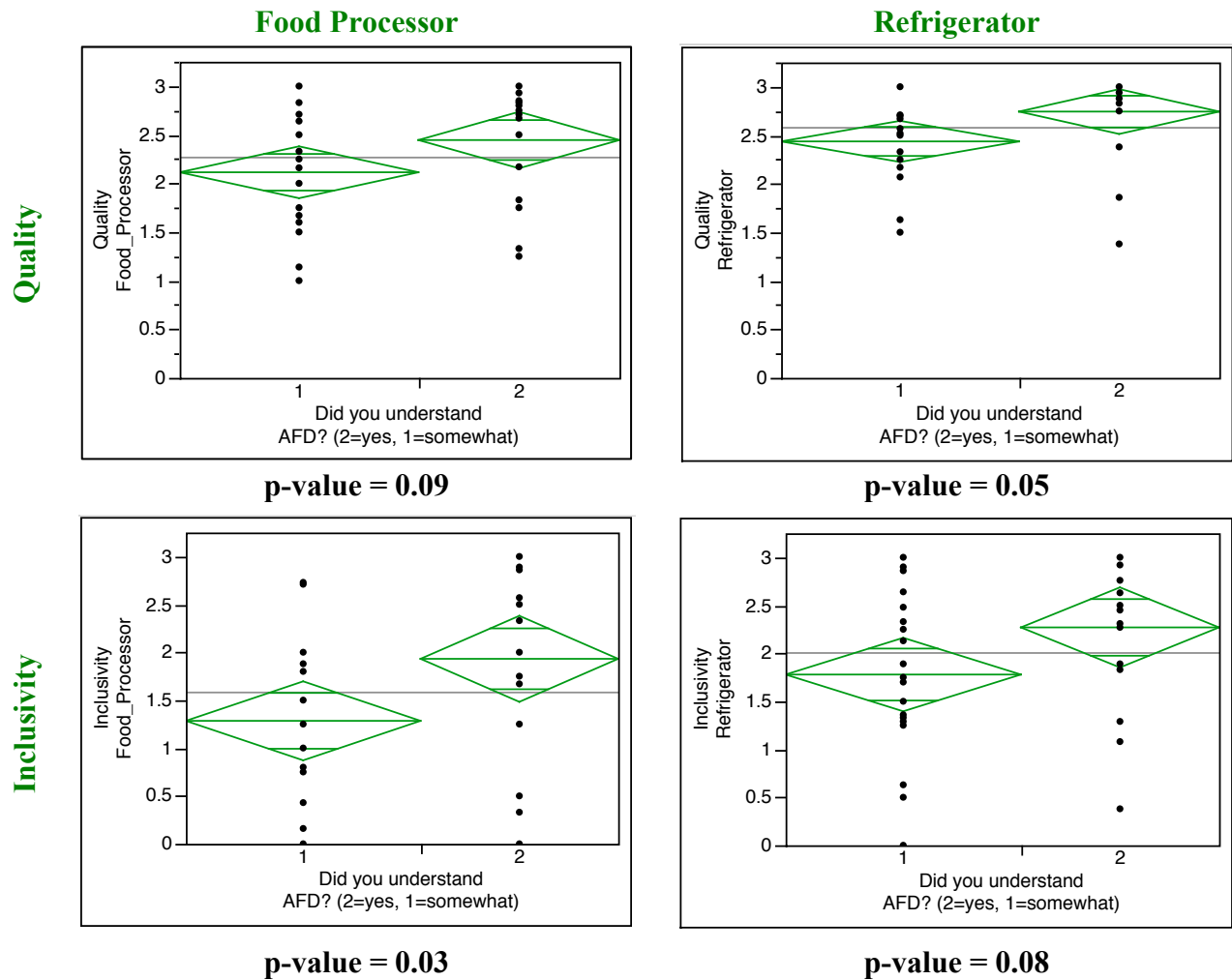


Figure 65. Results of two tailed t-test to compare the quality and inclusivity for both design problems based on self-reported understanding of the actionfunction diagram (AFD)

The participants are divided into three groups based on their self-reported measure on whether an actionfunction diagram helps with inclusive design (2 = yes, 1= somewhat, 0 = no). A one-way ANOVA is performed among the groups for average quality and average inclusivity of ideas generated by the participants for both design problems. Figure 66 shows the trends in the average inclusivity and quality based on the

usefulness of actionfunction diagrams along with the p-values and F-ratio from the one-way ANOVA performed. Participants who reported that the actionfunction diagrams helps with inclusive design also have significantly higher inclusivity for both design problems. The average quality of ideas for the food processor problem also showed a significant increasing trend with the self-reported measure on usefulness of the actionfunction diagram.

The results from Figure 65 and Figure 66 indicate that the participants' perception about the understanding and usefulness of the representation scheme correlates to the quality and inclusivity of the ideas they generate for a design problem. In other words, participants generate better ideas if they believe they understand the representation scheme. Thus, more thorough training will likely improve the implementation of the representation scheme, thereby generating better ideas for inclusive design.

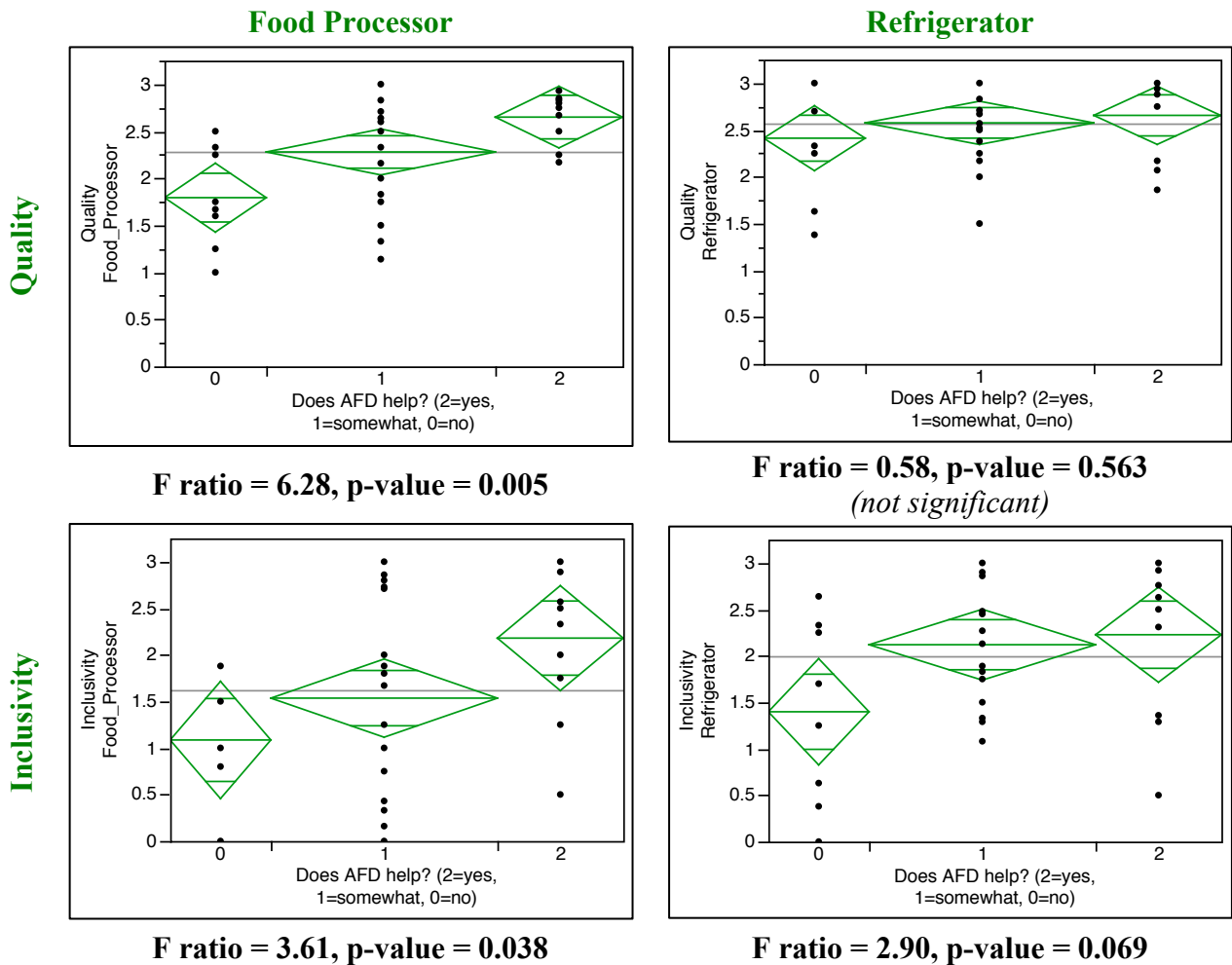


Figure 66. Plot showing the trends of average quality and inclusivity based on self-reported usefulness of actionfunction diagram (AFD) for both design problems

Summary

This chapter presents an exploratory study to investigate an inclusive design representation scheme. Participants are asked to design an inclusive product with and without the inclusive design representation scheme. Participants are trained to apply the representation scheme before the experimental condition. The two products studied in the research are a food processor and a refrigerator, which are functionally dissimilar

products. The effectiveness of the representation scheme is measured by comparing the quality and inclusivity of ideas generated by participants in the two conditions. Feedback is collected at the end of the study about participants' understanding and the perceived usefulness of the representation scheme.

The participants' self-reported feedback on usefulness of the representation scheme shows positive results. The analysis of the feedback indicates that the participants who believe they better understand the representation scheme also generate ideas with higher quality and inclusivity. Additionally, 78% of participants reported that the representation scheme helps for inclusive design. The participant's perception of the representation scheme indicates that the representation scheme has potential to influence inclusive design.

The results of the study show that there is problem dependency in the application of the representation scheme. The representation scheme helps in the design of products that can be categorized as architectural products. For architectural products, the overall position of the user with respect to the product is an important factor influencing inclusive design. It cannot be stated with certainty at this point if the representation scheme is helpful for designing consumer products, where only the activities related to the upper body are involved. Since, the inclusive design rules are empirically derived from a data set, adding more and diverse products to the dataset might yield rules that improve applicability of the representation scheme for the consumer product domain. Adding more and varied product domains in generation of the rules itself might also reduce the domain dependency of the inclusive design representation scheme.

CHAPTER X

VALIDATION STUDY

A detailed validation study with specific focus on the sub-functions of the product is conducted to know which part of the actionfunction diagram inspires an idea. The design activity is split into 3 tasks. The control condition of the exploratory study did not include any design aid while the experimental condition includes both the actionfunction diagram and inclusive design rules, and it is still uncertain which of the two influences the inclusiveness of ideas. Hence, functional models are provided in the control condition of the validation study to gain a better insight into the advantage of the actionfunction diagram over the functional model for inclusive design.

Besides, a study with a larger sample size might help to minimize any confounding factors such as the GPA of the participants. In the validation study the 20-minute pre-recorded training is replaced by a 50-minute lecture on inclusive design. The module to teach the inclusive design representation scheme aims to better train students on how to apply the representation scheme. This module is taught in MEEN 402 at Texas A&M University in the Spring 2013 semester. MEEN 402 is the second semester of a two-semester sequence of a senior design course at Texas A&M University. The study is split over two days and conducted as in-class activities before and after the lecture.

Validation Study Setup

This part describes the study procedure, characteristics of the study population, and the module developed for teaching inclusive design for the validation study.

Study Procedure

Table 20 explains the outline of the validation study procedure. Participants are randomly divided into two groups, red and blue. There are 53 participants in Red and 55 in Blue group. Two product design problems are considered for the validation study: design of an inclusive blender and design of an inclusive bread toaster. Similar to exploratory study, the design problems are swapped in the experimental and control condition for the 2 groups. The Red group receives blender problem in the experimental condition and the Blue group receives toaster problem in the experimental condition.

The study is conducted on Tuesday (Day 1) and Thursday (Day 2) of the same week. Day 1 begins with an introduction to the study, and the consent procedure. In the control condition, the participants are introduced to the problem statement followed by a concept generation activity. The total time of the control run is 40 minutes. After the control, participants are lectured for 20 minutes on the general concepts of human factors and the introduction to inclusive design.

On the Day 2, participants are lectured for 30 minutes on the concept of actionfunction diagram and inclusive design rules. The experimental run follows the lecture on second day. A training questionnaire is administered before the experimental condition to test whether participants understand the inclusive design representation scheme. Total time for the experimental run is 40 minutes including the time allowed for

studying the problem statement and the actionfunction diagram. Participant feedback and demographics are collected at the end of the study.

Table 20. Outline for the validation study

Task	Group Red	Group Blue	Time (min)	Description
Day 1	-----	-----	-----	Tuesday of the same week
Introduction	-----	-----	15	Consent form, General instructions Introduce inclusive design
Control Condition	Toaster	Blender	40	Introduction and Concept Generation
			(15)	Task 1 – Input & Controls
			(15)	Task 2 – Add & remove food
			(10)	Task 3 – Cleaning
Lecture	-----	-----	20	Human Factors
Day 2	-----	-----	-----	Thursday of the same week
Lecture	-----	-----	30	Inclusive Design Teaching Module
Experimental Condition	Blender	Toaster	40	<i>(With inclusive design rules)</i> Introduction and Concept Generation
			(15)	Task 1 – Input & Controls
			(15)	Task 2 – Add & remove food
			(10)	Task 3 – Cleaning
Participant Feedback	-----	-----	5	Participant demographics Feedback on the study
Total time	-----	-----	150	Two 75 min class periods

The problem statements given to the participants state: “Assume you are a product designer for a home appliances manufacturer. Your company is planning to launch an inclusive design of a [product name],” where the [product name] is “Blender” or “Bread toaster”. The problem statement for toaster and blender are shown in Figure 67 and Figure 68, respectively.

Problem Description

Assume you are a product designer for a home appliances manufacturer. Your company is planning to launch *an inclusive design of a “Bread Toaster”*

Definition

Toaster is a small electric kitchen appliance used to toast multiple types of bread products.

Figure 67. Problem statement for the toaster design

Problem Description

Assume you are a product designer for a home appliances manufacturer. Your company is planning to launch *an inclusive design of a “Blender”*

Definition

Blender is an electric kitchen appliance used to mix or puree food.

Figure 68. Problem statement for the blender design

General instructions

- Generate as many ideas as you can for each sub task.
- *Preferably sketch your idea* and write any notes if you feel necessary.
- Record all the concepts that occur to you, even if they might be infeasible.
- Partial solutions are allowed. Complete solution is not required.
- Your goal is to come up with commercially viable solution.
- Please feel free to write any comments that you might have on the last page.

Figure 69. General instructions given to the participants for both the design problems

For all conditions, the participants are instructed to generate as many ideas as possible, record all the concepts that occur to them, include partial solutions, preferably

sketch the idea, and write any notes they feel necessary. The general instructions are shown in Figure 69.

Both control and experimental runs are divided into three tasks. In the Task 1, participants are asked to generate ideas related to the controls of the devices. The controls include turning the device on/off as well as setting speed on the blender and temperature on the toaster. In the control condition, participants receive the functional model of the device as shown in Figure 70.

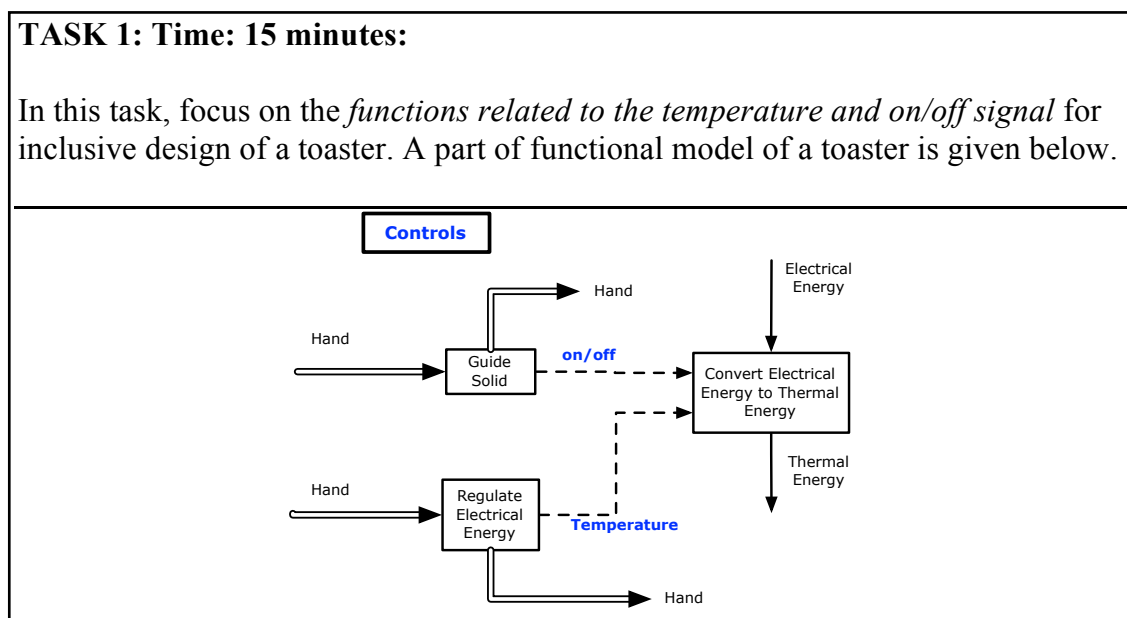


Figure 70. Task 1 for the control condition of the toaster design

In the experimental condition participants receive the actionfunction diagram of the product along with the related inclusive design rules and definitions of the ICF terms, refer Figure 71. Task 2 consists of generating ideas for either adding or removing food

from blender jar, and adding or removing bread from the toaster. Task 3 consists of ideas related to cleaning of the device. All task descriptions are listed in the Appendix B. Time allotted for task 1, task 2 and task 3 is 15 minutes, 15 minutes and 10 minutes, respectively

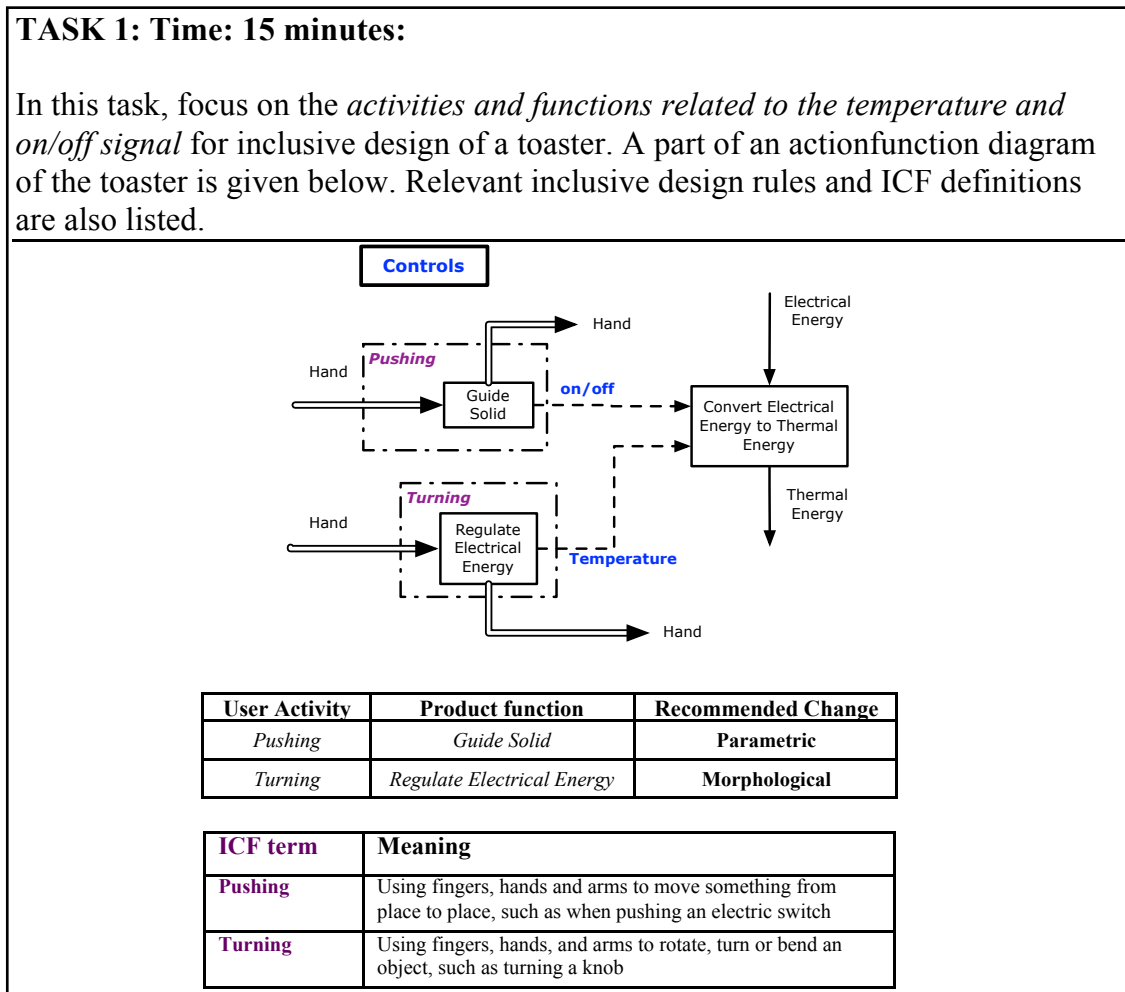


Figure 71. Task 1 for the experimental condition of the toaster design

The validation study investigates the inclusive design of blender and toaster, which are functionally dissimilar products. Both blender and toaster are devices that are used on the kitchen counter. Only the products involving upper body activities are selected for the validation study since the results of exploratory study shows that such products need further exploration. The design problems are from distinct domains so the product designed in the control condition should not influence the design of product in the experimental condition.

Study Population

The sample size for the validation study is 114. The participant group consisted of 80 males, 25 females, and the remaining did not report. Six participants who participated in only one condition are excluded from the analysis. The participants recruited for the validation study are senior mechanical engineering students who had prior training in basic design methods and functional modeling. The participants received in-class activity credit for participation in the validation study.

Teaching Module for Inclusive Design

The lecture on day 1 consists of introduction to human factors and considerations in human factors analysis. The lecture on day 1 gradually leads into importance of inclusive design. The lecture on day 2 started with overview of inclusive design and revision from the previous class. One specific method to practice inclusive design is illustrated in the class; the method consisting of actionfunction diagram and inclusive design rules. The three types of design differences are explained. Further, the application of inclusive design rules to actionfunction diagram is also explained.

Results of the Validation Study

This part compares the results from the control and experimental condition. The criteria for statistical significance is set at $\alpha = 0.05$. The average number of ideas generated per participants is 12 for both the design problems. While rating the ideas, 5 participants from blender study and 11 from toaster study are not considered in the analysis on account of illegible handwriting. Table 21 shows the number of participants in the control and experimental condition of both the design problems that are included in the analysis.

Table 21. Number of participants in the validation study in the control and experimental condition

	<i>Control</i>	<i>Experimental</i>
Toaster	46	51
Blender	52	55

Inter-rater reliability is tested for validation study where twenty-three percent of the toaster data and twenty eight percent of the blender data is rated by the second rater. The data tested includes ideas from both the control and experimental condition that are selected randomly across all of the three tasks. The linear-weighted Cohen's Kappa values are reported in Table 22. The weighted Cohen's Kappa values show reasonably reliable agreement between the raters for the validation study.

Table 22. Weighted Cohen’s Kappa (linear-weighted) for the validation study

	<i>Quality</i>	<i>Inclusivity</i>
Toaster (23% of the data)	0.66	0.44
Blender (28% of the data)	0.57	0.59

In the validation study analysis, non-parametric tests are applied to check difference in the count of ideas generated by the participants in each category of the quality and inclusivity rating. For instance, the count of inclusivity rating of 0 is the sum of ideas with inclusivity of 0 in each of the condition. Table 23 and Table 24 shows the Chi-Square tests of the observed counts, the expected counts and the deviation of inclusivity ratings by the control and experimental conditions for the toaster and blender problem, respectively. As shown in Table 23, the results of the chi-square tests show that the inclusivity ratings of ideas generated by the participants is significantly different in the two conditions of the toaster problem.

Note that, count of ideas with inclusivity of 0 is significantly higher than the expected value in the control condition. Also, the count of ideas with inclusivity of 3 is significantly higher than the expected value in the experimental condition. Thus, the inclusive design representation scheme helps to shift the designer’s focus away from the internal functions of the device and helps in generating better inclusive ideas. Similar trend is observed for the blender problem but the difference is not significant.

Table 23. Chi-Square tests of the observed and the expected average percentage of inclusivity ratings by the control and experimental conditions for the toaster problem (Chi-square = 22.18)

		Inclusivity Rating			
Condition		0	1	2	3
Control	Count	76	156	8	339
	<i>Expected</i>	53.5	157.5	7.5	360.5
	<i>Deviation</i>	22.5	-1.5	0.5	-21.5
Experimental	Count	31	159	7	382
	<i>Expected</i>	53.5	157.5	7.5	360.5
	<i>Deviation</i>	-22.5	1.5	-0.5	21.5
Chi Square		9.46	0.01	0.03	1.28

Table 24. Chi-Square tests of the observed and the expected average percentage of inclusivity ratings by the control and experimental conditions for the blender problem

		Inclusivity Rating			
Condition		0	1	2	3
Control	Count	80	197	23	287
	<i>Expected</i>	71.1	196.8	24.6	294.6
	<i>Deviation</i>	8.9	0.2	-1.6	-7.6
Experimental	Count	50	163	22	252
	<i>Expected</i>	58.9	163.2	20.4	244.4
	<i>Deviation</i>	-8.9	-0.2	1.6	7.6

Table 25 and Table 26 shows the Chi-Square tests of the observed counts, the expected counts and the deviation of the quality ratings by the control and experimental conditions for the toaster and blender problem, respectively. The quality ratings do not depend on the condition. In other words, inclusive design representation scheme does not affect the quality of ideas generated by the participants.

Table 25. Chi-Square tests of the observed and the expected average percentage of quality ratings by the control and experimental conditions for the toaster problem

		Quality Rating			
Condition		0	1	2	3
Control	Count	21	56	114	385
	<i>Expected</i>	<i>18.3</i>	<i>50.9</i>	<i>121.6</i>	<i>385.2</i>
	<i>Deviation</i>	<i>2.7</i>	<i>5.1</i>	<i>-7.6</i>	<i>-0.2</i>
Experimental	Count	16	47	132	394
	<i>Expected</i>	<i>18.7</i>	<i>52.1</i>	<i>124.4</i>	<i>393.8</i>
	<i>Deviation</i>	<i>-2.7</i>	<i>-5.1</i>	<i>7.6</i>	<i>0.2</i>

Table 26. Chi-Square tests of the observed and the expected average percentage of quality ratings by the control and experimental conditions for the blender problem

Condition		Quality Rating			
		0	1	2	3
Control	Count	17	50	116	417
	<i>Expected</i>	<i>22.1</i>	<i>46.4</i>	<i>113.2</i>	<i>418.4</i>
	<i>Deviation</i>	<i>-5.1</i>	<i>3.6</i>	<i>2.8</i>	<i>-1.4</i>
Experimental	Count	23	34	89	341
	<i>Expected</i>	<i>17.9</i>	<i>37.6</i>	<i>91.8</i>	<i>339.6</i>
	<i>Deviation</i>	<i>5.1</i>	<i>-3.6</i>	<i>-2.8</i>	<i>1.4</i>

With the results of the validation study, there is enough evidence to conclude that the quality of ideas generated by the participants is not influenced by the inclusive design representation scheme. The inclusivity ratings appears to be is problem dependent since it is significantly improved for the toaster problem but not for the blender problem. At this time, we cannot conclude the usefulness of the inclusive design representation scheme.

Participant Feedback

There is no significant difference in the self-reported GPAs of the participants in the two groups. Additionally, the quiz scores obtained in the post lecture quiz do not vary amongst the two groups. The quiz scores obtained by the participants are normally distributed across the sample. Figure 72 shows the responses of participant feedback collected at the end of the study to assess the effectiveness of the training. Most

participants responded “yes” or “somewhat” confident about their understanding of the inclusive design rules, actionfunction diagram, and the design differences based on the lectures.

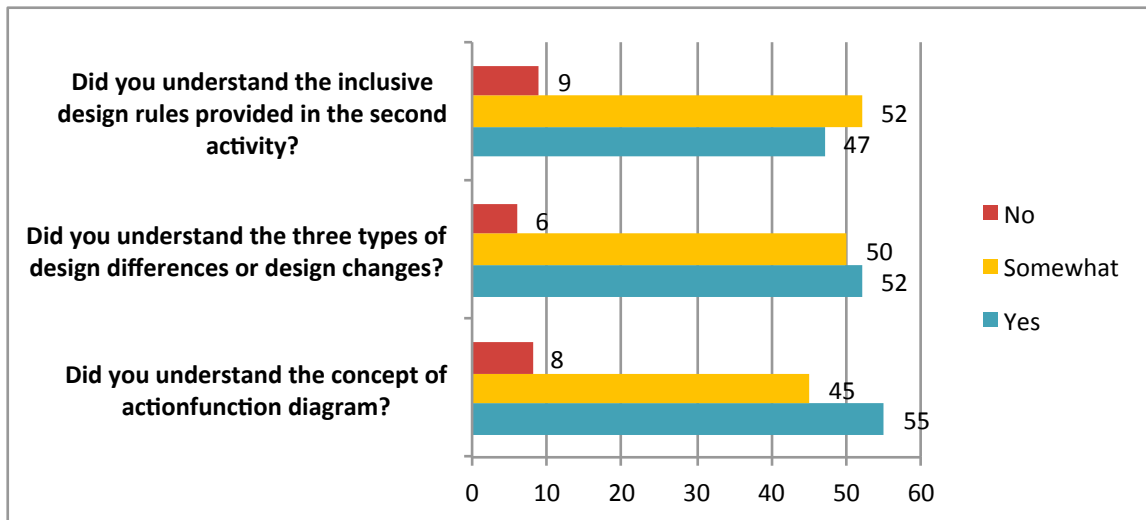


Figure 72. Participant feedback on effectiveness of the lecture for inclusive design representation scheme

The general trends in the responses for the self-assessment are roughly the same in both blue and red groups except for the response on the question “Did you understand the inclusive design rules provided in the study?” Table 27 shows responses from both groups for this question. The students from blue group who had toaster in experimental condition appear to be more confident about the training. This is interesting to note since the average inclusivity of ideas generated by participants is improving for the toaster design but not improving for the blender design with the inclusive design representation scheme.

Table 27. Participants' perception on understanding the inclusive design rules across the red and blue group

Did you understand the inclusive design rules provided in the study?	No	Somewhat	Yes
Red group (blender in experimental condition)	8 (15%)	28 (53%)	17 (32%)
Blue group (toaster in experimental condition)	1 (2%)	24 (43%)	30 (55%)

Figure 73 shows the participants' perception on inclusive design collected at the end of the study. Almost 70% participants feel confident about designing inclusive product in the future. The main purpose of the teaching module is to introduce participants to the concept of inclusive design and increase the confidence for designing inclusive products. The purpose of teaching module seems to be achieved based on the participant feedback.

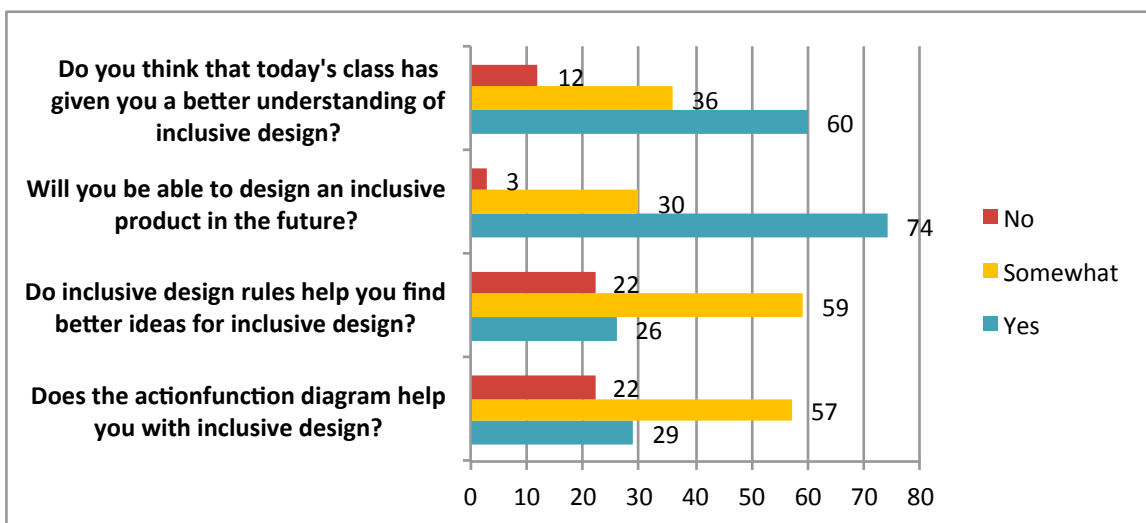


Figure 73. Participant feedback on the perception of inclusive design and effectiveness of the method

Few other questions were asked at the end of the study to check if the experiment's objectives are met, refer Figure 74. This feedback helps to know if the participants are aware of what they are designing. Thirty-two percent of the participants knew about the other design problem from their peers even when they were instructed to not discuss the design problems. The participants who are aware of the second design problem are not excluded from the study since it would drastically reduce the sample size.

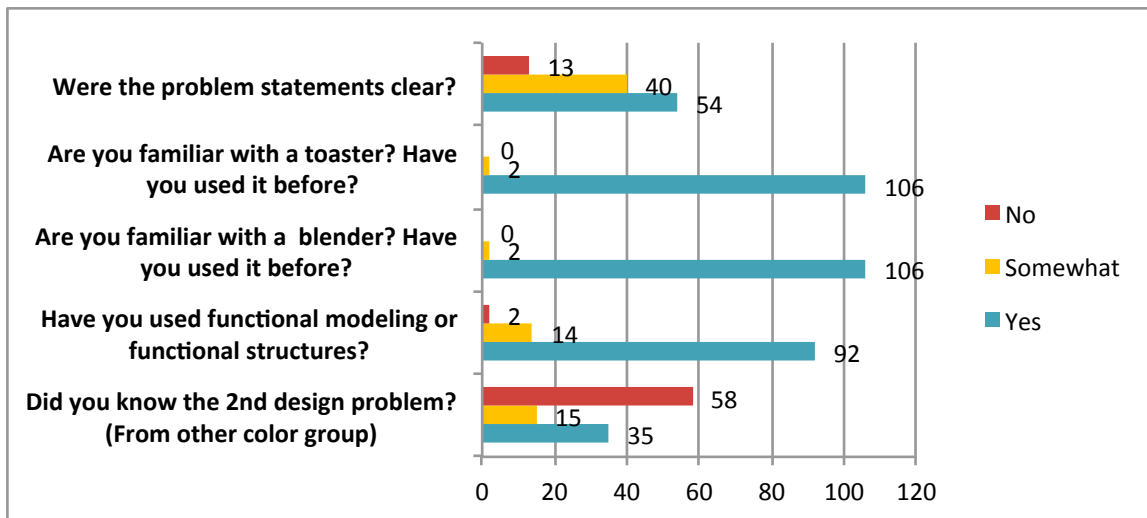


Figure 74. Participant feedback collected to check if the experiment's objective is met

Summary

This chapter discusses the validation study to test the inclusive design representation scheme. The application of the inclusive design rules to the actionfunction diagram is a promising method for inclusive design of products. Further development of

the inclusive design rule set can improve the applicability of the representation scheme for diverse product domains. Self reported feedback indicates that the teaching module is effective in increasing the awareness and confidence about inclusive design. Results show that the inclusivity ratings of ideas generated by participant is problem dependent. However, there is not enough evidence to support the hypothesis that the inclusive design representation scheme is helps for inclusive design.

CHAPTER XI

CONCLUSIONS AND FUTURE WORK

This research proposes to create knowledge and methods for engineering design that enable engineers to better create products to satisfy the needs of persons with disabilities. The broader impact of this research reaches the underserved population. These activities provide an opportunity for engineers to realize the power of design and what it can do for an underserved community.

Contributions

The specific contribution of this research is a method to derive heuristics for inclusive design. The function-based comparison data contains complex relational information; the graphical representation effectively captures all the pertinent information and preserves the relations. The graph mining serves as an automated heuristic extraction process that has the capability to efficiently mine the design repositories. Particularly, graph representation and graph data mining can be applied for the expansion of the design repository to find new heuristics for the design practice. Graph mining could also be applied for engineering grammar induction.

The application of graph mining for product design is an important contribution to the research in computational design synthesis. The graphical representation can be extended for many other functional representations like functional interaction modeling [25]. In general, graph mining can be applied to extract design heuristics from any discrete and relational design data that can be represented as graphs.

The heuristics derived in this research are validated to test its applicability for inclusive design. However, while applying these rules, designers should be mindful that the rules given here are design suggestions which may or may not be applicable to a specific design problem. The term ‘rule’ follows the data mining terminology. In addition, this research creates an inclusivity rating to evaluate the inclusiveness of solutions in conceptual stage of design. Inclusivity rating serves as a foundational block for benchmarking various methods for inclusive design. Further, a module to teach inclusive design methods is designed and tested as a contribution to the capstone design curriculum. This module can be adopted at other universities for similar capstone design course.

Other contributions of the research include a formal representation method for modeling user-product interactions known as the actionfunction diagram. The procedure to create actionfunction diagram is documented to allow repeatability and to allow expansion of inclusive design repository. An inclusive design repository is made available to other researchers and designers. A design difference classification scheme is established to compare a product pair. A bipartite graph provides an appropriate mathematical format for modeling discrete.

Overall Results

The research uses the ICF to formally describe a user activity, the Functional Basis to describe a product function, and the actionfunction diagram as a framework to create a detailed understanding of the interaction between a user and a product. Classification of changes as parametric, morphological, or functional provides a clear

framework for comparing a typical and an inclusive design. The rule generation allows statistical mining of the design guidelines from existing inclusive products.

Formalization results show that, the rate of rule generation decreases as more products are added to the dataset. Adding more product pairs to the dataset improves the statistical significance of the rule in terms of better values of confidence and support. Thus, the inclusive design knowledge can be captured and formalized by a set of association rules obtained from functional representations of an arbitrarily large set of products. Additionally, the clustering results indicate that the rules, and product clusters based on the rules, may provide opportunities to create product families based on sharing common components that make the product inclusive.

Graph data mining has the capability to efficiently search for new design heuristics from the updated repository of inclusive products. The advantages of graphical representations over flat-attribute type representations are discussed in the context of function-based representation framework. Particularly, graph representation and graph data mining can be applied for the expansion and mining of the design repository with some alterations in the graphical format. The automated method consisting of the GraphSynth, the FSG and the JUNG along with the JAVA GUI is particularly helpful in the developmental stages of graph mining applications for product design. The automated graph grammar method can derive the grammar rules that are originally derived by the experts. In addition, the automated process can be used to improve existing grammar rules.

The results of the exploratory study and validation show that there is problem dependency in the application of the representation scheme. It cannot be stated with certainty at this point if the representation scheme is helpful for designing consumer products, where only the activities related to the upper body are involved. Self reported feedback indicates that the teaching module is effective in increasing the awareness and confidence about inclusive design. Results show that the inclusivity ratings of ideas generated by participant is problem dependent. However, there is not enough evidence to support the hypothesis that the inclusive design representation scheme is helps for inclusive design.

Limitations

This research is empirical, thus extensions of findings have some limitations based on the data set studied. The framework used for analysis includes functional representation for engineered products and the ICF for user activity. Both these representations are general. Functional product representation has broad use in engineering practice and research. The ICF is developed specifically to provide application neutral representation of human health states. Thus, the findings and methods presented here should be generalizable. However, cases in which products are dissimilar, to the extent that there are no common functions and no common activities found in the actionfunction diagrams, may not support any common design guidelines in the form of those found here. Such a result is both a positive and limiting conclusion of the research. As a limiter of the approach, rule sets are only applicable to products that share some functions and user activities. As a positive conclusion, the formalism provides a

categorization framework for identifying products that have sufficient similarity that the products could implement and benefit from common strategies for inclusive design.

Future Work

More useful format of inclusive design guideline might be; given the disability, what change to a set of functions or a product module makes the product inclusive. Consider designing a product for an individual with reduced hand dexterity; addition of electrical energy would eliminate the fine hand controls required to operate the product. It is also interesting to know addition of which design feature in a product avoids exclusion of users with what kind of disability. For example, addition of electrical energy eliminates the activities requiring fine hand use thus including people with reduced hand functioning.

Performing a detailed study to quantify product similarity and rule transferability remains future work. Product similarity could be measured in a number of ways. Quantitative ratings to identify similar products based on a measure that includes product functionality are reported in McAdams and Wood [107]. More detailed classification of design differences would better serve product platform design. Performing a formal study for the inter-rater agreement for discerning the difference between the types of design changes remains as future work. Also, a rigorous product platform study for the inclusive product remains as future work. The dataset of products continues to be expanded with more product pairs.

REFERENCES

- [1] Brault, M. W., 2009, "Review of Changes to the Measurement of Disability in the 2008 American Community Survey," U.S. Census Bureau, July 16, 2013, http://www.census.gov/people/disability/files/2008ACS_disability.pdf
- [2] Field, J., and Jette, A. M., 2007, *The Future of Disability in America*, The National Academies Press, Washington, DC.
- [3] WHO, 2001, *International Classification of Functioning, Disability and Health*, World Health Organization (WHO), Geneva, Switzerland.
- [4] Vanderheiden, G., and Tobias, J., 2000, "Universal Design of Consumer Products: Current Industry Practice and Perceptions " Proceedings of the IEA, pp. 6-19.
- [5] Shah, J. J., Smith, S. M., and Vargas-Hernandez, N., 2003, "Metrics for Measuring Ideation Effectiveness," *Design Studies*, **24**(2), pp. 111-134.
- [6] Linsey, J. S., Clauss, E. F., Kurtoglu, T., Murphy, J. T., Wood, K. L., and Markman, A. B., 2011, "An Experimental Study of Group Idea Generation Techniques: Understanding the Roles of Idea Representation and Viewing Methods," *Journal of Mechanical Design*, **133**(3), p. 031008.
- [7] Linsey, J., Green, M. G., Murphy, J., Wood, K. L., and Markman, A. B., 2005, "Collaborating to Success: An Experimental Study of Group Idea Generation Techniques," ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Long Beach, CA.
- [8] Connell, B. R., Jones, M., Mace, R., Mueller, J., Mullick, A., Ostroff, E., Sanford, J., Steinfeld, E., Story, M., and Vanderheiden, G., 1997, "The Principles of Universal Design", July 14, 2009, http://www.design.ncsu.edu/cud/about_ud/udprincipleshtmlformat.html#top

- [9] Preiser, W. F. E., and Ostroff, E., 2001, *Universal Design Handbook*, McGraw-Hill Inc., New York, NY.
- [10] Vanderheiden, G., 1997, "Design for People with Functional Limitations," eds., Salvendy, G. in *Handbook of Human Factors and Ergonomics*, John Wiley & Sons, Hoboken, N.J.
- [11] Danford, G. S., 2010, Center for Inclusive Design and Environmental Access, June 15, 2010, <http://www.ap.buffalo.edu/idea/Home/index.asp>
- [12] Danford, G. S., 2003, "Universal Design People with Vision, Hearing, and Mobility Impairments Evaluate a Model Building," *Generations*, **27**(1), pp. 91-94.
- [13] Clarkson, P. J., Coleman, R., Keates, S., and Lebbon, C., 2003, *Inclusive Design: Design for the Whole Population*, Springer-Verlag, London.
- [14] Langdon, P. M., Clarkson, P. J., and Robinson, P., 2008, *Designing Inclusive Futures*, Springer Verlag, London.
- [15] University of Cambridge, 2013, "Inclusive Design Toolkit," May 13, 2013 <http://www.inclusivedesigntoolkit.com/>.
- [16] Duncan, R. C., 2011, "Universal Design Summit 4: A National Conference Focused on Universal Design in Housing and Communities," <http://uds4.org/>.
- [17] Mann, W., 2009, "The International Conference on Aging, Disability and Independence (ICADI) ", October 15, 2011, <http://icadi.php.ufl.edu/index.php>.
- [18] Yamamoto, T., 2010, "The 3rd International Conference for Universal Design in HAMAMATSU 2010," October 15, 2011, <http://www.iaud.net/en/index.php>.
- [19] Julian, A., 2004, "Designing for the 21st Century III: An International Conference on Universal Design ", October 15, 2011, <http://www.designfor21st.com/pg.cfm?nid=204&l=en>.

- [20] Glier, M. W., Tsenn, J., Linsey, J. S., and McAdams, D. A., 2011, "Methods for Supporting Bioinspired Design," Proceedings of the ASME International Mechanical Engineering Congress & Exposition, Denver, CO.
- [21] Glier, M. W., Tsenn, J., Linsey, J. S., and McAdams, D. A., 2012, "Evaluating the Directed Method for Bioinspired Design," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, IL.
- [22] Shah, J., 2003, "Metrics for Measuring Ideation Effectiveness," *Design Studies*, **24**(2), pp. 111-134.
- [23] Nelson, B. A., Wilson, J. O., Rosen, D., and Yen, J., 2009, "Refined Metrics for Measuring Ideation Effectiveness," *Design Studies*, **30**(6), pp. 737-743.
- [24] Srivathsavai, R., Genco, N., Hölttä-Otto, K., & Seepersad, C. C., 2010, "Study of Existing Metrics Used in Measurement of Ideation Effectiveness," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Montreal, QC.
- [25] Ramachandran, R., Caldwell, B. W., and Mocko, G. M., 2011, "A User Study to Evaluate the Function Model and Function Interaction Model for Concept Generation," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Washington, DC.
- [26] Coleman, R., Clarkson, J., Dong, H. A., and Cassim, J. I., 2007, *Design For Inclusivity: A Practical Guide to Accessible, Innovative and User-Centred Design*, Gower Publishing Company, Burlington, VT.
- [27] Mullick, A., 1999, "Measuring Universal Design: Case of the Bathroom," Proceedings of the Human Factors and Ergonomics Society Annual Meeting, **43**(8), pp. 557-562.

- [28] Talley, A., Crawford, R. H., and Talley, K., "Engineering Applicability of a Universal Design Performance Measure," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Washington, DC.
- [29] Braha, D., 2001, *Data Mining for Design and Manufacturing: Methods and Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands.
- [30] Li, J., Liu, Y., Yang, J., and Yao, J., 2006, "*Research on Acquiring Design Knowledge Based on Association Rule*," Knowledge Enterprise: Intelligent Strategies in Product Design Manufacturing, and Management K. Wang, G. Kavacs, M. Wozny, and M. Fang, eds., Springer, Boston, pp. 103-108.
- [31] Fayyad, U., and Stolorz, P., "Data Mining and KDD: Promise and Challenges," Proc. Future Generation Computer Systems, Elsevier Science, pp. 99-115.
- [32] Shahbaz, M., Srinivas, Harding, J. A., and Turner, M., 2006, "Product Design and Manufacturing Process Improvement Using Association Rules," Journal of Engineering Manufacturing, **220**(2), pp. 243-254.
- [33] Jiao, J., and Zhang, Y., 2005, "Product Portfolio Identification Based on Association Rule Mining," Computer-Aided Design, **27**(2), pp. 149-172.
- [34] Agrawal, R., and Srikant, R., 1994, "Fast Algorithms for Mining Association Rules in Large Databases," 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile.
- [35] Lawrence, B. H., Diane, J. C., and Surnjani, D., 1994, "Substructure Discovery in the SUBDUE System." Proceedings of the Workshop on Knowledge Discovery in Databases.
- [36] Campbell, M. I., Rai, R., and Kurtoglu, T., 2009, "A Stochastic Graph Grammar Algorithm for Interactive Search," Proceedings of the ASME International Design

Engineering Technical Conferences and Computers and Information in Engineering Conference, San Diego, CA.

[37] Patel, J., and Campbell, M. I., 2008, "An Optimization Approach/Technique for Solving Graph Based Design Problems," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, New York, NY.

[38] Vempati, C., and Campbell, M. I., 2007, "A Graph Grammar Approach to Generate Neural Network Topologies," Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Las Vegas, NV.

[39] Kurtoglu, T., and Campbell, M. I., 2009, "Automated Synthesis of Electromechanical Design Configurations from Empirical Analysis of Function to Form Mapping," *Journal of Engineering Design*, **20**(1), pp. 83 - 104.

[40] Sridharan, P., and Campbell, M. I., 2005, "A Study on the Grammatical Construction of Function Structures," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **19**(3), pp. 139-160.

[41] Zhang, W. Y., Tor, S. B., and Britton, G. A., 2004, "A Graph and Matrix Representation Scheme for Functional Design of Mechanical Products," *Int J Adv Manuf Technol*, **25**, pp. 221-232.

[42] Siddique, Z., and Rosen, D. W., 2001, "On combinatorial design spaces for the configuration design of product families," *Artificial Intelligence for Engineering Design, Analysis & Manufacturing*, **15**(2), pp. 91-108.

[43] Siddique, Z., and Rosen, D. W., 1999, "Product Platform Design: A Graph Grammar Approach," Proceedings of the ASME International Design Engineering

Technical Conferences and Computers and Information in Engineering Conference, Las Vegas, NV.

[44] Shai, O., 2003, "Transforming Engineering Problems through Graph Representations," *Advanced Engineering Informatics* 17 (2003) 77–93.

[45] Goel, A., 1997, "Design, Analogy and Creativity," *IEEE Expert Intelligent Systems and Their Applications*, 12(3), pp. 62-70.

[46] Rao, R. V., and Padmanabhan, K. K., 2007, "Rapid Prototyping Process Selection Using Graph Theory and Matrix Approach," *Journal of Materials Processing Technology*, 194(1-3), pp. 81-88.

[47] Rao, R. V., 2006, "A Material Selection Model Using Graph Theory and Matrix Approach," *Materials Science and Engineering A*, 431(1-2), pp. 248-255.

[48] Zhang, H. C., and Kuo, T. C., 1996, "A Graph-Based Approach to Disassembly Model for End-of-Life Product Recycling," *Electronics Manufacturing Technology Symposium*, 1996, Nineteenth IEEE/CPMT, pp. 247-254.

[49] Ates, K., Kukluk, J., Holder, L., Cook, D., and Zhang, K., 2006, "Graph Grammar Induction on Structural Data for Visual Programming," 18th IEEE International Conference on Tools with Artificial Intelligence, pp. 232-242.

[50] Inokuchi, A., Washio, T., and Motoda, H., 2000, "An Apriori-Based Algorithm for Mining Frequent Substructures from Graph Data," *Principles of Data Mining and Knowledge Discovery*, pp. 13-23.

[51] Kuramochi, M., and Karypis, G., 2001, "Frequent Subgraph Discovery," *Proceedings IEEE International Conference on Data Mining*, pp. 313-320.

[52] Yan, X., and Han, J., 2002, "gSpan: Graph-based Substructure Pattern Mining," *Proceedings IEEE International Conference on Data Mining*, pp. 721-724.

- [53] Russell, S. J., Norvig, P., Canny, J. F., Malik, J. M., and Edwards, D. D., 1995, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Englewood Cliffs, NJ.
- [54] Chakrabarti, A., Shea, K., Stone, R., Cagan, J., Campbell, M., Hernandez, N. V., and Wood, K. L., 2011, "Computer-Based Design Synthesis Research: An Overview," *J. Comput. Inf. Sci. Eng.*, **11**(2), pp. 1-9.
- [55] Stiny, G., 1980, "Introduction to Shape and Shape Grammars," *Environment and Planning B*, **7**(3), pp. 343-351.
- [56] Helms, B., Shea, K., and Hoisl, F., 2009, "A Framework for Computational Design Synthesis Based on Graph-Grammars and Function-Behavior-Structure," *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, San Diego, CA.
- [57] Sridharan, P., and Campbell, M. I., 2004, "A Grammar for Functional Structures," *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Salt Lake City, UT.
- [58] Schmidt, L., and Cagan, J., 1997, "GGREADA: A Graph Grammar-Based Machine Design Algorithm," *Research in Engineering Design*, **9**(4), pp. 195-213.
- [59] Eloy, S., and Duarte, J. P., 2011, "A Transformation Grammar for Housing Rehabilitation " *Nexus Network Journal*, **13**(1), pp. 49-71.
- [60] Orsborn, S., Boatwright, P., and Cagan, J., 2008, "Identifying Product Shape Relationships Using Principal Component Analysis," *Research in Engineering Design*, **18**(4), pp. 163-180.
- [61] Orsborn, S., Cagan, J., and Boatwright, P., 2008, "A Methodology for Creating a Statistically Derived Shape Grammar Composed of Non-Obvious Shape Chunks," *Research in Engineering Design*, **18**(4), pp. 181-196.

- [62] Stone, R. B., 2007, "Design Repository," April 15, 2013, <http://designengineeringlab.org/delabsite/repository.html>.
- [63] Bohm, M. R., Vucovich, J. P., and Stone, R. B., 2008, "Using a Design Repository to Drive Concept Generation," *Journal of Computing and Information Science in Engineering*, **8**(1), p. 014502.
- [64] Bryant, C. R., McAdams, D. A., Stone, R. B., Kurtoglu, T., and Campbell, M. I., 2005, "A Computational Technique for Concept Generation," *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Long Beach, CA.
- [65] Xu, Q. L., Ong, S. K., and Nee, A. Y. C., 2006, "Function-Based Design Synthesis Approach to Design Reuse," *Research in Engineering Design*, **17**(1), pp. 27-44.
- [66] Shooter, S. B., Simpson, T. W., Kumara, S. R. T., Stone, R. B., and Terpenney, J. P., 2005, "Toward an Information Management Infrastructure for Product Family Planning and Platform Customization," *International Journal of Mass Customization*, **1**(1), pp. 134-155.
- [67] Xu, Q., and Jiao, J. R., 2009, "Design Project Modularization for Product Families," *Journal of Mechanical Design*, **131**(7), pp. 071007-071010.
- [68] Ki Moon, S., and McAdams, D. A., 2012, "A Market-Based Design Strategy for a Universal Product Family," *Journal of Mechanical Design*, **134**(11), pp. 111007.
- [69] Moon, S. K., and McAdams, D. A., 2010, "A Platform-Based Strategic Design Approach for Universal Products," *International Journal of Mass Customisation*, **3**(3), pp. 227-246.
- [70] Moon, S. K., and McAdams, D. A., 2010, "Universal Product Family Design Valuation in an Uncertain Market Environment," *Proceedings of the ASME 2010*

International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Montreal, QC.

[71] Moon, S. K., and McAdams, D. A., 2009, "Universal Product Platform and Family Design for Uncertain Markets," Proceedings of the International Conference on Engineering Design ICED 09, Palo Alto, CA.

[72] Tucker, C. S., and Kim, H. M., 2011, "Trend Mining for Predictive Product Design," Journal of Mechanical Design, **133**(11), pp. 111008-111011.

[73] Tucker, C. S., and Kim, H. M., 2008, "Optimal Product Portfolio Formulation by Merging Predictive Data Mining With Multilevel Optimization," Journal of Mechanical Design, **130**(4), pp. 041103-041115.

[74] Otto, K., and Wood, K., 2001, *Product Design: Techniques in Reverse Engineering, Systematic Design, and New Product Development*, Prentice-Hall, New York, NY.

[75] Hirtz, J., Stone, R., McAdams, D., Szykman, S., and Wood, K., 2002, "A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts," Research in Engineering Design, **13**(2), pp. 65-82.

[76] Kurfman, M., Stone, R., Rajan, J., and Wood, K., 2003, "Experimental Studies Assessing the Repeatability of a Functional Modeling Derivation Method," Journal of Mechanical Design, **125**(4), pp. 682--693.

[77] Ahmed, S., and Wallace, K., 2003, "Evaluating a Functional Basis," Proceedings of Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, IL.

[78] Caldwell, B. W., Sen, C., Mocko, G. M., Summers, J. D., and Fadel, G. M., 2008, "Empirical Examination of the Functional Basis and Design Repository " in *Design Computing and Cognition '08: Proceedings of the Third International Conference on*

Design Computing and Cognition, vol. 3, Eds. Gero, J. S., and Goel, A. K., Springer Netherlands, Dordrecht, pp. 261-280.

[79] Sen, C., Caldwell, B. W., Summers, J. D., and Mocko, G. M., 2009, "Topological Information Content and Expressiveness of Function Models in Mechanical Design," ASME Design Engineering Technical Conference and Computers In Engineering Conference, San Diego, CA.

[80] Nagel, R. L., Hutcheson, R. S., Stone, R., McAdams, D., and Donndelinger, J., 2008, "Function Design Framework (FDF): Integrated Process and Function Modeling for Complex System Design," ASME International Design Engineering Technical Conference, **24**(2), pp. 115-131.

[81] Sangelkar, S., and McAdams, D. A., 2010, "Adapting ADA Architectural Design Knowledge to Product Design: Groundwork for a Function Based Approach," Proceedings of Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Montreal, QC.

[82] Kostovich, V., McAdams, D. A., and Moon, S. K., 2009, "Representing User Activity and Product Function for Universal Design," Proceedings of the 2009 ASME Design Engineering Technical Conferences & Computers and Information in Engineering, San Diego, CA.

[83] Sangelkar, S., 2010, "Adapting ADA Architectural Design Knowledge to Product Design: Groundwork for a Function Based Approach," Master of Science, Texas A&M University, College Station.

[84] Sangelkar, S., Cowen, N., and McAdams, D. A., 2012, "User Activity – Product Function Association Based Design Rules for Universal Products " Design Studies, **33**(1), pp. 85-110.

- [85] Dorst, K., and Vermaas, P., 2005, "John Gero's Function-Behaviour-Structure model of designing: a critical analysis," *Research in Engineering Design*, **16**(1), pp. 17-26.
- [86] Gero, J. S., and Kannengiesser, U., 2004, "The Situated Function-Behaviour-Structure Framework," *Design Studies*, **25**(4), pp. 373-391.
- [87] McAdams, D. A., 2011, "Product Synthesis Engineering Lab: UD Repository," June 10, 2012 <http://www.prosedesign.org>.
- [88] Han, J., Pei, J., Yin, Y., and Mao, R., 2004, "Mining Frequent Patterns without Candidate Generation," *ACM SIGMOD* **29**(2). pp 1-12.
- [89] Zaki, M. J., 2000, "Scalable Algorithms for Association Mining," *IEEE Transactions on Knowledge and Data Engineering*, **12**(3), pp. 372-390.
- [90] Rakotomalala, R., 2004, "TANAGRA project," April 5, 2010, <http://eric.univ-lyon2.fr/~ricco/tanagra/en/tanagra.html>.
- [91] Yun, H., Ha, D., Hwang, B., and Ryu, K. H., 2003, "Mining Association Rules on Significant Rare Data Using Relative Support," *Journal of Systems and Software*, **67**(3), pp. 181-191.
- [92] Cowen, N., 2010, "Universal Design Rules from Product Pairs and Association Rule-Based Learning," Master of Science Masters, Texas A&M University, College Station.
- [93] Clark, J., and Holton, D. A., 1991, *A First Look at Graph Theory*, World Scientific, New Jersey, NJ.
- [94] Karypis, G., 2011, "PAFI Software Package for Finding Frequent Patterns in Diverse Datasets," Jan 10, 2012, <http://glaros.dtc.umn.edu/gkhome/pafi/overview>.

- [95] Kurtoglu, T., Campbell, M. I., and Linsey, J. S., 2009, "An Experimental Study on The Effects of a Computational Design Tool on Concept Generation," *Design Studies*, **30**(6), pp. 676-703.
- [96] Fu, Z., Pennington, A. D., and Saia, A., 1993, "A Graph Grammar Approach to Feature Representation and Transformation," *International Journal of Computer Integrated Manufacturing*, **6**(1-2), pp. 137-151.
- [97] Bissell, 2013, "CleanView® Deluxe Corded Hand Vacuum 47R5-1," May 10, 2013, <http://www.bissell.com/clean-view-deluxe-corded-hand-vacuum/>.
- [98] Campbell, M., 2006, "The Official Graphsynth Site." April 15, 2013, www.graphsynth.com/
- [99] Huan, J., Wang, W., and Prins, J., 2003, "Efficient Mining Of Frequent Subgraphs in the Presence of Isomorphism," *Third IEEE International Conference on Data Mining*, pp. 549-552.
- [100] Worlein, M., Meinl, T., Fischer, I., and Philippsen, M., 2005, "A Quantitative Comparison of the Subgraph Miners MoFa, gSpan, FFSM, and Gaston," *Knowledge Discovery in Databases: PKDD 2005*. Springer Berlin Heidelberg, pp. 392-403.
- [101] Kang, U., Tsourakakis, C. E., and Faloutsos, C., "Pegasus: A Peta-Scale Graph Mining System Implementation and Observations," *Ninth IEEE International Conference on Data mining*, pp. 229-238.
- [102] Deshpande, M., Kuramochi, M., Wale, N., and Karypis, G., 2005, "Frequent Substructure-Based Approaches for Classifying Chemical Compounds," *IEEE Trans. On Knowledge and Data Engineering*, **17**(8), pp. 1036-1050.
- [103] White, S., O'Madadhain, J., Fisher, D., and Boey, Y.-B., 2004, "JUNG: Java Universal Network/Graph Framework," April 13, 2013, <http://jung.sourceforge.net/index.html>.

[104] O'Madadhain, J., Fisher, D., White, S., and Boey, Y., 2003, "The JUNG (Java Universal Network/Graph) Framework," University of California, Irvine, California.

[105] Campbell, M., 2013, "The Rule Set," May 8, 2013,
http://www.me.utexas.edu/~adl/fs_grammar.htm.

[106] Sangelkar, S., and McAdams, D. A., 2012, "Adapting ADA Architectural Design Knowledge to Product Design Using Association Rule Mining: A Function Based Approach," *Journal of Mechanical Design*, **134**(7), p. 071003.

[107] McAdams, D. A., and Wood, K. L., 2002, "A Quantitative Similarity Metric for Design by Analogy," *Journal of Mechanical Design*, **124**(2), pp. 173-182.

APPENDIX A

INCLUSIVE DESIGN REPRESENTATION SCHEME

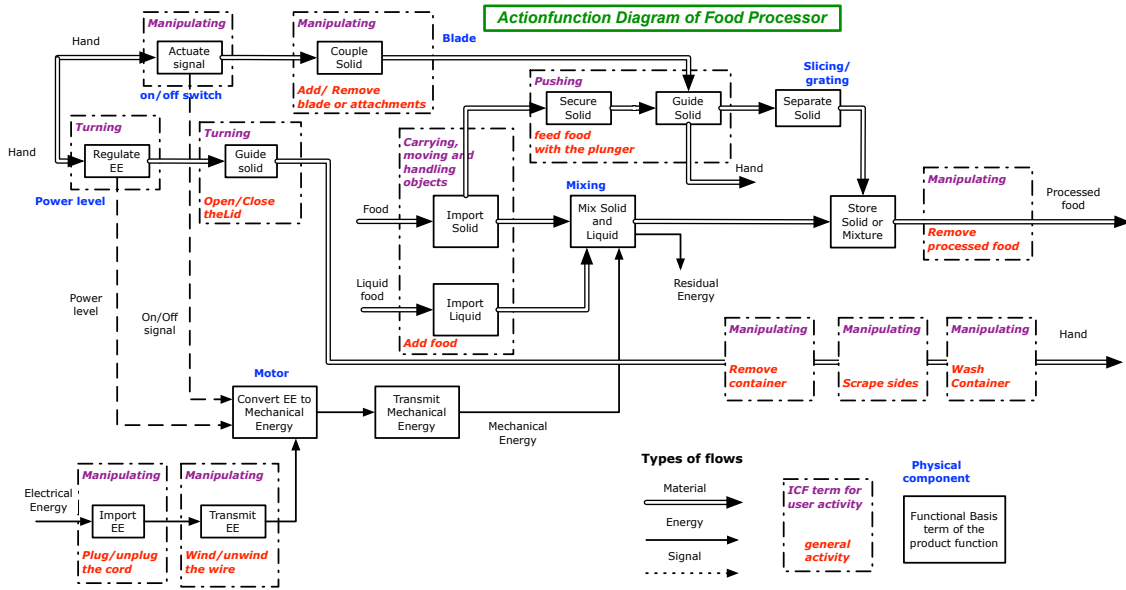


Figure A1. Actionfunction diagram of the food processor

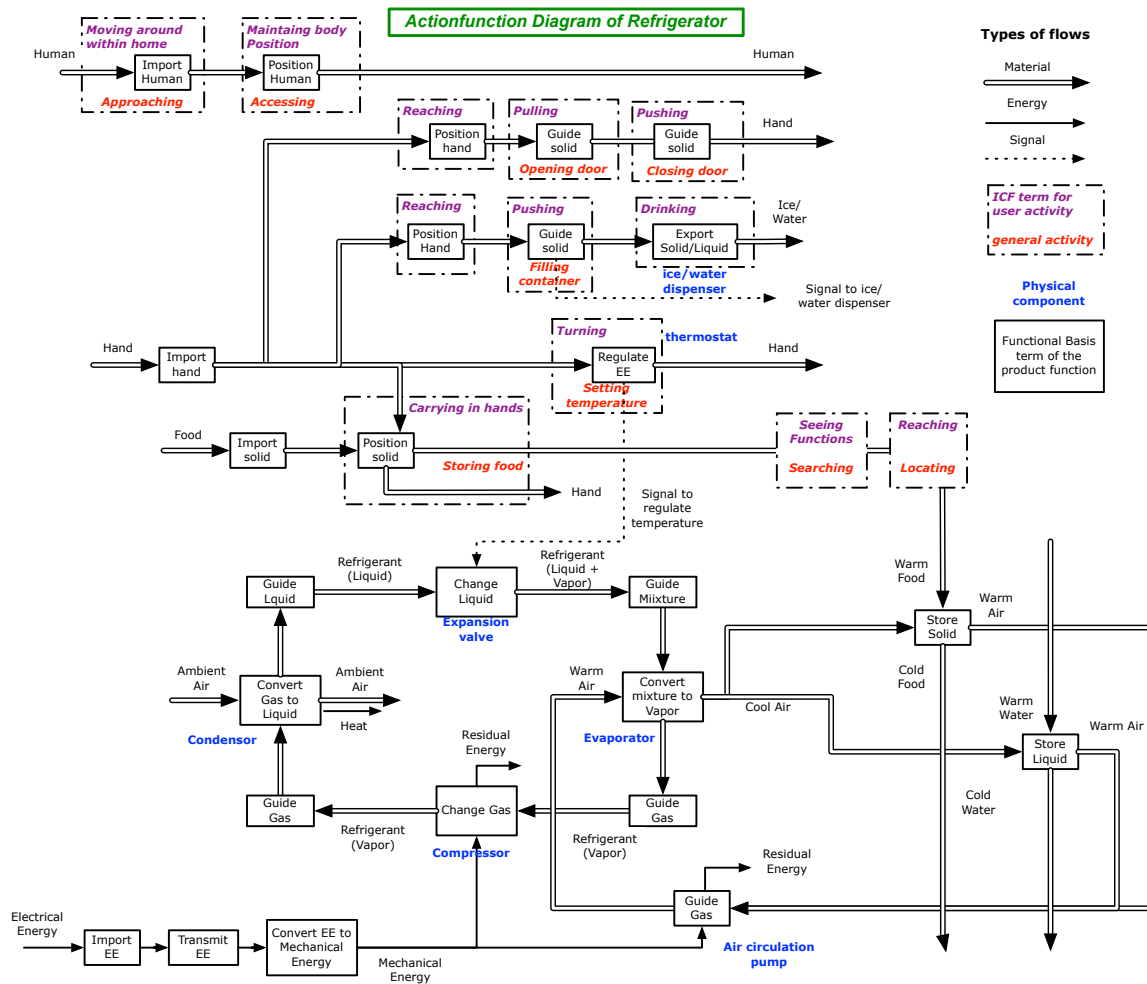


Figure A2. Actionfunction diagram of the Refrigerator

Table A1. Inclusive design rules listing the recommended change to user-product interactions

User Activity	Product function	Recommended Change	User Activity Change
Hand Activities			
Carrying, moving and handling objects	Import Solid	No change	Same as Typical
Carrying, moving and handling objects	Position Hand	Parametric	Easier
Carrying, moving and handling objects	Position Solid	Parametric	Easier
Picking up	Import Hand	No change	Same as Typical
Picking up	No function	No change	Same as Typical
Grasping	Position Hand	Parametric	Easier
Grasping	Secure Hand	Functional	Easier
Manipulating	Actuate Signal	Morphological	Pushing with fingers
Manipulating	Guide Solid	Parametric	Easier
Manipulating	Guide Solid	Morphological	Pushing with fingers
Manipulating	Position Hand	Parametric	Easier
Manipulating	Separate Solid	No change	Same as Typical
Manipulating	Store Solid	No change	Same as Typical
Manipulating	No function	No change	Same as Typical
Manipulating	Couple Solid	Parametric	Easier
Pulling	Guide Solid	Parametric	Easier
Pulling	Guide Solid	Morphological	No Activity
Pushing with hand	Guide Solid	Parametric	Same as Typical
Pushing with fingers	Guide Solid	Parametric	Same as Typical
Reaching	Position Hand	Parametric	Easier
Reaching	Import Hand	No change	Same as Typical
Reaching	No function	No change	Same as Typical
Turning	Guide Solid	Morphological	Pushing with hand
Turning	Regulate Electrical Energy	Parametric	Pushing with fingers
Communication			
Communication Written	Indicate Status	Parametric	Easier
Communication Written	Indicate Status	Morphological	Communication Braille
Hearing functions	Indicate Status	Parametric	Easier
Hearing functions	Indicate Status	Morphological	Better
Seeing functions	Indicate Status	Parametric	Easier
Seeing functions	Indicate Status	Morphological	Better
Gross Body Movements			
Moving around	Import Human	Parametric	Easier
Moving around building other than home	Import Human	No change	Same as Typical
Moving around	Secure Human	Functional	Better
Maintain Body Position	Position Human	Parametric	Easier
Transferring oneself	Import Human	Morphological	Better
Transferring oneself	Import Human	Parametric	Easier
Pushing with lower extremities	Guide Solid	Morphological	Pushing with hand
Sitting	Guide Human	Functional	Better
Standing	Guide Human	Functional	Better

APPENDIX B

DESIGN TASKS FOR THE VALIDATION STUDY

TASK 1: Time: 15 minutes:

In this task, focus on the *activities and functions related to the temperature and on/off signal* for inclusive design of a toaster. A part of an actionfunction diagram of the toaster is given below. Relevant inclusive design rules and ICF definitions are also listed.

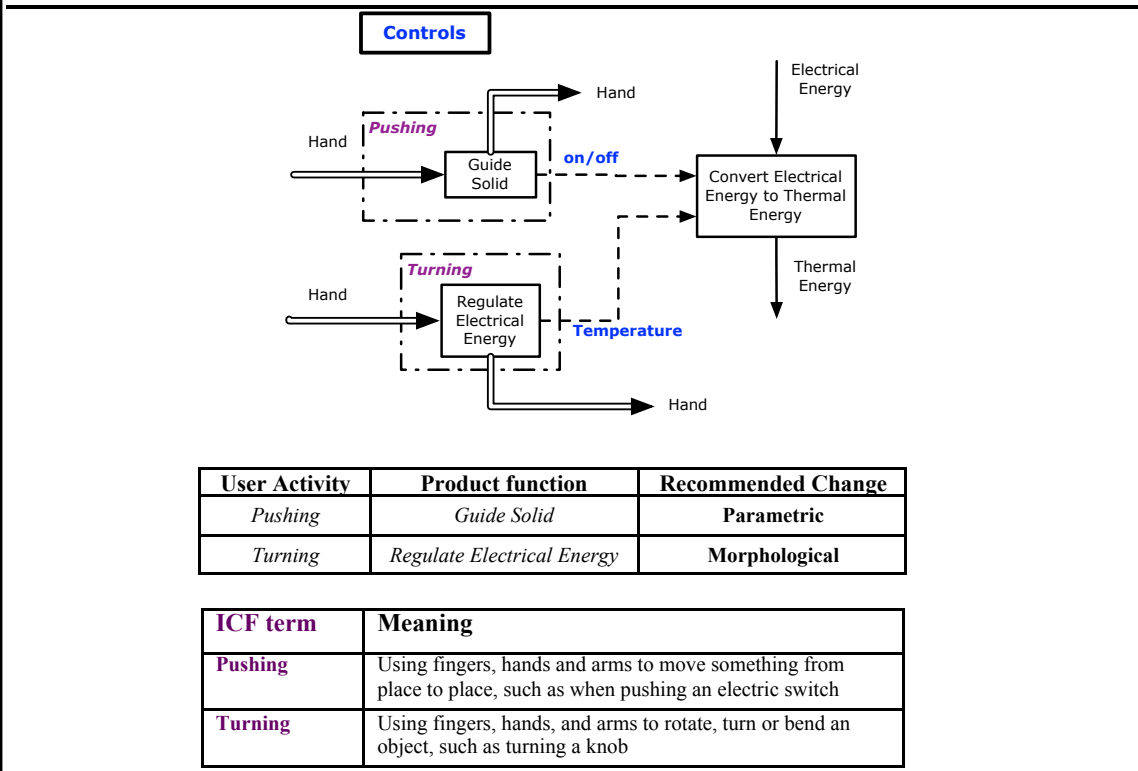


Figure B1. Task 1 for the experimental condition of toaster design

TASK 2: Time: 15 minutes

In this task, focus on the *activities and functions related to getting bread in and out* of the inclusive design of a toaster. A part of an actionfunction diagram of the toaster is given below. Relevant inclusive design rules and ICF definitions are also listed.

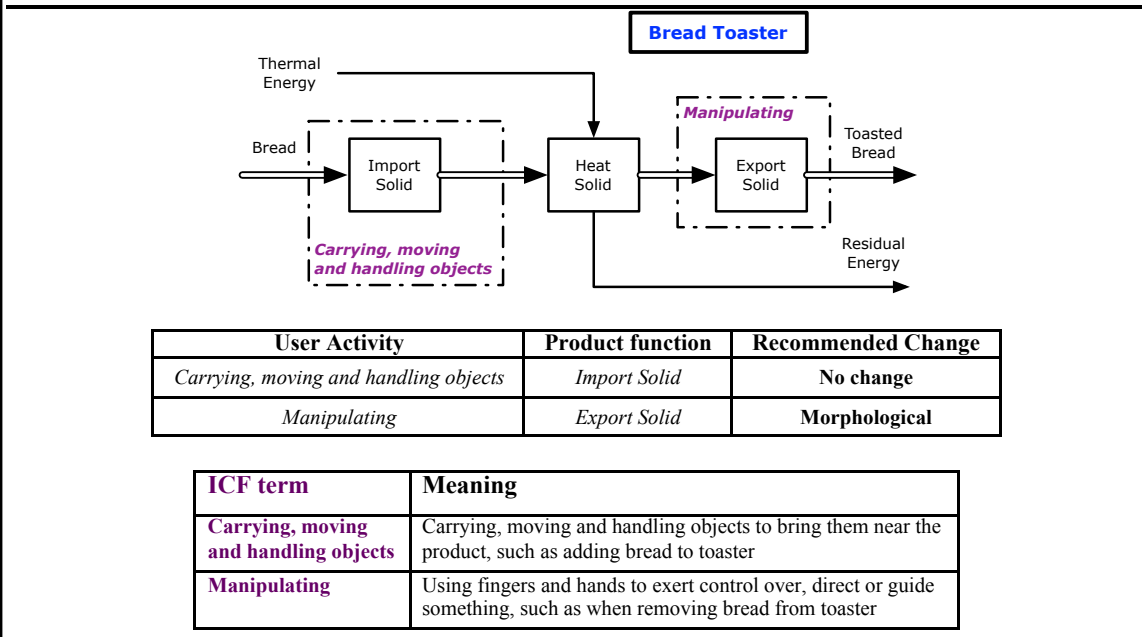


Figure B2. Task 2 for the experimental condition of toaster design

TASK 3: Time: 10 minutes

In this task, focus on the *activities and functions* related to the customer need “*easy to remove bread crumbs*” for the inclusive design of a toaster. During or after toasting bread crumbs fall inside the toaster which needs to be cleaned periodically. A part of an actionfunction diagram of the toaster is given below. Relevant inclusive design rules and ICF definitions are also listed.

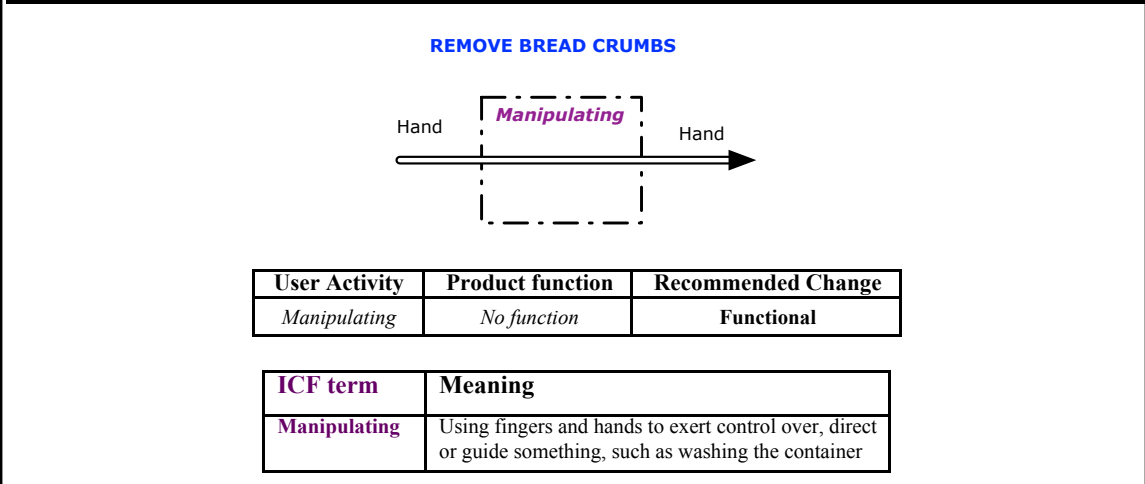


Figure B3. Task 3 for the experimental condition of toaster design

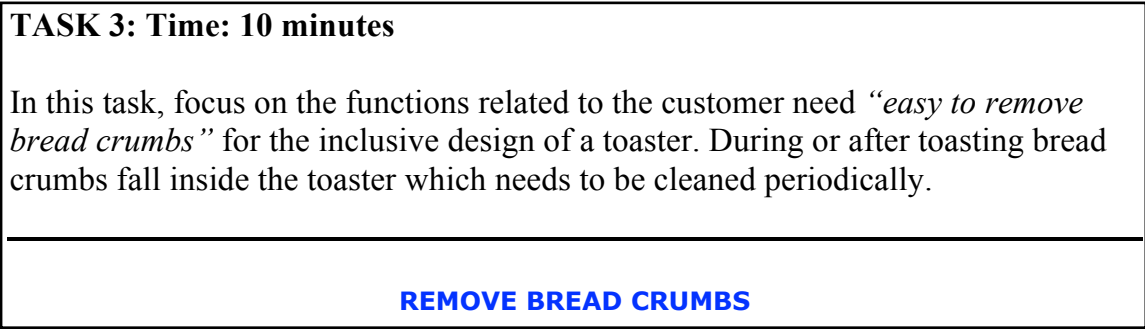
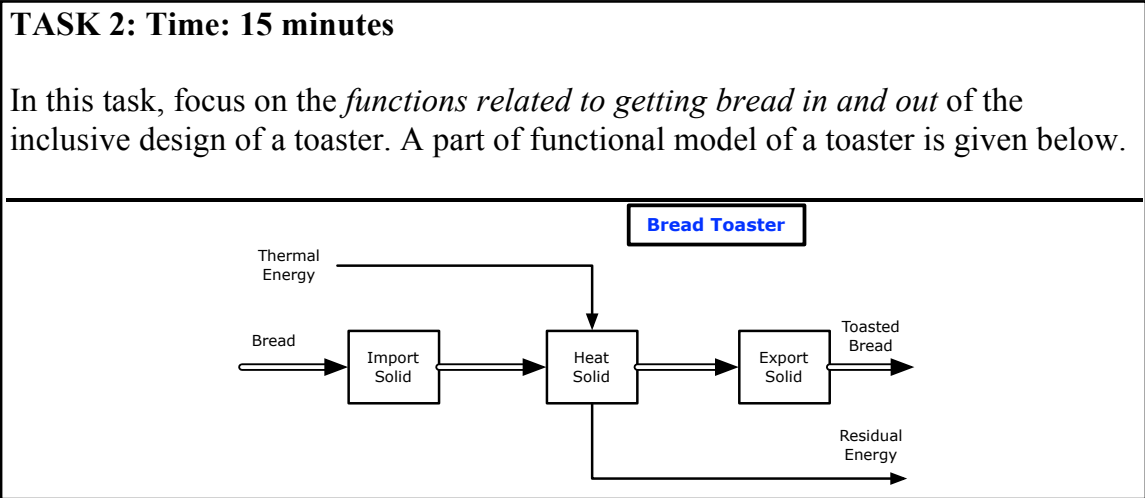
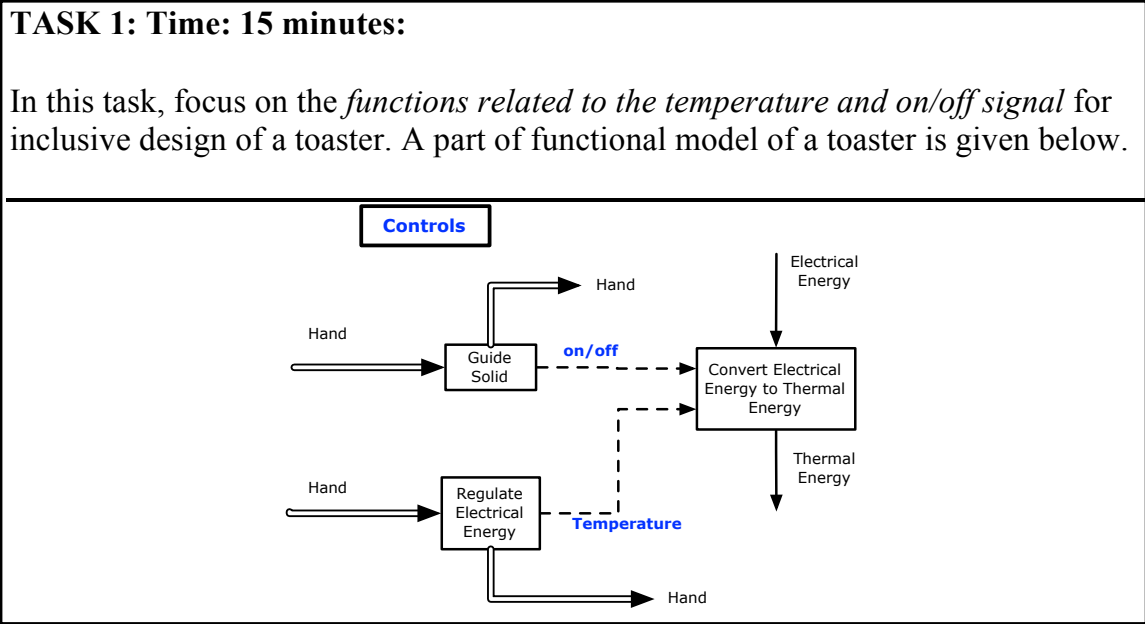
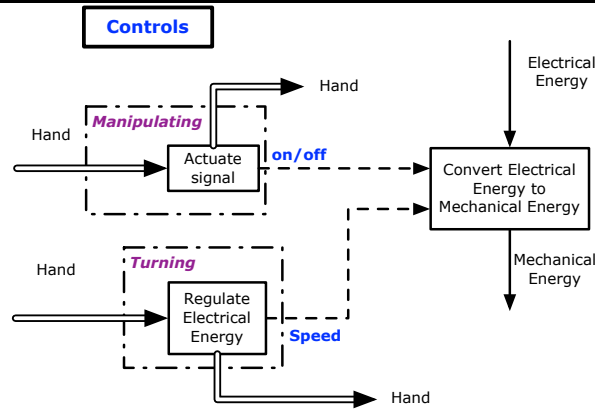


Figure B4. Task 1, Task 2 and Task 3 for the control condition of toaster design

TASK 1: Time: 15 minutes:

In this task, focus on the *activities and functions related to the speed and on/off signal* for inclusive design of a blender. A part of an actionfunction diagram of the blender is given below. Relevant inclusive design rules and ICF definitions are also listed.



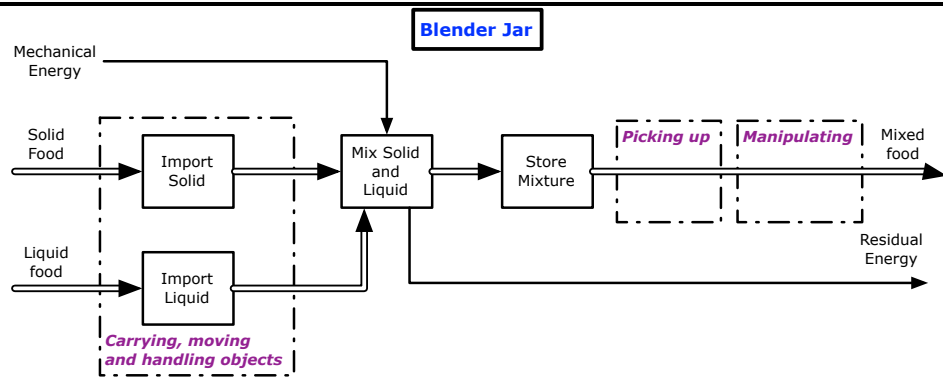
User Activity	Product function	Recommended Change
<i>Manipulating</i>	<i>Actuate Signal</i>	Morphological
<i>Turning</i>	<i>Regulate Electrical Energy</i>	Parametric

ICF term	Meaning
Manipulating	Using fingers and hands to exert control over, direct or guide something, such as when handling coins or other small objects
Turning	Using fingers, hands, and arms to rotate, turn or bend an object, such as turning a knob

Figure B5. Task 1 for the experimental condition of blender design

TASK 2: Time: 15 minutes

In this task, focus on the *activities and functions related to getting food in and out* of the inclusive design of a blender. A part of an actionfunction diagram of the blender is given below. Relevant inclusive design rules and ICF definitions are also listed.



User Activity	Product function	Recommended Change
<i>Carrying, moving and handling objects</i>	<i>Import Solid</i>	No change
<i>Picking up</i>	<i>No function</i>	Parametric
<i>Manipulating</i>	<i>No function</i>	Functional

ICF term	Meaning
Carrying, moving and handling objects	Carrying, moving and handling objects to bring them near the product, such as adding food into the container
Manipulating	Using fingers and hands to exert control over, direct or guide something, such as when removing food from the container
Picking up	Lifting or taking up a small object with hands and fingers, such as when picking up a container

Figure B6. Task 2 for the experimental condition of blender design

TASK 3: Time: 10 minutes

In this task, focus on the *activities and functions* related to the customer need “*easy to clean*” for the inclusive design of a blender. Cleaning of blender jar involves soap and water. A part of an actionfunction diagram of the blender is given below. Relevant inclusive design rules and ICF definitions are also listed.

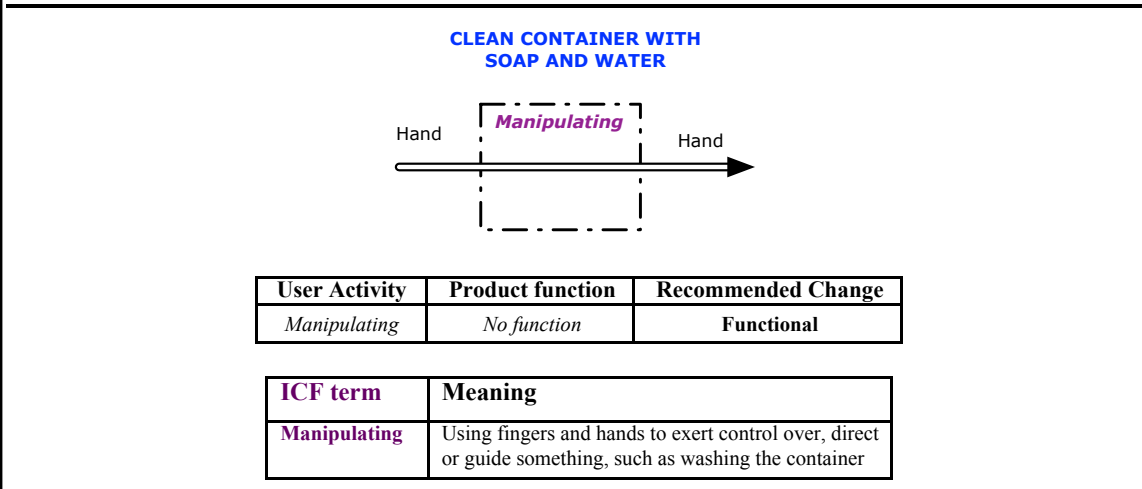


Figure B7. Task 3 for the experimental condition of blender design

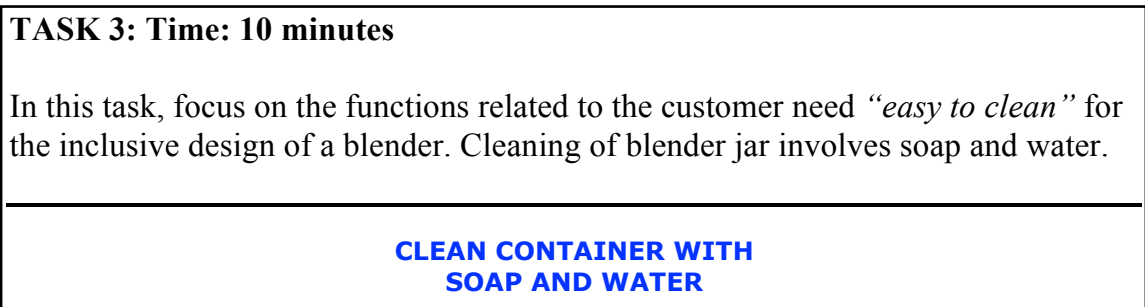
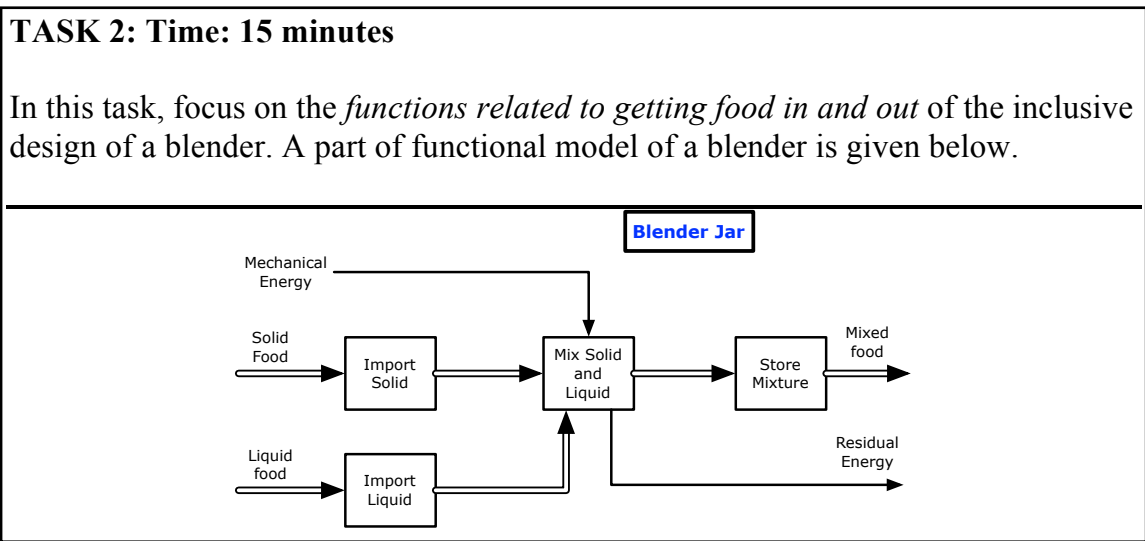
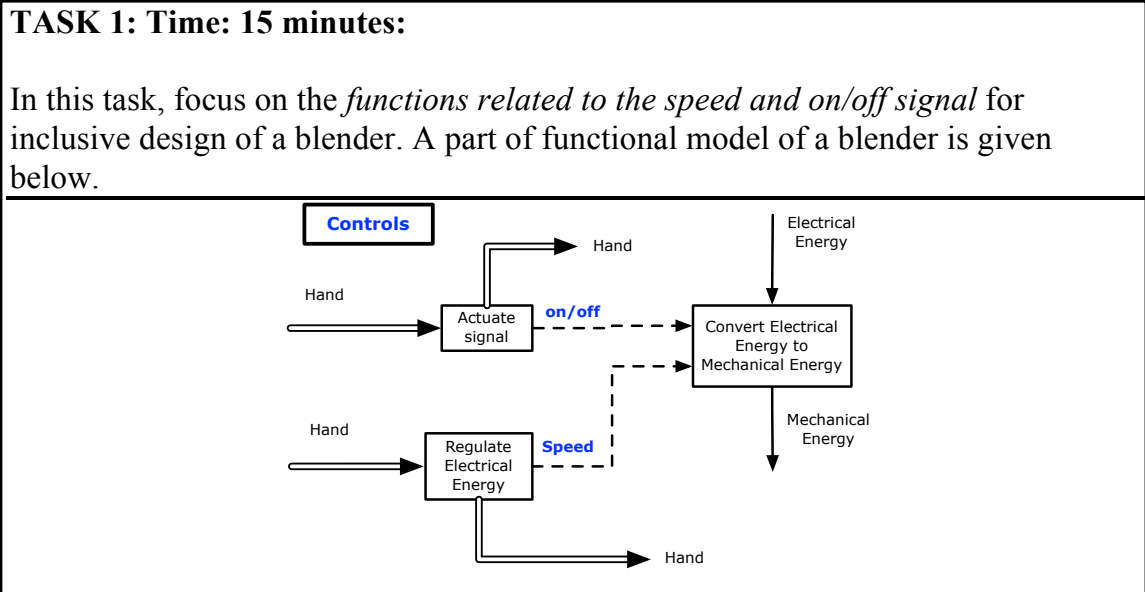


Figure B8. Task 1, Task 2 and Task 3 for the control condition of blender design

APPENDIX C

EXACT MATCHING GRAPH GRAMMARS

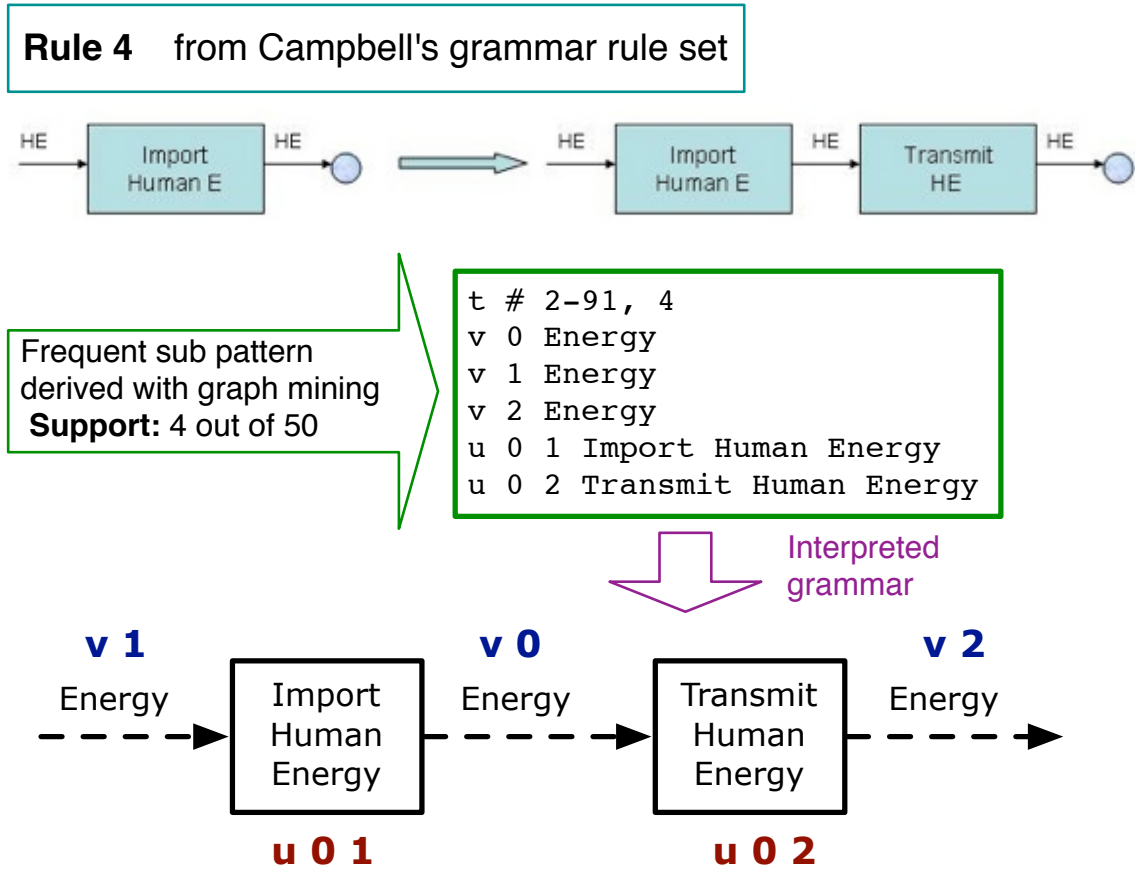


Figure C1. Rule 4 from Campbell's graph grammar rule and the corresponding frequent subgraph with a frequency of 4 and size 2

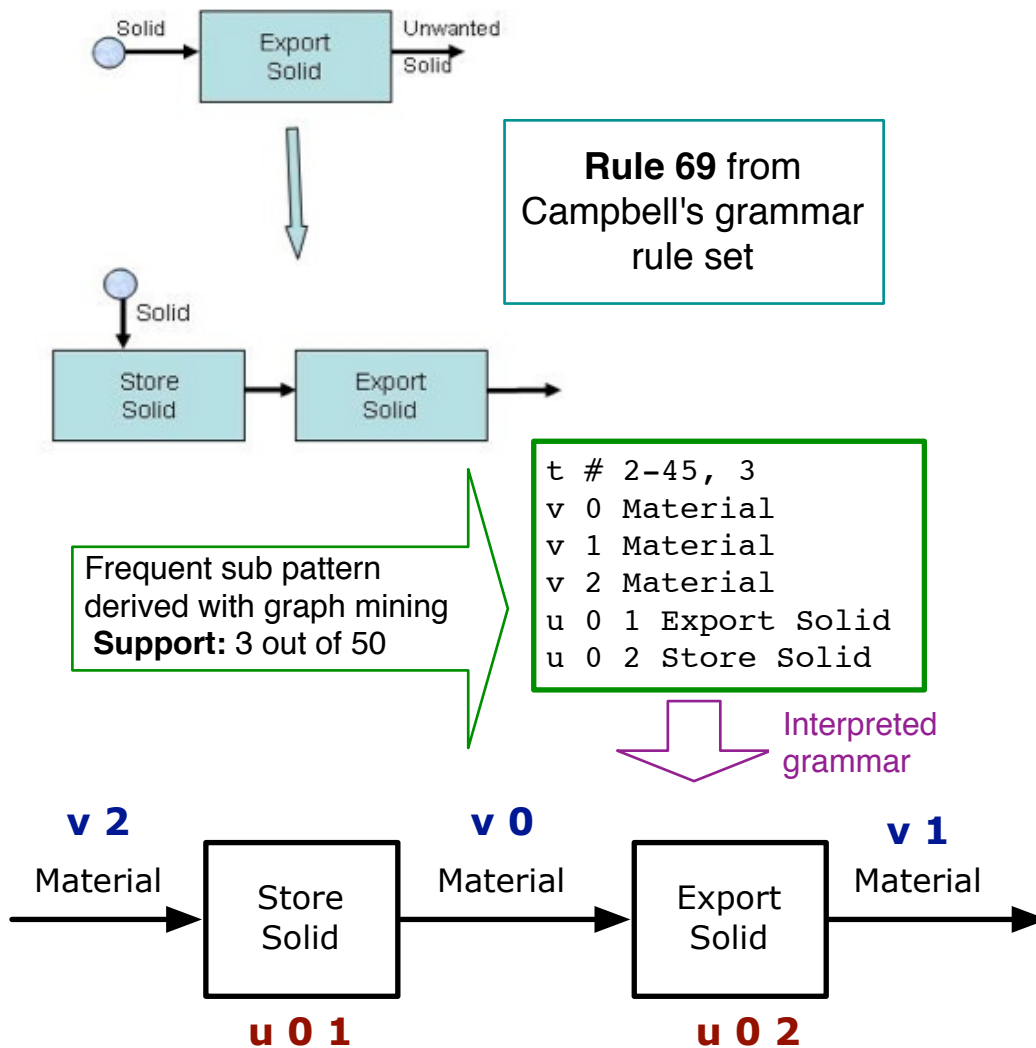


Figure C2. Rule 69 from Campbell's graph grammar rule and the corresponding frequent subgraph with a frequency of 3 and size 2

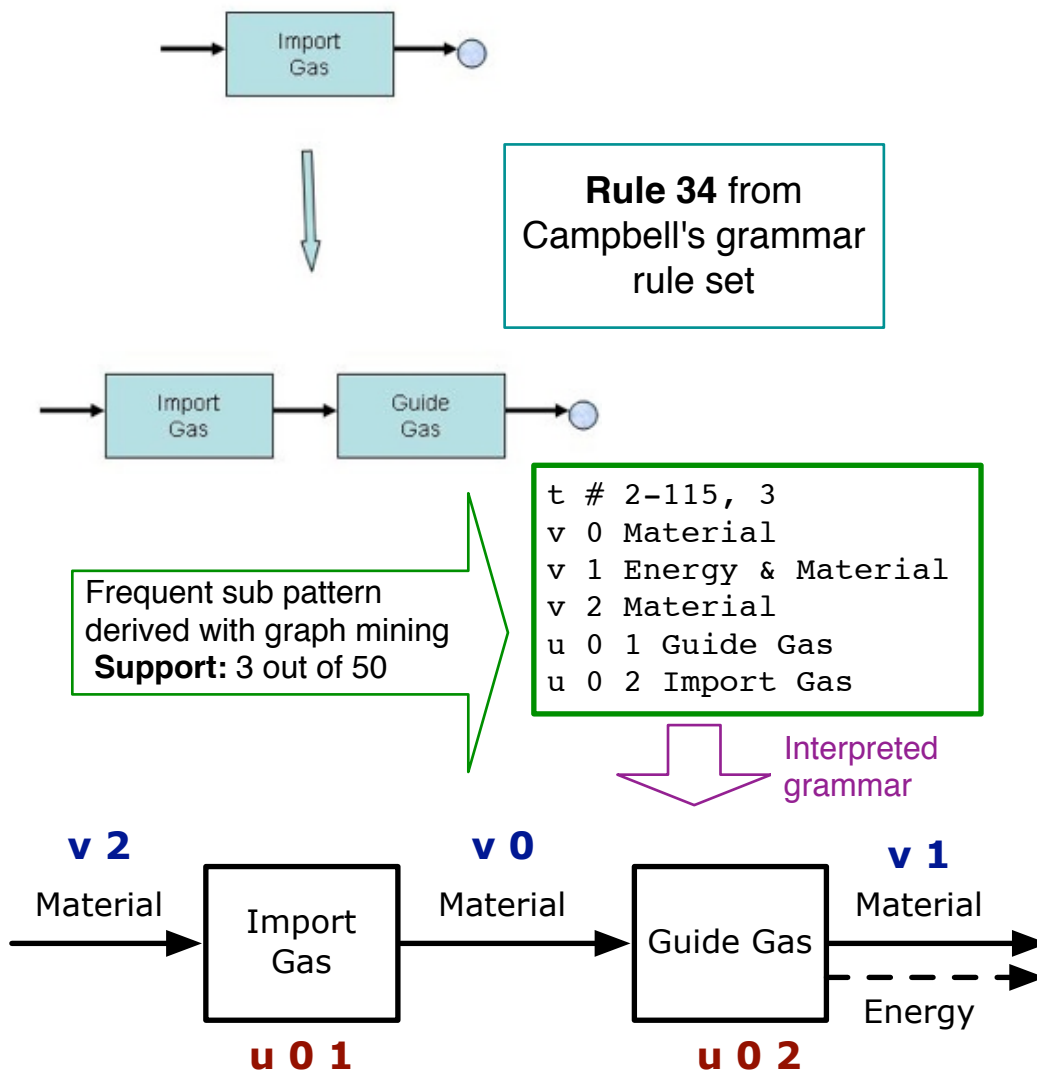


Figure C3. Rule 34 from Campbell's graph grammar rule and the corresponding frequent subgraph with a frequency of 3 and size 2

APPENDIX D

INCLUSIVE DESIGN REPOSITORY OF 65 PRODUCT PAIRS

Table D1. Database of 65 product pairs studied













	Product	Typical	Universal
1	PT Cruiser		
2	Box cutter		
3	Seat Belt Adaptor		
4	Ford Focus		
5	Wash Basin		
6	Bicycle		

Table D1. Database of 65 product pairs studied (continued)




	Product	Typical	Universal
7	Cutting Board		
8	Food storage box		
9	Tooth-brush		
10	Trashcan		
11	Arm Chair		
12	Chopping Bowl		
13	Closet		

Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
14	Cook Top		
15	Pizza Cutter		
16	Tooth-brush Dispenser		
17	Power Doors		
18	Eyewear		
19	Faucet		
20	Jar Opener 1		

Table D1. Database of 65 product pairs studied (continued)


	Product	Typical	Universal
21	Nail File		
22	Refrigerator		
23	Hammer		
24	Hoe		
25	Iron		
26	Door Knob		
27	Lamp		

Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
28	Recliner Lever		
29	Bottle Cap		
30	Fountain Drink Lid		
31	Microwave		
32	Touch Faucet		
33	Blood Pressure Monitor		
34	Can Opener		

Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
35	Oven		
36	Telephone		
37	Plug		
38	Garlic Press		
39	Pruner		
40	One-Hole Hole Punch		
41	Razor		

Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
42	Recliner		
43	Remote		
44	Kitchen Scale		
45	Scissors		
46	Toilet Auto Seat		
47	Easy Reach Seat Belt		
48	Shovel		

Table D1. Database of 65 product pairs studied (continued)








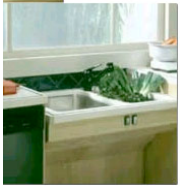












	Product	Typical	Universal
49	Hand Held Shower		
50	Braille Signs		
51	Jar Opener -2		
52	Kitchen sink		
53	Adjustable height Sink		
54	Spatula		
55	Syringe		

Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
56	Thermometer		
57	Toilet		
58	Ice Cube Tray		
59	Trowel		
60	Bathtub		
61	Dish-washer		
62	Washer		

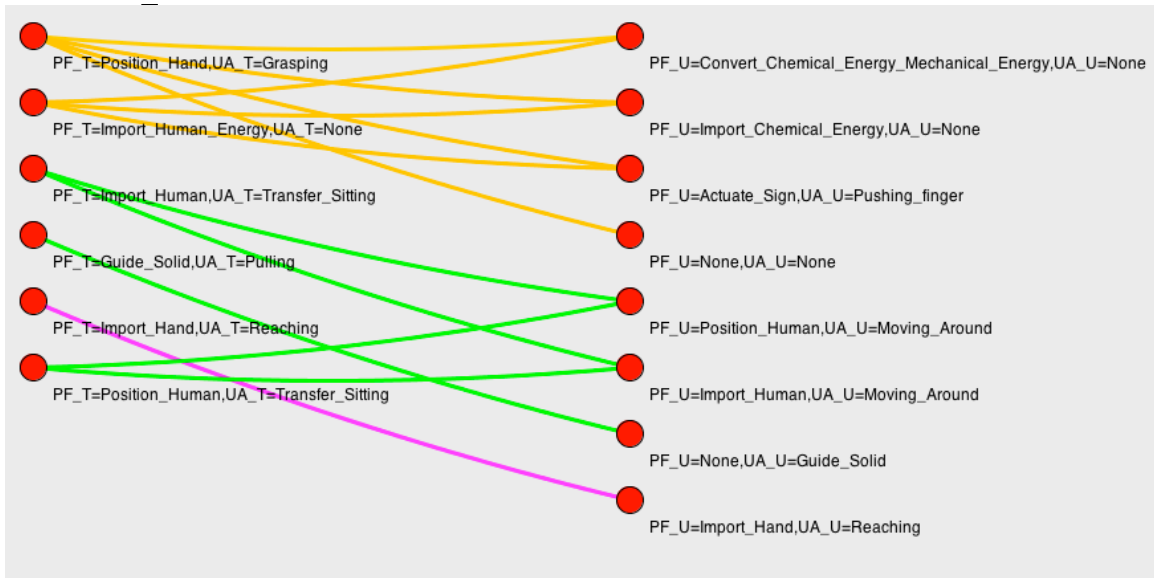
Table D1. Database of 65 product pairs studied (continued)

	Product	Typical	Universal
63	Wrist Watch		
64	Wine Opener		
65	Car window controls		

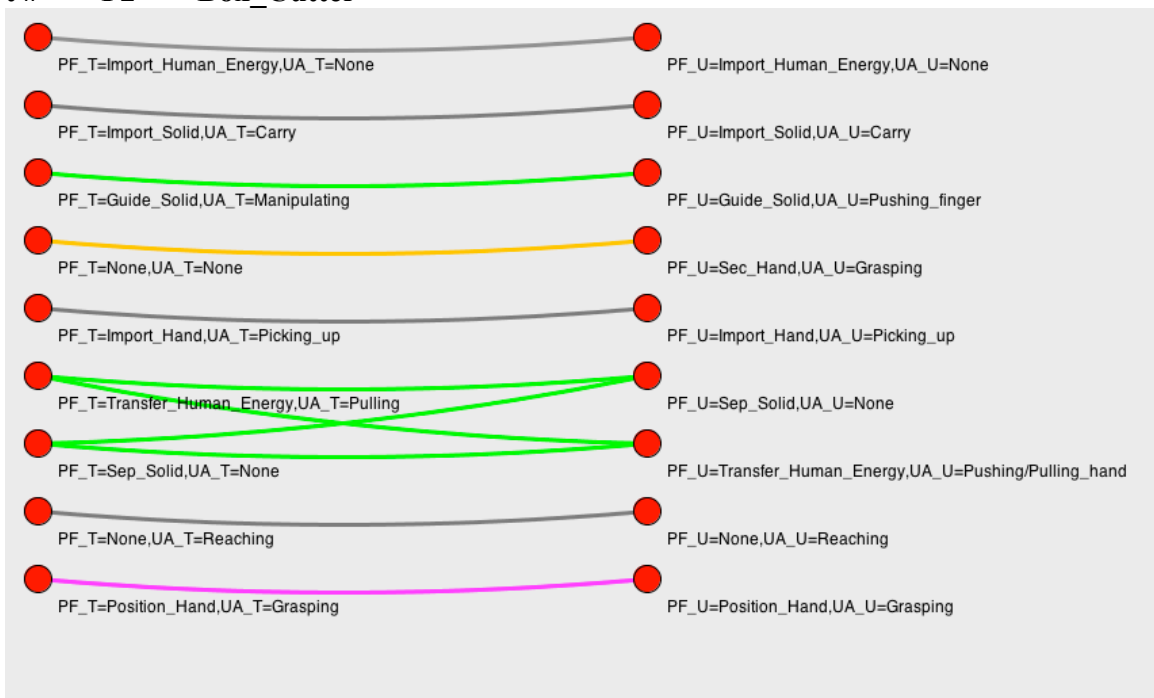
APPENDIX E

GRAPHS OF 65 PRODUCT PAIRS

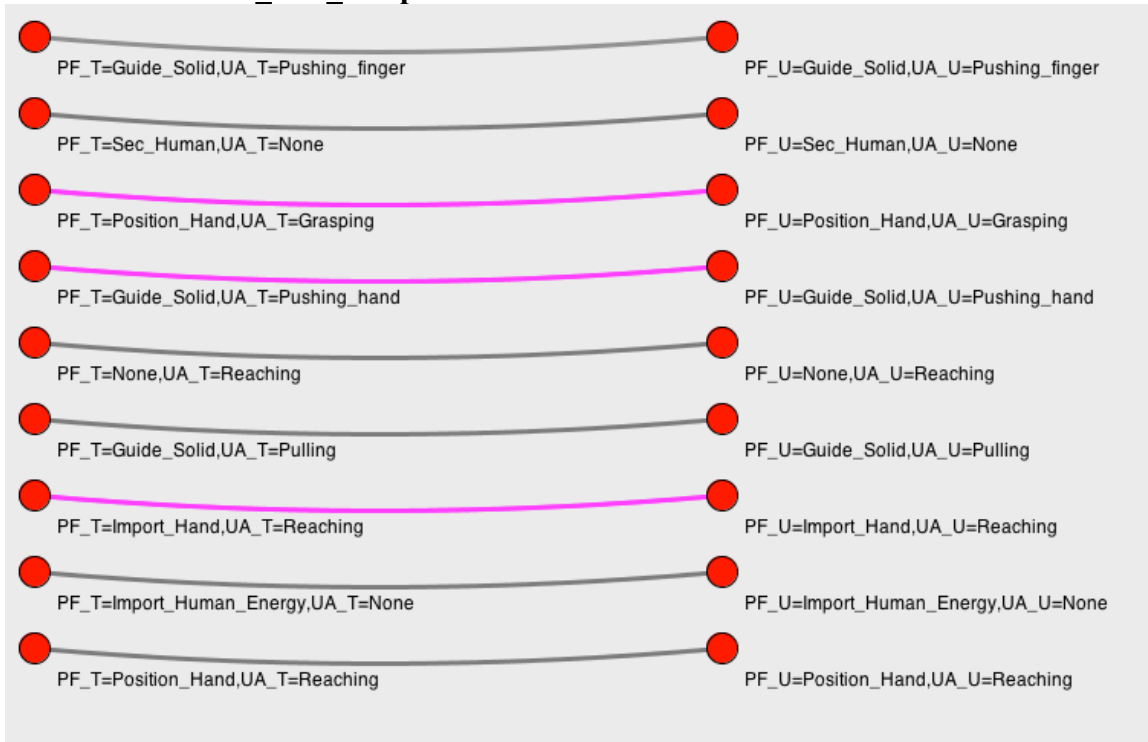
t # P1 PT_Cruiser



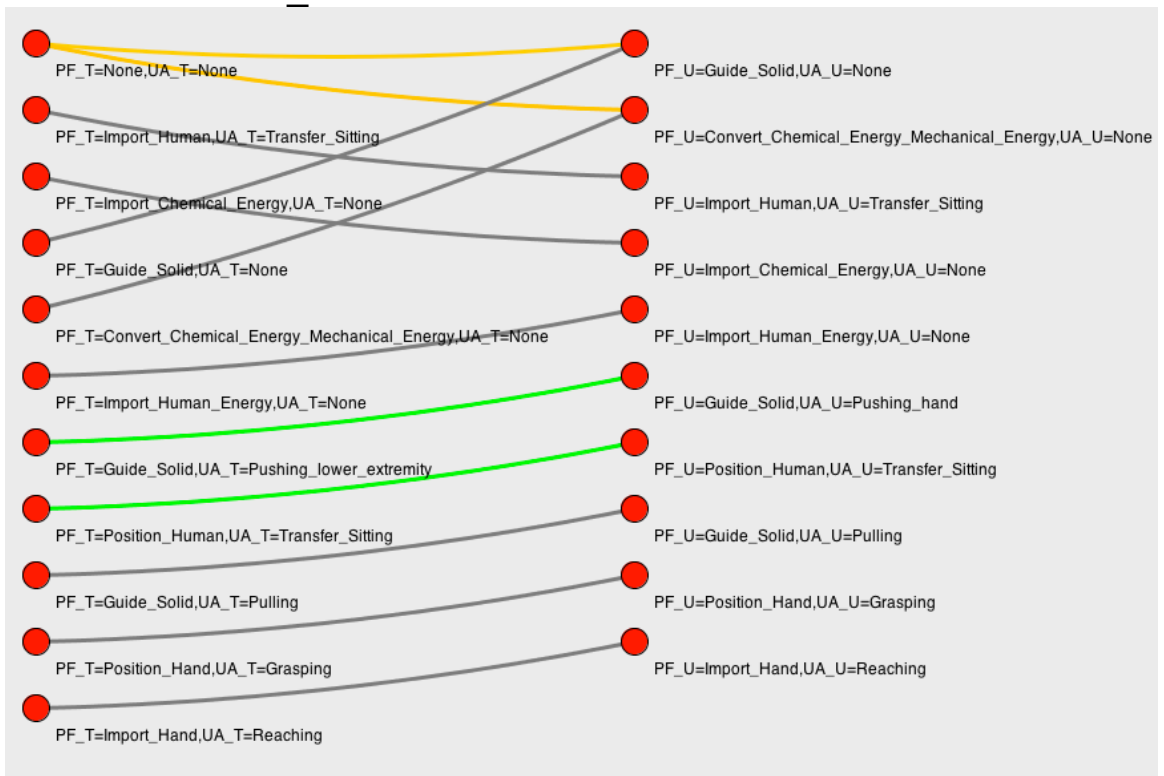
t # P2 Box_Cutter



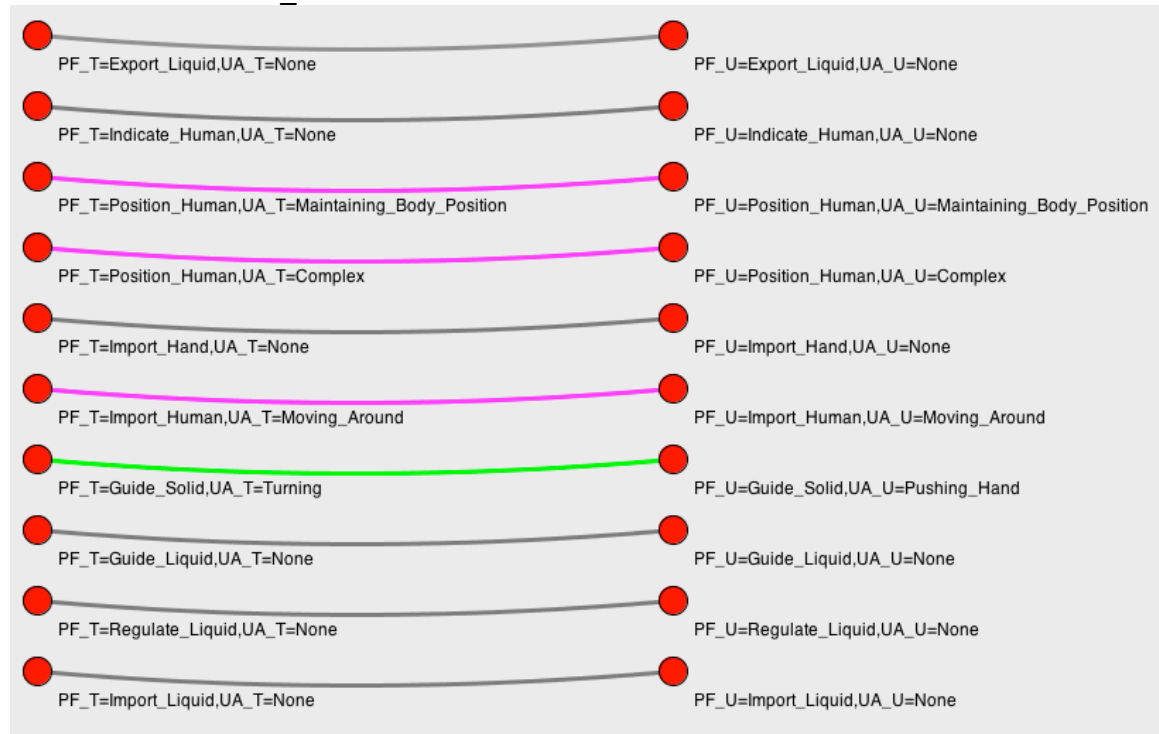
t # P3 Seat_Belt_Adaptor



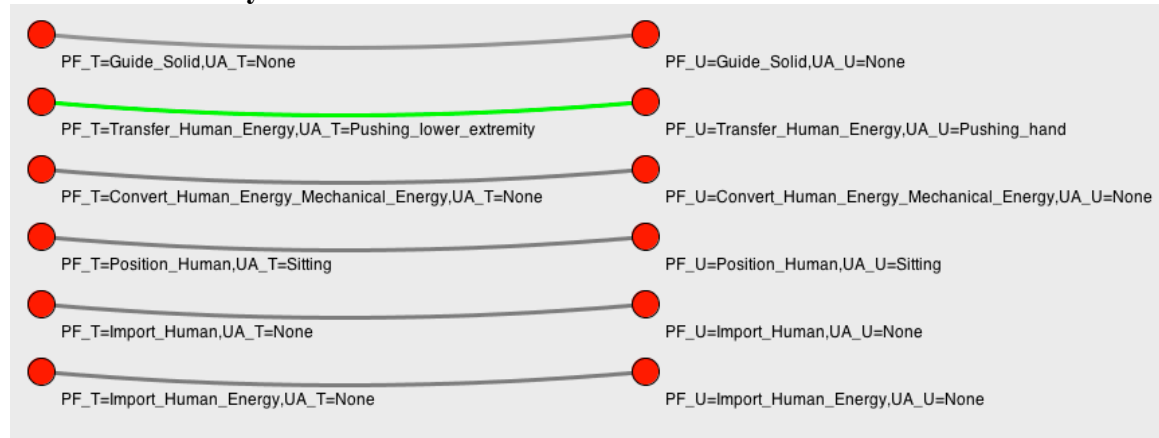
t # P4 Ford_Focus



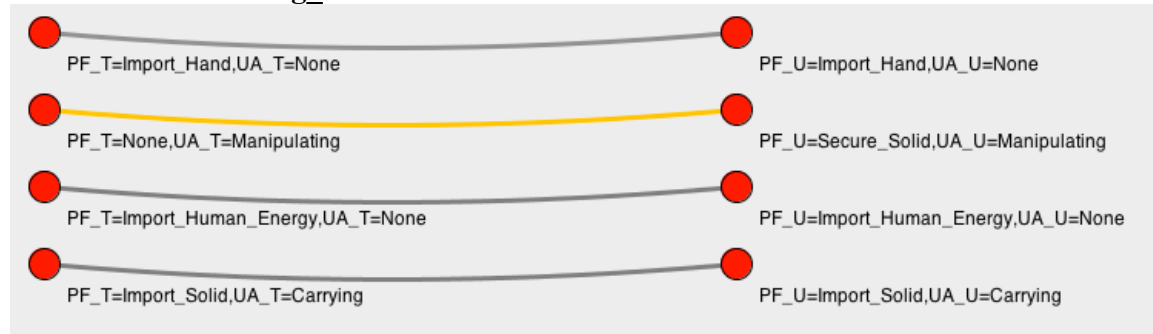
t # P5 Wash_Basin



t # P6 Bicycle



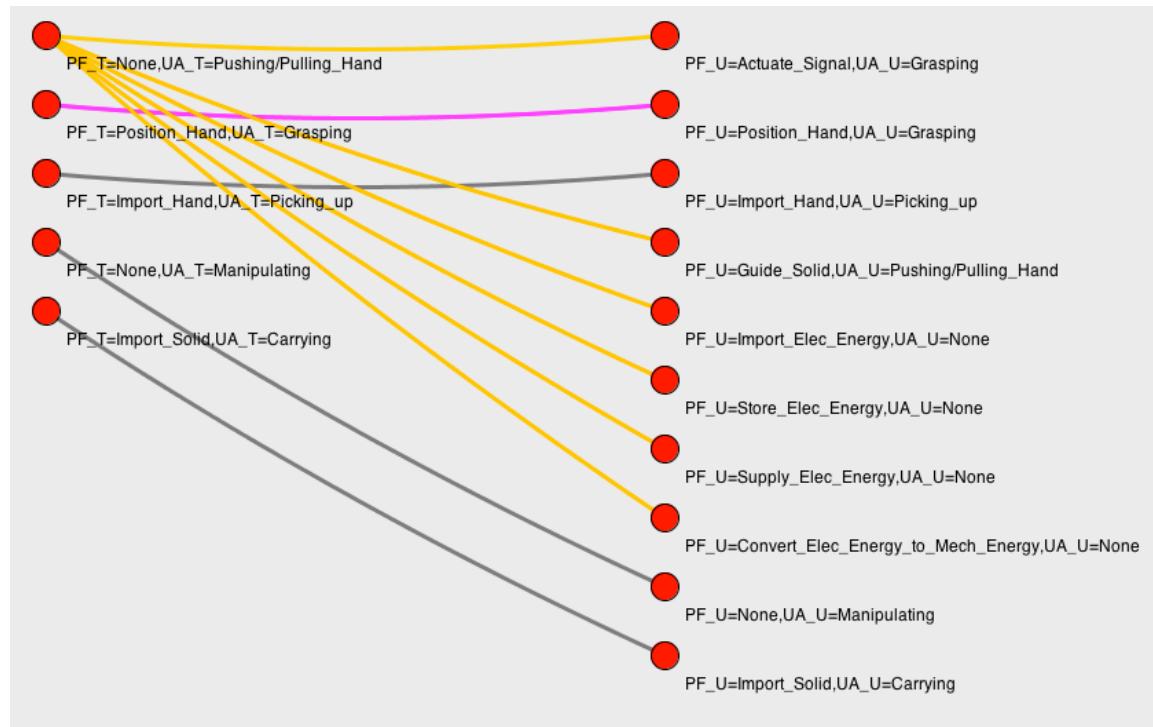
t # P7 Cutting Board



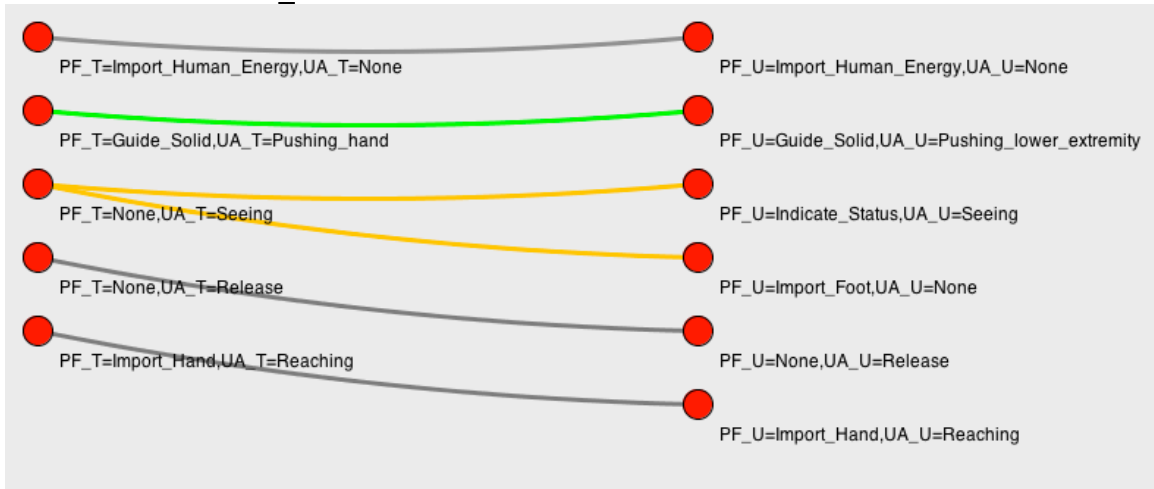
t # P8 Tupperware



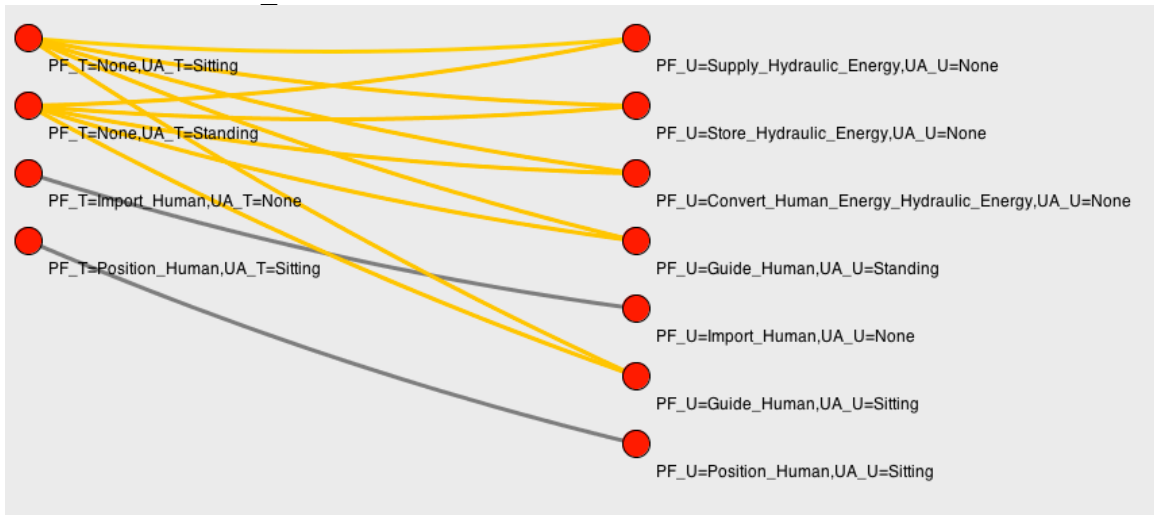
t # P9 Toothbrush



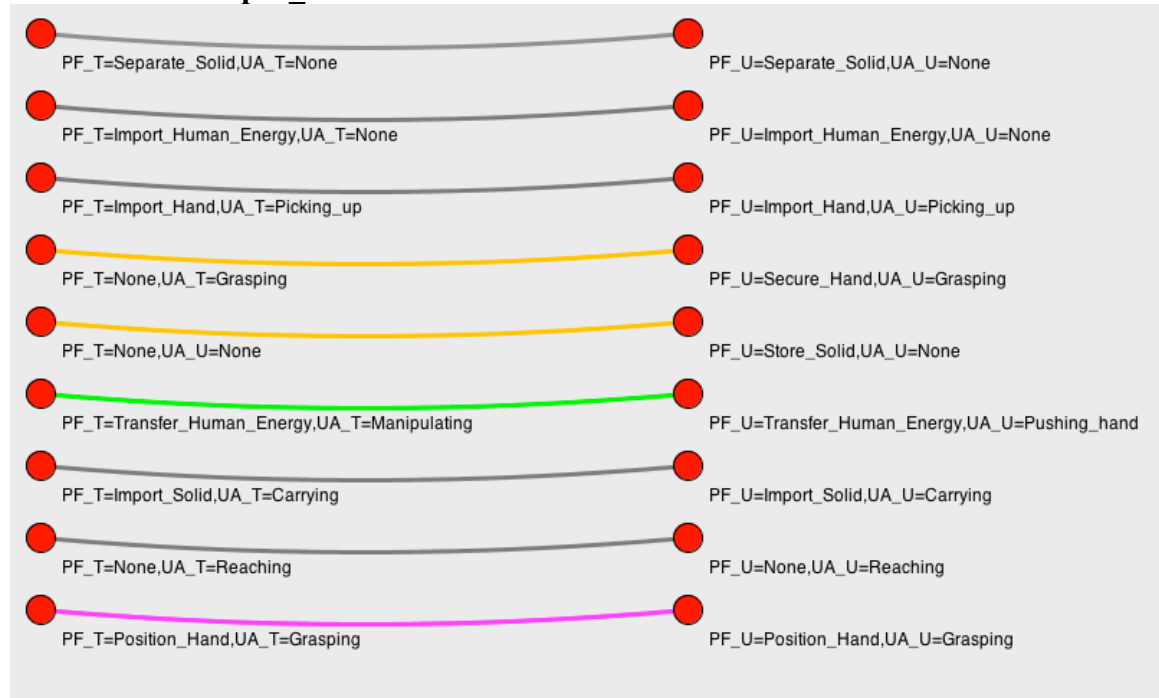
t # P10 Trash_Can



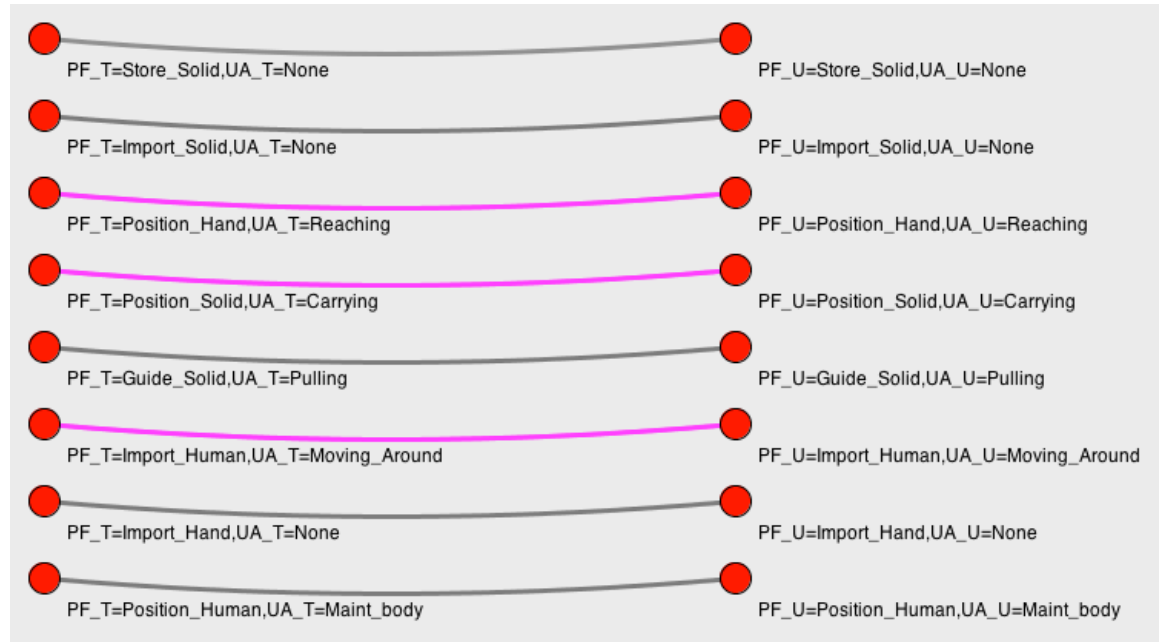
t # P11 Arm_Chair



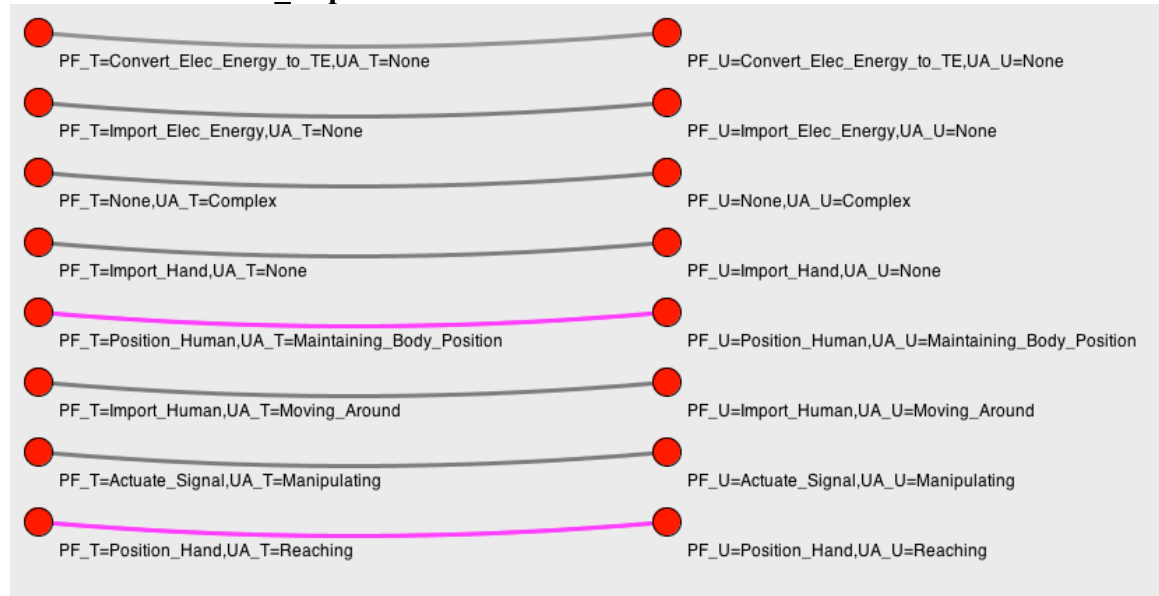
t # P12 Copco_Bowl



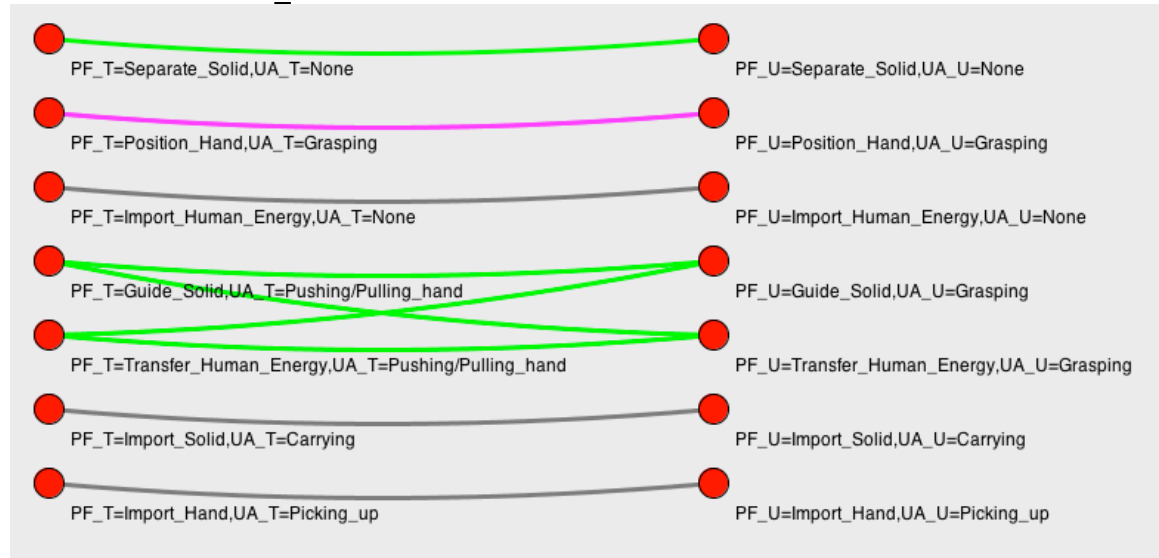
t # P13 Closet



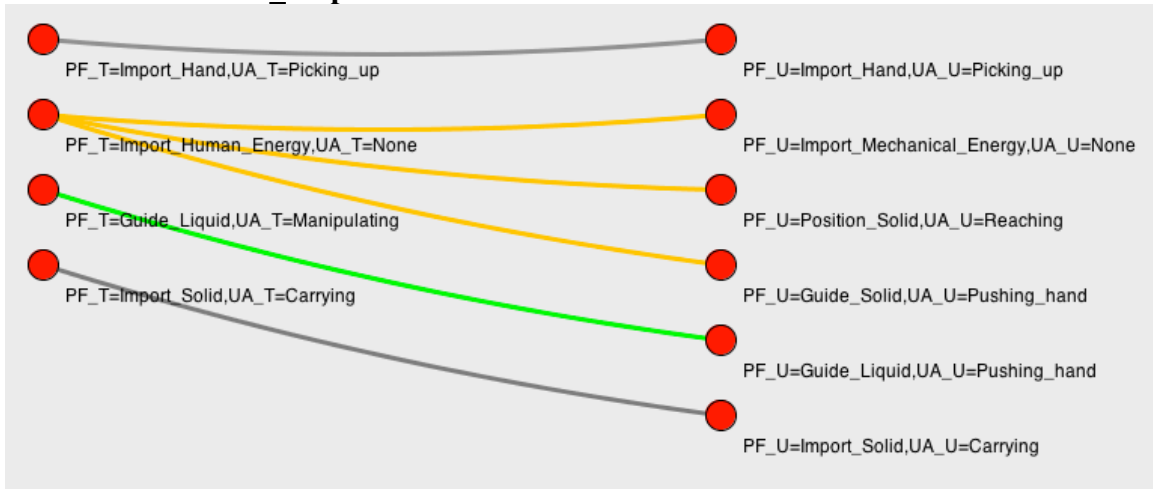
t # P14 Cook_Top



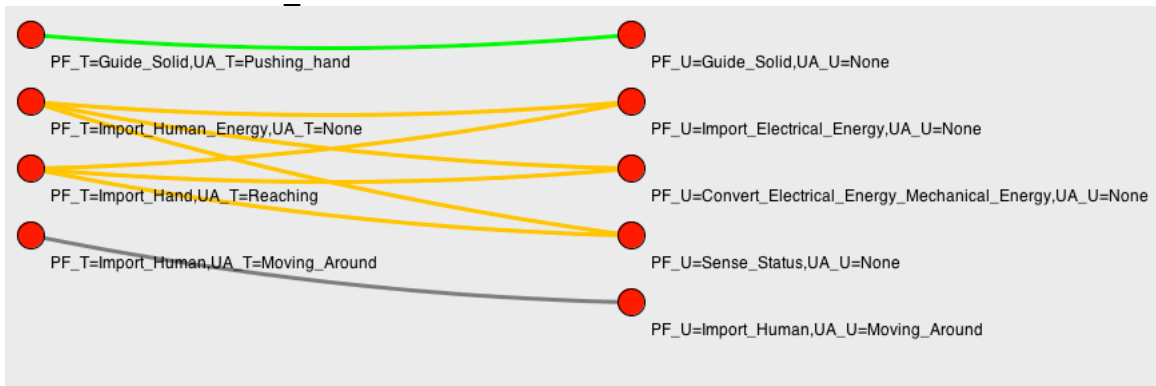
t # P15 Pizza_Cutter



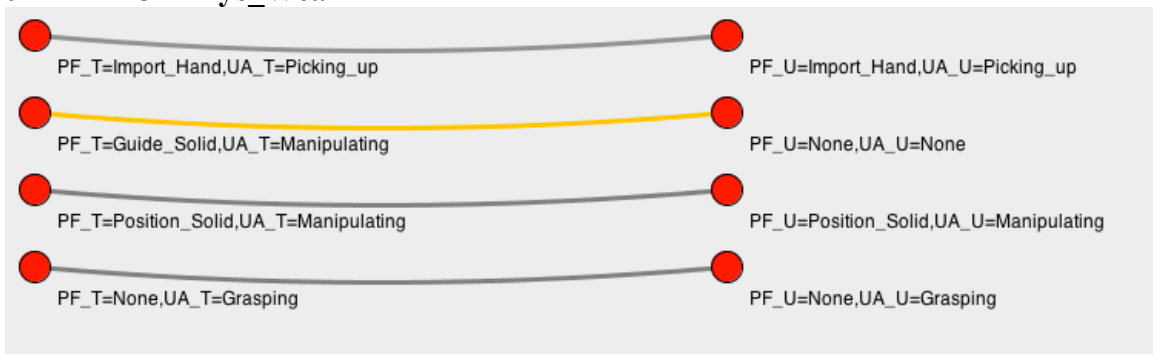
t # P16 Paste_Dispenser



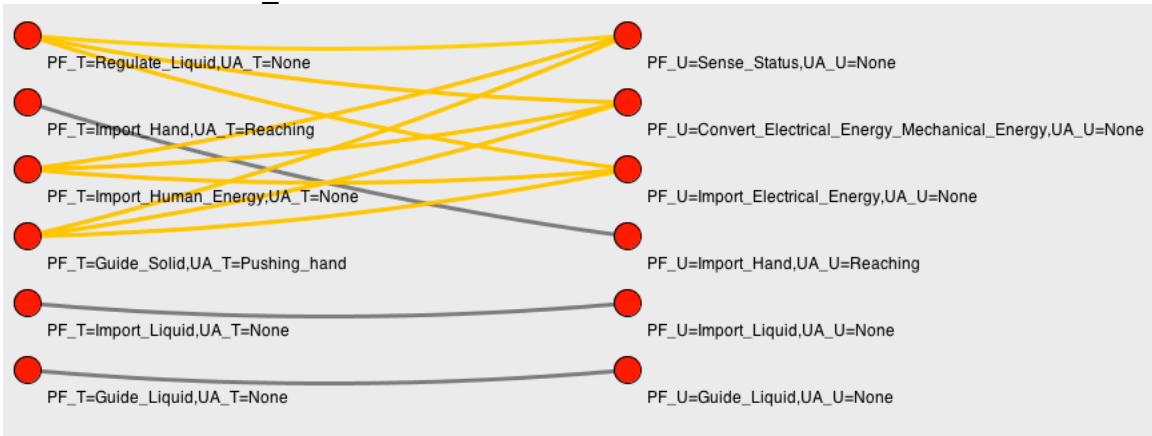
t # P17 Power_Door



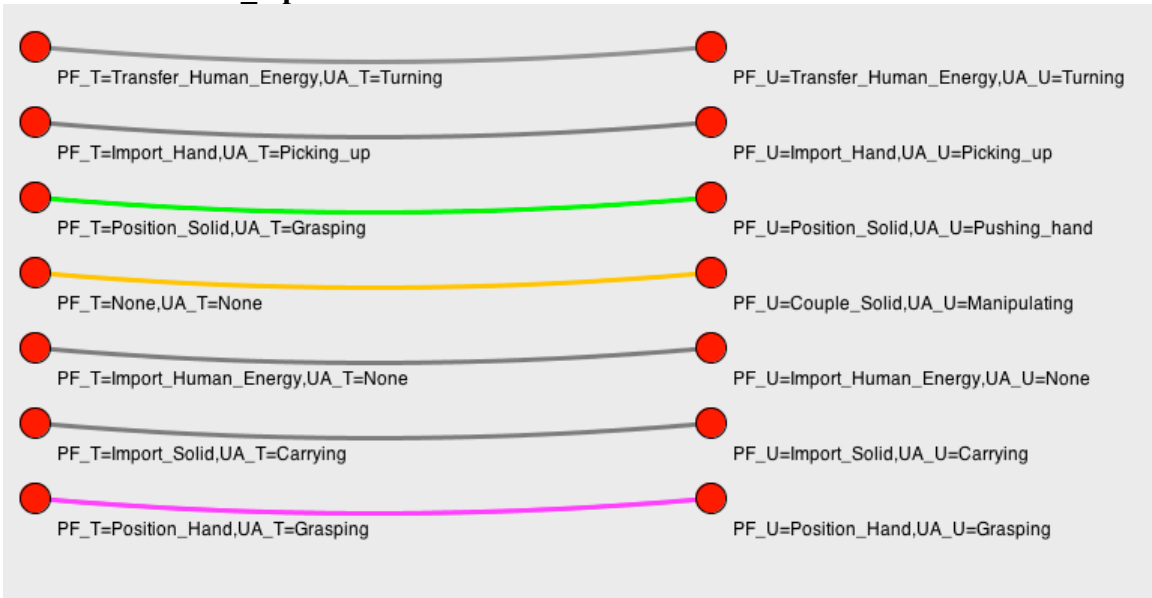
t # P18 Eye_Wear



t # P19 Auto_Faucet



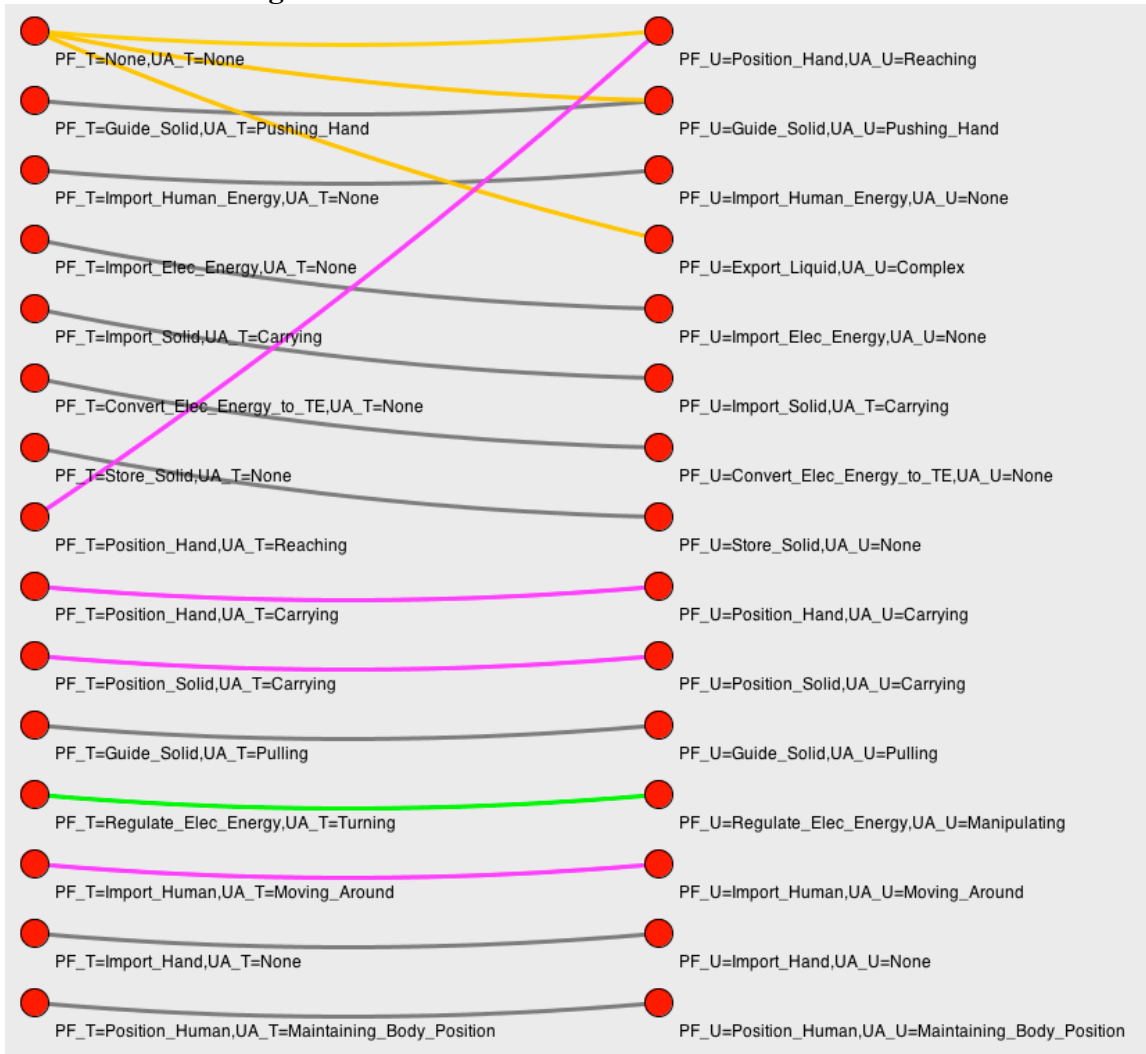
t # P20 Jar_Opener



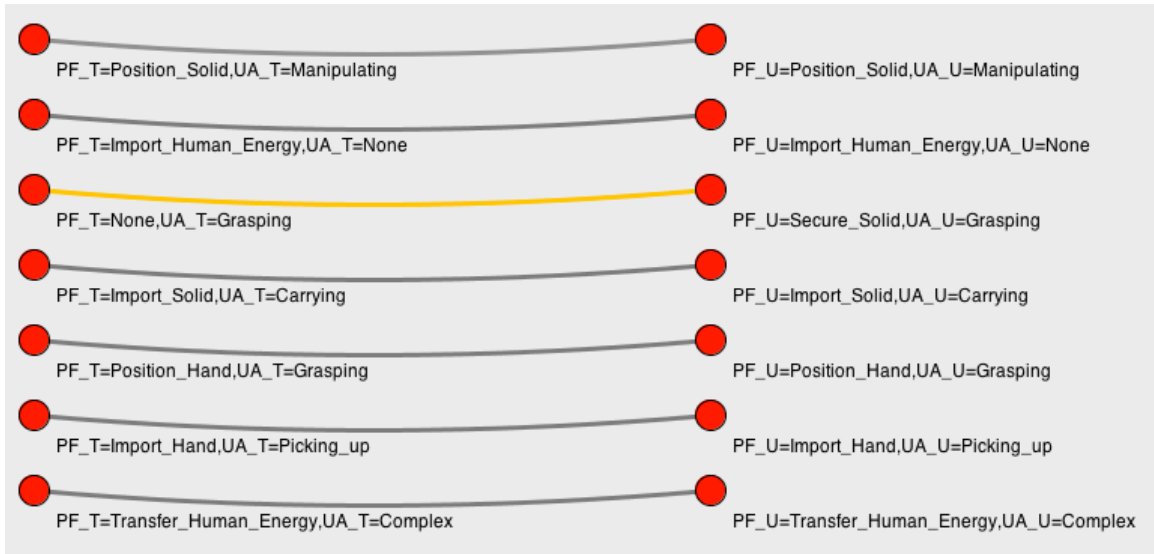
t # P21 Nail_File



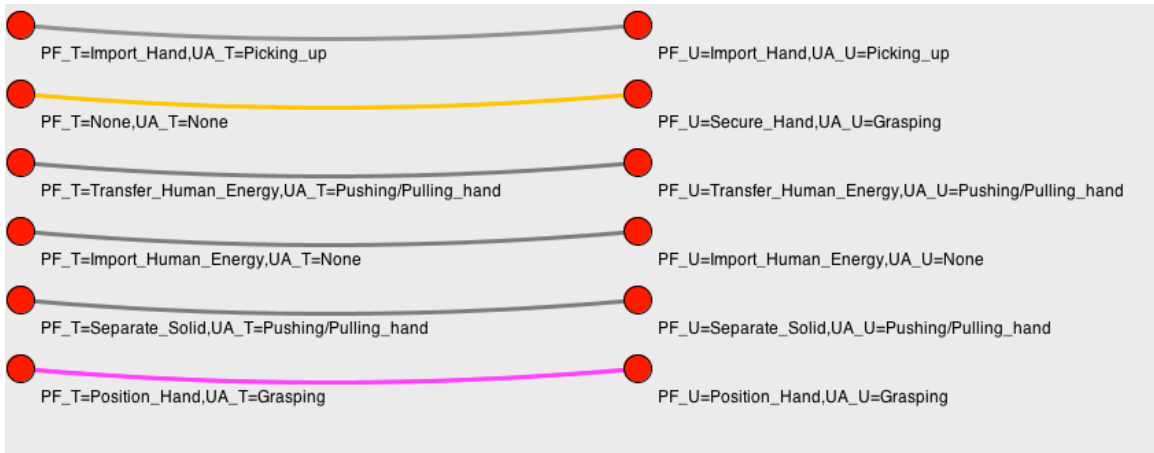
t # 22 Refrigerator



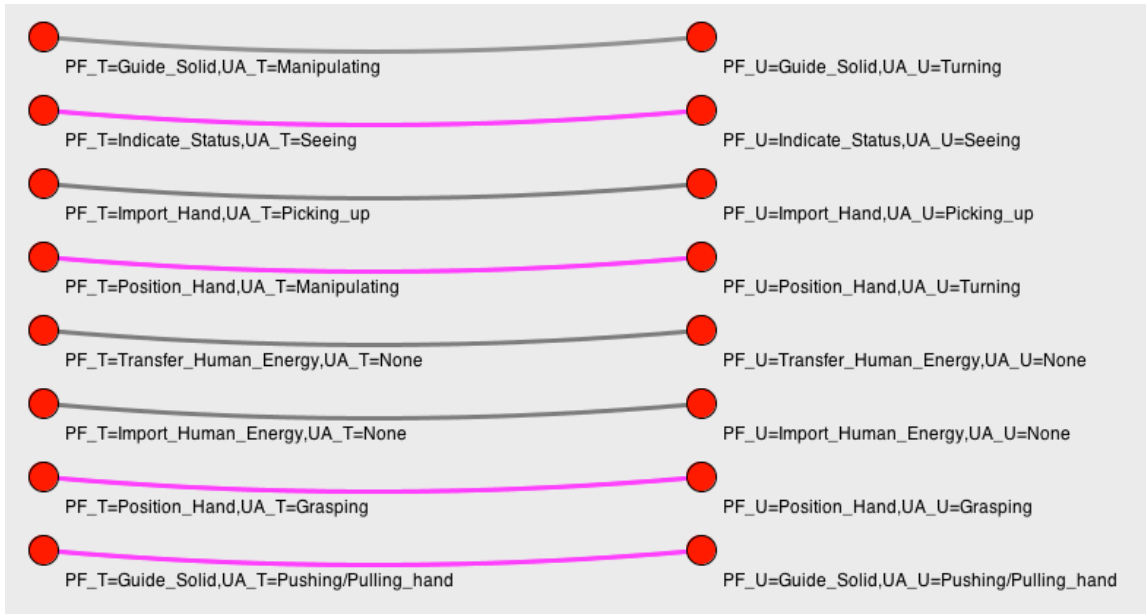
t # 23 Hammer



t # 24 Hoe



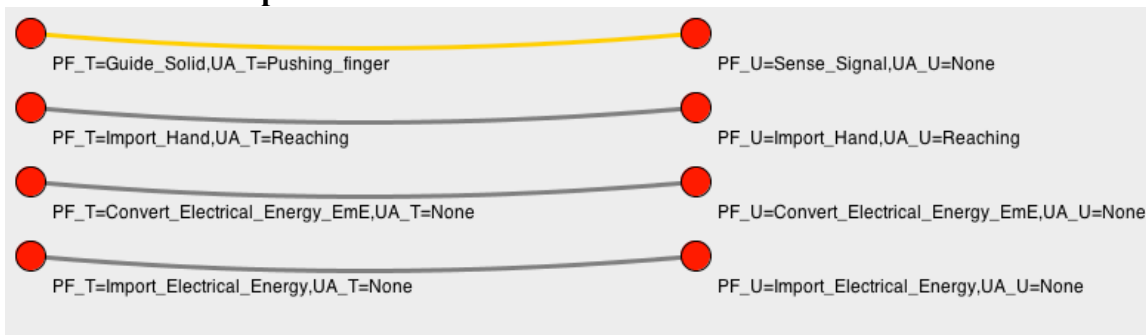
t # 25 Iron



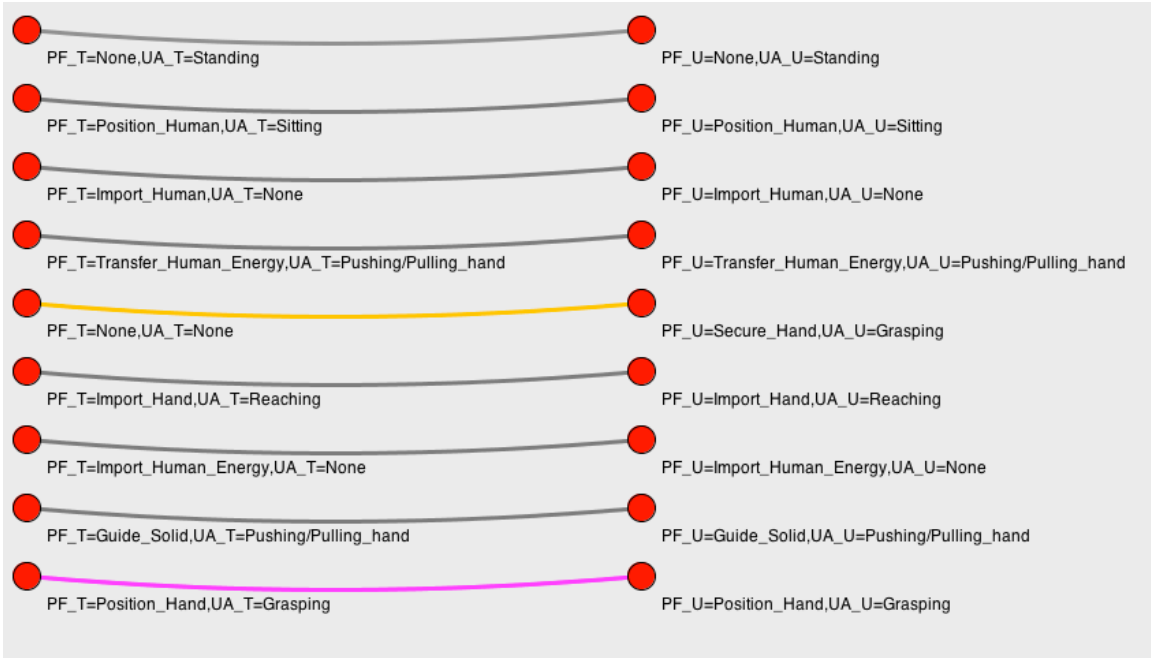
t # 26 Door Knob



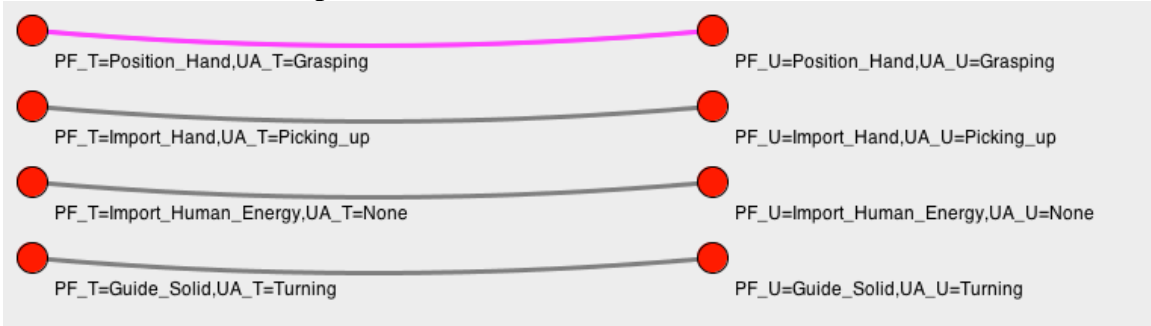
t # 27 Lamp



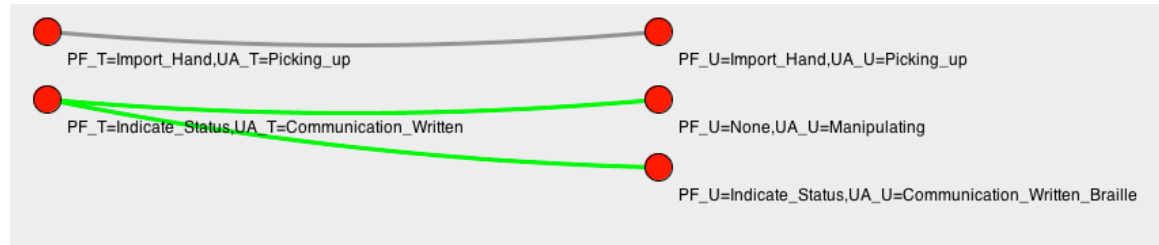
t # 28 Recliner with extended lever



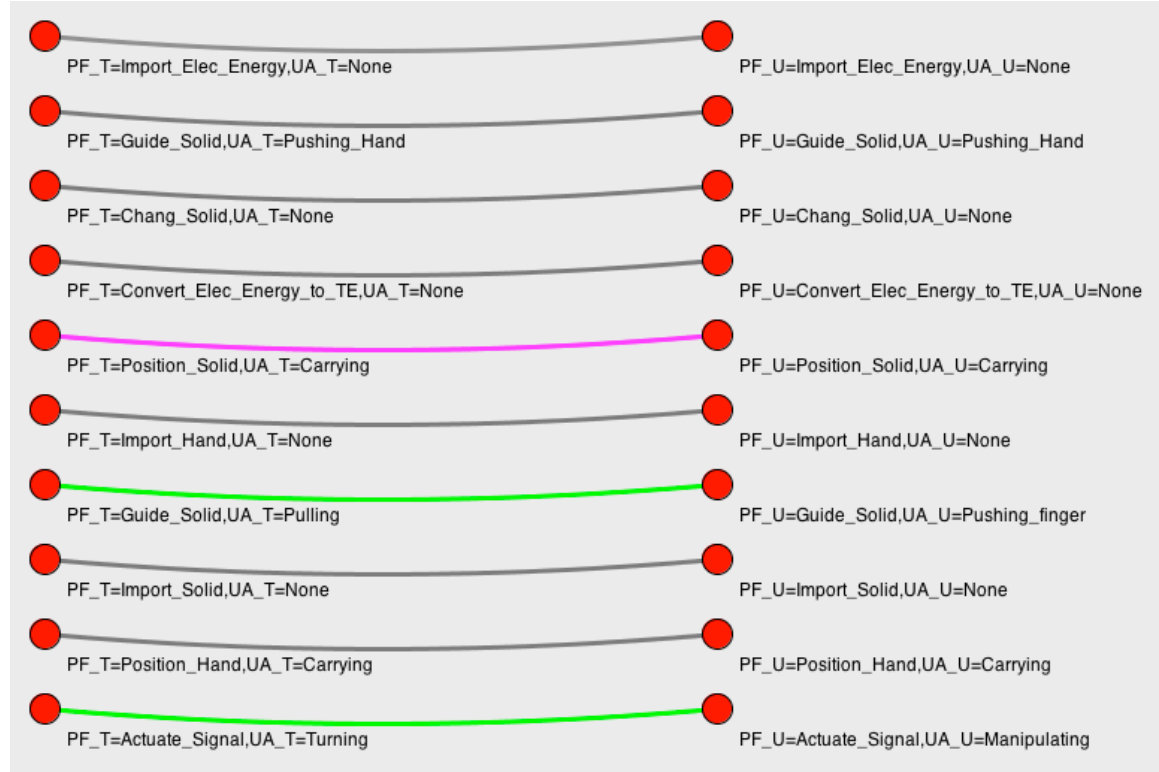
t # 29 Bottle cap



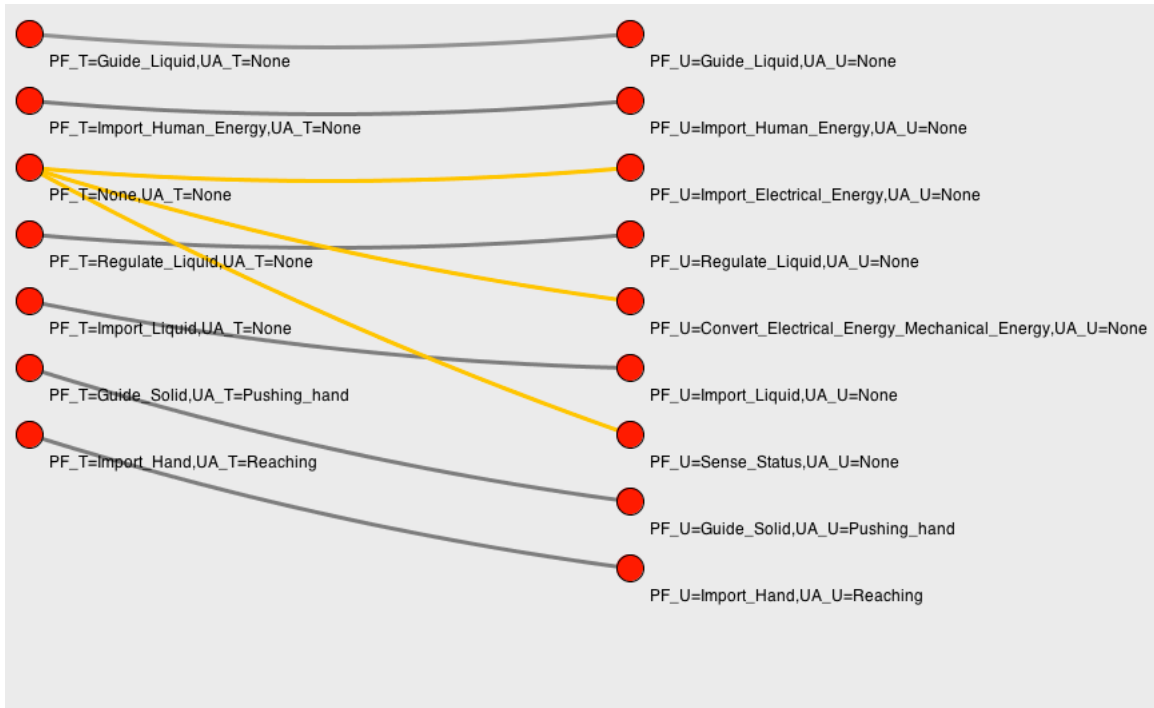
t # 30 Braille Lid



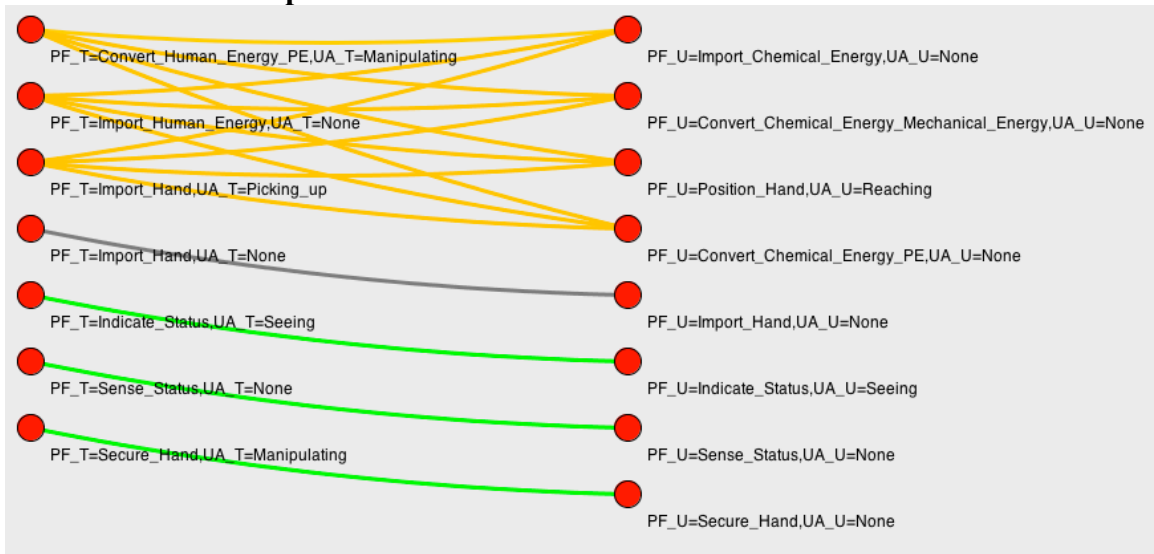
t # 31 Microwave



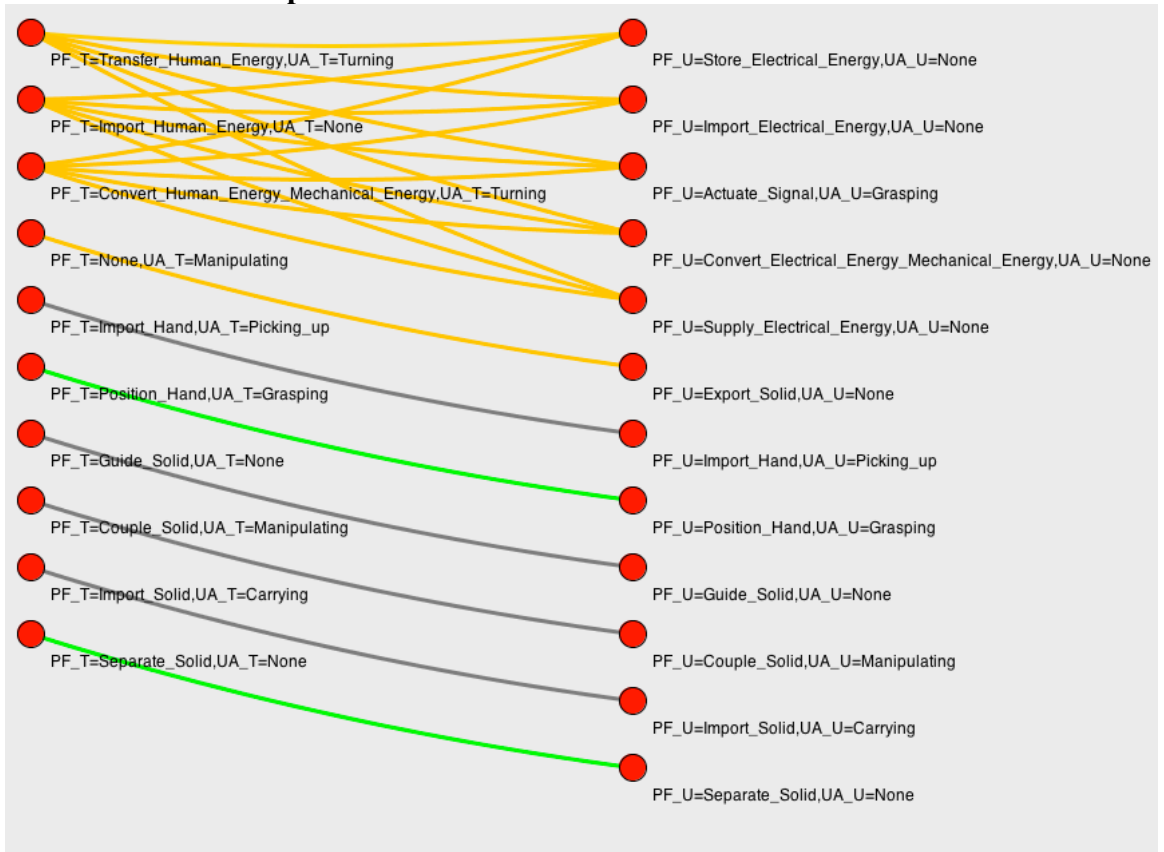
t # 32 Touch Start Faucet



t # 33 Blood pressure monitor



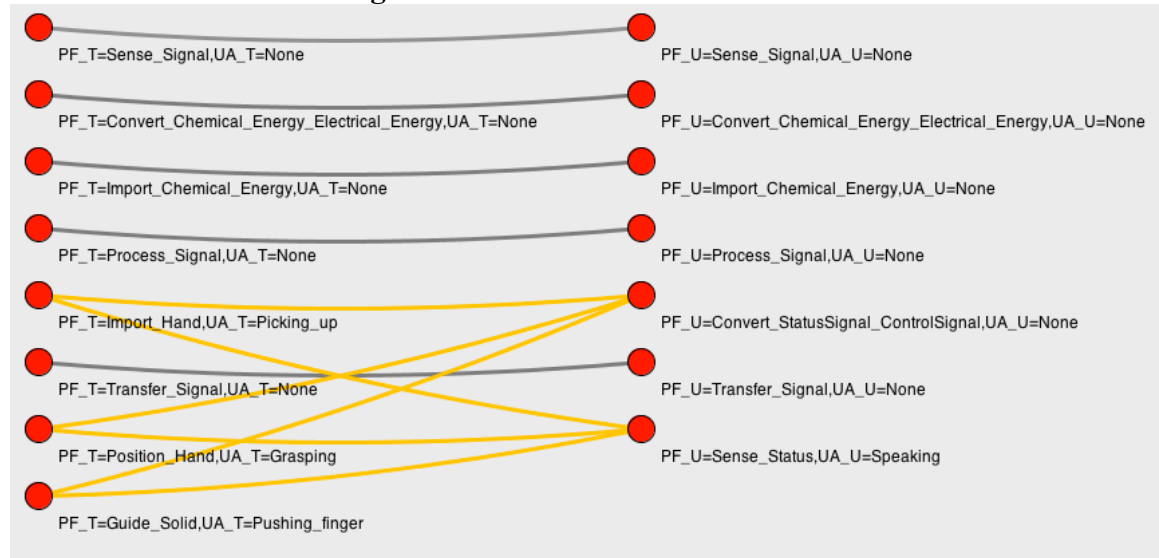
t # 34 Can Opener



t # 35 Oven



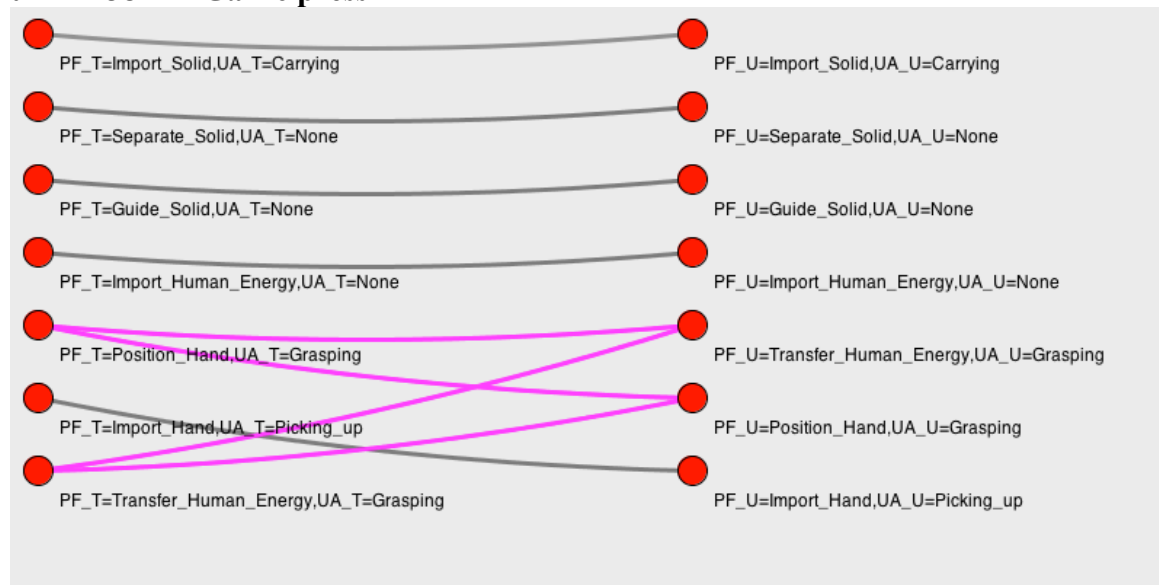
t # 36 Voice Dialing



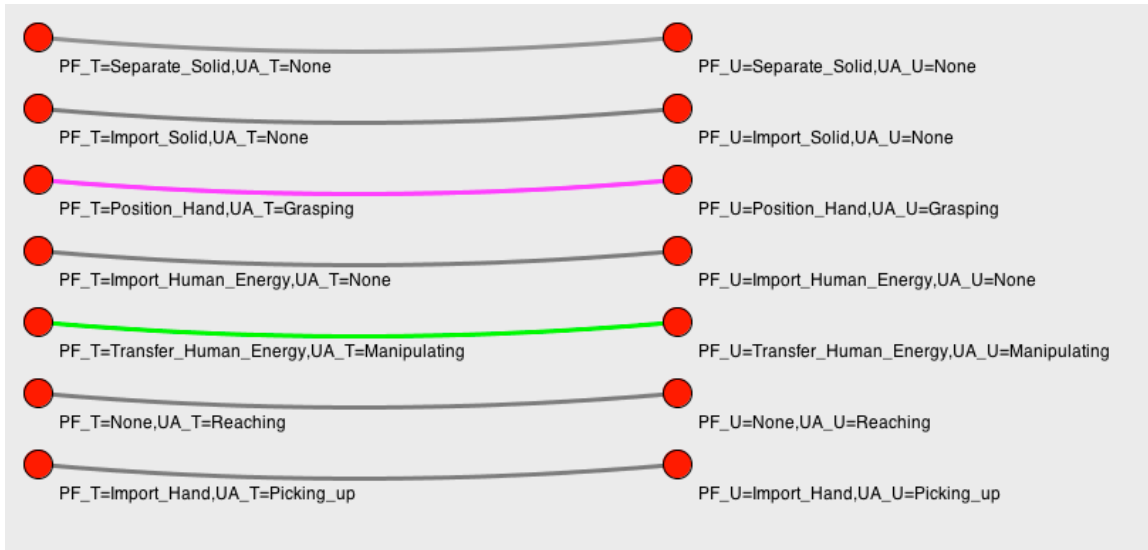
t # 37 Electric outlet plug



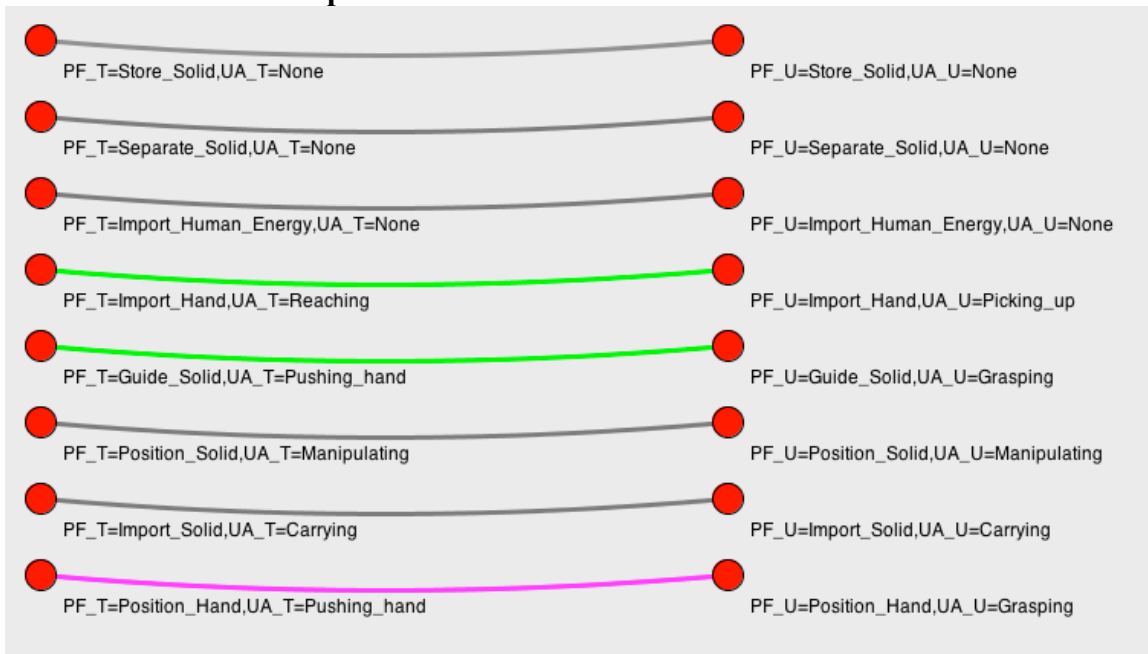
t # 38 Garlic press



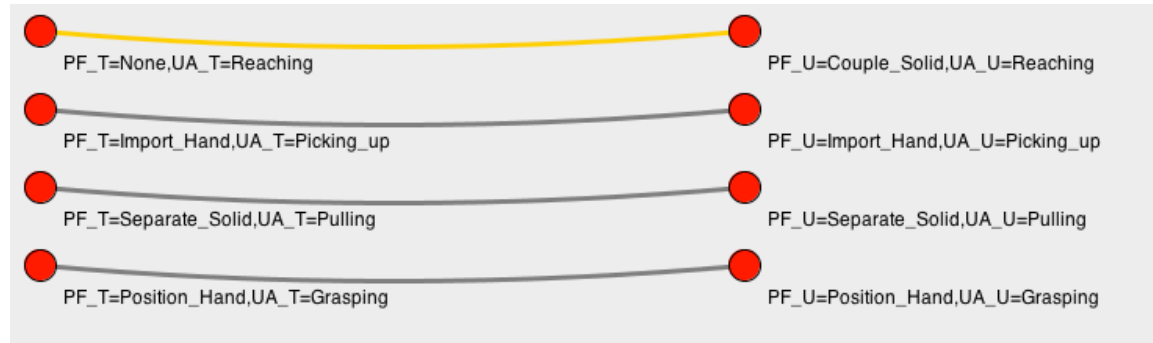
t # 39 Pruner



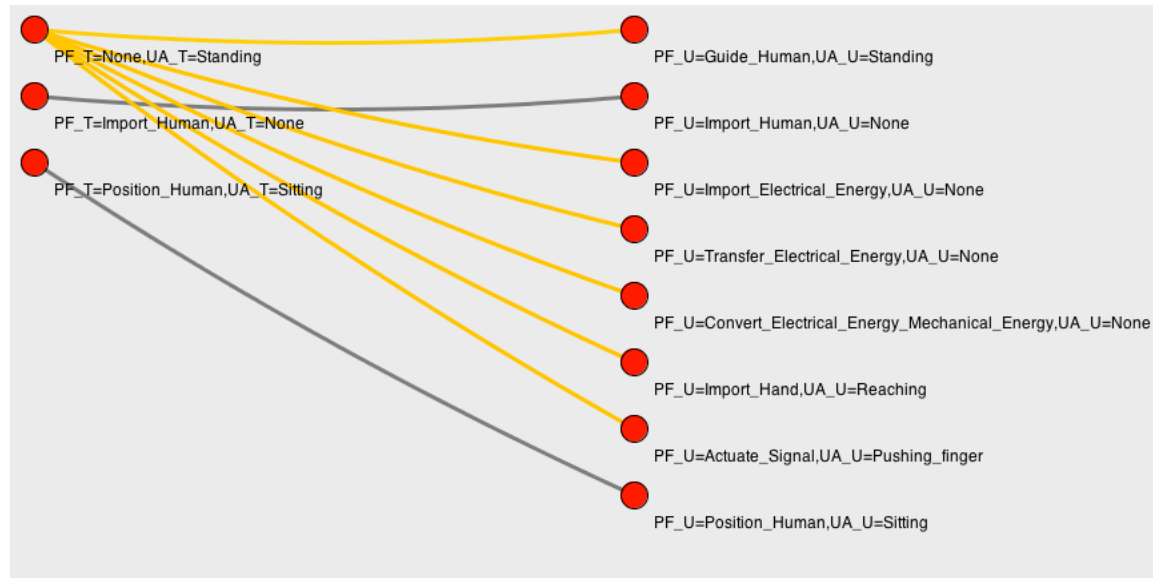
t # 40 One hole punch



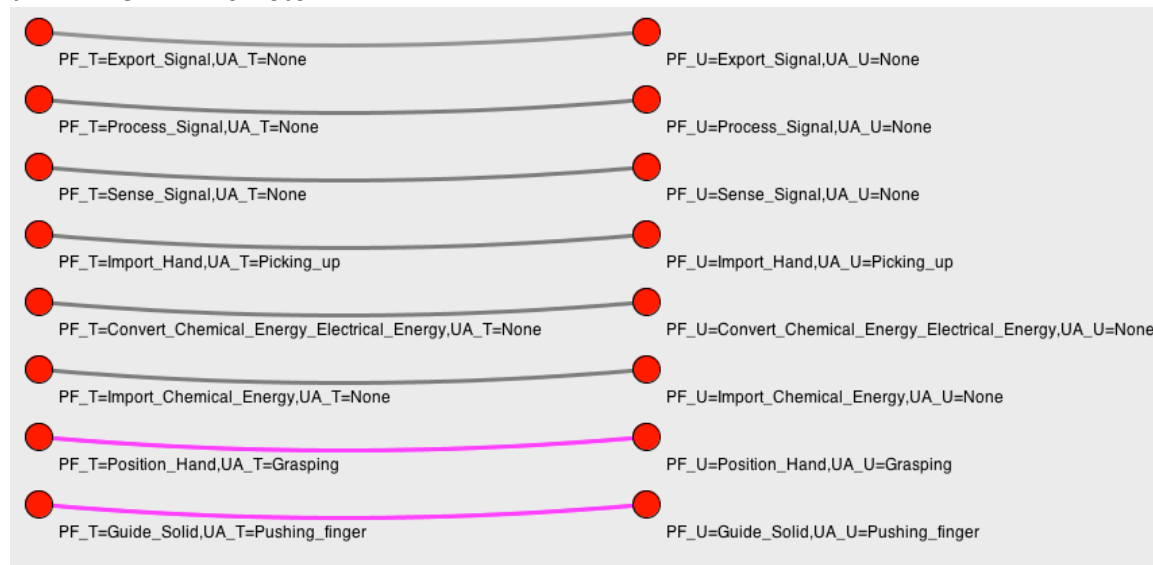
t # 41 Razor Reach



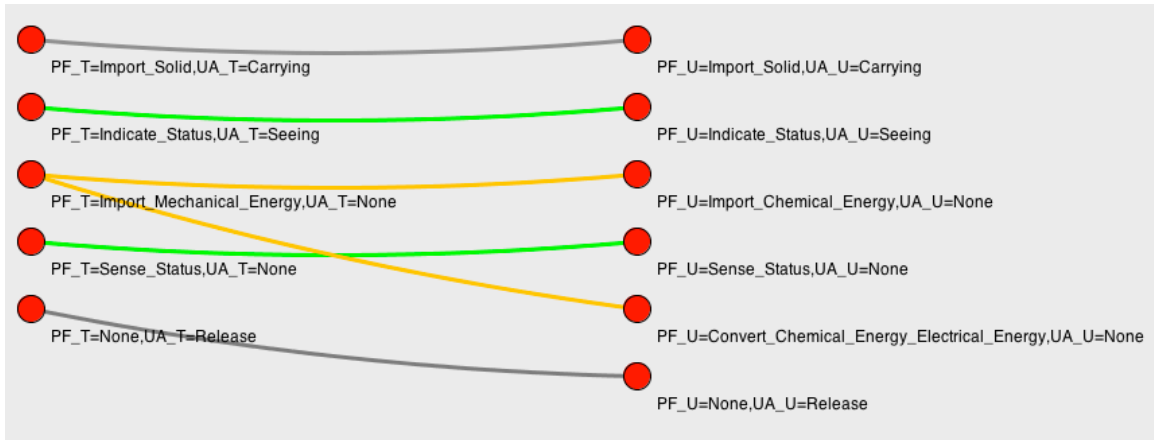
t # 42 Recliner



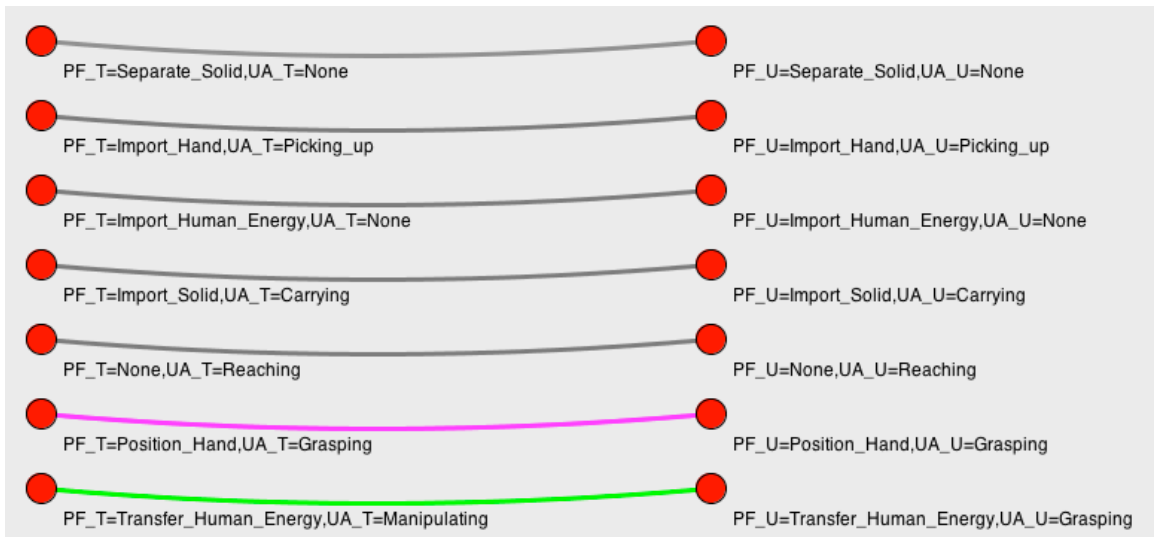
t # 43 Remote



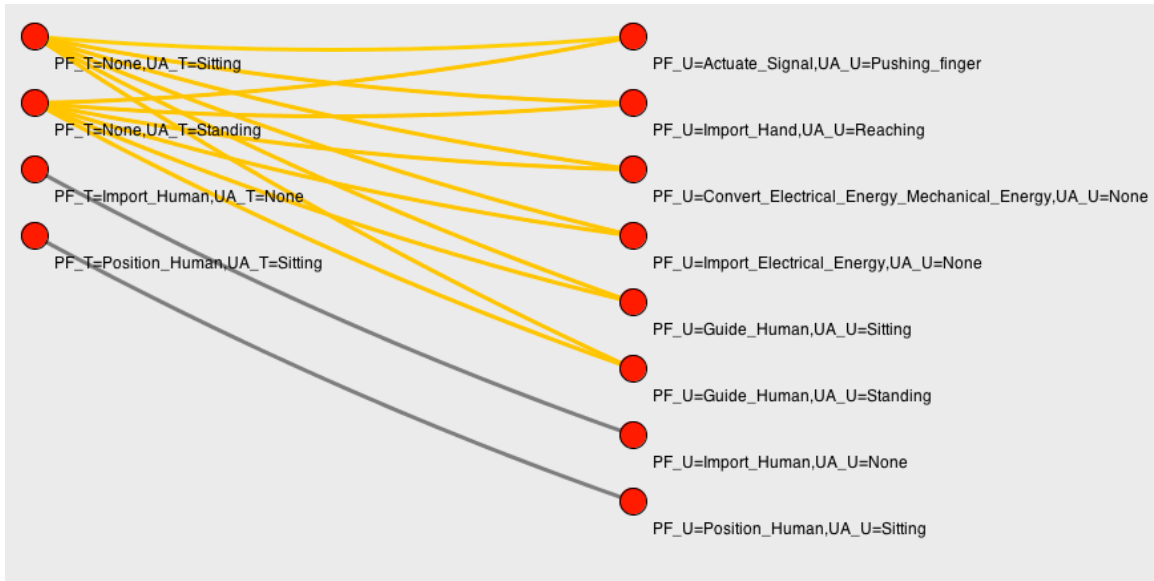
t # 44 Kitchen Scale



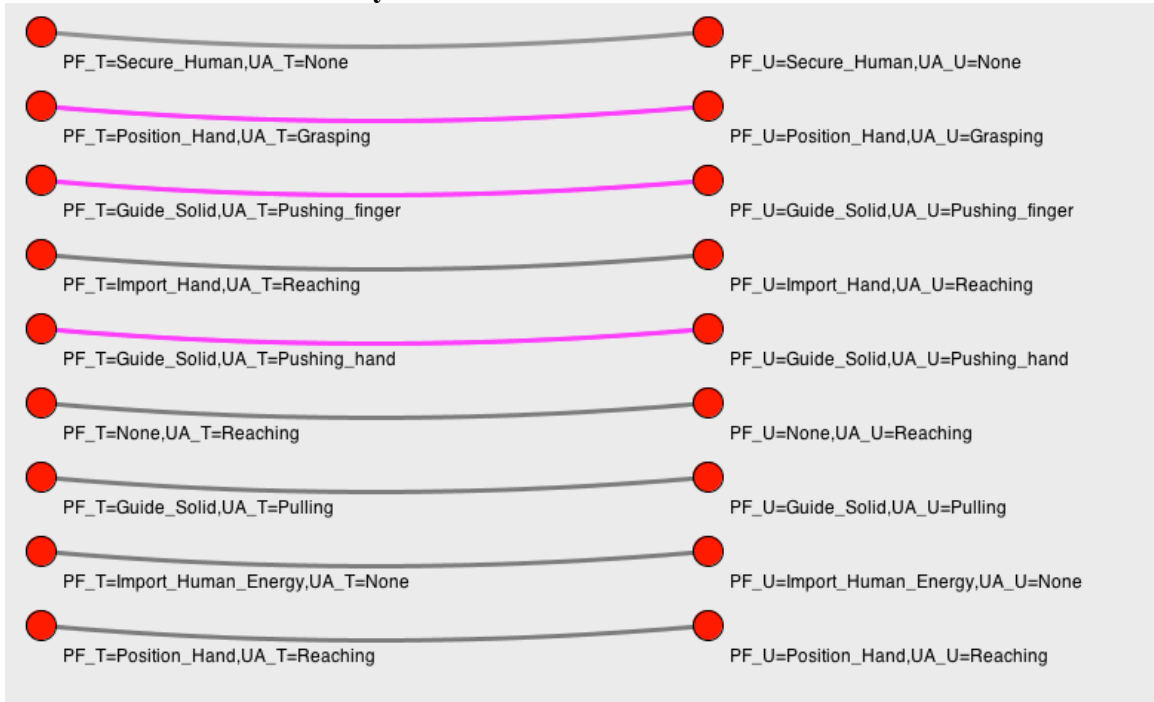
t # 45 Scissors



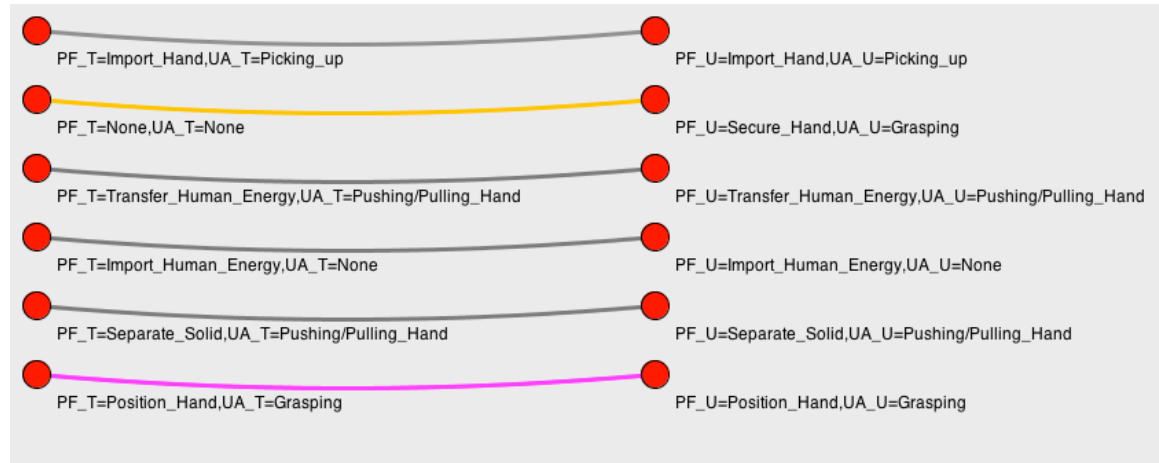
t # 46 Toilet Seat Auto



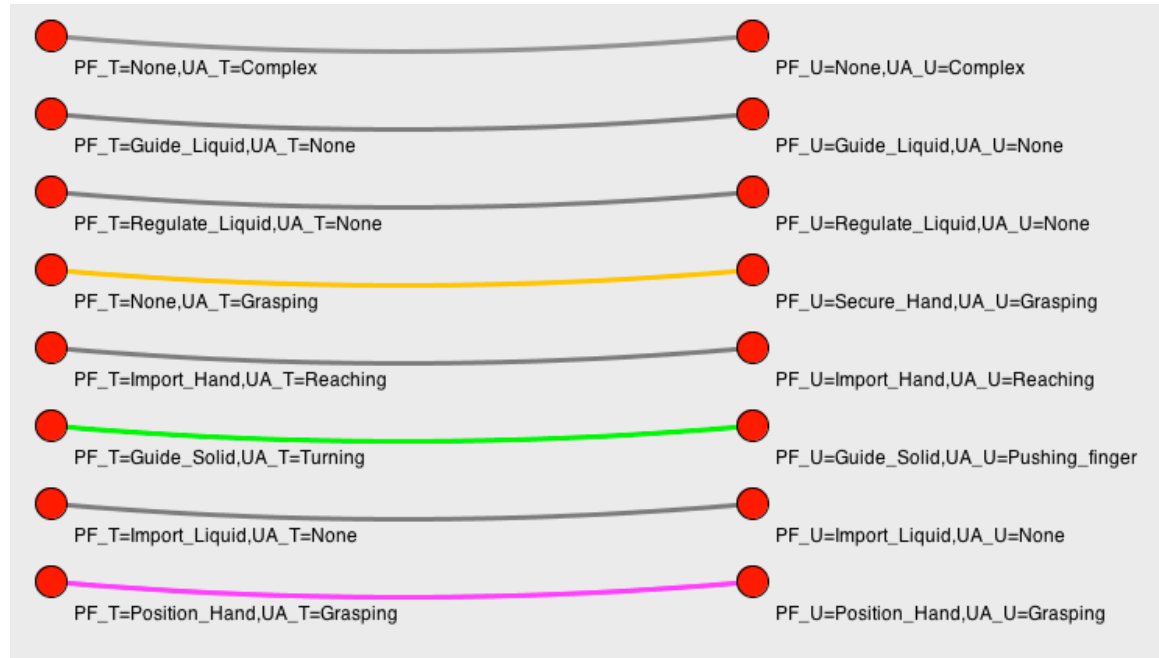
t # 47 Seat belt Easy Reach



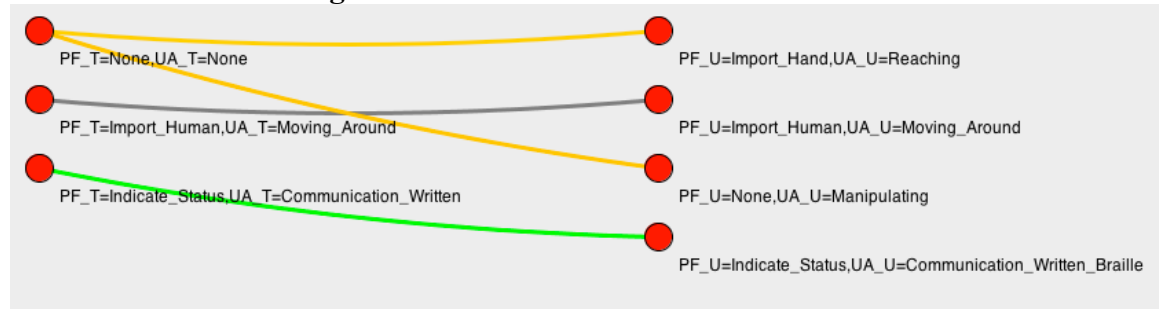
t # 48 Shovel



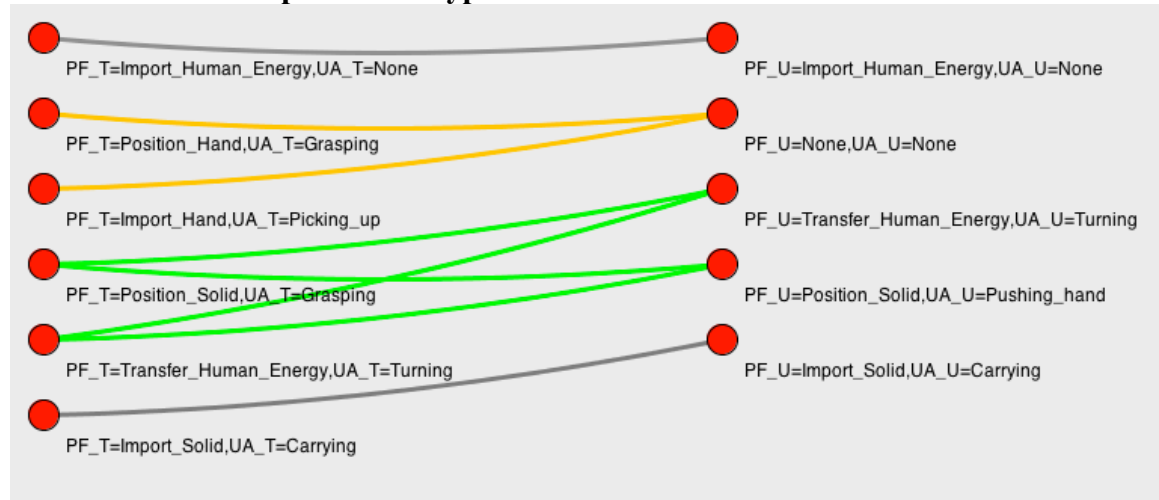
t # 49 Handheld Shower



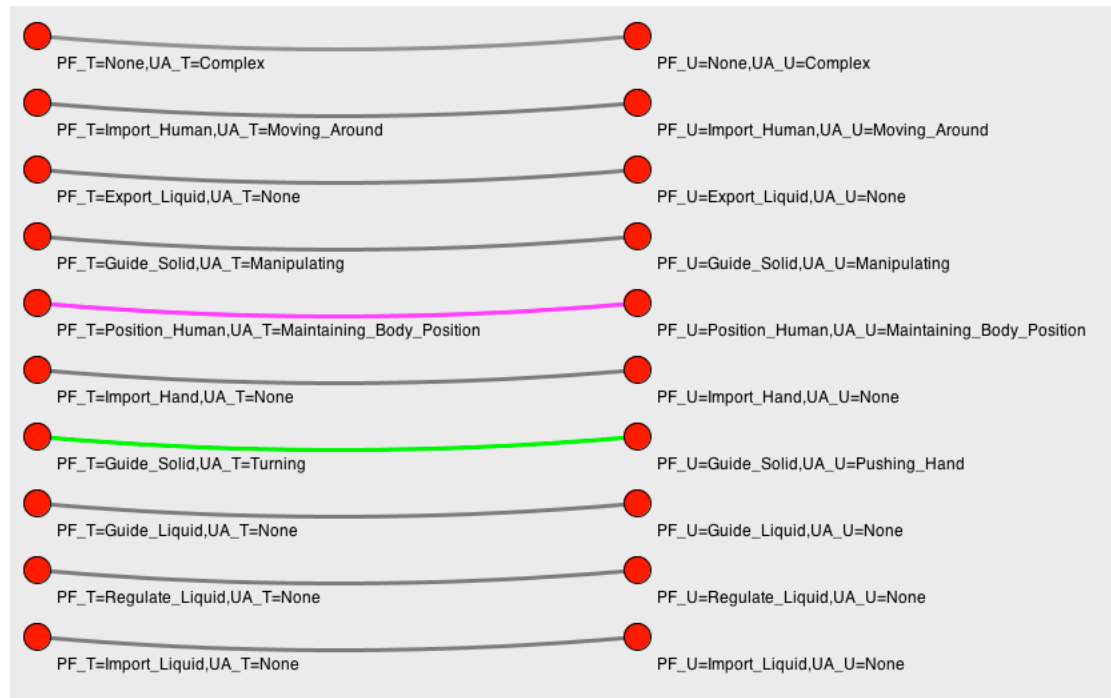
t # 50 Braille Signs



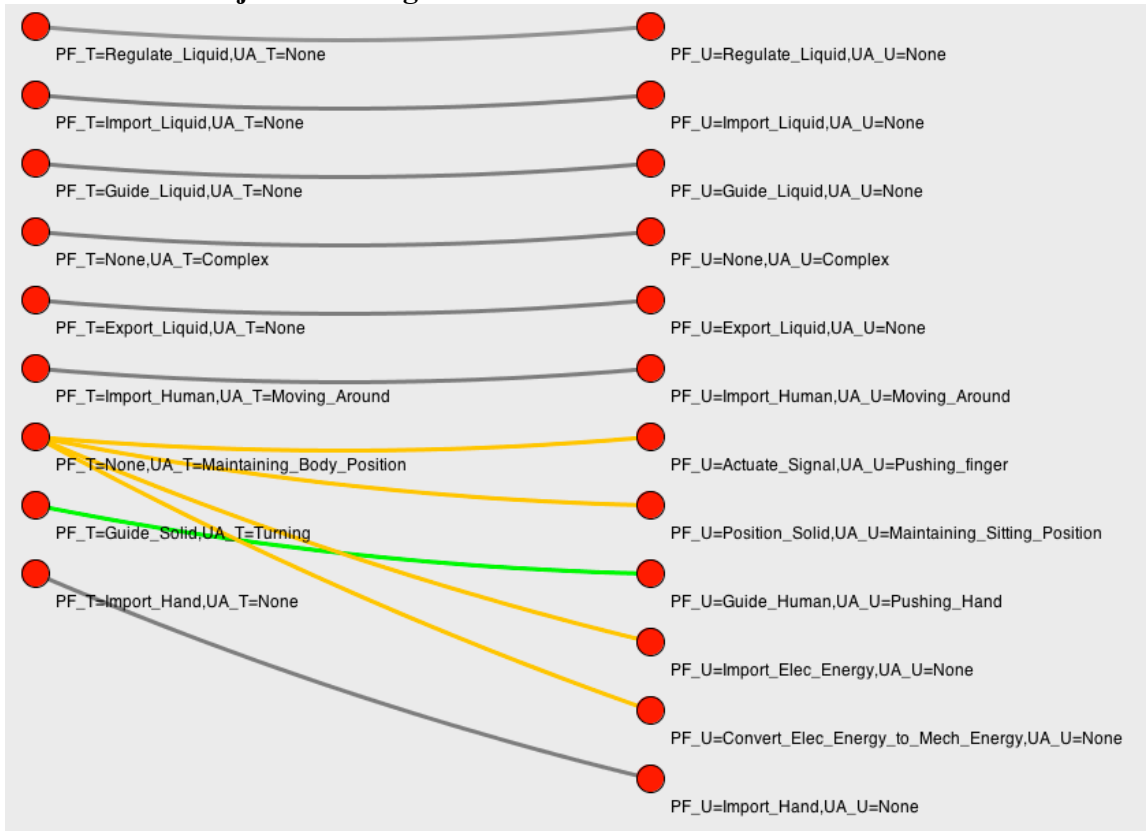
t # 51 Jar Opener Vice type



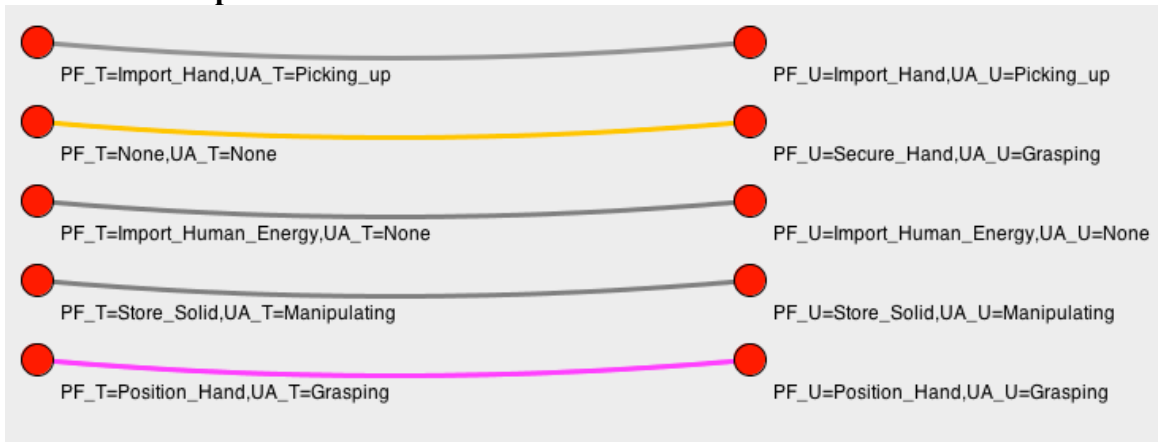
t # 52 Kitchen Sink



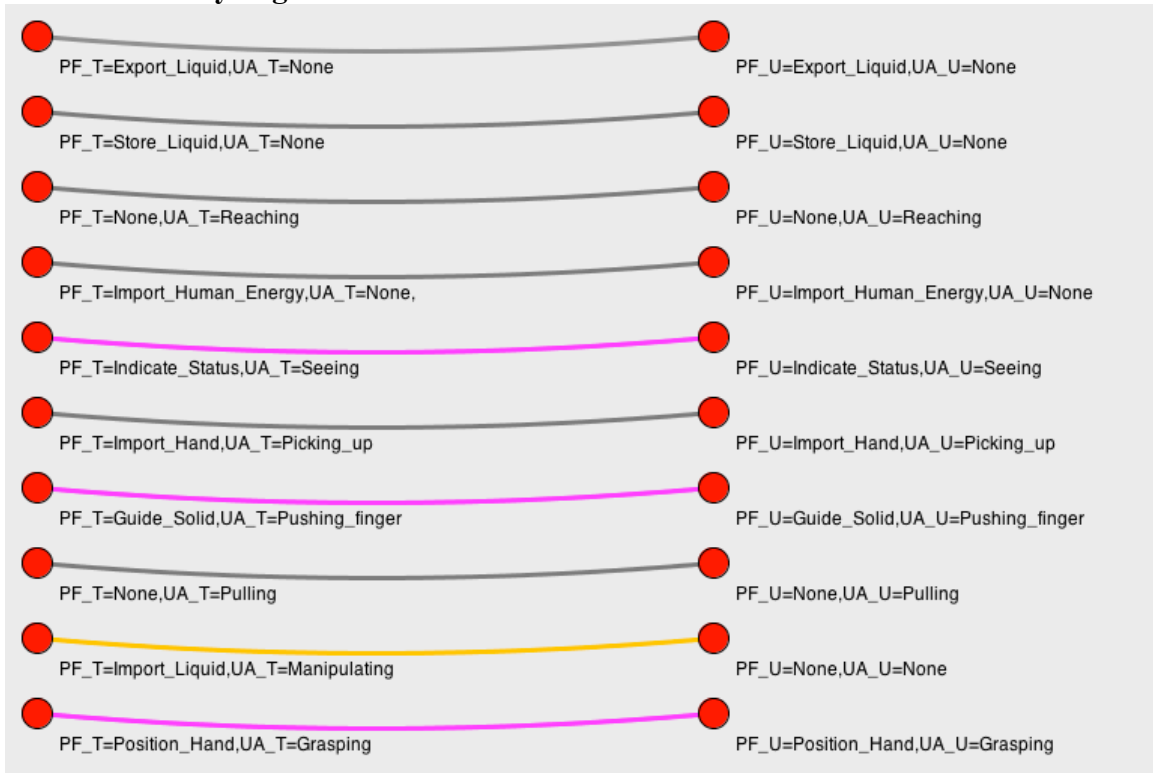
t # 53 Adjustable Height Sink



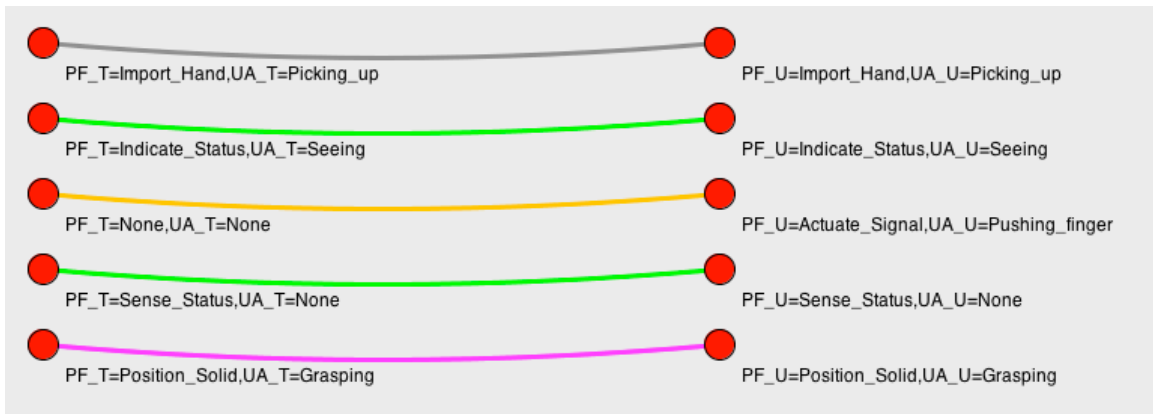
t # 54 Spatula



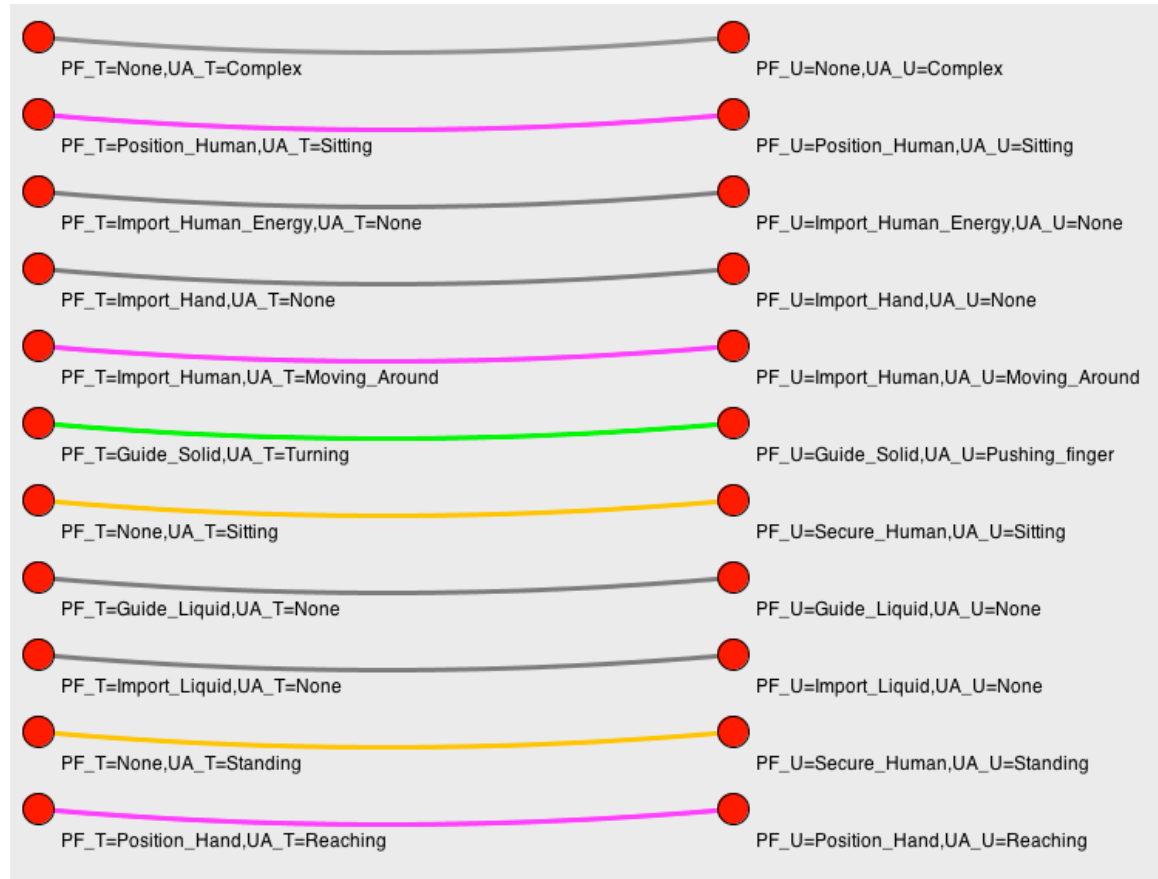
t # 55 Syringe



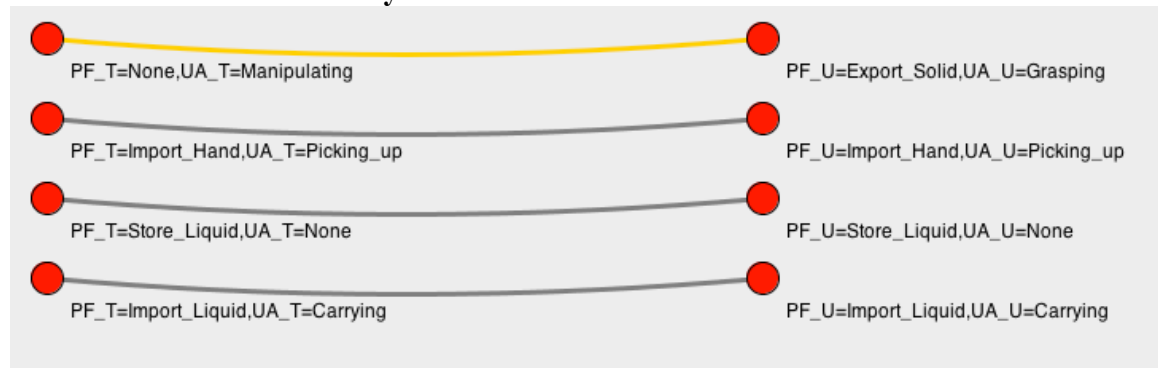
t # 56 Thermometer



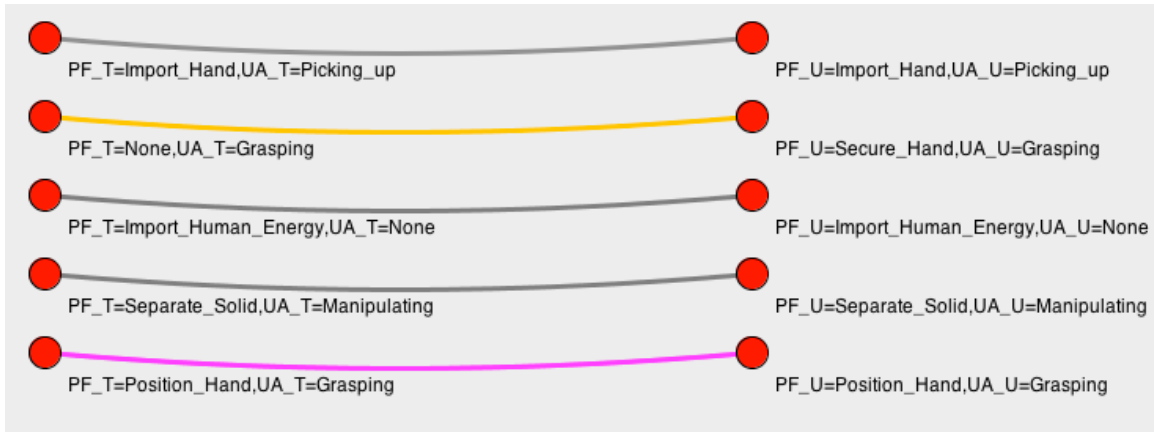
t # 57 Toilet



t # 58 Ice Cube Tray



t # 59 Trowel



t # 60 Bath tub



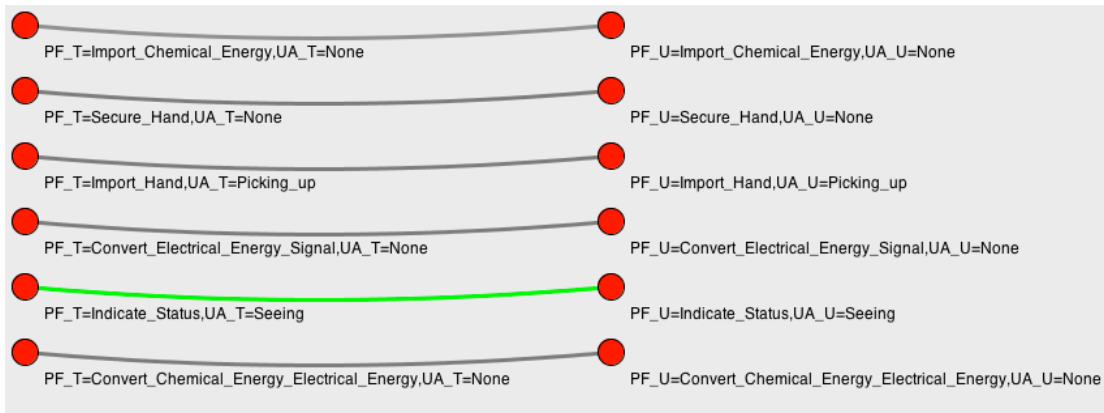
t # 61 Dishwasher



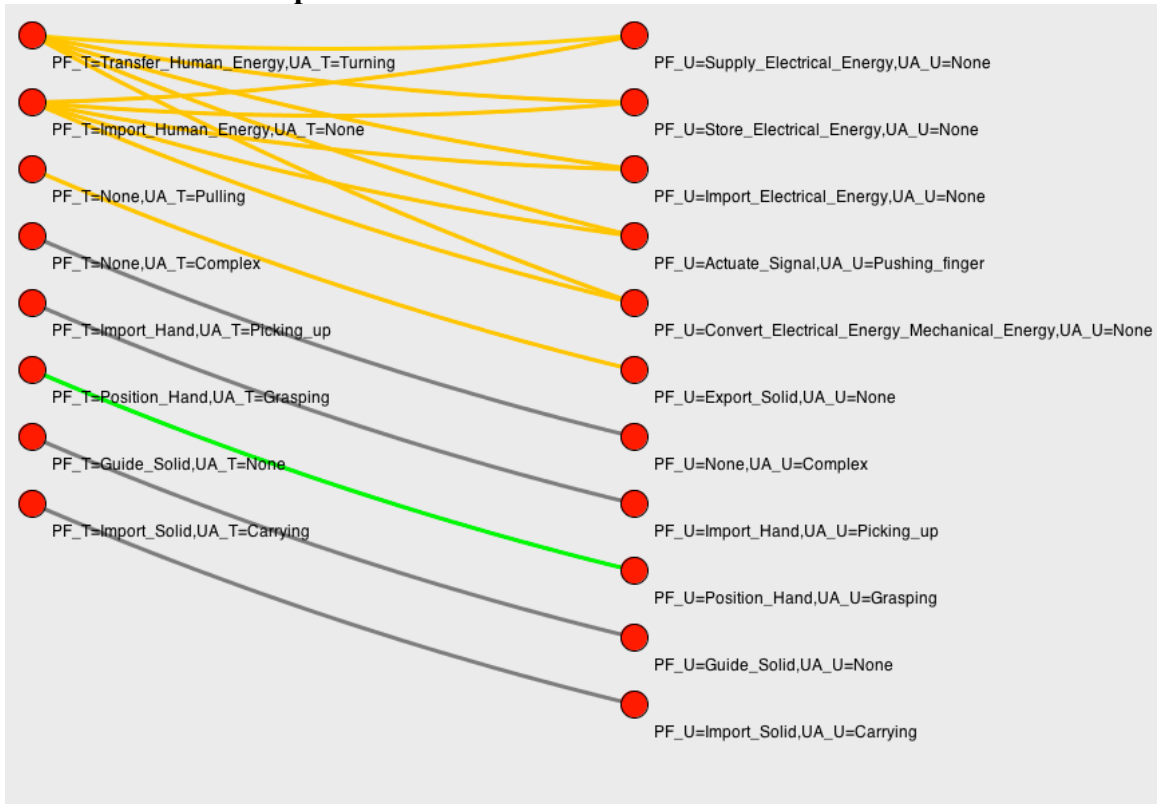
t # 62 Washer



t # 63 Wrist watch



t # 64 Wine opener



t # 65 Car Window

