Heuristic-based approach to customer-driven conceptual design

Chong, Yih Tng

2006

Chong, Y. T. (2006). Heuristic-based approach to customer-driven conceptual design. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/5285>

<https://doi.org/10.32657/10356/5285>

Nanyang Technological University

Downloaded on 26 Apr 2023 17:01:58 SGT

A HEURISTIC-BASED APPROACH TO CUSTOMER-DRIVEN CONCEPTUAL DESIGN

CHONG YIH TNG

**SCHOOL OF MECHANICAL & AEROSPACE ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY**

2006

A Heuristic-based Approach to Customer-driven Conceptual Design

Chong Yih Tng

School of Mechanical and Aerospace Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Master of Engineering

2006

Acknowledgement

Foremost, the author wishes to highlight the strong support he receives from his family.

The author would like to take this opportunity to express his heartfelt gratitude to his supervisor, A/P Chen Chun-Hsien for his invaluable advices and guidance. Despite of his hectic work schedule, he has been patient and encouraging throughout the course of this project.

The research scholarship granted to the author has made his attempt to pursue this higher degree possible. It came from Nanyang Technological University, with the support of Prof Khoo Li Pheng, A/P Chen Chun-Hsien and A/P Samuel Lim.

Special thanks go to Dr Yan Wei for the advices he constantly gave. A/P Hoon Kay Hiang has rendered the author support in handling various issues, including MAE Alumni Association matters. The first-year examiner of this work, A/P Leong Kah Fai, has provided the author with a series of pointers, therefore improving the results of this research. The author is thankful to him. The assistances from Mr Chia, Mr Soh and Mr Koh, the laboratory technicians, are greatly appreciated. Committee members of the MAE Graduate Students' Club, and the club advisor Prof Lye Sun Woh have supported the author in managing both the club activities and his research endeavors. Fellow researchers Mak, Darius, Zhiming and Lo Shuan have engaged the author in sessions of mind intriguing discussions on design topics. The author has certainly benefited from them.

The author would like to express his sincere appreciation to all those who have contributed in one way or another, be it a simple word of encouragement or constructive criticisms out of concern. Thanks to all.

Table of Contents

Acknowledgement	i
Table of Contents	ii
List of Figures	vi
List of Tables	viii
Abstract	ix

	Page Number
Chapter 1 Introduction	1
1.1 Background	1
1.1.1 The Nature of Conceptual Design	3
1.1.2 The Assistances of Information Technology	4
1.2 The Objectives	4
1.3 The Scope	5
1.4 Significances of the Project	5
1.5 Organization of the Thesis	6
Chapter 2 Literature Review	7
2.1 Design Theories	8
2.1.1 A Structure of Design Theories	8
2.1.2 An MTS of Conceptual Design Theories	10
2.1.2.1 Philosophical Views of Conceptual Design	10
2.1.2.2 Knowledge in Conceptual Design	12
2.1.2.2.1 Taxonomy of Conceptual Design Knowledge	12
2.1.2.2.2 Representations of Design Solutions	14
2.1.2.2.3 Multiple Abstraction Levels of Knowledge	15
2.1.2.3 Processes in Conceptual Design	16
2.1.2.3.1 Customer-Driven Design	17

Table of Contents

2.1.2.3.2	Exploration as Conceptual Design Process	19
2.1.2.3.3	Guided Conceptual Exploration	22
2.2	Artificial Intelligence in Design	29
2.2.1	Overview of CACD Research Efforts	30
2.2.2	AI in Conceptual Design	32
2.2.2.1	Knowledge Modeling	33
2.2.2.2	Knowledge Reasoning	34
2.3	Chapter Summary	36
2.3.1	The Literature Reviewed	37
2.3.2	The Objectives	38
2.3.3	The Hypotheses	39
2.3.4	The Tasks Required	39
Chapter 3	A Conceptual Design Knowledge Organization Structure	41
3.1	System Architecture	42
3.2	Conceptual Design Modeling and Representations	44
3.2.1	Knowledge Objects	45
3.2.2	Knowledge Relationships	48
3.2.3	Design Objects Representation	52
3.2.4	Design Space Framework	53
3.3	Discussions on DSF	58
3.3.1	DSF as a Knowledge Organisation Structure	58
3.3.2	Multi-Leveled Conceptual Design Approach	58
3.3.3	Systematic Customer-Driven Design	59
3.3.4	DSF Supporting Creativity	59
3.3.5	DSF in Capturing Design Rationale	60
3.4	Chapter Summary	60
Chapter 4	A Conceptual Design Exploration Approach	62
4.1	Of Design and Heuristics	62
4.2	Overview of Heuristic Algorithms	67
4.2.1	Notations and Assumptions of GBF Algorithm	69

Table of Contents

4.2.2	The General Best-First Algorithm	70
4.3	The Conceptual Design Heuristic Algorithm	71
4.3.1	Notations and Assumptions of CDH Algorithm	73
4.3.2	The Sub-Functions of CDH Algorithm	74
4.3.2.1	The Heuristic Function	74
4.3.2.2	The Evaluation Function f_1	77
4.3.2.3	The Evaluation Function f_2	78
4.3.2.4	The Solved-labeling Function	79
4.3.3	The CDH Algorithm	80
4.3.4	Nodes Retaining for Further Applications	82
4.4	Discussions on CDH Algorithm	83
4.4.1	The Role of CDH Algorithm	83
4.4.2	Managing Design Space	84
4.4.3	Representing Solution Insights	85
4.5	Chapter Summary	87
Chapter 5	Case Studies	88
5.1	Conceptualizing a Disposable Coffee Cup	88
5.2	Conceptualizing a Remotely Operated Vehicle	93
5.3	Conceptualizing a Bicycle Brake System	100
5.4	Chapter Summary	107
Chapter 6	Conclusions	108
6.1	The Research Summary	108
6.1.1	The Tasks Accomplished	108
6.1.2	The Significances of the Accomplished Tasks	110
6.2	The Limitations and Remedies	111
6.2.1	The Heuristic Assumption	111
6.2.2	CDH Algorithm Exiting with Failure	112
6.2.3	Design Coupling	113
6.3	The Envisaged Future Developments	113
6.3.1	Embedding Cost-Performance Parameter	114

Table of Contents

6.3.2	Center-out Solution Forming and Chaining	114
6.3.3	The Product Redesign Function	115
6.3.4	Specialization for Domain Specific Applications	115
6.3.5	Integrating CDS to Conventional CAD System	116
6.3.6	Case-Based Reasoning in CDS	116
6.3.7	Comparison with Existing Models	117
6.3.8	Collaboration with the Industry	117
6.4	Final Comments on the Project	117
References		120
Appendix		131

List of Figures

Figure	Page
Figure 2-1: Steps in Conceptual Exploration	20
Figure 2-2: Pattern of Conceptual Exploration	26
Figure 2-3: Liu et al.'s Proposed Pattern of Conceptual Exploration	27
Figure 2-4: Design Approach Characterized by Cross	27
Figure 2-5: Pugh's Proposed Model of Conceptual Exploration	27
Figure 2-6: Fricke's 3 Types of Concept Development Pattern	28
Figure 3-1: Modeling Phenomenon through Theories	41
Figure 3-2: Architecture of the Conceptual Design System	43
Figure 3-3: Conceptual Design Knowledge Representations	44
Figure 3-4: Relationships between C , P and S	49
Figure 3-5: A Novel Model of Conceptual Design Process	51
Figure 3-6: A Case of Graph G in Space D	55
Figure 4-1: An Instance of a Context	73
Figure 4-2: Flow Chart of the CDH Algorithm	81
Figure 5-1: Graphs G' of Auburn	89
Figure 5-2: Solutions for Product Requirement p_1	89
Figure 5-3: Solutions for Product Requirement p_2	90
Figure 5-4: Chemical Heat Source as the Solution s_6^1	91
Figure 5-5: The Concept of the Attachable Handler	91

List of Figures

Figure 5-6: User-Interface of CDS	94
Figure 5-7: Graphs G' of Marcie	95
Figure 5-8: Graph G' of Stallion	101
Figure 5-9: Rim Type Braking System	103
Figure 5-10: Disc Type Braking System	103

List of Tables

Table	Page
Table 3-1 <i>C</i> and <i>P</i> Object Frame	52
Table 3-2 <i>S</i> Object Frame	53
Table 3-3 <i>U</i> , <i>V</i> and <i>W</i> Object Frames	53
Table 3-4 <i>F</i> , <i>B</i> and <i>A</i> Object Frames	53
Table A-1: Object Class <i>C</i> and <i>P</i> in <i>G</i> '	132
Table A-2: Object Class <i>S</i> in <i>G</i> '	133
Table A-3: Relational Object Class <i>U</i> , <i>V</i> and <i>W</i> in <i>G</i> '	134
Table A-4: A Braking System Conceptualization Case (1 of 7)	135
Table A-4: A Braking System Conceptualization Case (2 of 7)	136
Table A-4: A Braking System Conceptualization Case (3 of 7)	137
Table A-4: A Braking System Conceptualization Case (4 of 7)	138
Table A-4: A Braking System Conceptualization Case (5 of 7)	139
Table A-4: A Braking System Conceptualization Case (6 of 7)	140
Table A-4: A Braking System Conceptualization Case (7 of 7)	141

Abstract

Research has shown that about 80% of a product's lifecycle cost is determined during the conceptual design phase. Despite its importance, the use of computer-aided design during this phase is uncommon. This research seeks to contribute to the theoretical foundation that supports the realization of a computer-aided conceptual design (CACD) system. In-depth reviews are carried out on design theories, artificial intelligence and other related areas of research.

In the inherently large space of design, it is known that explicating all possible concept variants is astronomically costly, if at all possible. A strategy that can effectively assist designers in exploring and ascertaining solutions within this vast space is therefore necessary. This work innovatively adapts a General Best First heuristic algorithm, termed as Conceptual Design Heuristic (CDH) algorithm, to specifically operates on conceptual design problems. The prescribed algorithm is essentially an explication of human's problem solving cognition, based on the principles of heuristic. As a function residing in a CACD system, CDH algorithm builds up design concepts by strategically managing the navigation in the vast space of design, thus semi-optimizing for a *satisficing* solution. Many design methods have 'feed-back loops' drawn to indicate design iterations procedures, without distinct instructions on *what*, *when*, *why* and *how* to proceed. In this work, the mechanisms and triggers for design iterations and backtrackings are modeled mathematically.

CDH algorithm is tailored to operate on a postulated knowledge organization structure, called Design Space Framework (DSF). Based on set and graph theories, DSF

Abstract

articulates and codifies the knowledge dealt with in conceptual design process. The range of information captured in DSF is proposed to exist in a space that can be characterized along three dimensions. Leveraging on a novel axiom, DSF's multiple abstraction levels provides many advantages over the existing models. DSF is capable of representing solution insights and capturing design rationales. Relative to its counterparts, DSF is appropriate to represent concepts of more complicated engineering products. The structure of DSF ensures that relevant customer opinions are constantly sought and assimilated in the entire product conceptualization process. Another distinct benefit of DSF is that it supports the design of innovative products.

Three case studies are presented in this thesis to illustrate the applicability and the logics of the propositions. The limitations of the work done are identified, along with their respective proposed remedies. Finally, several promising tracks of future research are identified, based on the foundation of this project.

– Chapter One –**Introduction**

This introductory chapter provides the project's background, objectives, scope and its significances. In the last section, the organization of this thesis is outlined.

1.1 Background

Product design can be perceived as the process of conceiving the attributes of artifacts. The well being of a product development organization has direct relationships with the products they design. According to Pahl and Beitz [1], the design process has to be systematically executed to increase the success rate of a new venture. This belief has driven the progressions of an important branch of design science over the past few decades, advocating certain prescriptive approach [2] towards design. In this field of study, a range of design methodology has been proposed. The prescribed methodologies that are directed to aid in the front end of the design process, in effect, support the conception of product's abstract attributes. This front end of the prescriptive process is generally known as *conceptual design*.

Data derived from the industry indicates that about 80% of a product's lifecycle cost is determined during the conceptual design phase [3]. Industries have acknowledged that detailed design of the highest standard cannot compensate for a poor design formulated at the conceptual design phase. Due to the importance of this design phase, proponents of design science have paid considerable attention to this front-end process.

As technology advances along with the establishment of design science discipline, parts of product design endeavors are progressively transferred to computers. The strengths of computers are capitalized to support the overall product design activities, including the conceptual design segment.

It has been noted that existing computer-aided design (CAD) systems have generally focused on modeling physical products using data available only at detail design stage, and have largely ignored the supports required during the conceptual design phase [4 - 7]. Such development trend of CAD systems is somewhat ironic, since the conceptual design phase has been widely recognized as vital, because, as aforementioned, it determines about 80% of the product's lifecycle cost. Although the research community has recognized the need to work on this area, and has made significant progresses, much work remains to be done [8]. There is thus a need to research into this area to support the realization of a CACD (computer-aided conceptual design) system for the industry.

In short, the lack of CACD systems in the industry, coupled with the paramount importance of the conceptual design phase motivates this research project. The main objective of this project is therefore to contribute to the theoretical foundation required for the realization of a CACD system. This objective is abstract. However, it effectively serves to direct the scope of the literature review necessary to distinguish the gaps of the relevant knowledge. Having identified these gaps, the exact objectives of this project can then be formalized.

The classes of knowledge relevant to this project are two-tiered. The first tier includes design science and cognitive psychology, which facilitate the understanding of the phenomenon and nature of conceptual design process. The second tier encompasses the disciplines within computer science, in particular, artificial intelligence. This tier of knowledge directs the specification of a CACD system, based on the comprehension of the first-tier knowledge. Accordingly, Section 1.1.1 expresses the nature of conceptual

design phase while Section 1.1.2 outlines the information technologies that are applicable in this design phase. Two precise objectives of this project are henceforth positioned and stated.

1.1.1 The Nature of Conceptual Design

Araujo et al. [9] asserted that conceptual design process is highly dependent on designers' tacit knowledge. Knowledge, being a key competitive advantage these days, [9] is seen as a form of asset to product developers. For them to own this form of asset, there is a need to explicate the tacit knowledge residing within designers. Formalizing conceptual design knowledge would open the doors for computer assistances. Moreover, it would allow effective data integration between conceptual design phase and the downstream design processes [7]. Very often, effective collaborations and team decision-making are absent at this 'fuzzy front-end' [3]. Explicit knowledge and formalized processes are seen as the critical premises in supporting coordination in a cross-functional collaborative design environment [10]. The inherent imprecise and incomplete knowledge conversed in conceptual design process has posed difficulties for designers during knowledge handling and processing [4,5,8,11,12]. Hence, for conceptual design, the representation methods are expected to handle knowledge with such characteristics.

Customer requirement is a set of vital information utilized in the conceptual design phase. It helps enterprises to focus on creating products that customers want, starting right from the conceptualization phase. Engelbrektsson and Soderman [13] have identified the problems of customer requirements missed or discovered untimely leading to costly consequences. Thus it is essential that any conceptual design methodology should effectively assimilate customer voices into the product conceptualization process.

Liu et al. [14] emphasized the importance of 'managing solution space' during conceptual design process. In practical design problems, the solution space is open ended

[15]. Designers generate alternatives of the widest range, so as not to leave out potential concepts. Unfortunately, the time available to do so is limited due to the demand for the shortest development cycles. Designers are often unable to take enough time to consider all alternatives, especially when permutation of ideas demands more time to process. In view of this dilemma, it is worthwhile to investigate into a method that intelligently guides the exploration of conceptual solutions in product design.

1.1.2 The Assistances of Information Technology

Most recently in 2004, Holland et al. [16, 17] created a prototype CACD system, which was claimed to be seamlessly integrated to AutoDesk Inventor, a well-known conventional CAD system. Other than such a prototype CACD system, AI-based knowledge representations and reasoning techniques have been extensively applied on conceptual design problems [5,7,8,11]. A wide range of knowledge representation techniques has been proposed. Among the formalization techniques are mathematical assertions, predicate calculus, conceptual graphs, frames, scripts, objects, semantic networks and production rules. Researchers have used various computational methodologies to support assorted reasoning tasks in the conceptual design process. Among the techniques commonly used are neural network, genetic algorithms, case and rule based reasoning, fuzzy logics, and heuristics. In various research efforts, combinative of techniques are usually applied integrally to model and reason design knowledge [8].

1.2 The Objectives

As discussed above, the inherently fuzzy knowledge involved within the conceptual design phase has to be formalized as the premise of a CACD system. A CACD system ought to advocate customer-oriented design philosophy. Finger and Dixon [18] suggested that the subject of innovation or creativity often arises in connection with conceptual

design. A CACD system that supports innovative design should thus be researched and worked on. Due to the large design space, a method that guides the designers within this space will be contributive as a function of a CACD system. Based on this set of desirable characteristics of a CACD system, two objectives of this project are as follows.

1. To define a customer-oriented knowledge organization structure, as the data model of a CACD system.
2. To prescribe an intelligent methodology that guides designers during product conceptualization processes, based on the defined structure required by the first objective.

1.3 The Scope

The scope of the project is as follows:

1. To establish the architecture of a proposed CACD system.
2. To formulate a knowledge organization structure as required by the first objective.
3. To review and study the applicability of AI techniques on product conceptual design problems.
4. To establish a methodology that guides designers during conceptual design phase, as required by the second objective.
5. To validate the proposed CACD system using case studies.
6. To review the proposed CACD system and to recommend future work.

1.4 Significances of the Project

The fulfillment of the above stated objectives will potentially be contributive towards the realization of a CACD system in the following three ways.

- The attainment of the first objective will result in the establishment of a conceptual design representation scheme, as the premise to a CACD system.

- The product of the second objective will assist designers in navigating the design space strategically in a CACD environment.
- In accomplishing the first and second objectives, a basic system framework will be established. Based on this framework, it is expected that a range of promising future work can be initiated.

Overall, the novel conceptual design approach proposed in this work takes a contributive step towards the realization of a commercial CACD system.

1.5 Organization of the Thesis

This thesis is organized according to the following order.

- Chapter 1 introduces the background, objectives, scope and significances of this research.
- Chapter 2 reviews a range of conceptual design theories, as well as the applications of various AI techniques. The directional orientation and grounding to this project is established by the end of this chapter.
- Chapter 3 defines the proposed conceptual design knowledge organization structure.
- Chapter 4 prescribes the conceptual design exploration approach, based on the propositions established in Chapter 3.
- Chapter 5 presents three case studies based on the propositions made in Chapters 3 and 4.
- Chapter 6 concludes the thesis with research summary, discussions, and followed by further work proposals.

– Chapter Two –**Literature Review**

In the effort to develop an approach to handle conceptual design problems, this chapter reviews the state-of-the-art computer-aided conceptual design (CACD) systems and the multi-faceted knowledge underlying it, including design theories and artificial intelligence (AI).

CAD systems ought to be established on certain underlying design theories. Suh [19] pointed out that *many educators and engineers have often tried to use computer to deal with design subjects, rather than to comprehend the conceptual aspect of design process*. Thus it is prudent to approach this project first from the viewpoint of design theories. Specifically, design theories that concern the front-end of design process, i.e. conceptual design, are reviewed and analyzed in Section 2.1.

In Section 2.2, the applications of information technologies and computation techniques on conceptual design theories are discussed. In addition, several efforts made to realize the CACD systems are reviewed as well.

Several research interests emerged from the reviews and discussions are presented in Sections 2.1 and 2.2. Based on these identified areas of interest, Section 2.3 presents a sequence of tasks required to fulfill the stated objectives of this research.

2.1 Design Theories

Artificial intelligence and other computer-based techniques have been applied to conceptual design tasks, necessarily based on the understanding of the underlying design theories. As such, this section is committed to review a range of design theories.

These design theories exist in various levels of abstraction with imbalanced distribution. Section 2.1.1 presents a ‘meta-theoretical’ structure to facilitate the reviews of the bodies of design theories in separate levels. The inclusion of this proposed theoretical structure enables clear presentation on the discourse of design theories, which has been inherently complex. Based on the format of the structure, Section 2.1.2 systematically discusses the multitude of propositions made by proponents of design science.

2.1.1 A Structure of Design Theories

Love [20] asserted that theories about design have been developed by researchers using a wide variety of perspectives, from a large number of disciplinary and sub-disciplinary cultures, for example, engineering, philosophy, psychology, computer science and so on [21]. This progression has unfortunately resulted in a large collection of individual theoretical, analytical, conceptual and terminological elements that in many cases are contradictory, ambiguous or limited in scope [20]. Researchers have identified the need to structure existing theories, which will allow the theories to be analyzed meaningfully. Such step could *simplify the paradigm of design research*, which is extremely important for this field [22].

Love [23] proposed a higher-level perspective on design theories, so-called meta-theory. A meta-theoretical structure (MTS) was postulated to consist of ten levels of abstraction, namely: (1) Direct Perceptions of Realities, (2) Description of Objects, (3)

Behavior of Elements, (4) Mechanism of Choice, (5) Design Methods, (6) Design Process Structure, (7) Theories about the Internal Processes of Designers and Collaboration, (8) General Design Theories, (9) Epistemology of Design Theory, and (10) Ontology of Design. Human senses (at the lowest level) together with human values (at the highest level) encompass the abstraction spectrums of design theories. This structure can represent design theories in abstraction levels.

Popper [24] suggested a relatively coarser taxonomy to address the problem of theoretical confusion: (1) World 1 – Physical and material objects, (2) World 2 – The subjective world containing minds and their contents, and (3) World 3 – The objective world of theories, knowledge and problems.

Both Love and Popper have proposed categories to decompose design theories into levels of abstraction. Love's rationale of proposing an MTS is to reduce confusion, semantic proliferation and lack of coherency in design theories. Accordingly, an MTS is useful to explicate the work of previous theorists, without the need of using new language and new concepts. Overall, such meta-theories can be used to analyze, relate, position and validate concepts and theories that are in and from different theoretical stances. In this vein, the purpose of introducing an MTS into this thesis is to facilitate a systematic approach in review the range of design theories. Without such review structure, cross-referenced discussions between theories would be difficult, if at all possible.

A three-level structure can be generalized from Love's [23] MTS for application in this work. The generalizations are as below:

- Levels 1-3 relate to *knowledge* of design.
- Levels 4-7 relate to *processes* of design.
- Levels 8-10 relate to *philosophies* of design.

The above taxonomy allows the range of literature to be presented and discussed in logical order and divisions. Instead of contrasting theories in their wholes, contrasting

them on separated abstraction levels should achieve more meaningful discussions, and perhaps arriving at some significant conclusions.

2.1.2 An MTS of Conceptual Design Theories

Generally, each design theory has established their respective set of *design knowledge and design processes*. Zeng and Gu [21] termed *knowledge* of design as *language*, and *processes* as *law*. They viewed design science as having these two fundamental parts. Brunetti and Golob [7] stated that the procedure of design is supported by design methodologies via providing specific *design methods* and *design knowledge*. As such, *design knowledge* and *process* can be seen as two variables in the frameworks of design theories. Various researchers have given their philosophical views on CACD systems or conceptual design theories. Such values held by various theorists are of interest to this research. According to the MTS of design theory highlighted in the previous section, the constituents of conceptual design theories will be reviewed in the following order.

- Section 2.1.2.1: Philosophies of conceptual design.
- Section 2.1.2.2: Knowledge in conceptual design.
- Section 2.1.2.3: Processes of conceptual design.

2.1.2.1 Philosophical Views of Conceptual Design

Pahl and Beitz [1] asserted that design process is a variant of general problem solving. Xiong [25] perceived that conceptual design is to utilize the judging ability of human beings to solve basic problems in design and manufacture. Tay and Gu [26] indicated that information structure in conceptual design support system *should bear resemblance to user's thought structure*. Stempfle and Badle-Schaub [27] declared that the thinking process of designers is one of the most important issues in design research. They went on

to survey the cognitive issues empirically and theoretically. Meniru et al. [10] performed observational studies on eight designers who were conceptualizing a given product. Based on the findings on designers' acts and thinking processes (cognition), they drew up the specifications for a computer-aided conceptual building design system. Al-Salka et al. [2] identified 'abstraction', 'divergent and convergent thinking' as common principles underlying a wide range of conceptual design methodologies. These principles are related to human thought processes during conceptual design of products. Rosenman and Gero [28] defined design as a purposeful human activity in which cognitive processes are used to transform human needs and intent into an embodied object. Gero [29] asserted that design systems must be based on human design processes. Finger and Dixon [18] summed that the knowledge of how human design is required to create better CAD tools. The above perspectives of various theorists suggest the scrutiny of human problem solving cognition in prescribing theories for a computer-aided conceptual design (CACD) system.

In the studies of cognitive psychology, Newell [30] specified a system, so-called Soar (State, Operator And Result), which is the vehicle to verify their theories of human problem solving. Their theories are not domain-specific, prescribed to handle general problems. Sullivan [31] pointed out that in any conceptual design method or system, the human designers should have the freedom to approach the process in any way desired. Holland et al. [16,17] echoed this philosophy in the CACD system they have prototyped. They have asserted that it is important that technologies used in CACD systems should simply maintain the consistency of the designers' decisions, and not to forbid any form of action. This is a critical characteristic that CAD technologies for supporting conceptual design must exhibit [16, 17].

In developing Schemebuilder, a software tool for system design and evaluation, Bracewell and Sharpe's [32] tried to move away from conventional expert systems

approach of automated design to one embracing a more cooperative relationship between man and machine. As Gero [29] asserted, design systems must be based on human design processes. In general, to specify the assistances machine provides to man, the working characteristics of man have to be in prior grasped. In the case of this research, the cognitive characteristics of the designers during product conceptualization process have to be well understood before a CACD tool can be specified to assist them.

The aforementioned philosophies of conceptual design are vital, and would be referred to throughout this thesis. Among them, cognitive-orientation is highly regarded as one of the most important philosophical criteria in establishing design theories and specifying CACD systems [10,16,17,26,31-33]. On this note, it should be mentioned that certain propositions made in Chapters 3 and 4 are based on assertions from the field of cognitive psychology.

2.1.2.2 Knowledge in Conceptual Design

Based on the understanding that design knowledge is one of the fundamental elements defined in design theories [7,10], this section reviews the important viewpoints and perceptions on *design knowledge* pertaining to conceptual design phase.

Section 2.1.2.2.1 reviews the taxonomies and terminologies design theorists have used to represent information conversed in conceptual design endeavors. In Section 2.1.2.2.2, a set of information that can represent design solution is brought up and discussed. Finally, the prevalence of expressing design knowledge in multiple abstraction levels is highlighted in Section 2.1.2.2.3.

2.1.2.2.1 Taxonomy of Conceptual Design Knowledge

A range of terminologies has been used by design theorists to capture the information involved in the processes of product conceptualization.

In Ulrich and Eppinger's [34] prescription, design begins with identifying *customer needs*. Hauser and Clausing's [35] Quality Function Deployment (QFD), and Suh's [19] axiomatic design are also among the methodologies that treat customer voices as the seeds of design. Terminologies used for customer needs include *customer attributes* (CA) [19] as well as *customer requirements* (CR).

Common in several design methodologies, *product requirements* (PRs) are generated based on the elicited CRs. The process of establishing PRs has been commonly referred to as *problem definition* under various theories. Suh [19] recommended the definition of *functional requirements* (FR) upon eliciting and organizing the customer's needs. Sullivan [31] and Holland et al. [16, 17] defined design problems by declaring a set of *design principles*. Design principle, in the context of *function-means map*, is a set of functions that should be satisfied by the design solutions. Medland and Mullineux [36] defined design problems by listing a set of *rules* defined with reference to the product requirements. In QFD, CA are mapped to *engineering characteristics* (ECs) [35]. FR (in Suh's [19] notation), EC of QFD chart [35], *specifications* in Ulrich and Eppinger's literature [34], and *design principles* of function-means maps [16, 17, 31] are all defined based on customer needs and specified prior to solution exploration.

Having defined the design problems, Sullivan [31] and Holland et al. [16, 17] proposed the process of *solution* exploration, with the design principles as the basis. In Suh's [19] method, physical embodiment solution is generated, denoted as *design parameters* (DPs). Bracewell and Sharpe [32] suggested design solutions to be represented in terms of *schemes*. In representing solution-object, sub-objects, i.e., *function*, *behaviour* and *structure*, have been proposed and used by various researchers. These sub-objects are examined in the next section.

Customer requirement, product requirement and solution are three categories of the information acquired, created and utilized in conceptual design processes. These can

be regarded as three classes of design information, which ought to be represented in a CACD system. This issue will be addressed in the proposition made in Chapter 3.

2.1.2.2.2 Representations of Design Solutions

Ulrich and Eppinger [34] defined a conceptual solution as descriptions of forms, functions, and features of a product and are usually accompanied by a set of specifications. Zeng and Gu [21] pointed out that the proposed design language or objects are often not sufficient to support design processes, and hence proposed mathematical representations for the design objects.

Hsu and Woon [8] viewed the reasoning processes in conceptual design stage as the processes of mapping among the three product definitions, namely function, behavior and structure. They deemed mechanical products could be expressed in terms of its function, behaviour and structure. Qian and Gero [37] suggested a model, so-called Function-Behaviour-Structure (FBS), for design knowledge representation. On the other hand, Function-Behaviour-State model was proposed by Tomiyama et al. [38]. Deng et al. [39] described the importance of considering *environment* within the FBS paradigm. As the result, a Function-Environment-Behaviour-Structure (FEBS) was proposed. Relationships among decomposed functions, termed context relations [16,31], are explicitly documented in function-means map. Deng et al. [39] summed that it is now a consensus that design information should include physical structure, required function and implementing behaviours. It can be seen from the range of work that the FBS framework has been widely recognized.

2.1.2.2.3 Multiple Abstraction Levels of Knowledge

Everything we do in design has a hierarchical nature to it; proficient use of hierarchy is a prerequisite for design or organizational success – Suh [19].

Design methodologies have adopted hierarchy as the frameworks to organize both *knowledge* and structured *processes* [1, 5, 14, 19, 26, 31, 32, 40]. It is the hierarchical nature of design information that some theories assumed to organize design objects of all abstract levels, making design systematic and manageable.

Gershenson and Stauffer [41] stated, “*It is common knowledge that design information exists at various levels of abstraction all along the design process*”. They have proposed customer requirement’s taxonomy in three levels of abstraction, providing depths of description. Zou et al. [5] used Place/Transition (P/T) net to represent the hierarchical and multi-abstraction levels of conceptual design knowledge. Their proposed structure represents product concepts from abstract ideas to detailed descriptions. Based on Suh’s [19] multi-level abstractions of functional requirements (FRs), Tseng and Jiao [40] proposed a FR recognition method for use in developing designs based on existing designs.

Conceptual design process commences with high-level descriptions of requirements and proceeds with a high-level description of solution [1]. Bracewell and Sharpe [32] suggested a single functional requirement at the highest level of the function-means tree. Upon decomposition, the complete solution is represented in multiple abstraction levels. This philosophy of hierarchical design process is also seen in Suh’s [19] design methodology. In Suh’s framework [19], the design process is prescribed as mapping to and fro between two domains (FR and DP) in a zigzagging manner. This mapping procedure can be seen as a design exploratory process that progressively increases the details of the design [19]. Tay and Gu [26] proposed a system that represents the function-form relationships, and the recording of this mapping process

between abstraction levels of FRs and DPs. In QFD [35], the engineering characteristics (ECs) do not exist in levels of abstraction. However, the cascading Houses of Quality (HoQ) is perceptible as being levels of design knowledge abstractions. According to Hsu and Woon [8], the reasoning processes in conceptual design stage may involve transformations between planes of abstractions, arriving at sets of candidate solutions.

In Ottosson's [42] view, conceptual design should gradually proceed from greater roughness in dimensions and shape, towards smaller tolerances and well-defined surfaces. During the whole design process, the functions, features and other information are gradually elaborated from abstract sketches to detailed descriptions [5]. This process increases efficiency of product conceptualization, by forcing the engineers to build products from the totality of the concept down to the details [42]. In using conventional CAD, designers build up their new designs from parts into totality. For this reason, conventional CAD systems are better suited for re-engineering products, rather than new product developments [42].

For the conceptualization of new or re-designed products, the key is to think from the highest abstraction level and develop details from vague ideas. From this perspective, one of the specifications for any CACD system would be to require abstraction levels to represent knowledge, as well as to facilitate the processes of solution concept detailing. This will be addressed in the propositions made in Chapter 3.

2.1.2.3 Processes in Conceptual Design

Design methodology is the kernel of a design system. It drives and directs the design processes from concept generation to geometry creation - Wang et al. [4]

The prescriptive approach to design is the earliest and probably the most important area of design science [2]. 'Design methodology', 'design method' and 'systematic design' are all names, which refer to the prescriptive approach. Zeng and Gu

[21] termed *design processes* as *law*, and deemed the definition of them as the foundation of CAD system development. Brunetti and Golob [7] stated that the process of design is supported by design methodologies, by providing specific design *methods* and design *knowledge*. While design knowledge has been reviewed in the last section, this section focuses on the *methods of processing* them.

This section reviews various researchers' viewpoints on conceptual design processes. They are organized into three sub-sections. Firstly, in Section 2.1.2.3.1 the connotations of customer needs in various design methodologies are reviewed. It reflects how design methodologies handle customer requirements to impact on the designs. Section 2.1.2.3.2 illustrates some design methods that consider solution *exploration* process as the representative process of conceptual design. Lastly, Section 2.1.2.3.3 reviews how the exploration process can be strategically guided to achieve results more efficiently.

2.1.2.3.1 Customer-Driven Design

Engelbrektsson and Soderman [13] summed that previous studies have recognized the identification and implementation of customer requirements as a significant issue for successful product development. Situations where customer requirements are neglected are critical and carry negative consequences.

The first step in conceptual design prescribed by Ulrich and Eppinger [34] is to identify customer needs. In Suh's [19] method, functional requirement (FR) is based on customer attributes elicited from the end-users. QFD [35] has a procedure of generating engineering characteristics, parallel to the generation of product specifications in Ulrich and Eppinger's framework. Khoo et al. [43] identified the correlations between design specifications and product concepts in terms of individual customer's needs with the application of analytic hierarchy process (AHP) technique. The foundation of the HoQ is

the belief that products should be designed or engineered to reflect customer's desires and tastes. QFD [35] and axiomatic design [19] are among the methods that use customer voices to begin design process. Such approach, frequently known as customer-driven, market-driven [12], end-user driven, or customer-oriented design [43] process, helps enterprises to focus on creating products required by the customers. This method has been known to improve enterprise's market position [12].

End-users' voices should not only be elicited in the front end of the conceptual design process, but frequently at various junctions along the process. Based on today's spiral and stage-gate processes of design, inputs from customers should be iteratively elicited during the product development process. According to Dahan and Hauser [44], customers are surveyed during: (1) rapid evaluation of ideas early in the process, (2) the identification of important "delighter" features as the product concept is refined, (3) detailed measures of the importance of customer needs as the product is engineered, and (4) accurate evaluation of prototypes as the product nears pretest and test marketing.

Chen et al. [45] recognized the need to have a strategy to solicit customer requirements and henceforth proposed the application of laddering technique to aid the elicitation and representation of customer voices in a hierarchical structure. In the food industry, product development researchers Naes and Nyvold [46] prescribed a systematic concept exploration method that can handle a wide range of possible alternatives, supported by customer's voices during prototype testing sessions. The method takes care of the broadness of alternatives, but does not facilitate concept detailing or concept depth increments. The concept exploration works in a single level of abstraction in that case.

Ideally, end-users should be regarded as part of the design team, along side of designers, marketers, engineers, etc. The contributions of customers and designers should be alternating in the design process, if simultaneously is not practical. In this manner, the

opinions of customers can constantly affect the design of the products at all levels of abstractions.

At various levels of product concept detailness, customer voices are required to guide and support decisions. For instance, when concepts of a cellular phone design are abstract, customers may be asked on how a cellular phone can better satisfy them in general. Their answers, for instances, are lightweight, visually attractive, and has camera function. After the team has adopted the built-in camera feature during concepts exploration, finer customer requirements regarding the built-in camera may then be elicited at this point. In retrospective, it would have been inefficient to consult customers on their detailed wishes of the camera feature, prior to the adoption of this concept earlier during design. This type of inefficiency is in fact due to the unsystematic and non-coordinated elicitation of customer voices.

Engelbrektsson and Soderman [13] identified the problem of customer requirements missed or discovered untimely, leading to costly consequences. They studied this issue in terms of techniques used to elicit customer voices (such as questionnaires and clinics), and concept representation (such as sketches or mock-ups). Apart from considering the methods of elicitation, how the elicited customer requirements get internalized into the design process is not less critical. A design methodology that systematically assimilates customer requirements along the process of conceptual development may be useful to implement and streamline the process, gaining positive effects of customer-orientation. This will be address in the proposition made in Chapter 3.

2.1.2.3.2 Exploration as Conceptual Design Process

Conceptual design is an exploratory process [47], typically containing two types of phases, i.e. divergent and convergent [14, 48] (See Figure 2-1). The corresponding processes are *generation* and *selection* of concepts, respectively [49]. In the divergent

phase, a range of concepts is generated, while in the convergent phase, selections are made. Liu et al. [14] identified two conflicting goals in conceptual design process. On the one hand, designers generate alternatives of the widest range, so as not to leave out potential concepts. On the other hand, they need to keep the number of alternatives small, so as to make evaluating all permutated solutions possible.

Generation and selection are two processes involved in what some researchers termed as an exploration process [14]. Sim and Duffy [15] have classified a generic set of design activities. Among the classified activities, one is the *exploration* process.

Figure 2-1 Steps in Conceptual Exploration [14]

During design, designers need to keep their options open as the design emerges, so as not to crystallize the design too soon [15]. Intuitively, designers explore alternatives to certain depth before backtracking to explore the next alternative, while keeping the explored path documented. Liu et al. [14] states that designers do not discard undeveloped and solutions with less potential. Instead, these solutions are recorded so as to enable backtracking in more advanced stages when more information is available to justify the action. An observational study done by Meniru et al. [10] showed that some designers keep records of the design progress so that they can backtrack to an earlier point in the design session or branch off towards a different goal. Goel [50] described exploration as developing sparsely connected modules and developing them incrementally, possibly concurrently, without the need to make irrevocable commitment to any particular

solution. The exploration of alternatives in depth or detailness requires constraints to be assumed temporarily, hence backtracking is possible. In this way, exploration of numerous alternatives is possible without making commitment.

Liu et al. [14] proposed a prescriptive pattern of concept generation and selection to increase the effectiveness of exploring of concepts, with minimum compromise to the richness of the solution space explored. Holland et al. [16, 17] postulated an interactive conceptual design system using function-means map as the backbone of exploration. Deng et al. [39] considered exploration of functional design information as an important aspect of design work. In the process of exploring design space for possible solutions, they deemed it necessary to abstract, explore and organize information including function, behaviour, structure and working environment. Such organized information may constitute a design representation model, as a premise to a computer-based design tool. In the exploratory process, Suh [19] recommended zigzagging between two domains, namely FR and DP domains. Sullivan [31] suggested the decomposition of functions, followed by mapping them to physical entities with the considerations of behaviours. Zeng and Gu [21] proposed the mapping of design specifications to product descriptions as the concept exploration process. Roy et al. [51] found that a comprehensive exploration of non-geometric concepts in conceptual design phase is still not well established. They presented a structural framework to drive the design process, through defining a set of generic design objects representations. Bracewell and Sharpe [32] advocated the development of solutions from the first principles, exploring the myriad of alternative schemes in the process before choosing the best alternatives for pursue in detail. Medland and Mullineux [36], in their conceptual design model, derived rules (or constraints) from the requirements. This set of rules restricts the exploratory space, hence assisting in producing designs that satisfies the product requirements.

Nellore et al. [52] investigated the needs of a structured process to manage specifications that will shape the final product. Concept exploration helps designers to define the structure of the problem space and the potential design solutions. Deng et al. [39] asserted that structuring exploration process is necessary for developing a computer-based design tool. Stempfle and Badke-Schaub [27] assumed four cognitive operations of generation, exploration, comparison and selection in a laboratory study. Their subjects of study are designers in the process of exploring solution concept. Zou et al. [5] observed that current CAD's emphasis on detailed design has left concept exploration process in conceptual development inefficiently supported.

The above reviewed research efforts can be seen as taking exploration as a main process of conceptual design. Exploration is essentially a representative process of conceptual design. It is potentially beneficial to the design community if it can be effectively modeled and assisted by computer. Such an approach will be discussed in the next section, with interest in the possibilities of guiding the exploration process via exploiting AI techniques. In the next chapter, a framework for facilitating the exploration of design concepts is proposed.

2.1.2.3.3 Guided Conceptual Exploration

Other than having the basic function of *facilitating* concept exploratory process, it is also necessary for a CACD system to possess functions that intelligently guides the exploration process.

The exploration process, in conceptual design, consists of both concept generation and selection [14]. Traditionally, decisions on generation and selection are left to team dynamics and individuals' tacit experiences [9,14]. Applications of AI and other computer-based techniques may support both the generation and selection processes. A good CACD system is assumed to be the one that support and encourage designers to

generate a range of possible conceptual solutions, to select, and to strategically explore the designs. The following paragraphs review and discuss the three aspects of guided conceptual solution exploration.

- Computer-aided Concept Generation
- Computer-aided Concept Selection
- The Pattern of Exploration Process (Generation and Selection)

Computer-Aided Concept Generation

Researchers have used various AI techniques to represent knowledge and to reason it. Knowledge-based systems have been proposed to aid in concept generation [9,31,32,52-54]. Bracewell and Sharpe [32] developed a system, so-called Schemebuilder, to guide designers through the process of conceptual scheme synthesis. This system supports users by providing options of solutions from the database. In FuncSION [14], solutions are composed of building blocks, stored in database library. Activities for synthesis involve generating exhaustive variations in these compositions in terms of number and types of building blocks composed. The idea behind the development of FuncSION was to provide designers with the relevant concepts and to support the exploration of concepts into details. Similar to Schemebuilder, the assistance of FuncSION is in the form of process facilitation and prompting ‘what’ are the available options.

Matthew et al. [33] viewed producing computational tool for conceptual design as a challenge. They deemed rules and algorithm to be inevitable in such tool, which may not rest well with enhancing creativity. Innovative solutions and solutions based on newly found technologies (which do not exist in past designs) are not found in knowledge-based systems (KBS). For KBS to generate newer solutions, it should allow human designers to input new and innovative solutions into the system. This approach allows CACD system

to design without existing data, and allows the system to progressively learn domain knowledge as users utilize it.

Computer-Aided Concept Selection

Matthews et al. [33] highlighted the need for designers to know if their lines of thoughts in pursuing are worthwhile, or if their efforts would be better spent following a different path of solution. Seo et al. [11] pointed out that time in product design industry is critical; it limits the ability of designers to create larger number of detailed models for many different concepts. Many concepts have to be evaluated and selected without the establishment of their details. To choose path of concept exploration, in practice, designers implicitly in their mind discard infeasible solutions based on their experience, particularly at a more abstract level of solutions [14].

To limit the solution space, Schemebuilder's [32] users are recommended to *manually* supervise the development of the tree by closing down branches that go nowhere or are currently of little interest to the designer. This means, the navigation or exploration of the design space is undertaken with the discretion of the human designers. In the similar vein, Ulrich and Eppinger [34] advocated designers to prune unrealistic or less desirable solution in their concept classification tree. According to them, these acts of pruning can be based on design team's discussion and their experiences and judgments. The underlying understandings required of the designers to perform such judgment can be in the forms of underlying trends, existing solutions, tradeoffs, or awareness of possible technological alternatives [33]. The endeavor to 'wisely' explore the often unbounded design space, under time constraint, knowledge built up through many years of experiences, resided tacitly within individual designers is required.

Relying on human's assertions during solution selection has its disadvantages. Impedance arises when the number of branches or solution alternatives gets too

numerous, at various levels of abstraction, which is also increasing. Ulrich and Eppinger [34] suggested that although hundreds of concepts may be found in the concept generation step, only five to twenty will be seriously considered by designers. This could be due to the difficulty for designers to have an objective comparison between potential branches, and deciding the alternatives for further considerations. Human judgment may be affected due to the huge number of alternatives, fuzziness and complexity of such exploration process.

CACD systems should acquire knowledge and selection decisions from the designers as design progresses. Apart from that, the system should preferably support the users in selecting a most promising path for further exploration, amidst the wide range of alternatives at various levels of abstraction. This requirement will be addressed in Chapter 4 of this thesis.

The Pattern of Exploration Process (Generation and Selection)

Liu et al. [14] emphasized the importance of ‘managing solution space’ during the conceptual design process. Issues of solution space management include *what* concept to generate and select (as discussed), as well as *when* to perform generate and select. ‘When’ to diverge and converge or the pattern of exploration is another facet of implication on the proficiency of conceptual design, which will be discussed below.

Liu et al. [14] identified a pair of conflicting goals of concept development, i.e. generation of concepts have to go for the widest possible range, while their management must go for minimum possible number. Figure 2-2 (a) shows a case of concept divergence to the maximum before convergence effort is applied. To converge with large number of alternatives would be difficult. Liu et al. [14] suggested that repetitive pattern of diverge-converge being more practical, as shown in Figure 2-2 (b). They proposed a specific diverge-converge pattern of approach towards an effective way of exploration (Figure 2-

3) and claimed that such approach is effective in managing large concept space. Cross [48] deemed the exploratory process as a convergent one, overall, however divergent at times to widen searches (see Figure 2-4). Pugh [55] suggested that concept development process have to be conducted in a controlled series of divergence and convergence, as depicted in Figure 2-5.

Figure 2-2 Pattern of Conceptual Exploration [14]

In Cross's, Pugh's and Liu et al.'s [14] patterns of exploration, there are multiple convergence and divergence, in alternate fashion, progressing towards a final solution. In Pugh's and Cross's suggested patterns (Figures 2-5 and 2-4), the number of alternatives have overall trend of decreasing, along the process to solution arrival. In Liu et al.'s [14] approach, they argued that the concept range should be generally expanded till somewhere in the middle before decreasing.

Figure 2-3 Liu et al.'s Proposed Pattern of Conceptual Exploration [14]

Figure 2-4 Design Approach Characterized by Cross [48]

Figure 2-5 Pugh's Proposed Model of Conceptual Exploration [55]

Fricke [56] conducted an investigation to probe the effectiveness of different conceptual development divergent/convergent pattern. Figure 2-6 shows 3 types of concept generation and selection pattern. From left to right of Figure 2-6: (a) Systematic expansion of solutions, but not managed to decrease the solutions. This pattern is analogous to breath-first search in AI context. (b) Balanced search, with divergent and convergent alternating. This can be seen as best-first search, guided by some rules and criteria. (c) Unreasonable restriction of search space. Fricke's [56] observations imply that a balanced search with multiple divergence and convergence is most likely to lead to successful designs.

Figure 2-6 Fricke's 3 Types of Concept Development Pattern [56]

The overall idea, as described by Liu et al. [14], is to increase the effectiveness of exploring concepts with minimum compromise to the richness of the solution space explored. In this mode of exploration, rule and criteria for expansion and contraction of solution alternatives can be used to guide and regulate the conceptual development process. In the case of prescribing a CACD system, these rules and criteria, which may be residing tacitly in teams and individual designers, have to be made explicit, possibly in a form of algorithm or rule set for implementation using information technologies. This requirement of a CACD system will be addressed in Chapter 4.

2.2 Artificial Intelligence in Design

Gero and Sudweeks [57] asserted, “Computer-aided decision making will produce better designs”. Inappropriate applications of computer assistance or AI may however not result in better designs. The ability of AI in assisting or oppressing design performances depends on, among other factors, the chosen portion of conceptual design process to be supported, as well as the adopted methodologies.

Eagan [58] summarized in a report on design methodologies that *design is not an endeavor that can be totally automated*. This means that design processes may not be all hidden in a black box, with a start and a stop button. Along the process of product design, or conceptual design in our interest, certain portions are deemed more suitable for computer assistances, while other sub-procedures could be better off performed by human. This is probably why full automation of design cannot be justified and envisaged. Bracewell and Sharpe [32] suggested that a more appropriate approach to CACD system is for the computer to provide decision support while allowing human designers to apply judgment. Computer has three main strengths, namely high processing power, large data storage, and almost instant worldwide connectivity. Human cannot compete with computer along these three dimensions. On the other hand, human is relatively far more adaptable, assertive and creative. Design of CAD systems can be perceived as an attempt to optimize the configuration between artificial intelligence and natural intelligence. In retrospective, one may reason that in designing conventional CAD tools available today, such as Pro/Engineer, each of those computers and human’s strength has been strategically synergized. Computer does the memory work, complex and extensive calculations, as well as facilitating collaborations of geographically dispersed team. Among those tasks, human’s experienced judgments and innovativeness are invited at ingeniously selected junctions to co-drive the design process. Similar philosophy has been

detected in CACD systems, which are generally in the research and prototype stage currently.

Internet-based collaborative issues on CAD systems have been a worthwhile area for research. It is worthwhile to note that this project is focused to research on the fundamental framework of a CACD system, instead of taking up the Internet-based collaborative issues. Collaborative design systems introduced by vendors, such as Pro-Engineer's Windchill ProjectLink [59] are developed based on their well-established CAD framework. Relative to such conventional CAD systems assisting in detailed design phase, research efforts on CACD system are not in anyway unified and matured currently [60]. Collaborative CAD of conceptual design phase may be more suitable for future work when CACD systems are more established in the research field and in the industry. For the research and development status of collaborative CAD software, the reader may refer to the most recent work by Li et al. [61].

As a recap, the following sub-sections report the state-of-the-art CACD research efforts (Section 2.2.1), and review the research initiatives aiming to support conceptual design by applying IT and AI techniques (Section 2.2.2).

2.2.1 Overview of CACD Research Efforts

Wang et al. [4] observed that there are many CAD tools to support detailed design, while there are only a few to support conceptual design [7]. The review of a range of CACD-related research efforts is arranged in a chronological fashion below.

In 1993, Bracewell and Sharpe [32] introduced a 'comprehensive' system that supports and guides designers through concept generation, selection, simulation and optimization. Nishioka et al. [62] proposed a Problem Interactive Clarification and Concurrent Solving System (PICCSS) to aid in product concept development. Elsas and Vergeest [63] created their version of CACD system, which consist of only sketching aid,

meant to assist industrial designers. In the literature published in 1998, they commented on their observation – systems assisting industrial designers are rapidly appearing. Al-Salka et al. [2] constructed the conceptual design support and analysis system (CODSAS), a domain-neutral framework for carrying out conceptual engineering design. In 1999, Zeng and Gu [21] integrated design objects and design processes, during an attempt to set up a formal framework for architectural conceptual design. Later, they have proposed a mathematical based conceptual design process.

In 2000, Nortel created a simple CACD system. Mitzi and Tony [64] reported that this system is a structured approach for idea development and evaluation for the fuzzy front-end process. This system is however relatively simple, in terms of functions, compared to the CACD systems proposed during that time. In the same year, Deng et al. [39] used a prototype system to illustrate the process of functional design information exploration during conceptual design. Zhang et al. [54] presented a knowledge-based conceptual synthesizer (KBCS) that support the synthetic phase of conceptual design.

In 2002, Tay and Gu [26] developed a conceptual design support system which allows designers to view design objects at various level of abstraction. Sullivan [31] established an interactive conceptual design support system, based on the notion of scheme generation. This system, similar to Holland et al's [16, 17], is based on function-means map. In the domain of architectural design, Szuba et al. [65] proposed a CACD system, so-called GraCAD. It is a high-level design tool, which adds on to the well-known commercial program ArchiCAD. Khoo et al. [43] investigated on a prototype customer-oriented information system (COIS) for product concept development. Their proposition increased customers' involvement using analytic hierarchy process (AHP).

In 2004, Meniru et al. [10] established a list of specifications for the creation of a computer-aided conceptual building design system. Through observational studies of eight designers during their building conceptualization process, they identified the main

requirements of a CACD system. Numerous research efforts have been made to facilitate product conceptualization processes using computers. Most recently in 2004, Holland et al. [16, 17] developed a CACD prototype system that claimed to be seamlessly integrative to AutoDesk Inventor. The prototype CACD systems' data structures are capable only to accommodate concepts laterally, not vertically. Hence, exploration using these systems does not seem to be able to penetrate solutions into further details, which ironically is the course of conceptual design. It is domain-specific, for mechanical system design.

Limited success has been achieved in using agent-based design systems to support conceptual design [66]. According to Gero and Kannengiesser [66], this is due to an ignored challenge of conceptual design - the requirements are not all known at the outset of a design task.

It has been noticed from the trend that CACD system research has received constant attention and has been progressive in the last decade. The constantly urging industry needs could have propelled domain specific applications to be studied, such as building and mechanical design conceptualization. Thus a practical system is required in the field to lay the groundwork, as opposed to a sophisticated and complex one. The CACD system recently prototyped by Holland et al. [16, 17] has taken such step. As Wang et al. [8] have perceived, much work remains to be done in spite of the great advances in this field. To this end, this thesis will henceforth propose a set of theories and corresponding computational framework that aims to address the issues raised and discussed in this chapter.

2.2.2 AI in Conceptual Design

AI requires tacit and implicit knowledge formalized, followed by codifying them for computer's reasoning and processing. This is commonly known as knowledge representation or modeling. Researchers have defined design knowledge or objects of

conceptual design using various taxonomies and terminologies, as described in Section 2.1.2.2. Section 2.2.2.1 reviews computationally *how* (as opposed to ‘*what*’ in Section 2.1.2.2) are these design objects modeled in AI. Having reviewed how is design knowledge modeled, Section 2.2.2.2 covers the run-down of the range of AI reasoning techniques that are applied on the knowledge.

2.2.2.1 Knowledge Modeling

Wang et al. [4] deemed that a design concept is difficult to capture, visualize and communicate electronically due to its ‘soft’ nature. A wide range of knowledge representation techniques has been proposed in the AI literature. Among the formalization techniques used are mathematics, predicate calculus, conceptual graphs, frames, scripts, objects, semantic networks and production rules. In the range of the proposed CACD systems reviewed in the previous sections, proponents have used a variation of representation schemes.

Roy et al. [51] captured customer’s specifications and its derived technical specification in an object-oriented (O-O) environment using UML and C++/Java. Khoo et al. [43] employed O-O techniques in their customer-oriented information system, enabling efficient data management. Tay and Gu [26] proposed a multileveled O-O structure to model products in the conceptual design phase. Zhang et al. [54] integrated O-O and production rules representations to solve conceptual synthesis problems. Brunetti and Golob [7] proposed a feature *schema* to store, manage, and retrieve product semantics including conceptual data. McKay et al. [67] defined a product data model to capture product specifications structurally. This data model was proposed to be used in the early design process. To support engineering conceptual design, Sullivan [31] utilized Galileo, a constraint programming language that is a high-level language, based on frames and first order predicate calculus. An architectural conceptual design system, so-called

GraCAD, is developed by Szuba et al. [65]. It adopts a graph-based knowledge representation approach. Moulianitis et al. [49] used mathematical vectors to represent scores in a concept evaluation model. Zou et al. [5] described product model mathematically in terms of functions, features, attributes and relationships. Zeng and Gu [21] used set theory to represent knowledge, in their approach to formalize design process. Zou et al. [5] represented conceptual model using hierarchical structure, on top of employing mathematics in a relational formalization. A mathematical structure has been proposed by Braha and Reich [47] to model engineering design process. Chen et al [45] presented a so-called CAH (Customer Attribute Hierarchy) to organize voices of customers.

It can be summed that the multitude of researchers have used not one, but a set of representation schemes to integrally model design knowledge. Among them, graph languages, predicate calculus (first order), rules and mathematical notations are widely adopted to bridge between design theories and computational objects.

2.2.2.2 Knowledge Reasoning

In the endeavor to automate or aid various segments of the conceptual design process, an array of AI techniques has been proposed. Developing efficient computational tools for product conceptualization is widely accepted as the principal issue of improving current CAD systems [14,68].

The research on automated design alternative generation and evaluation was started with the development of knowledge-based system [69]. Zhang et al. [54] presented a knowledge-based conceptual synthesizer (KBCS), developed using an expert system shell called CLIPS. Sullivan [31] specified a knowledge-based system with function-behavior reasoning engine for product conceptualization. Holland et al.'s [16,17] CACD prototype system possesses requirement-reasoning capability, based on constraints

inference process. Bracewell and Sharpe [32] designed Schemebuilder system, a ‘comprehensive and intelligent’ knowledge-based conceptual design package. Ratchev et al. [69] proposed a knowledge acquisition and sharing system for customer’s requirements engineering. Researchers [9, 70] have proposed approaches to support conceptual design based on expert systems, using the framework of QFD to manage design. Han and Lee [71] presented a case-based approach to represent and reuse the underlying design concepts in the existing mechanisms, in order to synthesize mechanisms for function-generation and motion-transmission. On the premise that technical systems can be designed by chaining physical laws, Zavbi and Duhovnik [72] developed an algorithm for design conceptualization. The algorithm draws information from a database of 139 physical laws.

In general, neural networks are good for classification tasks and for performing associative memory retrieval. Adaptive resonance theory, specifically ART2 neural network has been employed by Chen et al. [45] for customer and market analysis. Sun et al. [73] developed a systematic approach to identify the best design alternative, as the basis of a CACD system. This approach is supported by neural network-based fuzzy reasoning. Seo et al. [11] proposed an approach that estimates product lifecycle cost in conceptual design phase using neural networks. Neural networks often require large set of training data [6], hence may not be directly supportive in new product development projects.

Qiu et al. [74] and Lin [75] have separately proposed the use of genetic algorithms (GAs) to support intelligent automated conceptual design systems. An idealized framework for conceptual design has been presented by Goldberg [76], who argued that human designers should perform at least as well as GAs. In view of the inability of sorting technique to handle imprecision and uncertainty inherent in customer voices, Yan et al. [77] presented an integrated approach based on fuzzy evaluation and repeated

single criterion sort. Other researchers [12,78-80] have respectively proposed the use of fuzzy sets to aid in conceptual design activities.

In 1972, marketing researcher, Tauber [81] proposed the use of simple heuristics to generate concepts, specifically food product. Matthews et al. [33] have recently proposed a heuristic-based knowledge extraction method, mining knowledge in the explicit form (as opposed to neural network's implicit knowledge) from existing designs located in the database. Most often, heuristics are applied to established design spaces, searching for design solutions. Li et al. [82] proposed to use best-first heuristic search to generate feasible design solutions in conceptual design phase. This method, like other heuristic-based methods, assumes existing product knowledge is available.

The above paragraphs have reviewed various computational methodologies researchers have used to aid in assorted tasks in the conceptual design process. Among the techniques used are neural network, genetic algorithms, case and rule based reasoning, fuzzy logic, and heuristics. It is envisaged that a combination of these techniques may be required to specify a CACD system to intelligently support human during product conceptualization.

2.3 Chapter Summary

CAD systems call for a set of well-defined design theories to underline the computational facets. According to the adopted meta-theoretical structure (in Section 2.1.1), the three levels of theory addressed are the knowledge, the process, and the philosophical levels. Based on these 3 levels of premises, the knowledge modeling and reasoning techniques are called upon to implement the theories in a CAD system. (One pitfall at this junction is probably to work the other way round, that is, to customize the design theories according to the available AI techniques. These mistakes are observable in research works that failed to discuss design theories before embarking on implementation using IT and AI.)

Research in CACD system has been deemed necessary, and urgently. Recall that the abstract objective of this project is to approach the realization of a CACD system. With this in mind, this chapter has identified and reviewed the classes of research necessary to accomplish this objective. As rationalized in the preceding paragraph, the relevant literature includes conceptual design theories (at knowledge, process, and philosophical levels), the state-of-the-art CACD systems, as well as the knowledge modeling and reasoning aspects of the CACD system.

Section 2.3.1 summarizes the literature reviewed and the issues raised in this chapter. Gaps are identified among the bodies of knowledge, leading to the objectives stated in Section 1.2. In accordance to the requirements of the stated objectives, the respective hypotheses are formulated (in Section 2.3.2). In the last section (Section 2.3.3), sets of tasks that can bring about the soundness of the hypotheses (and therefore the attainment of the two stated objectives) are listed, guiding the development of this project.

2.3.1 The Literature Reviewed

The following summarizes the issues discussed and gaps identified during the literature review.

1. *Philosophical Level*: Human-orientation is highly regarded as one of the most important philosophical criteria in specifying a CAD system [10,16,17,26,31-33]. Instead of aspiring to build a highly independent system to automate design, focus has been shifted to create a supportive system around human's capability boundaries. The strengths of natural and artificial intelligence should be strategically deployed in a CACD system. Apart from this, the characteristics of human thoughts (i.e. cognition) and acts (i.e. behaviour) during product conceptualization should be a factor of the

system design. As such, references from the studies of cognitive science will be adopted in this project.

2. *Knowledge Level*: Information involved in conceptual design is inherently disorganized, imprecise, uncertain and fuzzy [4,5,10,12]. The taxonomies and terminologies conversed during conceptual design are proliferated. A set of design objects or taxonomy to represent conceptual design information is required in this work. The relationships amongst the objects, such as customer requirements, product structures and behaviours have to be accordingly formalized as well. Apart from these, it was summed from the literature that one of the specifications for CACD system would be to require abstraction levels to represent knowledge, as well as to facilitate the processes of solution concept detailing.
3. *Process Level*: It has revealed through reviewing that the identification and implementation of customer requirements is a significant issue for successful product development. A design methodology that systematically assimilates customer requirements along the process of concept development is required to implement and streamline the process, to gain positive effects of customer-orientation philosophies. It was also found that concept *exploration* could be a representative process of conceptual design. The envisaged system to be proposed in this project is able to support the users in selecting the most promising path for exploration, amidst the wide range of alternatives at various levels of abstraction. This form of assistance is deemed to be potentially beneficial to the design community.

2.3.2 The Hypotheses

This project assumes that certain configurations of IT and AI techniques are contributive to the attainment of the two objectives stated in Section 1.2. Two hypotheses are respectively set out as follows.

1. A family of modeling methods, which may include mathematical assertions, frames, graphs and objects, is able to integrally represent different facets of design knowledge. The envisaged customer-oriented knowledge organization structure can be formulated based on these modeling methods.
2. An intelligent design exploration method can be devised by explicating human's design cognition into an algorithmic format, which is henceforth applicable in a proposed CACD system. This method can be based on the principles of heuristic.

The affirmation of the above assumptions will lead to the attainment of the two-tiered objectives.

2.3.3 The Tasks Required

To verify the hypotheses set out above, the following series of tasks are devised.

1. To establish the framework of a proposed conceptual design system.
2. To formulate a knowledge organization structure as required by the first objective.
3. To review and critically study the applicability of heuristic algorithms on product conceptual design problems.
4. To prescribe a set of heuristic-based conceptual design exploration algorithms, as required by the second objective.
5. To exemplify the propositions made using design cases. The logics and the pragmatic values of the propositions are to be assessed. From these cases, the integrity of the hypotheses is to be discussed.

Tasks 1 and 2 will be addressed in the Chapter 3, while Tasks 3 and 4 in Chapter 4. Finally, Task 5 will be reported in Chapter 5.

By the end of this project (in Chapter 6), the implications of the work done in this research project with regards to the reviewed literature (essentially summarized in Section 2.3.1) will be discussed. This would, as a conclusion, put the results achieved in this

project back into the context of the original research, where the gaps were initially identified.

– Chapter Three –

A Conceptual Design Knowledge Organization Structure

A knowledge base can be thought of in terms of a *mapping* the objects and relations in a problem domain to the computational objects and relations in a program environment [83]. Knowledge of design, as the language of design communication [21], has to be *formalized* in order for a system to function. These *mapping* and *formalizations* processes can be perceived as putting physical phenomenon through a filtering theory [84]. The result is a model of the physical phenomenon. Figure 3-1 shows the process of modeling the real world using theories. Friedman [85] quoted Sutherland (1975) describing theory as ‘an ordered set of assertions about a generic behaviour or structure assumed to hold throughout a significant broad range of specific instances.’

Figure 3-1 Modeling Phenomena through Theories

In this chapter, a theoretical structure is proposed to organize the knowledge involved in the conceptual design process. The structure serves as a model to the real world phenomenon. This model will be the premise to a proposed computer-aided conceptual design (CACD) system, so-called Conceptual Design System (CDS). The architecture of CDS is presented in Section 3.1.

In Section 3.2, a set of theory is proposed to model the world of discourse in the product conceptualization processes. Drawing on the parsimonious strengths of set theories, the model is represented mathematically and computationally. A so-called Design Space Framework (DSF) is proposed to put the numerous definitions made into a congruent perspective. Based on graph theories, DSF facilitates as a data structure to support product conceptualization in a knowledge intensive environment. By the end of this chapter, the propositions made are discussed and concluded in Sections 3.3 and 3.4, respectively.

3.1 System Architecture

Conceptual Design System (CDS) is proposed as a CACD system in this project. It provides a context for clearer elucidation of the propositions made in this work. The proposed architecture of CDS is depicted in Figure 3-2. It contains the necessary modules, as described below, to support the functions of the propositions made in this project.

Design Exploration Module: The central control module where procedures of computation resides. This module carries prescriptive algorithms, capable of invoking functions in other modules.

Design Space Module: A record of logical flow and status of conceptual design problems. This is the ‘short-term or working memory’ of the system [30, 86].

Database: For storage and retrieval of design data, facilitating knowledge reuse. This is the ‘long term memory’ of the system.

→ : Data Transfer

Figure 3-2 Architecture of the Conceptual Design System

User Interface: This module bridges the Design Exploration Module, Design Space Module and the system users. Data are exchanged between the users and the system via this module. A screenshot of the prototyped CDS is presented in Chapter 5.

Research in knowledge-intensive CAD systems is multi-faceted, with numerous focuses and fronts. The CDS architecture proposed above does not necessarily satisfies the requirements of a full-fledged CACD system. It is nevertheless constructive in this thesis, since it elucidates how are the propositions made (in Chapter 3 and 4) related to a CACD system. When this research progresses beyond this thesis, the system framework should be reviewed and improved to adapt to any new requirements.

3.2 Conceptual Design Modeling and Representations

Zeng and Gu [21] termed knowledge of design as language, and processes as law. Brunetti and Golob [7] stated that the procedure of design is supported by design methodologies via providing specific design methods and design knowledge. Based on these models, this section attempts to formalize conceptual design knowledge, thus defining the primitive objects for the proposed CDS.

Instances of knowledge conversed during conceptual design processes include examples like: ‘this shaft transmits energy to the flywheel’, ‘an I-beam is between the blocks’, ‘this product has to weigh under 3 kg’, ‘the springs absorb the shock’, ‘customers want to feel comfortable’, etc. Such knowledge ought to be represented in the CDS, as depicted in Figure 3-3. On this note, the first definition is postulated in Definition 3.1.

Figure 3-3 Conceptual Design Knowledge Representations

Definition 3.1: All design knowledge exists in the universe of discourse within the CDS domain, denoted as D .

Nishioka et al. [62] commented that knowledge is characterized by the connections amongst them. Apart from identifying the primitive objects (in Section 3.2.1), the connections, or the relationships amongst them have to be formalized (Section 3.2.2). Intrinsic knowledge of design objects can be further organized and represented

using frames, which will be proposed in Section 3.2.3. A knowledge organization structure, so-called Design Space Framework (DSF) can be established to order the numerous definitions into a congruent perspective (Section 3.2.4).

3.2.1 Knowledge Objects

The instances of information utilized in conceptual design (as seen in the left portion of Figure 3-3) seem disorganized and daunting at the first glance. Especially if they ought to be fully represented in CDS. Fortunately, proponents of design science have created versions of taxonomy to classify and systemize them (as discussed in Section 2.1.2.2.1). Unfortunately, according to Love [23], the versions are however unnecessarily extensive. These multitude versions of classification schemes challenge the adopters to review and possibly reorganize them according to their own requirements.

In the following paragraphs, primitive classes of design information captured in domain D are formalized based on mathematical sets theory. This method of formalization is chosen for its parsimonious characteristics and its compatibility with object-oriented programming technique, which can be adopted during implementation.

Definition 3.2: Let N be the set of finite design objects considered in D .

In the customer-driven design paradigm, customer needs are perceived as an important piece of design information in the product conceptualization phase. It is based on to determine design specifications under various theories [19,34,35]. This class of information can be obtained via market research efforts such as focus groups and interviews [34]. In this work, it is assumed that every aspect of a product can be duly traced back to the voices of customers. May it be a rounded edge, or the choice of colour, they can be reversely traced to the related customer requirements. The definition of *customers*, in this work, encompasses all stakeholders in the product lifecycle, not solely referring to the end-users.

Definition 3.3: Let the customer requirements be represented by $c_i \in C$,

$$C = \{c_i | i = 1, \dots, I\} \quad \forall c_i \in C \subseteq N \quad (3.1)$$

where i is the sequential order for element c_i , and I is the total number of element c_i in a design problem.

Product requirements are often formalized in design process to document the required product characteristics that can satisfy the customer needs. Other terminologies used in literature include functional requirements, engineering characteristics, and design principles [16, 19, 31]. This class of information, in customer-oriented design philosophy, bridges customer needs and design solutions.

Definition 3.4: Let the product requirements be represented by $p_j \in P$,

$$P = \{p_j | j = 1, \dots, J\} \quad \forall p_j \in P \subseteq N \quad (3.2)$$

where j is the sequential order for element p_j , and J is the total number of element p_j in a design problem.

Design solutions can be characterized by *design parameters* [19] and *schemes* [32] among other types of representations. Solution to design is regarded as another class of information in this work. Unlike the former two classes (C and P), solutions may have alternatives in the context of product design. Hence, alternative solutions, if any, have to be represented in this framework. It should be noted that within a set of solution alternatives, only one is valid to the main solution at any one time.

Definition 3.5: Let the design solutions be represented by $s_k^l \in S$,

$$S = \{s_k^l | k = 1, \dots, K \wedge l = 1, \dots, L_k\} \quad \forall s_k^l \in S \subseteq N \quad (3.3)$$

where k is the sequential order for element s_k ; K is the total number of element s_k in a design problem; l is the sequential order for alternative solutions of s_k ; L_k is the total number of alternative solutions of s_k .

Any complete product (for instances a bicycle, or a mug) is a form of solution, to a given problem. From this perspective, a special case of $s_k^l \in S$, where $k=l=0$ (i.e. $s_0^0 \in S$), is predefined to denote the solution of the highest abstraction level within a given case. In other words, s_0^0 represents the most abstract solution in a given problem.

From Definition 3.2 to 3.5,

$$N=\{C, P, S\} \quad (3.4)$$

Equation 3.4 states that objects C , P and S are the subsets of finite design objects N , considered in D .

In representing concepts, sketches and abstract descriptions are used, especially in the preliminary stages such as brainstorming. In research, three types of objects have been widely used for reasoning during design process. These objects include function, behavior and structure (FBS) [8, 31, 39]. Some proponents of FBS have used F (function), as Suh [19] has used FR , to describe functional requirements, and used S (structure) to describe solutions, which is in the form of physical structure, in the mechanical design domain. However, in the proposed CDS model, FBS is wholly adopted to describe conceptual solutions $s_k^l \in S$. In CDS, it is proposed that F represents the actual functions of solutions, as opposed to FR (functional requirement), which describes the desired function of the solution. A represents the arrangement, configuration, or the structure of conceptual solutions. B represents the behaviour that is required to achieve the stated function F , based on a solution structure A .

Definition 3.6: Three elements, one from each set of F , B and A , describe the respective solution element $s_k^l \in S$. The sets F , B and A are defined as follows:

$$F = \{f_k^l | k = 1, \dots, K \wedge l = 1, \dots, L_k\} \quad \forall f_k^l \in F \subseteq S \quad (3.5)$$

$$B = \{b_k^l | k = 1, \dots, K \wedge l = 1, \dots, L_k\} \quad \forall b_k^l \in B \subseteq S \quad (3.6)$$

$$A = \{a_k^l \mid k = 1, \dots, K \wedge l = 1, \dots, L_k\} \quad \forall a_k^l \in A \subseteq S \quad (3.7)$$

From Equation 3.4 and Definition 3.6,

$$N = \{C, P, S\} = \{C, P, \{F, B, A\}\} \quad (3.8)$$

3.2.2 Knowledge Relationships

Customer requirement (C), product requirement (P) and solution (S) are three general types of information classes acquired, created and utilized in the conceptual design process. It is known that knowledge is characterized by the connections amongst them [62]. This section establishes a novel relational structure amongst C , P and S , which will facilitate the establishment of the DSF in the later section.

Definition 3.7: Let E be the set of finite design relationships amongst the design objects considered in D .

In various design methodologies [19,34,35], direct relationships exist between customer requirements and product requirements as well as between product requirements and solutions. These two classes of relationships are respectively defined as follows.

Definition 3.8: Let the relationships between domain C and P be represented by $u_{i,j} \in U$.

$$U = \{u_{i,j} \mid \forall i \in \{1, \dots, I\}, \forall j \in \{1, \dots, J\}\} \quad \forall u_{i,j} \in U \subseteq E \quad (3.9)$$

Remark 3.1: Equation 3.9 reflects that $\forall p_j \in P$ has a corresponding parent, $c_i \in C$, related by element $u_{i,j} \in U$.

Element $u_{i,j}$ can be represented in terms of the emanating and incidence nodes, as an ordered pair [87],

$$u_{i,j} = (c_i, p_j) \quad (3.10)$$

Definition 3.9: Let the relationships between domain P and S be represented by $v_{j,k}^l \in V$.

$$V = \{v_{j,k}^l \mid \forall j \in \{1, \dots, J\}, \forall k \in \{1, \dots, K\} \wedge \forall l \in \{1, \dots, L_k\}\} \quad \forall v_{j,k}^l \in V \subseteq E \quad (3.11)$$

Remark 3.2: Other than $s_0^0 \in S$ (as explicitly excluded in Equation 3.11), this equation reflects that, $\forall s_k^l \in S$ has a corresponding parent $p_j \in P$, related by element $v_{j,k}^l \in V$.

Element $v_{j,k}^l$, can be represented in terms of the emanating and incidence nodes, as an ordered pair [87],

$$v_{j,k}^l = (p_j, s_k^l) \quad (3.12)$$

Figure 3-4 depicts the two sets of relationships, U and V .

Figure 3-4 Relationships between C , P and S

This work proposes that there exist another set of *relationship*, apart from U and V , in the discourse of conceptual design. *'The solution to one problem becomes the basis of the next one,'* as described by Davidson and Sternberg [88]. Concept development can be treated as a special case of the human problem solving process. The latter has been intensively studied in cognitive psychology where this study has taken references. In real life problem solving, it is factual that every solution is also a problem, possibly awaiting further solutions. For instance, the decision to buy a car may solve a traveling problem, but will inevitably lead to further problems such as which make and model should be purchased and so on. This sub-problem would require further analysis and resolutions. In

this fashion, the problem-solution pattern can be repetitive. By going through this pattern repetitively, the result would be the progressive detailing of the solution to the initial problem. But when does one stops going through this cycle? One may relate that only when one needs to clarify the details of a solution, one would then consider the solution as a problem. In solving design problems, this logic can be similarly applied.

Axiom 3.1: Design solutions themselves are design problems if and only if more detailed level of solutions are required.

Axiom 3.1 states that elements $s \in S$ can possibly be perceived as design problems. For instance, a dynamo may be a solution to the electrical requirements of a bicycle. The dynamo itself could be then perceived as a sub-problem when the designers intend to look at its functional concept. Common to various design methodologies, the step after problem definition is to elicit customer requirements based on the defined problem [34]. This common procedure suggests that a defined problem can be the premise for object C (Definition 3.3). Since Axiom 3.1 (conditionally) equates design problem to design solution, it can be inferred that object S (design solution) is the premise of object C (customer requirement), under the condition stated in Axiom 3.1. Based on such inferences, the causal relationship between S and C in the process of product conceptualization can be defined as follows.

Definition 3.10: Let the relationships between domain S and C be represented by $w_{k,i}^l \in W$.

$$W = \left\{ w_{k,i}^l \mid [\exists k \in \{1, \dots, K\} \wedge \exists l \in \{1, \dots, L_k\}] \vee k = l = 0, \forall i \in \{1, \dots, I\} \right\}$$

$$\forall w_{k,i}^l \in W \subseteq E \quad (3.13)$$

Remark 3.3: Equation 3.13 reflects that $\forall c_i \in C$ has a corresponding parent, $s_k^l \in S$ (including $s_0^0 \in S$), related by element $v_{j,k}^l \in V$. However, not all $s_k^l \in S$ has a child $c_i \in C$.

This means that $s_k^l \in S$ can be a terminal node, as later seen in Chapter 4. It can be understood by comprehending that not all design solutions are further explored, such as the less favoured alternatives and the non-qualified solutions. Hence these cases will form terminal nodes in the context of product design.

Element $w_{k,i}^l$, can be represented in terms of the emanating and incidence nodes, as an ordered pair,

$$w_{k,i}^l = (s_k^l, c_i) \quad (3.14)$$

From Definition 3.7 and Equations 3.9, 3.11, 3.13,

$$E = \{U, V, W\} \quad (3.15)$$

The relationships amongst the represented classes design information have been formalized in this section. Axiom 3.1, in this work, facilitates the modeling of the conceptual design endeavor – a cyclical process of design knowledge generation, progressing towards the solidification of design concepts. Such model is illustrated in Figure 3-5.

Figure 3-5 A Novel Model of Conceptual Design Process

3.2.3 Design Objects Representation

Intrinsic knowledge of design objects can be further organized using network representations, based on further sub-objects and more links. Semantic network is one such method to represent knowledge in *object-attribute-value* triples. Apart from graphical network representations, knowledge can as well be represented efficiently using schemas or frames. It is known that when stereotyped structure in knowledge can be identified, frames will come in neatly [87]. As the information structures within the defined object classes are typical, they are suitable for frame-based representations.

Frame-based knowledge representations supports object-oriented (O-O) programming, widely used in AI implementations [87]. Frames are classes of data structures, consisting of slots holding attributes-value pairs pertaining to a certain represented object. Slots may also contain pointers and demons (or procedural knowledge). For the latter, a set of instructions may be invoked to acquire values from users. Default values may be predefined for various slots, which is an important aspect for a CACD system where missing information in conceptual design phase is common [89]. As in the fundamentals of O-O programming, frames may represent object classes, inheriting properties of the associated types. Inheritances here help to systemize knowledge and economize data storage space. For these reasons, frames are employed to represent classes of objects in the domain D . They are illustrated in Tables 3-1 to 3-4.

Name	string
Superclass	{[N]}
Class	{[C, P]}
ID	integer
Contribution fraction	float
Solve status	{[1, 0]}
Description	string

Table 3-1 C and P Object Frame

Name	string
Superclass	{[N]}

Class	{[S]}
ID	integer
<i>l</i> _ID	integer
MR	float
Contribution_fraction	float
Description	string
Heuristic_h	float
Solve_status	{[1, 0]}
f_ID	integer
b_ID	integer
a_ID	integer

Table 3-2 *S* Object Frame

Superclass	{[N]}
Class	{[U, V, W]}
Parent_ID	integer
Child_ID	integer
<i>l</i> _ID	integer

Table 3-3 *U*, *V* and *W* Object Frames

Name	string
Superclass	{[S]}
Class	{[F, B, A]}
ID	integer
Description	string

Table 3-4 *F*, *B* and *A* Object Frames

3.2.4 Design Space Framework

Researchers have identified two types of operations between sketches during conceptual design process [10,50], viz. lateral transformation and vertical transformation. Lateral transformations bring sketching from one idea to another, i.e. evolving alternatives. Vertical transformation brings sketching work to a different depth or abstraction level. Sketching, as a product conceptualization activity, can be perceived as an act of exploring design space vertically and laterally. It is the reflection of designers' thoughts and

working processes. As discussed in Chapter 2, a key philosophical requirement in specifying a CACD system is to comply with the designer's cognition. Along this line, this work proposes two *dimensions* in the design space D (Definition 3.1) - vertical and lateral. A third dimension is further postulated to hold the components (or the divisions) of objects.

D , first raised in Definition 3.1, is further defined as a space that can be characterized along three dimensions. It will accommodate design objects in N , where $N = \{C, P, S\}$ (Equation 3.8).

Definition 3.12: The three dimensions of space D are X , Y and Z , where

- X : Sibling objects along dimension X are the decomposed constituents related to their parents, having AND relationships amongst them.
- Y : Sibling objects along dimension Y are alternatives available for the parent, where only one is relevant to the parent at any time, hence having OR relationships amongst them.
- Z : Objects generated along dimension Z are of incremental detailing.

Thus far, three formalisms have been established:

1. The three dimensions of D (Definition 3.12),
2. Objects in D (Definition 3.8),
3. Objects' inter-relationships (Equation 3.15).

On the basis of these definitions, any instance of a particular design concept can be represented as a type of graph.

Definition 3.13: Design Space Framework (DSF) is a graph, denoted as G , existing in space D .

Figure 3-6 shows an instantiated design concept. This representation is a type of network model [87], where nodes represent objects and the arcs represent associations or relations among them. In the following sub-section, the characteristics of graph G (*with*

respect to product conceptual design application) are rigorously discussed. Before this section ends, graph G will accordingly be formalized based on graph theory [90].

Figure 3-6 A Case of Graph G in Space D

The Characteristics of Graph G

Graph G is proposed to exist in space D , characterized by the defined three dimensions, i.e. X , Y , Z . It is constructed by elements of set N and set E . It has been proposed that a special case solution node $s_0^0 \in S$ denotes the solution of highest abstraction level within a given design problem. Remarks 3.1, 3.2, and 3.3 state that every node in set N has one parent each, with the exception for element $s_0^0 \in S$ (Remark 3.2). It is defined that *rooted graph* [87] has a unique node, from which all paths in the graph originate. That is, this root has no parent in such graph. In graph G 's case, $s_0^0 \in S$ is the bespoke root. G can be considered as a tree graph. Tree is a special case of directed graph, holding two conditions: (1) there is no cycle, and (2) root exists in the graph. Stated in Remarks 3.1, 3.2, and 3.3, all nodes have one parent each (except the root node), indicating that looping or cycling in the graph G is absent, since no path can contain any node more than once [87, 91]. Since the above 2 conditions hold for graph G , it is a tree graph.

A tip or leaf node is a node that has no children. As described in Remark 3.3, although all nodes have a parent (except root node), not all nodes have children. Those

nodes without children are the leaves of graph G . Solutions that do not have children are the leaf nodes of graph G .

G and other trees used in AI applications are drawn with roots on top and arrows going down, with heads of arrows omitted since it is understood that they point downwards [90]. Tree, as a type of directed graph has been known to be useful for representing knowledge and operations performed on knowledge [90].

G can also be considered as an AND/OR graph, which is an important tool in AI problems. Graphs have been proven to be an ideal vehicle for formalizing associationists' theories of knowledge. Graph-based reasoning has its advantages over predicate calculus [87]. AND/OR graphs are capable of representing expressions such as $q \wedge r \Rightarrow p$ and $q \vee r \Rightarrow p$. Nodes in AND/OR graphs, G in this case, are connected by k -connectors [87], where k is the cardinality of the set of descendent nodes. For nodes with $k=1$, the descendents lie along Y -axis of G , where only 1 descendent is relevant to the solution $s_0^0 \in S$, at any one time (defined as an *algorithmic cycle* in the next chapter). The descendents are of alternatives to the parent. Such graphical representation is equivalent to the $\vee$ operator in predicate calculus. For node with $k>1$, the descendents lie along X -axis, which all are parts of their parents at all times. Descendents of such node are decomposed components from their parents. To represent such relationships graphically, traditionally, curved arcs between edges are drawn. The operator $\wedge$ is used in predicate calculus to represent AND relationships among operands.

In G , a 3rd dimension is used to differentiate AND and OR inter-relationships among nodes. Differing from putting all mandatory and optional nodes in a 2-dimensional plane, the 3-dimensional representation enhances human's perception of graph G . That is, the final solution can be presented on the two dimensional Z - X plane, isolating all unselected alternatives (hence irrelevant to final solution) behind Z - X plane, along Y -axis.

CACD system users are able to intuitively relate the design solution structure and the data structure, which are both structured similarly by definitions of graph G . In representing this *AND/OR tree graph*, notations of graph theory are used, as illustrated below.

Formalizing DSF as a Mathematical Graph

Generally, a directed graph is a quadruple (P, L, f, b) , where P is the set of nodes, L is the set of edges, and f and b are the forward and backward function respectively [90]. Graph G , being a tree graph, is a special case of a directed graph.

Definition 3.14: Let Design Space Framework (DSF) be an AND/OR tree graph G , a quadruple where

$$G = (N, E, \Phi, \Gamma) \quad (3.16)$$

N is the set of finite design objects (Equation 3.8); E is the set of finite design relations (Equation 3.15), while Φ and Γ are the forward and backward pointer functions [91] respectively.

It is noted that computer “sees” graph G differently than human, who tends to pay attention to graphical representation such as Figure 3-6 [90]. Since machines are symbol-oriented, all manipulations of G are accomplished using only the quadruple defined above. Navigations from node to node can be achieved via pointer functions Φ and Γ . Since the root node of G , i.e. s_0^0 , has no parent,

$$\Phi: E \xrightarrow{1-1} N - \{s_0^0\} \quad (3.17)$$

Definition 3.15: In graph G , the degree of nodes $n \in N$ (where $N = \{C, P, S\}$) is defined as $d(n)$. In the case where $d(n)=1$, n is a leaf node (excluding special case node s_0^0). In a tree graph, computer may identify leaf nodes judging from their degrees.

In this section, DSF is established to organize the generated primitive design objects and the constraining relationships between them. It effectively describes product concepts and the rationales of their existences.

3.3 Discussions on DSF

Reviews and discussions presented in Chapter 2 have led to the objectives of this research. Working towards the first of the two objectives, this chapter has formulated a novel knowledge organization structure, according to the hypothesis made in Chapter 2. The usefulness of this structure is discussed in the following sub-sections.

3.3.1 DSF as a Knowledge Organisation Structure

Design Space Framework (DSF), or graph G , is a data structure that has been defined to handle concept development knowledge. It contains a set of taxonomy represented based on set theory, including customer requirements, functions and structures, etc., to overall capture information utilized in this process (from discussion Section 2.1.2.2). The relationships among these elements are formalized such that they can collectively characterize a full product conceptual solution. DSF is mathematically represented as an AND/OR tree graph, with nodes as instances of design objects, existing in the defined three-dimensional space. The objects are in turn further represented by means of frames. This assembly of representation techniques integrally *articulates* and *codifies* [84] tacit knowledge dealt with in the conceptual design process.

3.3.2 Multi-leveled Conceptual Design Approach

In using conventional CAD system, engineers build up their new designs from parts into totality. On the other hand, in conceptual design, designers build products working from

the totality of the concepts down to the details [42]. DSF allows designers to work towards design solutions from general ideas. Liu et al. [14] deemed generating concepts in several steps of abstraction is less complex than generating a solution with details in a single step. Conceptual design methodologies that prescribe single-step solution generation may be impractical for considerably complicated products.

As summarized in Chapter 2, one of the specifications for a CACD system would be to require abstraction levels to represent design information. DSF leveraged on Axiom 3.1, modeling the conceptual design progression as a problem-solution cyclic process. The result is the division of design information into multiple abstraction levels. These multiple levels facilitate the handling of more complicated products.

3.3.3 Systematic Customer-Driven Design

Studies have recognized the identification and implementation of customer requirements as a significant issue for successful product development [13]. Customer voices should not only be elicited at the front end of the conceptual design process, but frequently at various junctions along the product conceptualization process. Unsystematic and non-coordinated elicitations of customer voices may lead to inefficiencies in design and can result in producing a product not specifically desired by the consumers. The proposed DSF promotes systematic customer requirements elicitation by having customer requirements (object *C*) acting as bridges intermediately along the product conceptualization process. This structure ensures that the *relevant* customer opinions are *constantly* sought and applied in the *entire* product conceptualization process.

3.3.4 DSF Supporting Creativity

The subject of innovation or creativity often arises in connection with conceptual design [18]. An innovative product can be designed if there are new needs, new solutions, or

both. The structure of DSF presents the designers a framework to systematically scrutinize the concept of a product, and possibly identifying an area to make a difference. Appending to the nodes in the X or Y dimensions may create an innovative product. For instance, when the marketing team discovers a new market need, a new *C* node can be added to graph *G* at a certain level of abstraction. From that node, a series of new product requirements and solutions will be generated from that node to result in a product that caters to the new need. In another example, if there is a technological advancement leading to a new alternative solution, a new *S* node can be appended to a current solution along the Y dimension, at a certain level of the sub-solutions. The result can be an innovative product. Cagan and Agogino [92] asserted that innovative designs require reasoning from the first principles – as facilitated in the DSF.

3.3.5 DSF in Capturing Design Rationales

In knowledge intensive CAD system, Mantyla [89] advocated the availability of a full trace from product structural element back to its justification in terms of customer-requested function. This notion is parallel to a philosophical assumption underlying the proposed DSF: *customer requirement is the source of every resulted design feature of a given product*. It is envisaged that CDS can be mapped to conventional CAD software for the continuation of product development – detail design. This would facilitate the capturing of design rationales as raised by Mantyla [89].

3.4 Chapter Summary

Tomiyama et al. [84] considered the intensity of knowledge not only refers to the amount in a chunk, but more importantly, how well systemized is the knowledge in that chunk. The attempt to systemize objects and their relations in the novel conceptual design

knowledge organization structure provides a new problem-solving paradigm, in the context of product conceptualization. In this model, systematic elicitation and assimilation of customer voices is facilitated, contributing to the first objective of this project. Apart from that, DSF facilitates other requirements such as capturing design rationale and supporting creativity.

DSF is proposed as a backbone information structure for application in concept development process. It outlines the taxonomies of objects, and their relationships, which by no means has completely represented all types and levels of information required in a knowledge intensive environment. Its generic fundamentalism, however, offers a platform for further developmental efforts.

In all, this chapter proposes and establishes a foundational knowledge organization structure, according to the first hypothesis (Section 2.3.3), as required by the first objective (Section 2.3.2). It forms the premises for further work. One of them, a heuristic-based conceptual design approach is presented in Chapter 4. Apart from this, other promising propositions are projected in Section 6.3.

– Chapter Four –

A Conceptual Design Exploration Approach

Based on the foundation laid in the Conceptual Design System (CDS) presented in Chapter 3, this chapter introduces a heuristic-based conceptual design approach. The prescriptive algorithms aim to support designers through the conceptual design process.

Section 4.1 points out the different nature between conceptual design problems and other classical problems that heuristic algorithms can deal with. From this gap, indications on how heuristic algorithms can be adapted to work on conceptual design problem are picked up. Section 4.2 presents an overview of heuristic algorithms, with one representative analyzed in detail. Having considered heuristic algorithms' background and their applicability in conceptual design problems, Section 4.3 proposes a novel conceptual design approach. Section 4.4 discusses the significances of the proposed approach. The last section, Section 4.5, summarizes the conclusions reached in this chapter.

4.1 Of Design and Heuristics

This section discusses the applicability of heuristic algorithms on conceptual design problems.

Pahl and Beitz [1] stated that design is a variant of general problem solving. Conceptual design is to utilize the judging ability of human being's thinking to solve basic problems in design and manufacture [25]. Gero [29] asserted that design systems must be based on human design processes. Finger and Dixon [18] summed that the knowledge of how human design is required to create better CAD tools. The above

perspectives of various theorists suggest the scrutiny of human problem solving cognition in prescribing theories for a computer-aided conceptual design (CACD) system.

Human problem solving (which is a generalization of product design task) can be represented by search space, as illustrated by Newell in his *Unified Theories of Cognition* [30]. The symbolic computations within this space capture the essence of problem solving processes. Solutions to problems are affirmed within this search space. Newell [30] created a problem solving system, so-called SOAR, establishing relationships between AI and cognitive science. AI and cognitive science are closely knitted and of an iterative nature [93]. Academics have proposed search algorithms, such as breath-first and depth-first search, to systematically uncover solutions in design space. To increase the efficiency, best-first algorithms [87, 91, 94, 95] were suggested where heuristics, as explicit rules derived from human experiences and tacit knowledge, play decisive roles.

Heuristics are rule-of-thumb that have been successful in producing “acceptable”, not necessarily “optimal” solution to a type of problem [82]. They are also criteria, methods, or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goals. Like a chess grand master who does not always win, heuristics does not guarantee to identify the most effective course of action, but do so sufficiently often [81, 91, 96]. To guide the exploration of concepts, in the case of design, a priori knowledge must be employed – heuristics.

Among several classical game problems used to explicate heuristic principles, such as the 8-Queens, 8-Puzzle, Road Map, and Traveling Salesman problem [91], the Counterfeit Coin problem bears most resemblance to the configurations in Conceptual Design System established in Chapter 3. Essentially, conceptual design problems and Counterfeit Coin problem can both be approached using problem reduction technique.

In a chess game, Newell [30], as part of his theories of cognition, identified a non-random thought pattern, called progressive deepening: *search down in a depth-first fashion; then return all the way back to the initial situation; then search back down again following an old path for a while, before either going deeper or branching out to the side.* In the case of product design, one often keeps other unexplored alternatives in mind, while exploring into one selected alternative. Anytime when the current solution path ‘seems’ (based on the adopted heuristic) to have its potential falling below those unexplored alternatives, backtracking is initiated. In solving both chess problem [30] and conceptual design, progressive deepening patterns are observed in human. Without computer’s aid, limited short-term memory of human restricts remembering unexplored alternatives. However, with computer to aid in product conceptualization, complex and massive paths and tracks of thoughts can be registered or remembered.

Product conceptual design, like most other practical problems, requires an immense number of possibilities to determine an exact solution. The time required is often more than a lifetime [91]. What makes a problem a problem is not when large amount of search is required, but when large amount is required and the requisite level of intelligence is not applied [93]. Heuristics play an effective role in problem solving, such as in product design, by indicating a way to reduce the number of evaluations and to obtain product solutions within reasonable time constraints.

Product design’s solution is not expected to be an exact one. This may be due to inherent ambiguities in the problem statement, as well as the incomplete predefined constraints for solutions. Hence, product design does not have an exact *goal state*, which the so-called General Problem Solver (GPS) [30, 86] has been designed to search for, in a state-space representation. Much research has developed algorithms that are suitable for searching in state-representation spaces, where start-state, operators and end-states are

well defined. These types of heuristic algorithms are thus not applicable to product design problems.

Like most game problems that heuristics can handle, product conceptualization is associated with combinatorial explosion due to the permutation of alternatives. Product design, as an ill-defined problem [62], does not have a finite solution space. Ill-defined problems may have more than one possible solution [96]. Product design is unlike chess game problems, which the latter may also have problem of combinatorial explosion, but at least, the solutions are still finite in terms of possibilities. Due to the infinite solution nature of product design, it is closer to the “generate and test” paradigm of AI [93] than the “split and prune” paradigm inclined towards OR (Operations Research). In the former paradigm, new objects are created in process of problem solving (as in product design), while the latter, objects are preexisting, which non-solutions are pruned [91]. According to Pearl [91], the OR approach is more effective in guarantee completeness and optimality. However, as product design is inclined towards creating new objects and knowledge, optimality achievable by OR methods may not be necessarily realized in product design problems.

In problems where best quality solution is the only concern, optimization techniques can be used. Pure traveling salesman problem (NP-hard) is one such example. However, problems that have time and other resources as constraints are likely to accept relaxed quality solution. In product conceptualization, designers may be forced to relax optimality and settle for “good enough” solutions, after expending certain amount of search effort. In heuristic literature, this is described as semi-optimization [91]. Research in cognitive psychology, however, has long proven that in their natural thinking, human rarely strived for optimum solutions, but rather for satisficing solutions [27]. According to Simon [97], satisficing solution is one that performance level exceeds certain threshold of acceptability.

As reviewed in Chapter 2, product conceptualization process is conducted along the *dimension of incremental detailing*. Product designers conceptualize from abstractions to detail. On the other hand, most problems that heuristic algorithms attempt to solve, such as the Traveling Salesman problem and Road Map problem, proceed along the *dimension of incremental quantity*, such as distances for the two cases mentioned. Algorithms (such as A*, HPA, B, C, BF* and D) for solving such problems have *path length functions* which add up quantities of all sub-solutions to get the overall solution [94]. For this difference, the path length functions of heuristic algorithms cannot be directly applied to product conceptualization problem.

In product design, problems are often decomposed into sub-problems, which are to be collectively solved, before the main problem can be considered solved. In many heuristic algorithms, such logic is not found. Specifically, algorithms meant only for OR graph is not directly suitable for product design use. In an OR graph, all nodes are OR type [91]. Graphs with both AND as well as OR nodes are called AND/OR graph. Another category of algorithms that operates on AND/OR graphs would be more appropriate for conceptual design problems.

In the next section, an overview of heuristic algorithms is presented, with GBF (general best-first) algorithm in more detail. It should however be noted that the role of such heuristic algorithm in this research project is not the same as in most research efforts, e.g., for intelligent knowledge search automation or for knowledge extraction on expert or knowledge-based system. Newell [30] has clarified that: *There are 2 separate searches going on in intelligence. One is **problem search** in problem space. The other is **knowledge search**, which is the search in the memory of the system for knowledge to guide the problem search.* In this research project, the application of heuristic algorithm is according to the former case distinguished by Newell [30]. That is to facilitate exploration of product design concepts.

4.2 Overview of Heuristic Algorithms

Designing heuristic search algorithms has been a core concern of AI research. It has been discussed in literature that the two tenets of systematic search of ideas or concepts are: (1) leave no stone unturned, and (2) not to turn over any stone more than once [91]. These principles are the philosophies behind breadth and depth-first search algorithms, which hunt for solutions in a systematic, albeit brute manner [87]. Breadth-first search guarantees to terminate with solution (if it exists). The first of the two tenets is not practical in real-world applications, such as in the case of product design. This is because, in product design, solutions are often limitless in product's conceptual level, on top of the extremely tight time constraints present in every product development cycle today. That is, breadth-first search is systematic, but not the most efficient while trying to *leave no stone unturned*. The proposed depth-bound criteria of depth-first search may prevent endless progression, but may also unwisely overlook good solutions beyond the bounded depth.

Hill-climbing strategy [95] is designed to engage problems in the immensely large search space, cleverly ignoring the first tenet of systematic search. It selectively explores the solution space based on some heuristic merits. One imperfection of this strategy is that the maxima may not be the solution found, instead, only a local optimal solution may be found by climbing the wrong hill, without any function to switch hills. In short, it finds the steepest ascent to only optimize locally. In view of this, a strategy closer to human natural cognition is developed. That is to keep a record of the exploration frontier, or remembering alternatives along the past paths, so as to enable backtracking for development of other paths. This is commonly known as Best-First algorithm, or simply BF [91]. Two points setting BF apart from hill-climbing algorithm is that BF keeps a record of previous alternatives, and it constantly considers all these alternatives, no matter where in the space the search is located.

A basic BF strategy outlines the common characteristics of all other heuristics algorithms. It operates in state-space representation graphs, where all nodes are OR type. A General Best-First (GBF) Strategy is designed to explore AND/OR graphs [91]. In an OR graph, solution can be represented by tip of a path. The tip solely is adequate to ascertain the validity of the solution. On the other hand, in the case of AND/OR graphs, which GBF strategy is capable of handling, the complete solution is represented by a solution graph (usually a sub-graph), formed by a set of nodes. These two fundamental types of strategies are used to solve different types of problems.

From BF and GBF strategy, BF* and GBF* are evolved respectively, by delaying termination to ensure optimality [91]. Specialized algorithms such as Z*, A* [91], AO and AO* are additionally proposed. Among these algorithms, Z, Z* and A* are based on BF [95], hence limited for searching only OR graphs. All specialized BF and GBF strategies consist of specialized functions.

According to Pearl [91], state-space representations or OR graphs are suited for problems which their solutions can be specified in the form of a path or a single node. On the other hand, AND/OR graphs representations are suited for problems solved via reduction using trees structure. For the purpose of this research, GBF strategy is reviewed, for its capability to handle Graph G , which is an *AND/OR tree graph* as described in Chapter 3. GBF consists of fundamental techniques to explore AND/OR graph; it has not restrictedly specialized with functions yet. Since it is non-specialized, this project is able to tailor certain functions pertaining to the needs of conceptual design problem solving.

In the following two sub-sections, notations used in GBF algorithm, and the algorithm itself will be introduced.

4.2.1 Notations and Assumptions of GBF Algorithm

Following are the notations and related assumptions used in GBF heuristic strategies [91], introduced for the purpose of reviewing the highlighted algorithm. The definitions in this section do not constitute the definitions of the proposals of this research, unless later formally defined.

1. An AND/OR graph, G , consist of nodes, n , with a start node, s .
2. **Node generation** is the computing of node's representation code from its parent.
A new successor is said to be **generated** and its parent is said to be **explored**. An **expanded** node is a node which has its successor *all* generated. **Pointers** are normally set up from each generated successor back to its parent node.
3. The set of nodes in the graph can be at any given time be divided into 4 disjoint subsets: (1) nodes that have been expanded, (2) nodes that have been explored but not yet expanded, (3) nodes that have been generated but not yet explored, and (4) nodes that are still not generated.
4. Nodes that were expanded are called **closed**, while nodes that were generated and are awaiting expansion are called **open**. Two separate lists called **CLOSED** and **OPEN** are used to keep track of these two sets of nodes.
5. At a certain phase of the search, the **explicated portion** of the underlying AND/OR graph G is represented by the subgraph G' of G .
6. Candidates of solution, or the **solution base**, is subgraph G'' of G' .
7. The promise of a solution base, G'' , is estimated numerically by a heuristic evaluation function f_1 .
8. The promise of a node, n , is estimated numerically by a heuristic evaluation function f_2 .

4.2.2 The General Best-First Algorithm

The best-first search algorithm GBF for AND/OR graph is extracted from literature [91]:

The GBF Algorithm:

1. Put the start node s on OPEN.
2. From the explicit search graph G' constructed so far, (initial just s), compute the most promising solution-base graph G_0 , using f_l and the heuristics h provided in Step 4.
3. Using f_2 , select a node n that is both on OPEN and in G_0 ; remove n from OPEN and place it on CLOSED.
4. Expand node n , generating all its immediate successors, and add them to OPEN and to the search graph G' with pointers back to n . For each successor n' , provide heuristic information h that characterize the set of solution graphs rooted at n' .
5. If any successor n' is a terminal node, then
 - a. Label node n' “solved” if it is a goal or “unsolvable” if not.
 - b. If the label of n' induces a label on any of its ancestors, label these ancestors “solved” or “unsolvable” using the labeling procedure that follows.
 - c. If the start node is solved, exit with G_0 as a solution graph.
 - d. If the start node is labeled “unsolvable”, exit with failure.
 - e. Remove from G' any nodes whose label can no longer influence the label of s .
6. Go to Step 2.

The “Solve” Labeling Procedure

1. A terminal node is labeled “solved” if it is a goal node (representing a primitive subproblem); otherwise it is labeled “unsolvable” (representing a subproblem that cannot be reduced any further).

2. A non-terminal AND node is labeled “unsolvable” as soon as *one* of its successors is labeled “unsolvable”; it is labeled “solved” if *all* its successors are “solved.”
3. A non-terminal OR node is labeled “solved” as soon as *one* of its successors is labeled “solved”; it is labeled “unsolvable” if *all* its successors are “unsolvable”.

The Frontier Nodes, n'

In Step 4 each node n' on the frontier of G' is assumed to be characterized by information regarding the set of solution graphs rooted at n' . This set (possibly empty) consists of subgraphs $g(n')$ of G where each $g(n')$ satisfies:

1. n' is the root node of $g(n')$.
2. n' can be labeled “solved” by applying the labeling procedure to $g(n')$.

Of course, since n' is not yet expanded, the subgraphs $g(n')$ are not available explicitly and can only be assessed using heuristic information from **outside of G** . The role of this information is to assess both the difficulty of finding a solution for the subproblem represented by n' and the quality of the solution when found [91].

4.3 The Conceptual Design Heuristic Algorithm

Product conceptual design can be seen as a type of problem that heuristics are applied implicitly in practice. Making the process of concept development systematic and explicit via the aid of computer, that is, via a CACD system, is very much anticipated currently in the industry [4, 6, 7]. This research attempts to apply the ideologies, as well as the established formalities of heuristic problem solving algorithm to the area of design conceptualization problems.

In most problems of practical interest, the full representative graph relevant to the problem is often too large for the computer to consider completely [91]. In solving these problems, a technique in terms of rules should selectively extract data from the database,

as well as from human, hence generate the graph from the start node in any desired direction incrementally. This generated graph is the *explicit portion* constructed, keeping most other parts unexplored. The objective of a guided exploratory algorithm is to find a solution graph by explicating only a minimal portion of the implicit graph.

GBF algorithm is strategically adapted to act as a core process of the Conceptual Design System (CDS), as defined in Chapter 3. The adapted GBF algorithm, called Conceptual Design Heuristic (CDH) algorithm, operates on the AND/OR graph G , in the Design Space Module, which essentially contained structured data defining design objects and their inter-relationships for respective design problems. As seen in the framework of CDS (Figure 3-2), processes installed in the Design Exploratory Module would retrieve past design data from the database (long-term memory), and elicit designers' input via the user interface. The user inputs would include forms of conceptual design heuristics, inherent in designers (and naturally absent in computer), based on their experiences and tacit knowledge. As mentioned in the last paragraph of the previous section, Pearl [91] explained that GBF algorithm expects heuristic information outside graph G . That is, in the context of CDS, CDH algorithm would obtain heuristic information from designers via the user interface.

Section 4.3.1 states the notations and assumptions of the CDH algorithm. Four sub-functions of the CDH algorithm are defined in Section 4.3.2, prior to the presentation of CDH algorithm itself in Section 4.3.3. Section 4.3.4 raises the issue of CDH algorithm retaining all nodes in G , as opposed to the philosophies underlying GBF algorithm.

4.3.1 Notations and Assumptions of CDH Algorithm

The proposed conceptual design heuristic (CDH) algorithm operates on a Design Space Framework (graph G) as established in Chapter 3. Recalling Definition 3.14, graph G is a quadruple where s_0^0 is the start node.

Definition 4.1: At certain phase of the product conceptualization process, the *explicated or explored portion* of graph G is represented by the subgraph G' of G . A candidate solution, or a *solution base*, is subgraph G'' of G' . The most promising solution candidate G'' is denoted as G_θ . Three separate conditions hold for G'' at all times.

1. G'' contains the start node s_0^0 .
2. If an expanded node n in G'' has its successors along dimension X , then all n 's successors are also in G'' .
3. If an expanded node n in G'' has certain successor with siblings along dimension Y , then only one of the siblings is in G'' .

Definition 4.2: Let a *Context* represents a set of nodes generated in one CDH algorithmic cycle. This set of nodes contains C , P and S in sequence. Due to the recursive nature of object relationships (Axiom 3.1), *Contexts* continue to stack from below. Figure 4-1 shows a case of a *Context*. Here, a *Context* is a description of how a sub-problem (i.e. the parent node of the Context's C node) is solved.

Figure 4-1 An Instance of a Context

Definition 4.3: For any node generated from the expansion step, it will be a terminal node, if it is either: (1) unable to produce any successor, its heuristic value will be assigned $-\infty$, or (2) if no further concept detailing is required (in other words, no successor node is necessary), the terminal node shall be labeled ‘*Solved*’.

In this CDH algorithm, like other heuristic algorithms, nodes that are expanded are called *closed*, while nodes that are generated and are awaiting expansion are called *open*. Two separate lists called **CLOSED** and **OPEN** are used to keep track of these two sets of nodes in this approach.

4.3.2 The Sub-functions of CDH Algorithm

Four sub-functions are prescribed to support the CDH algorithm, viz. the heuristic function h , evaluation functions f_1 and f_2 , and lastly the Solved-labeling function.

4.3.2.1 The Heuristic Function

Araujo et al. [9] suggested that conceptualizing product solutions largely depends on people and on their tacit knowledge. In the same vein, Liu et al. [14] asserted that designers implicitly discard infeasible solutions based on their experiences. As in the case of QFD (Quality Function Deployment), filling ranged numbers into matrix to reflect team’s tacit knowledge helps to build team consensus as well as to facilitate team’s decisions. In effect, these numbers represents the team’s domain knowledge.

Definition 4.4: Let the heuristic function $h(s_k^l)$ represents the *promise* of solution s_k^l has to its (only) parent.

$h(s_k^l)$ captures the design team’s confidence level of the solution s_k^l , based on its next level of detailness. When the details or child nodes of s_k^l are yet to be explored, the

value of $h(s_k^l)$ is derived from the team's experiences and their tacit domain knowledge on s_k^l . This measurement, implicit in practice, motivates further exploration or termination of ideas. The value of $h(s_k^l)$ propagates and contributes to the collective worthiness of all its ancestors, from its direct parent to the graph root s_0^0 . Such propagation is enabled by the Merit Ratio equation defined in Equation 4.3. In the case where s_k^l is considered for further exploration into details, its worthiness (or Merit Ratio (Definition 4.5)), beyond the concreteness of its initial heuristic h value, is tuned and progressively affirmed by the heuristic values of its descendents. The merit ratio of any non-terminal node is influenced by its descendants' best alternatives (if any alternative is present) in the explicated graph G' . The best alternative is defined as the one with the highest merit ratio. This form of mechanism is derived from the fact that *promises* (as used in Definition 4.4) can only be evaluated retrospectively. Despite of the uncertainty of *promises*, they are extremely useful in problems where decisions have to be made in vague situations to further support or refute itself. The role of heuristic in this work is similar, that is to make progressions in conceptual design, based on the rules-of-thumb.

The value of h has the corresponding classified denotation as follows.

$$h(s_k^l) : \begin{cases} h(s_k^l) > 1 & \text{If } s_k^l \text{ promises more than it is required.} \\ h(s_k^l) = 1 & \text{If } s_k^l \text{ promises as required.} \\ 0 < h(s_k^l) < 1 & \text{If } s_k^l \text{ promises less than it is required.} \\ h(s_k^l) = -\infty & \text{If } s_k^l \text{ is unpromising.} \end{cases} \quad (4.1)$$

According to Al-Ayyoub and Masoud [98], the determination of the estimate h is relied on heuristic information available from the problem domain. Heuristic h value can be derived from abstract knowledge. For instance, if motor X is *conceptually* known to produce 80% of it required power, it can be assigned $h=0.8$ (this is if the level of power is the only requirement). It should be noted that the knowledge dealt in conceptual design phase is often tuned along the process. Knowledge exists in levels of abstraction or “conceptualization”, as opposed to the constraints in detailed design, which have to be distinct. Hence, the 80% power of Motor X can be accepted as a guide, even if not highly accurate, in the context of heuristic methodologies. When the design of Motor X gets more detailed, the judgment on Motor X may be morphed. Alternatively, a scoring system can be prescribed in the team environment to arrive at a weighted average h value.

It is assumed that the value of h is derived from designers’ best knowledge and experiences. (See Section 6.2.1 for discussions on this assumption). By definition of heuristics, h needs not be exact, but should be an estimated figure guided by some criteria [81, 91, 96, 98], as required by the heuristic algorithms. It is the virtue of heuristic methods to solve problems, when adequate concrete data is not available (usually due to time/cost constraints), as in the nature of product conceptualization problems here. Like all other heuristic applications, rules of thumb are based on to make decisions, which is actually what human does in situations when complete information is lacking.

The root node of the graph, or the highest-level solution has an assumed heuristic value of 1.0. This assignment states that the design starts off with an overall concept that promises to fully satisfy the design problem. For instance, in a project to design a bicycle, a *conceptual* “ideal bicycle” promises to fully solve the defined problem. Thus,

$$h(s_0^0) = 1.0 \quad (4.2)$$

4.3.2.2 The Evaluation Function f_l

The explicated portion of graph G , G' , consists of sets of solution base graph, so-called G'' , each of which is a set of solutions to the problem. The most promising G'' is denoted as G_θ . With every expansion exercise in a CDH algorithmic cycle, G_θ is refreshed, based on the new data available in G' . Function f_l computes the merit ratio (Definition 4.5) of every solution node from the bottom to the top. Starting from the generated tip nodes, the computation propagates upward impinging on their respective ancestors' merit ratios. The merit ratio of any non-terminal node is based on its descendants' best alternatives (if any alternative is present) in the explicated graph G' . The best alternative is defined as the one with the highest merit ratio. G_θ is physically a set of structured data (a quadruple $G = (N, E, \Phi, \Gamma)$ from Equation 3.16) consisting of the selected alternatives in every option junction in G' . As G_θ varies with each algorithmic step, it will overwrite G_θ of the previous algorithmic cycle.

Definition 4.5: Merit ratio (MR) of $s_{k_1}^l$ is the degree of promise $s_{k_1}^l$ provides to its parent. $MR(s_{k_1}^l)$ measures the heuristic performances of the solution based on the proportional sum of merit ratios of its constituents. Mathematically,

$$MR(s_{k_1}^l) = \begin{cases} h(s_{k_1}^l) & \text{If } s_{k_1}^l \text{ is an OPEN node.} \\ h(s_{k_1}^l) \sum_i \Lambda(c_i) \sum_j \Lambda(p_j) \sum_k \max_l [\Lambda(s_k^l) MR(s_k^l)] & \text{If } s_{k_1}^l \text{ is a CLOSED node;} \\ & c_i \text{ is a child of } s_{k_1}^l; p_j \text{ is a} \\ & \text{child of } c_i; \text{ and } s_k^l \text{ is a child} \\ & \text{of } p_j. \end{cases} \quad (4.3)$$

where $\Lambda(n)$ is the contribution fraction of node n . It is the share of impact n has on its parent amongst its siblings along X dimension of D . (i.e. $0 < \Lambda(n) \leq 1$)

Definition 4.6: Function f_l provides the most promising solution graph G_0 , given the explicated graph G' and the merit ratios (MRs) of solutions in graph G' .

$$f_l : [G', \text{MR}(s_k^l)] \rightarrow G_0 \quad (4.4)$$

As G_0 varies with each algorithmic cycle, its values are updated every cycle.

4.3.2.3 The Evaluation Function f_2

It is argued that since each and every node in a solution base graph (AND graph) is relevant to the overall solution (whereby all its tip nodes on the OPEN list must eventually be expanded), the order of expansion is rendered immaterial [91]. However, it may be sensible to first expand the node that has the highest chance of revealing an impasse, so as to possibly either declare the problem “unsolvable”, or to prompt seeking of alternative solutions at the earliest possible time. In the context of product conceptualization, this argument is similar. Amongst the constitutions of a particular product concept, where every component is a requisite, the logical priority is to probe and identify the most fallible component. If instead designers jump into developing the most exciting or the perceived most promising component, they may in certain cases, uncover the weakest link, or worst, broken link only later in the design process. This may unfortunately render parts of their developmental efforts irrelevant and wasted.

The purpose of function f_2 is to single out a node for the expansion exercise in every CDH algorithmic cycle. The chosen node n^* should be from the OPEN list as well as being part of the ascertained G_0 . For the arguments described above, probing the weakest element to refute G_0 's superiority as soon as possible is the rationale for evaluation function f_2 .

Definition 4.7: Function f_2 identifies an optimal node n^* for expansion in the CDH algorithm.

$$f_2(s_k^l \in G_0 \cap \text{OPEN}) = \min_k [h(s_k^l \in G_0 \cap \text{OPEN})] = n^* \quad (4.5)$$

As the heuristic value $h(s_k^l \in G_0 \cap \text{OPEN})$ represents the designers' confidence level on the next level solution, the one with the least value is the one that has the highest possibility of revealing the lowest h or even an “unsolvable solution” (i.e. $h(s_k^l) = -\infty$) in the next level. For this reason, the solution with the minimum h is selected as the next node for expansion.

4.3.2.4 The Solved-labeling Function

Solved-labeling function is invoked by CDH algorithm when any of the newly explored nodes is declared “solved” (i.e. $solve_status=1$). It signifies that further solution depth is not required for that particular node. Designers should moderate the depth probed in the product conceptualization stage. As a guide, too deep would render the process as detail design, where procedures would have been different [17]. On the other hand, conceptual freeze at a shallow depth may not have considered issues adequately, consequently leaving too much unconstrained aspects for the processes in detail design stage. This may cause reworks and additional iteration back to conceptualization phase, therefore reducing product developmental efficiencies.

Another scenario when a solution node may be declared “solved” is when the solution is *out-sourced*. In this case, further details of the out-sourced solution can be generated by the vendors, and submitted to the principal design team for the merging of the partial solutions.

This function takes in a node as an operand, edits the OPEN list, and propagate the “solved” label bottoms up in G' . Upon exit of this function, the CDH algorithm would continue from the point of suspension.

Definition 4.8: Solved-labeling Function is recursive. It is defined as:

1. Remove the operand, its siblings along dimension Y (its alternatives), and all their descendents if any from OPEN.
2. *If* all of the operand's siblings (if any) along dimension X have $solve_status=1$, update the parent of the operand with $solve_status=1$ and let this newly labeled parent be the operand. Go to Step 1. *Else* exit to Step 5 of CDH algorithm.

4.3.3 The CDH Algorithm

The CDH algorithm is assumed to be installed in a system, where it exchanges data with a user-interface (UI) and a database sub-system. The Conceptual Design Heuristic (CDH) algorithm is graphically depicted in Figure 4-2 and is defined as follows.

1. Put the start node s_0^0 on both OPEN and G_0 .
2. Using f_2 , select a node n^* (if initial, just s_0^0). Remove n^* from OPEN and place it on CLOSED.
3. Expand node n^* - generating its immediate *Context* (Definition 4.2) successor by using data from the UI and/or database. Add all generated nodes, as well as their relational pointers (leading back to n^*) to the search graph G' . Place all leaf nodes on OPEN.
4. For each new leaf node s_k^l , from UI, acquire heuristic information h that characterizes the set of solution graphs rooted at s_k^l . Let new leaf node s_k^l 's $solve_status(s_k^l)=1$, if instructed by UI.
5. If any of the new leaf node s_k^l on OPEN has been marked $solve_status(s_k^l)=1$ in Step 4, invoke Solved-labeling function with s_k^l as the operand. Else go to next step.

6. Base on the explicit search graph G' constructed so far, compute the most promising solution base graph G_0 using function f_l and the heuristics h provided in Step 4. Refresh G_0 .
7. If $solve_status(s_0^0) = 1$, exit with G_0 as the solution graph. Else if $MR(s_0^0) = -\infty$, exit with failure. Else go to Step 2.

Figure 4-2 Flow Chart of the CDH Algorithm

4.3.4 Nodes Retaining for Further Applications

In heuristic algorithms, nodes that are deemed not contributive from a point onwards are removed to reduce computational and memory load (see Step 5e of the reviewed GBF algorithm in Section 4.2.2). In GBF, “unsolvable” nodes, and their siblings under the AND parents are deemed removable after they have helped to label their parents “unsolvable”. Similarly, “solved” nodes and their siblings (alternatives) under the OR parents are removed instantly from G' once they have facilitated in labeling their parents “solved”.

In product design problems, “unsolvable” and “solved” nodes should however not be discarded from G' . “Unsolvable” nodes could be resurrected for further propagations in the future when newly found solutions appear due to new discoveries or technological improvements. “Unsolvable” nodes are also valuable for further analysis, especially if CDH algorithm exits with failure (Section 6.2.2 describes the possible exploitations of the “unsolvable” nodes). Similarly, “solved” nodes should also remain in G' , so as to present the final solutions to the designers. For the above reasons, the typical step that removes these useful nodes found in heuristic algorithms (GBF’s Step 5e) is not included in the CDH algorithm.

Having omitting the node removal step as mentioned above, it should be noted that it was the presence of Step 5e in the GBF algorithm that ensures the function f_2 of GBF would never pick a “unsolvable” node and its siblings (which is inducted irrelevant) under an AND parent, nor will it pick “solved” node and its siblings under a OR parent for the node expansion exercise. In CDH algorithm, the step to remove nodes not qualified for expansion is not included for the reasons given in the preceding paragraph.

The current dilemma is that the CDH algorithm preserves the irrelevant nodes for further uses, but need to know that they are not eligible for further expansion. This dilemma is solved via two ways as follows:

1. The function f_2 's criterion for selecting an expansion node (which is common in both GBF and CDH) is “it must be in the range of $\text{OPEN} \cap G_\theta$ ” (Equation 4.5). Hence selectively removing the “solved” node and its irrelevant siblings along Y dimension (its alternative) from only OPEN, (instead of deleting indiscriminately from G) will solve part of the problem. This removal process is additionally embedded in the Step 1 of the Solved-labeling Function proposed in Section 4.3.2.4.
2. For “unsolvable” nodes, CDH assigns $h(s) = -\infty$ to them. By the virtue of CDH's function f_1 defined above, unsolvable nodes will not be found in a valid G_θ (except in case if $\text{MR}(s_0^0) = -\infty$ when CDH algorithm will exit with failure). If the “unsolvable” nodes cannot be found in a valid G_θ , CDH's function f_2 cannot possibly select them for expansion due to the criterion of “ $\text{OPEN} \cap G_\theta$ ” in Equation 4.5.

With the above 2 methods, CDH algorithm (as opposed to GBF algorithm) will be able to retain all its nodes for further applications, as required by the product conceptualization types of problem.

4.4 Discussions on CDH Algorithm

This chapter has formulated a heuristic-based conceptual design exploratory algorithm, based on the second hypothesis made in Section 2.3.3. The following sub-sections discuss the role and significances of the CDH algorithm.

4.4.1 The Role of CDH Algorithm

The CDH algorithm presented in Section 4.3 facilitates as a core procedure to build up conceptual product models, based on the data structure defined in Chapter 3. It gradually

elaborates the conceptual product model from abstractions to details. Through the progressive applications of constraints, the solution is crystallized as the result. This procedure is analogous to the steps prescribed to construct a 3D product model in conventional CAD systems. For instance, to create a 3D cube model in a conventional CAD system, the typical steps are: appoint a reference plane, sketch on the plane, followed by specifying the protrusion dimension. These users' supplied data are in accordance to the data structure of the representing model (e.g. feature-based model) prescribed in that CAD system. An algorithm in such a CAD system specifies the sequences to build the 3D cube model. In this vein, CDH algorithm together with the DSF (the knowledge organization structure defined in Chapter 3) can accordingly be applied in a computer-aided conceptual design (CACD) system.

4.4.2 Managing Design Space

As raised by Liu et al. [14], solution space has to be managed properly, keeping it under control. Impedance arises when the number of branches or solution alternatives gets too numerous at various levels of abstraction. Without the aid of computer, designers' inherently limited cognitive capacity may not be able to fully handle assessment tasks on complex solutions with overwhelming details. Cognitive overloading may affect designers' judgment, thus increasing the possibility of producing sub-optimal or incorrect conclusions of the assessment tasks.

Selection of concepts is necessary in due time, to restrict their number from getting too large when comparisons becomes tedious, if at all possible. Concept selection is a process where constraints are being assumed [15]. Sim and Duffy [15] asserted that the aim of assuming constraints (or selections) is to limit the exploration of design space. Indeed, selecting a concept for further detailed development is deeming further exploration of the unselected alternative concepts unnecessary (not immediately

necessary). In the approach of CDH algorithm, constraints are being temporarily applied during concept exploration process, that is, options are kept opened (literally, kept in the OPEN list). CDH algorithm keeps the design space as small as possible (without compromising the richness of solution space) by selectively expanding design frontiers heuristically.

4.4.3 Representing Solution Insights

Researchers have highlighted the need for designers to know if their lines of thoughts they are pursuing are worthwhile, or if their efforts would be better spent following a different path of solution [33]. During conceptual design, solution candidates are likely to be highly abstract, without detailed characteristics. Ulrich and Eppinger [34] raised the concern: *“how can the team chooses the best concept, given that the designs are still quite abstract?”* Liu et al. [14] posed a similar challenge: *“Solutions represented at an abstract level can be hard for the designers to understand. It is a question of how to screen an abstract solution space”*. To handle this challenge, Holland et al. [16, 17] recommended the designers to specify the scheme of solutions to a detailed extent, which permit it to be evaluated against the constraints in the design specifications, and compared against any alternative schemes that have been developed. Similarly, Liu et al. [14] suggested that solutions with less potential should be pruned as soon as they are made detailed enough to be considered against the major requirements. For the sake of argument, say when 20 alternatives are available, the designers would intuitively not generate all 20 sets of adequately detailed solutions for comparison, especially if time disallows them to do so. They would probably pick the ones that seem to have the best potential, based on their tacit knowledge and experiences. At the end of the day, designers still need to perform selection among the abstract solutions. That is, during concept development, designers need to evaluate alternatives based on concepts, which are still abstract.

In practice, designers probe details (both explicitly and implicitly) of each alternative before the act of selection. For instance, when using *concept combination table* [1], the team may think (implicitly) or discusses (explicitly) into deeper details of each alternative solutions before selecting from the table. Liu et al. [14] commented that individual designers discard infeasible solutions implicitly based on their experiences. The scoring or appraisals of concepts by each designer is based on their *insights* into the individual concepts. Feldman defined *insight* as the *awareness of relationships among elements, due to prior experiences* [99]. The unique insight that each designer possesses is derived from their knowledge, experiences and judgments found tacitly within them. This may account for the differences of viewpoints regarding the assessment of concepts among the designers. To clarify these differences, and to create a consensus, verbal exchanges of individual designer's tacit judgments are usually attempted, sometimes ineffectively, probably due to the informal and subjective language used.

To build team consensus while concepts are still abstract (often the case in conceptual design phase), CDH employs a heuristic parameter h to reflect the promises of respective alternative solutions. It can be said that CDH algorithm senses the viability of each solution via h . The value of this parameter is estimated, based on the team's *insights*. After a solution is selected (based on h) for further exploration, its detailness would progressively increase. Along this process, the promises of the selected solution may drift from that as initial perceived, due to the differences between the explicated insights (i.e. the explored details) and the insights earlier based on. Due to these differences, the merit ratio (MR) of the initial selection is adjusted via propagations, possibly firing a backtracking for alternative solution exploration in a higher abstraction level. This mechanism of CDH algorithm identifies that exploring selected solutions into details may shed light on previously unknown pros and cons, which may affect the previous selection results.

4.5 Chapter Summary

This chapter has in Section 4.1 explored the applicability of heuristic methods on conceptual design problems. From the studies, it is deemed that an adaptation of the GBF algorithm can be applied on the problem. A range of heuristic algorithms is reviewed with GBF algorithm examined in detail. According to the second hypothesis presented in Section 2.3.2, an intelligent design exploration method, so-called CDH algorithm, is devised based on the principles of heuristic. It is formulated to operate on the Design Space Framework (graph G) established in Chapter 3. As discussed in Section 4.4, CDH algorithm can potentially be employed to facilitate conceptual design processes in a CACD system environment. It serves to guide the designers in managing the inherently vast design space.

– Chapter Five –**Case Studies**

Three cases, incremental in details, are presented in this chapter in bid to demonstrate the propositions made in this work. Through the first case, designing a simple disposable coffee cup, the concept of the Design Space Framework (DSF) is clarified. The second case involves the conceptualization of a more technical product – a remotely operated vehicle. This case aims to exhibit the roles of CDH algorithm and its four sub-functions. The third case is on the conceptual design of a bicycle brake system. It focuses on presenting the technical mechanisms within the Conceptual Design System (CDS) in details.

5.1 Conceptualizing a Disposable Coffee Cup

To illustrate the concept of DSF, a simple case study on the conceptualization of a disposable coffee cup is presented in this section. The product was codenamed *Auburn*. It was required to be stylish, environmentally friendly, able to keep the content warm, and portable. As such, four requirements listed below formed the child nodes of Auburn (i.e. s_0^0), as shown in Figure 5-1.

1. c_1 : Stylish image
2. c_2 : Environmentally friendly
3. c_3 : Ability to keep beverage warm
4. c_4 : Portable

The designers generated three product requirements, viz. elegant shape (p_1), beautiful motif (p_2) and appropriate colour scheme (p_3), to satisfy the *stylish image* (c_1) required by the customers. Four solutions were proposed to satisfy p_1 , as shown in Figure 5-2. The designers proposed three possible types of motifs for Auburn, as sketched in Figure 5-3. Two types of colour schemes to address the requirement p_3 were specified. They are (1) s_3^1 (classic- shades of brown) and (2) s_3^2 (trendy – shades of blue and silver).

Figure 5-1 Graph G' of Auburn
 s_1^1 - Slim s_1^2 - Stout s_1^3 - Bamboo s_1^4 - Curvy
Figure 5-2 Solutions for Product Requirement p_1

s_2^1 - Fuzzy s_2^2 - Sunny s_2^3 - PopFigure 5-3 Solutions for Product Requirement p_2

To address c_2 (environmentally friendly), the designers deemed that the product has to be made of readily recyclable material (i.e. p_4). The design team jotted several suitable recyclable materials, which include: (1) s_4^1 : Polypropylene (PP), (2) s_4^2 : Polystyrene (PS), (3) s_4^3 : Polyethylene Terephthalate (PET), and (4) s_4^4 : Certain food-grade paper. It is noted that solutions such as s_4^4 can be relatively abstract. Incomplete information in conceptual design stage is known to be common, and should be tolerated in the any CACD system. In this case, s_4^4 may later be further concretized, in accordance to Axiom 3.1 (see Chapter 3).

To minimize heat loss from the cup (i.e. p_5), the designers proposed several alternative ideas as follows: (1) s_5^1 : Insulative cup material, (2) s_5^2 : Insulative cup sleeve, and (3) s_5^3 : insulative sleeve over cup made of insulative material. The design team asserted that other than having the usual insulative characteristics (i.e. p_5) of coffee cups, having *heat addition function* (p_6) would be an innovation to sustain the high temperature of the beverage (i.e. c_3), if it is possible. Chemical heat source (s_6^1) was identified as the solution to p_6 . The designers were vaguely aware that the release of an activating solution into contact with a type of metallic compound would produce enough heat to warm up a cup of coffee. They drew up a conceptual sketch, as shown in Figure 5-4. As the finer details of this concept require expertise, this sub-solution was out-sourced to a vendor for further conceptualization.

Figure 5-4 Chemical Heat Source as the Solution s_6^1

To design Auburn for portability (c_4), two product requirements were identified: no spillage during transportation (p_7) and easy to carry (p_8). For product requirement p_7 , the designers deemed that a suitable lid would be the solution (s_7^1). As for p_8 , the team narrowed down to two alternative solutions – either a carrier tray (s_8^2) or an attachable handler (s_8^1).

Customer requirements of the attachable handler (s_8^1) were elicited from the client. They are: (1) c_5 : aesthetic, and (2) c_6 : modular. Accordingly, the relevant product requirements were generated, followed by the solutions. The conceptual sketch of the attachable handler applying on solution s_1^1 is shown in Figure 5-5. It may also be applied on other cup shapes, perhaps with some modifications to the cup and/or handler. As required by the requirements, the solution is simple, stylish and modular. The designers proceeded to generate the details of the carrier tray design before selecting one of these two solutions to satisfy requirement p_8 .

Figure 5-5 The Concept of the Attachable Handler

Sub-solutions of p_1 to p_5 were deemed to be detailed enough, as the results of the conceptualization phase of design. These solution nodes were therefore considered ‘solved’. This means that the team was not interested to further explore into these solutions, in this (conceptual) stage of design. Further exploration of solution s_6^1 was also terminated, as the team did not have the expertise to work on it. The conceptualization of the heat addition function was tasked to a vendor, who may submit their concept in the DSF format for merge with the principal graph G' . From this point onwards, the design team may further explore certain sub-solutions of Auburn into greater details, such as s_7^1 , the lid of Auburn. They may also expand graph G' by appending new requirements and/or alternative solutions to the existing ones, at various levels of abstraction.

The conceptual design of Auburn took form as graph G' , which captures the design outcomes and their rationales. It can be used for communication amongst designers, for presentation to the stakeholders of the projects, for design improvement (design iterations), and as the basis for detail design utilizing downstream CAD software. Coffee cup design is simple, relative to other sophisticated engineering designs. It is adopted here to concisely demonstrate the concept of DSF. However, when DSF is applied on more complicated engineering products, designers would need to generate relatively wider and deeper (i.e. more abstraction levels) graphs G' to represent the concepts.

Discussions

In this case study, the concept of DSF has been demonstrated. The following are some of the points noted from the case.

- An instance of DSF tolerating incomplete information is shown.
- DSF facilitates the out-sourcing of designs.
- The structure of DSF facilitates the ideation of innovative product (i.e. the cup warming function).

- Sketching, a dominant activity of conceptual design is coherent in the structure of DSF. Sketching is done with respect to the components of DSF, hence can be systematically carried out and documented.
- The structure of DSF facilitates communications among designers and other stakeholders.
- DSF is able to capture design outcomes and design rationales.

5.2 Conceptualizing a Remotely Operated Vehicle

A case study on the conceptual design of a more technical product, a remotely operated vehicle (ROV) [100], is presented in this section. The proposed Conceptual Design System (CDS), and its built-in algorithm (i.e. CDH algorithm) are applied in this case. This case aims to exhibit the roles of the CDH algorithm and its four sub-functions.

A design team was given the challenge to conceptualize a vehicle that is required to operate on the planet Mars. The ROV, codenamed *Marcie*, is required to travel across the Mars surface to collect various types of data over a period of three months. The team adopted CDS (see Figure 3-2), a CACD package to assist team members in their conceptualization process. It was assumed that this product is new; hence no previous data related to ROV is available in the database of the CDS. This means that the CDS in this case, was unable to suggest solutions based on the past designs. The user-interface of a prototype CDS is depicted in Figure 5-6.

The design team started a new project in the CDS environment. As in the Step 1 of the CDH algorithm, the designers let s_0^0 be *Marcie*, a solution in the form of a ROV. Corresponding to the Step 2 and Step 3 of the CDH algorithm, CDS elicited the customer requirements (CRs), the product requirements (PRs) and the solutions of *Marcie* from the system users.

Figure 5-6 User-interface of CDS

According to the user-needs, the designers identified four CRs (i.e. what the end users demand Marcie to accomplish) as follows.

1. c_1 : To move across Mars surface (rugged terrain).
2. c_2 : To sustain over the period of the mission (3 months) .
3. c_3 : To survey key indices of the environment on Mars.
4. c_4 : To receive and transmit data.

With references to the stated CRs, the design team discussed and stated the product characteristics that can lead to the satisfaction of the CRs. The specified product requirements (PRs), as listed below, are fed into the CDS.

1. p_1 : Ability to move over uneven surface.
2. p_2 : Stable, would not flip over.
3. p_3 : Ability to move up slope.
4. p_4 : Sustainable energy supply.
5. p_5 : Reliable system.
6. p_6 : Ability to sense the environment.
7. p_7 : Data processing capability.
8. p_8 : Data reception capability.
9. p_9 : Data transmission capability.

Figure 5-7 illustrates the relationships between the CRs and the PRs.

Based on the above listed PRs, the designers brainstormed to specify a set of solutions. The relationships between the solutions and the PRs can be seen in Figure 5-7.

1. s_1^1 : Tracked system
2. s_1^2 : Wheeled system
3. s_2^1 : Low center of gravity
4. s_3^1 : Broad-based
5. s_4^1 : High torque
6. s_5^1 : Gripping contact with ground
7. s_6^1 : Large power storage capacity
8. s_7^1 : Solar Panel
9. s_8^1 : Energy efficient system
10. s_9^1 : Redundancy of critical parts
11. s_{10}^1 : Hygrometer
12. s_{11}^1 : Thermometer
13. s_{12}^1 : Video Camera with light source
14. s_{13}^1 : Barometer
15. s_{14}^1 : Microprocessor
16. s_{15}^1 : Antenna
17. s_{16}^1 : Data transmitting system

Figure 5-7 Graph G' of Marcie

For each of the solutions listed above, the designers assigned a heuristic (h) value to it, indicating the degree of promise it can satisfy its corresponding PR. For the case of p_1 (ability to move over uneven surfaces), the designers asserted that s_1^1 (tracked system) promises higher capability to move across uneven terrain, as compared to s_1^2 (wheeled system). For the heuristic value of s_1^1 is higher than s_1^2 , CDS (according to Step 6 of the CDH algorithm) selected s_1^1 for inclusion in the graph G_0 , the most promising solution base.

Among the set of first-leveled solutions, the designers decided to out-source some of the modules, such as the hygrometer, thermometer and the barometer. Therefore, these sub-solutions were labeled ‘solved’ in the CDS environment. The conceptual designs of these sub-solutions were to be done by their vendors. Solution such as s_{12}^1 (video camera) was deemed unnecessary for further conceptualization work, due to its simplicity and availability off-shelf. It was therefore labeled ‘solved’ as well. The detailed design of s_{12}^1 would be continued using conventional CAD software that imports the CDS generated file.

Owing to the numerous physically interacting parts inherent in a tracked system, the designers had doubts on its reliability. They asserted that repair work could not be conducted once Marcie is sent onto Mars. Among all the sub-solutions in graph G_0 , s_1^1 had the lowest heuristic value. Therefore, the tracked system of Marcie (s_1^1) was singled out as the most fallible sub-solution in G_0 , by CDS (the Step 2 of CDH algorithm). Among the set of first-level solutions, s_1^1 was selected for further probe.

The designers proceeded to elicit opinions regarding the tracked system from the end-users, i.e. the ROV operators and the rest of the mission team. The end-users demanded the tracked system to have the following two qualities:

1. c_5 : Provides Marcie with high mobility

2. c_6 : High operability without repair support

As identified by Gero and Kannengiesser [66], one distinguish feature of conceptual designing is that not all requirements are known at the outset of a design task. From this case thus far, it might be seen that certain user-requirements cannot be elicited without the interim work of the designers. The end-users of Marcie was not likely to demand the specific qualities of the tracked system, without the designers recommending the idea of using such system in prior. More likely, they might simply state the general requirements regarding Marcie's mobility. In effect, CDS advocates user-centered design philosophy by providing the end-users with systematic junctions in the design process to input their requirements. DSF allows the opinions of the end users to be constantly sought along the entire product conceptualization process. This helps to produce the product that is close to the user requirements. From this perspective, DSF can be employed accordingly to facilitate customer voices elicitation for product development.

If past designs of a tracked system were available in CDS, the system would attempt to suggest the relevant PRs. However in this design case, it is assumed that this ROV is a new product being developed. Based on the demands (CRs) of the tracked system, the designers worked to specify the required product characteristics of the tracked system solution:

1. p_{10} : Speed of at least 8 km per hour
2. p_{11} : Low Vehicle Cone Index (VCI) ~ A low VCI equates to good soft-soil mobility, good performance on slopes, over sandy terrain, and over obstacle/gaps [101]
3. p_{12} : Reliable mobility system

Details, or the sub-solutions of s_1^1 (tracked system) were conceptualized based on the product requirements:

1. s_{17}^1 : A DC motor driving a pair of sprocket wheels via a gear reduction box.

2. s_{18}^1 : Two wide tracks to maximize the footprint area
3. s_{19}^1 : Numerous parts (support rollers, sprocket wheels, road wheel, tracks, idle wheel, track pins)

The tracked system was analyzed against the PRs. The design team found that though p_{10} and p_{11} were reasonably promised by sub-solutions s_{17}^1 and s_{18}^1 respectively, the inherent numerous parts of the tracked system may compromise the reliability of Marcie's mobility system. For this, the team assigned a low heuristic value to s_{19}^1 .

The low heuristic value of s_{19}^1 affected the Merit Ratio (MR, Equation 4.3) of s_1^1 (tracked system). CDS computed the new solution base G_0 where s_1^1 was excluded. Further exploration into s_1^2 (wheeled system) is henceforth recommended by CDS. This scenario exemplifies the possibility of the value of a solution fluctuating upon further exploration into details. If the value (i.e. MR) of one solution drops lower than its next best alternative, a backtracking would be recommended by the CDS.

The same set of CRs and PRs were used to further explore and conceptualize s_1^2 , the wheeled system. The designers generated the sub-solutions of the wheeled mobility system:

1. s_{20}^1 : Six DC motors driving six wheels with gear reduction boxes.
2. s_{21}^1 : Six wheels with larger diameter and width to increase footprint area
3. s_{22}^1 : Six wheels- simpler construction, less parts for greater reliability

Upon conceptualizing the wheeled system, the designers were satisfied by the sub-solutions. Higher heuristic values were given to the sub-solutions of the wheeled system, compared to the tracked system.

From this point onwards, the design team may further explore the sub-solutions of the wheeled mobility system, or any other nodes of graph G' into greater details. They

may also expand graph G' by appending alternative solutions to the existing ones, at various levels of abstraction. Appending an innovative solution to one of the graph tree branches may result in an overall innovative product (as discussed in Section 3.3.4). In addition, when the design team has approved the designs done by their vendors, they may henceforth merge them with their principal design in the CDS environment.

The above tasks are the common challenges of conceptual design, which may be done in an ad hoc and at times unaccountable manner. In the CDS environment, they may be formally (with accountability) and systematically performed. The conceptual design of Marcie takes form as a graph (G'), which captures the design outcome, and their design rationales. It can be used for presentation to the stakeholders of the projects, for design improvement (design iterations), or for the purpose of detailed design using downstream CAD software.

Discussions

This case has introduced the roles of the CDH algorithm and its four sub-functions. The following are some of the points noted from the case.

- The values of heuristic function h should reflect the team's consensus. This value helps CDS in computing the best configuration of the possible solutions.
- The case shows that not all requirements are known at the outset of a design task. DSF provides designers with systematic junctions along the design process to elicit and assimilate the relevant requirements.
- Merit Ratio enables CDS to recommend backtracking or design iteration. It also points to where it should backtrack and iterate.

5.3 Conceptualizing a Bicycle Brake System

This third case study aims to elucidate the working mechanisms of the proposed CDS. As such, the detailed progressions of the algorithm, the data, and the mathematical calculations are documented here. The design problem and the assumptions are introduced prior to the stepwise demonstration of the CDH algorithm.

The Design Problem

A bicycle, codenamed *Stallion*, is required to satisfy the customer group X . The design case started with an abstract solution s_0^0 (*Stallion*), promising to solve the given design problem. A sample of customer group X was identified for the purpose of eliciting customer requirements. By surveying this sample group, four highest-level needs of *Stallion* are established, namely: (1) safety, (2) comfort, (3) high efficiency and (4) ease of usage. They are respectively being assigned as c_1 , c_2 , c_3 and c_4 in the CDS environment.

In accordance to the proposed CDH algorithm, the CDS proceeds to establish the *Contexts* (Definition 4.2) for each of the customer requirements. Under customers' desire for 'safety' (c_1), two product requirements (p_8 and p_9) are identified. They are respectively 'quality braking system' and 'high presence'. The latter requirement needs *Stallion* to have obvious presences, and may have later generated solutions such as having audio horn and flickering lightings. The other product requirement 'quality braking system', denoted as p_8 , awaits further exploration. The case is focused from this point onwards, neglecting further propagations of nodes c_2 , c_3 , c_4 and p_9 (as seen in Figure 5-8).

The Assumptions

A group of representative customers forming the sample of the market target is assumed to be present, as the source of customer requirements (c_i) for the design process. A team

of designers from various segments of the product lifecycle is assumed to be the users of the proposed CDS.

Figure 5-8 Graph G' of Stallion

CDS is an architectural constitution as depicted in Figure 3-2. With the proposed CDH algorithm resided in the Design Exploratory Module, it interacts with Design Space Module, the database, and the CDS' users (via the user interface (UI)). The extraction and storing functions of data from/to the database are excluded in the scope of this work.

For the purpose of computing Stallion s_0^0 's MR (Equation 4.3), the *contribution fractions* of 'safety' (c_1) and 'quality brake system' (p_8) components are set to 1.0 (as reflected in Table A-1 in the appendix). On the other hand, their siblings' (c_2, c_3, c_4, p_9) contribution ratios are set to 0.0. These assumptions render the siblings neglected in this case, so as to allow focus on the brake system sub-problem.

It was assumed in Section 4.3.2.1 that the heuristic function h collectively captures the judgments of the solution concepts, based on the designers' best knowledge and experiences. For instance, in this design case, a heuristic value of 0.7 is assigned for a rim-type braking system (i.e. $h(s_{15}^1) = 0.7$). According to the assumption, the value of 0.7 therefore reflects the design team's perception on the promises of the rim-type braking system. (This particular assumption is further discussed in Section 6.2.1.)

The Design Process

This sub-section presents the stepwise operations of CDH algorithm (see Section 4.3.3) on the design problem. Tables A-1, A-2 and A-3 in the appendix present the data gathered by the end of the case study. Table A-4 on the other hand, tabulates the technical procedures that can be cross-referenced to the following textual descriptions.

- 1) CDH Algorithm Step 1: The design process starts with the creation of s_0^0 , the *seed* of further developments. This instantiated S class object is shown in Figure 5-8 and presented in Table A-2.
- 2) CDH Algorithm Step 2: *Stallion* (s_0^0), the main problem is selected for expansion by CDH algorithm.
- 3) CDH Algorithm Step 3: Customer requirements (CRs) on *Stallion* are elicited. Four CR are identified, viz. 'safety', 'comfort', 'high efficiency' and 'ease of usage'. For each CR, the designers establish the respective product requirements (PR) that are capable of satisfying the parental CR. For the case of CR 'safety', two PRs are identified, namely: 'quality braking system' and 'high presence'. The former PR is the focus in this case. The propagations of nodes 'comfort', 'high efficiency', 'ease of usage' and 'high presence' will be excluded in this case study. For the PR 'quality braking system', two alternative solutions are identified. They are separately 'rim' and 'disc' types braking system, shown in Figures 5-9 and 5-10 respectively.

- 4) CDH Algorithm Step 4: For the rim and disc type braking system, the system users assigned the heuristic values of 0.7 and 0.5 respectively (see the above section regarding the assumptions on the basis of the h values). The design team asserts that the rim type system is more promising due to its higher braking power (longer moment arm for higher resistive torque), and due to its relatively low weight as well. The alternative disc type system is however known to provide better force modulation, according to the team's knowledge and experiences. As the team deemed that the conceptualized solution thus far is not adequately detailed, they further explore into it.

Figure 5-9 Rim Type Brake System [102] Figure 5-10 Disc Type Brake System [102]

- 5) CDH Algorithm Step 5: Since no new leave node is being labeled 'solved' by the designers, no action is taken in this step.
- 6) CDH Algorithm Step 6: The CDS computed the most promising solution base (G_0). Of the two alternative braking systems, the rim type is included in the current solution base (according to the calculations shown in Table A-4 (3 of 7)).
- 7) CDH Algorithm Step 7: No action in this step, since *Stallion* (s_0^0) is neither 'solved' nor having $MR(s_0^0) = -\infty$. The CDH algorithm is looped. Go to Step 2.
- 8) CDH Algorithm Step 2: Among the current most promising solution base, the CDS identifies the next logical node for expansion (see Table A-4 (4 of 7) for the selection logics). The rim braking system (s_{15}^1) is recommended to the system users for further expansion.

9) CDH Algorithm Step 3: The solution ‘rim braking system (s_{15}^1)’ is subjected to the sample customer group for customer voices elicitation. Four CRs arose from the survey (as shown in Figure 5-8 and Table A-1):

- Robust performance in rain and muddy conditions (c_5)
- Good force modulation (c_6)
- Minimal weight (c_7)
- High force leveraging ratio (c_8)

PRs ($p_{10} - p_{15}$) that are capable of satisfying the above CRs are generated either from the database or from the user prompts. Following that, the solutions ($s_{16}^1 - s_{21}^1$) for the respective PRs are generated (for details, see Figure 5-8 and Table A-1).

10) CDH Algorithm Step 4: CDS requested the heuristic values for the respective solutions generated. As solutions s_{16}^1 (aluminum material to increase friction coefficient [102]) and s_{21}^1 (V-brake design to increase the force ratio) promise to fully satisfy their respective PRs, they receive the heuristic value of 1.0. This is as anticipated by the team earlier (in Step 4 of CDH algorithm in the first loop) - the rim-type braking system is able to provide high force leveraging ratio and lightweight components.

However, the team discovered that the force modulation capability of the rim brake system is less adequate than they have assumed, which is required by racing and cross-terrain cyclists. Albeit using solution such as hydraulic force transmission s_{18}^1 (which will however increases the maintenance) to replace the traditional cable type (s_{18}^2), the heuristic value assigned is still as low as 0.4. This low value reflects the low confidence level of the designers in the capability of the rim brake system in satisfying the “force modulation” product requirement. In other words, it cannot fully *promise* as required.

At this junction, the users deemed that the idea (solution s_{19}^1) of using aluminum material to minimize the weight of the braking system requires no further detail. This node is hence labeled ‘solved’ by the designers.

- 11) CDH Algorithm Step 5: As the design team has declared solution s_{19}^1 ‘solved’ in the previous step, CDS attempts to propagate this status upwards, in accordance to the Solve-labeling Function. The operations of this function are shown in Table A-4 (5 of 7).
- 12) CDH Algorithm Step 6: In this step, the CDS computed the most promising solution base (G_0), and presented it to the system users graphically. Within this cycle of CDH algorithm, *the merit ratio of the rim brake system falls below that of the disc brake system* (as calculations shown in Table A-4 (6 of 7)). Due to this mathematical trigger, the latter system replaces the former in the solution base G_0 . From design point of view, CDS recommended this replacement due to the recognized potential in the alternative solution system (disc-type). Furthermore, upon solution detailing of the rim-type system, the design team discovered that the modulating capability of the rim braking system is poorer than they have thought (causing the decrease of MR of rim-type system), which has attributed to the replacement recommendation.
- 13) CDH Algorithm Step 7: No action in this step, since *Stallion* (s_0^0) is neither ‘solved’ nor having $MR(s_0^0)=-\infty$. The CDH algorithm is looped again. Go to Step 2.
- 14) CDH Algorithm Step 2: Among the current most promising solution base identified in the last algorithm cycle’s Step 6, the CDS identifies the next logical node for expansion. The disc braking system (s_{15}^2) is recommended to the system users for expansion.

15) ... (The case description is truncated from here onwards. However, Tables 5-1, 5-2, and 5-3 contains the data generated by another cycle of CDH algorithm for further references.)

Discussions

In this case, two alternative solutions are available to the designers, of which the rim type is firstly selected for exploration. Generally, bicycle hobbyists and makers consider the rim type brake as relatively light, and conceptually able to produce more braking force due to the frictional force application at a longer moment arm at the rim, as compared to application near the wheel hub in the disc type case [102, 103]. As the rim type system is explored, the designers realized that several sub-solutions are unable to meet the expected requirements. For instances, the rim type is susceptible to mud at the wheel rim, which affects the braking efficiency. Apart from that, braking force is not easy to modulate. These setbacks affected the *merit ratio* of the rim type system, and consequently initiated the exploration of its alternative solution, the disc type system.

This case study did not attempt to cover the illustration of conceptualizing the entire bicycle design solution. Instead, the case focuses on demonstrating the functions of the CDH algorithm and its sub-functions. Applying the proposed mechanisms in a practical (i.e. larger) scale would in fact demonstrate more of their values, since designers usually encounter difficulties when handling large and complicated design space.

This case has overall demonstrated the working mechanisms of CDH algorithm and its sub-functions by presenting the detailed progressions of the algorithm, the data generated, and the mathematical calculations.

5.4 Chapter Summary

In this chapter, three cases aim to progressively demonstrate the functions of the CDS. Design Space Framework (DSF) is elucidated in the first case study. The second case exhibited the roles of CDH algorithm and its four sub-functions, while the third demonstrated the working mechanisms embedded in the CDS.

In the first and second case studies, numerous advantages of CDS have been shown, as summarized at the end of Sections 5.1 and 5.2 respectively. In the third case study, it has been shown how DSF and CDH algorithm facilitate the crystallizations of product concepts. Customer voices affect design at intervals along the process, during every generation of nodes c_i in the Step 3 of the CDH algorithm. In this manner, customer opinions are systematically and constantly assimilated into the conceptual design process. This methodology effectively advocates the philosophies of customer-oriented design, which has been found vital in the literature review (Chapter 2). The third case study has also shown how the CDS initiated a backtracking of the design. While existing design methods have not looked into the conditions and mechanisms of triggering design iteration, CDS has modeled this mathematically.

CDH algorithm and DSF are designed to *model* and *assist* the mental and verbal reasoning activities carried out during the conceptual design phase. Overall, CDS functions as a support and a guide to the designers in building up the conceptual product model.

– Chapter Six –

Conclusions

This chapter concludes the research and point out the future developmental work. Section 6.1 recaps the work done in this research and the significances of the results. In Section 6.2, the limitations of the work done are identified, along with their respective proposed remedies. Finally, in Section 6.3, several promising future work are identified.

6.1 The Research Summary

This section recaps the objectives before describing the tasks accomplished and their significances.

6.1.1 The Tasks Accomplished

Based on the literature review, two objectives were identified:

1. To define a customer-oriented knowledge organization structure, as the data model of a CACD system.
2. To prescribe a methodology that intelligently guides designers during product conceptualization process, based on the defined structure required by the first objective.

This project assumes that certain configurations of AI techniques are contributive to the attainment of the objectives. Two hypotheses were set out as follows.

1. A family of modeling methods, which may include mathematical assertions, frames, graphs and objects, is able to integrally represent different facets of design

knowledge. The envisaged customer-oriented knowledge organization structure can be formulated based on these methods.

2. An intelligent design exploration method can be devised by explicating human's design cognition into an algorithmic format, which is henceforth applicable in a CACD system. This method can be based on the principles of heuristic.

At the end of Chapter 2, a list of intended tasks was put forward, in view of the above hypotheses. The following paragraphs summarize the corresponding tasks addressed within Chapters 3, 4 and 5.

Based on the implementation requirements of the DSF and the CDH algorithm, the architecture of the conceptual design system (CDS) is postulated in Chapter 3. A conceptual design knowledge organization structure, so-called DSF, is formalized in terms of an axiom, set theory, graph theory, and framed-based techniques. This data structure serves as the product model of CDS. As intended in the first objective, this model advocates the customer-oriented philosophy of design, as well as capturing the multiple abstraction levels of design concepts.

As discussed in Chapter 4, product conceptualization problem is perceptible as a special case of human problem solving. The latter is widely researched in both cognitive science and AI fields. In AI research efforts, heuristic algorithms have been proposed to intelligently define solutions, as accordance to studies in cognitive sciences. The feasibility and applicability of heuristic algorithms on conceptual design problem is studied. It is found that the nature of conceptual design problems is fundamentally similar to some representative problems that heuristic algorithm can solve. However, there are few points of disparity between the problem domains. For these differences, traditional heuristic algorithms cannot be directly applied onto product conceptualization problems. This project innovatively adapts an existing general-purpose heuristic algorithm to operate on the product conceptualization process. Specifically, a conceptual design

heuristic (CDH) algorithm is tailored based on the GBF algorithm [91], to operate on the DSF established in Chapter 3. As a prescriptive procedure, the proposed CDH algorithm manages the exploration of the design space economically.

Three cases are presented in Chapter 5, in bid to demonstrate the propositions made in Chapters 3 and 4. The concept of DSF was elucidated in the first case. The second case illustrates a practical application of CDS, while the third exemplified the working mechanisms embedded in the CDS. It has been shown how DSF and CDH algorithm function as a support to the designers in building up the conceptual model. Through the establishments and discussions documented in Chapters 3, 4 and 5, the two hypotheses made in Chapter 2 have been shown to be positive. More importantly, the results of this research have therefore fulfilled the two objectives of this project.

6.1.2 The Significances of the Accomplished Tasks

As clarified in Section 1.1, the main objective of this project is to *contribute to the theoretical foundation required for the realization of a CACD system*. This somewhat abstract objective is translated into the two precise sub-objectives, based on the gaps identified in the volume of literature review documented in Chapter 2. The following lists the significances of the accomplishments with respect to the main objective.

- As the data structure of a CACD system, DSF is defined to *articulate* and *codify* [84] tacit knowledge dealt with in the conceptual design process. It is envisaged that this data structure can be mapped to conventional CAD software for the continuation of product development process – detail design. This would facilitate the capturing of design rationales.
- DSF promotes close proximity between the developed concepts, and the actual needs of the customers at various levels of abstraction. This structure ensures that the

relevant customer opinions are *constantly* sought and applied in the *entire* product conceptualization process.

- An innovative product can be designed if there are new needs, new solutions, or both. The structure of DSF presents the designers a framework to systematically scrutinize the concept of a product, and possibly identifying an area to make a unique difference.
- Recognizing the astronomical cost of explicating all possible concept variants, CDH algorithm is defined to assist designers in navigating design space strategically. Leveraging on designers' insights, the heuristic-based method enables the designers to explore concepts in a systematic manner. Backtracking in design is inevitable. Many methodologies have 'feed-back loops' drawn to indicate iterations and backtracking procedures [104]. However, in this work, such mechanisms and their triggers are explicated mathematically.

6.2 The Limitations and Remedies

Having attempted the tasks listed in Chapter 2, two limitations are detected. In the next two sub-sections, they are respectively described, along with their proposed remedies.

6.2.1 The Heuristic Assumption

The methodology proposed in this work has assumed that the heuristic function h represents the level of promises of the conceptual solutions, which is based on the designers' insights on solutions (see Section 4.4.3 on *insights*). It provides the rule of thumb for navigations in the design space.

Heuristic algorithms require estimated h values for their operations. The accuracy of the h values, in this work, is not relevant to the functions of the propositions made in this work. Instead, the consistencies of h , and the strength of correlations between h and

designers' insights on solutions' value/cost were assumed to be high in this work. Methodologies such as QFD [35], Pugh's decision matrix [55], Pahl and Beitz's utility theory [1], and Saaty's analytic hierarchy process (AHP) [105] are some of the methods designed to infuse human's tacit knowledge or insights into decision support tools. They all hold the same assumption, explicitly or implicitly.

As part of the future work, it is proposed that a simple but consistent and holistic method to elicit designers' insights on solutions' value and cost is prescribed. Yoshioka et al. [106], in their proposal of combining the essences of FBS (function, behaviour, structure) and QFD, have prescribed a method to explicate designers' assertions on alternative solutions, in terms of ratings. This is a distinct example of how designers' assertions can be explicated and transferred into a system for analysis. An improvement and adaptation of their work can possibly be part of the future work as well.

6.2.2 CDH Algorithm Exiting with Failure

In practice, can a team of designers declare the product conceptualization process a failure? Unlikely. They would probably backtrack and/or brainstorm for further alternatives whenever an impasse [30] is encountered.

CDH algorithm does not conform to this reality. It inherited the notion of "exit with failure if $MR(s_0^0) = -\infty$ " from its origin, the GBF algorithm. Classical problems that typical heuristic algorithms (such as GBF) are demonstrated to solve can, on the contrary, justifiably accepts 'failure' as an output. For instance, games such as chess, being a well-defined problem has finite numbers of alternative paths. If all alternatives are deemed "unsolvable" with regard to the objective (i.e. to win), then exiting with failure as a permanent output is justifiable.

It is proposed that if a product conceptualization problem receives ‘failure’ as an output from the CDH algorithm, certain further prescriptive steps are available to locate the bottleneck. As in reality, inability to solve a problem is due to the failure of *all* alternatives (if any) at one or more points in the design space (a point here is defined by a specific part of the problem, at certain level of abstraction). Which means, there will be classes of bottlenecks, which solving any one within a class will improve the overall situation. Against this setting, it could be possible for AI to locate and select the *most potential* bottleneck within a class for designers’ focused attention, as a form of assistance from a CACD system. The term ‘most potential bottleneck’ here may be referring to the unsolvable subproblem which will give the overall problem solving process most mileage if it is solved.

6.2.3 Design Coupling

Design methods that decompose problems into smaller parts have an inherent challenge – the interactions among the parts greatly complicate the solving procedures. This is a limitation of Design Space Framework (DSF) that requires attention. Ideally, DSF should model the coupling relationships, e.g. as design constraints, for considerations during the product conceptualization process. The specific remedy for this issue requires further research efforts in further development of DSF.

6.3 The Envisaged Future Developments

DSF and CDH algorithm have laid the groundwork for further investigation to be conducted. The following sub-sections each describe a potential future work, which merits and feasibilities are subjected to additional research and studies.

6.3.1 Embedding Cost-Performance Parameter

Research in cognitive psychology has long proven that in their natural thinking, human rarely strive for optimum solutions, but rather for satisficing solutions [27]. Satisficing solution is one that exceeds some threshold value [97]. It is a challenge in this work to insert a set of adjustable threshold parameter into the CDH algorithm. It can be expected to reflect the designers' balance of emphasis between the valuation of solution's quality and available resources such as time and cost. These parameters embedded in the algorithm may help to effectively control the trade-off between cost and performances according to changing market conditions and individual project needs.

6.3.2 Center-Out Solution Forming and Chaining

Function-means map [16, 31, 32] is a design method that works in a top-down convention. Nishioka et al. [62] promoted *only* top-down function refinement in their prototype system, PICCSS, to support product conceptual design. Bracewell and Sharpe [32] however recommended the reversed to introduce rigorous thinking.

During concept development, it is unnatural and highly impractical for the team to strictly work from the solution of the highest abstraction level. It is human nature not to generate solutions only in a top-down (nor bottom-up) manner. Besides, design processes are usually not sequential in practice.

As summarized in Chapter 2, one philosophy of specifying a CACD system is to synchronize and complement designers' natural practices. A CACD system should support both top-down and bottom-up approaches. Commonly used terms in the AI literature are *function-driven*, *means-driven*, *data-driven* and *goal-driven*. In effect, this will facilitate a more natural 'middle-out' solution generation process. These logics would permit alternative and partial solutions to randomly and concurrently sprout in the design

space. These ‘islands’ of solutions may then be integrated in stages to give a progressively complete solution. It is proposed that in the future work, such humanistic and cognitive-driven approach to product conceptualization is to be established.

Ulrich and Eppinger [34] considered one common dysfunctions exhibited by design team is the failure to integrate partial solutions from other firms, such as sub-solutions from the vendors. Apart from supporting design, this bi-directional approach is potentially capable of applications in the VME (virtual manufacturing enterprise) paradigm. Infusion of competitor’s reverse-engineered technological concepts is yet another promising application via this approach.

6.3.3 The Product Redesign Function

Derivative products, technology-pushed products, and customized products [34] are some of the terms used to describe products that are not designed from scratch, but based on a parent or sibling product. The root differences between different generations of products could be in their customer requirements and/or solution types. In the terminologies of the DSF proposed in this work, product redesign can be carried out by adapting graph G' of the platform or original product. Node class C and S at the relevant abstract level can be modified (addition and/or deletion) according to, for instances, new market needs, improved technology, or new-found innovative solutions. By applying a specialized CDH algorithm to the new graph G' , a redesigned product could be effectively resulted. This specialized CDH algorithm could form one of the future contributions.

6.3.4 Specialization for Domain Specific Applications

Numerous conceptual design methodologies were prescribed specifically for mechanical designs [5, 14, 16, 65]. Few problem-solving frameworks, such as Simon and Newell’s

GPS [30] are domain-neutral [62]. The DSF proposed in this work is domain-neutral, which is applicable to a wide range of design problems. Just as most generic systems can be specialized to perform better, DSF can be more defined to serve certain problem domain. For instance, the slots of frames and other area of the data structures can possibly be customized towards certain classes of products. The future of this work may choose to narrow down to work on certain class of product, to demonstrate the usefulness of the propositions here after specializations.

6.3.5 Integrating CDS to Conventional CAD System

For a CACD system to be applicable in the industry, its integration with conventional CAD system is not an option. Mapping product data models of CDS to conventional CAD system can possibly be addressed as future work.

6.3.6 Case-based Reasoning in CDS

Behavioral experiments have shown that while novices try to solve design problem through deductive reasoning, experts prefer to apply their experience directly [107]. The former approach of problem solving is parallel to the node-by-node expansion of graph G in the CDS. This way of constructing solution (i.e. solving problem) may churn out new innovative solution, but is slow and inefficient. On the other hand, the latter approach can be seen as applying case-based reasoning, i.e. applying chunks of nodes representing proven solution from past cases. Like an expert solving problems, this approach is a reliable and fast albeit difficult to produce innovative solution.

The author deems that the best approach to solve design problems is to allow the system to flexibly adopt between case-based reasoning technique and node-by-node deductive reasoning method, in accordance to the respective problem's unique needs

between innovation and efficiency. Such philosophy of problem solving can potentially be translated into another artificial intelligence method.

6.3.7 Comparison with Existing Models

To prove that the proposed approach has advantages over the existing methods, comparison has to be carried out. It should be compared to well-known methods such as axiomatic design [19] and QFD [35]. Apart from direct cross-sectional analysis and comparison of the methods, the comparison could also be done through observational studies on a group of designers using various methods to solve a common problem.

6.3.8 Collaboration with the Industry

In the attempt to validate the system, this work has demonstrated the solving of three conceptual design problems. For more rigorousness, the problems involved should be of wider range, such as architectural and electronic, and preferably using product data from the industry. Refining the model through iterative applications on problems from the industry will improve the robustness of the model.

6.4 Final Comments on the Project

An issue of modeling, especially of engineering modeling, is the validation of the model itself. This is to prove that any model in question can be ‘reasonably accurate’ for reliable applications in certain situation. A classic example is the modeling of spring force using Hooke’s Law, accompanied by the assumption that the spring is linear. This model can be validated empirically. On the other hand, there are models, referred to as prescriptive models that are *evaluated through their pragmatic value*, according to Bell et al. [108]. Design methods such as Suh’s axiomatic design and QFD cannot be validated using

quantitative analysis alone, since much of the variable in such models are of subjective nature [109]. The models presented in this thesis, DSF and CDH algorithm, are two such models. As proposed by Olewnik and Lewis [109], such models should at least be logical. It can be seen that although quantitative validation may not be applicable to the prescriptive models proposed in this work, they must be logical (in both their derivations and applications), and above all, they need to hold pragmatic values. The propositions in this thesis have been developed from an axiom and through a series of logical inferences before putting through three case studies to exemplify their application logics. These three cases may have suggested the pragmatic values of these models, further research, improvements and applications will reinforce its values.

While computer can surpass human in terms of storage capacity, processing speed and network connectivity, human's creativity, flexibility and intuition are currently unchallenged by the computers. The Conceptual Design System proposed in this thesis prescribed the cooperation of AI and natural intelligence to co-drive the conceptual design process. Horvath [110] summed that the paradigm has now shifted to knowledge-intensive systems without built-in problem solving capabilities. Instead of aiming towards enabling design endeavor fully automatic (as deemed not possible by Eagan et al. [58]), it might be more pragmatic at this point to specify a semi-automatic CACD system to complement the strengths of computers using human's fortes.

Artificial intelligence and design science are two distinct disciplines. Suh [19] has cautioned the application of information technologies (IT) onto design tasks without comprehending the latter in prior. The results of such application may appear impressive, due to the sophisticated technologies utilized. It is however not necessarily improving the design processes. Avoiding this pit-fall, this research starts by rigorously reviewing, studying and postulating design theories that aim to give the proposed CACD system a scientific foundation. The henceforth applications of IT and AI, in this work, maps the

devised theories to the computational objects (data structures, functions and algorithm) as an implementation proposal. The studies of design science in this work is in bid to better understand design, and hence towards better design tools [110]. In addition, this work has scrutinized the cognitive models of general problem solving [30,50,86,88,93,97,99,111] which has been the central basis of design science.

References

1. Pahl, G. and Beitz, W. (1996) Engineering Design: A Systematic Approach. 2nd ed., London: Springer-Verlag.
2. Al-Salka M.A., Cartmell M.P. and Hardy S.J. (1998) A Framework for a Generalized Computer-based Support Environment for Conceptual Engineering Design, *Journal of Engineering Design*, 9(1), 57-88.
3. Smith, P.G. and Reinertsen, D.G. (1997) Developing Products in Half Time: New Rules, New Tools, 2nd ed., John Wiley & Sons.
4. Wang L., Shen W., Xie H., Neelamkavil J. and Pardasani A. (2002) Collaborative Conceptual Design – state of the art and future trends, *Computer-Aided Design*, 34, 981-996.
5. Zou G., Hu Y., Wang X. and Li C. (2005) The representation of conceptual product based on component-connector design feature with P/T net approach, *International Journal of Advanced Manufacturing Technology*, 26, 1193-201.
6. Hsu W. and Liu B. (2000) Editorial. Conceptual Design: issues and challenges, *Computer-Aided Design*, 32, 849-850.
7. Brunetti G. and Golob B. (2000) A feature-based approach towards an integrated product model including conceptual design information, *Computer-Aided Design*, 32, 877-887.
8. Hsu W. and Woon M.Y. (1998) Current research in the conceptual design of mechanical products, *Computer-Aided Design*, 30 (5), 377-389.

References

9. Araujo R.H., Possamai O. and Valentina L.D. (2001) Knowledge Decomposition for Conceptual Product Design: An Approach to Develop Specific Domain Expert Systems for Supporting Concurrent Engineering Projects. In: Monostri L., Vancza j. and Ali M. (eds.). IEA/AIE 2001, LNAI 2070, p.757-766. Berlin Heidelberg: Springer-Verlag.
10. Meniru K., Rivard H. and Bedard C. (2003) Specifications for computer-aided conceptual building design, *Design Studies*, 24, 51-71.
11. Seo K.-K., Park J.-H., Jang D.-S. and Wallace D. (2002) Approximate Estimation of the Product Life Cycle Cost Using Artificial Neural Networks in Conceptual Design, *International Journal of Advanced Manufacturing Technology*, 19, 461-471.
12. Harding J.A., Popplewell K., Fung R.Y.K. and Omar A.R. (2001) An intelligent information framework relating customer requirements and product characteristics, *Computers in Industry*, 44, 51-65.
13. Engelbrektsson P. and Soderman M. (2004), The use and perception of methods and product representations in product development: a survey of Swedish industry, *Journal of Engineering Design*, 15 (2), 141-154.
14. Liu Y.-C., Bligh T. and Chakrabarti A. (2003) Towards an 'ideal' approach for concept generation, *Design Studies*, 24, 341-355.
15. Sim S.K. and Duffy A.H.B. (2003) Towards an ontology of generic engineering design activities, *Research in Engineering Design*, 14, 200-223.
16. Holland A., O'Callaghan B., O'Sullivan B. (2004) Supporting Constraint-Aided Conceptual Design from First Principles in AutoDesk Inventor. In: R. Orchard (eds.) *IEA/AIE 2004, LNAI 3029*, pp. 905-914. Berlin Heidelberg: Springer-Verlag.
17. Holland A., O'Callaghan B., O'Sullivan B. (2003) A Constraint-Aided Conceptual Design Environment for AutoDesk Inventor. In: Rossi F. (eds.): *CP 2003, LNCS 2833*, pp.422-436. Berlin Heidelberg: Springer-Verlag.

References

18. Finger S. and Dixon J.R. (1989), A review of research in Mechanical engineering design. Part I: Descriptive, Prescriptive, and computer-based models of design processes, *Research in Engineering Design* 1:51-67.
19. Suh N.P. (1990) *The Principles of Design*, New York: Oxford University Press.
20. Love T. (2002) Constructing a coherent cross-disciplinary body of theory about designing and designs: some philosophical issues, *Design Studies*, 23, 345-361.
21. Zeng Y. and Gu P. (1999) A science-based approach to product design theory. Part 1: formulation and formalization of design process, *Robotics and Computer Integrated Manufacturing*, 15, 331-339.
22. Cross N. (1992) Research in design thinking. In: Cross N., Dorst K. and Roozenburg N. (eds.) *Research in Design Thinking*. Delft University Press, pp 3-10.
23. Love T. (2000) Philosophy of Design: a meta-theoretical structure for design theory, *Design Studies*, 21, 293-313.
24. Popper K. (2002) *Unended Quest*, London: Routledge.
25. Xiong M.Ch. (2003) Global manufacture by using conceptual design, *Journal of Material Processing Technology*, 139, 453-456.
26. Tay F.E.H. and Gu J. (2002) Product modeling for conceptual design support, *Computers in Industry*, 48, 143-155.
27. Stempfle J. and Badke-Schaub P. (2002), Thinking in design teams – an analysis of team communication, *Design Studies*, 23, 473-496.
28. Rosenman M.A. and Gero J.S. (1998), Purpose and function in design: from the socio-cultural to the techno-physical, *Design Studies*, 19, 161-186.
29. Gero J.S. and Coyne R.D. (1985), Knowledge-based planning as a design paradigm, *Design theory in computer-aided design, Proceedings of the IFIP WG 5.2 Working Conference 1985, Tokyo*, Yoshikawa, H. and Warman, E.A., eds., North Holland, Amsterdam, 261-295.

References

30. Newell A. (1990) Unified Theories of Cognition. Cambridge MA: Harvard University Press.
31. Sullivan B.O (2002) Interactive constraint-aided conceptual design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 16, 303-328.
32. Bracewell R.H. and Sharpe J.E.E. (1995) The Use of Bond Graph Reasoning for the Design of Interdisciplinary Schemes, *International Conference on Bond Graph Modelling and Simulation (ICBGM'95)*, Las Vegas, SCS Publishing, v27(1), pp116-121.
33. Matthews P.C., Blessing L.T.M. and Wallace K.M. (2002) The introduction of design heuristics extraction method, *Advance Engineering Informatics*, 16, 3-19.
34. Ulrich K.T. and Eppinger S.D. (2000). *Product Design and Development*. 2nd ed., McGraw-Hill.
35. Hauser J.R. and Clausing D. (1988) The House of Quality, *Harvard Business Review*, May-June, 63-73.
36. Medland A.J. and Mullineux G. (2000) A decomposition strategy for conceptual design, *Journal of Engineering Design*, 11(1), 3-16.
37. Qian L., Gero J. S., (1996) Function-behaviour-structure paths and their role in analogy-based design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10, 289-312.
38. Tomiyama T., Umeda Y., Yoshikawa H. (1993) A CAD for functional Design. *Annals of the CIRP*, 42(1), 143-146.
39. Deng Y.M., Tor S.B. and Britton G.A. (2000) Abstracting and Exploring Functional Design Information for Conceptual Mechanical Product Design, *Engineering with Computers*, 16, 36-52.

References

40. Tseng M.M. and Jiao J. (1997) A Variant Approach to Product Definition by Recognizing Functional Requirement Patterns, *Computers and Industrial Engineering*, 33 (3-4), 629-633.
41. Gershenson J.K. and Stauffer L.A. (1999) A Taxonomy for Design Requirements from Corporate Customers, *Research in Engineering Design*, 11, 103-115.
42. Ottosson S. (1998) Qualified Product Concept Design Needs a Proper Combination of Pencil-aided Design and Model-aided Design Product Data Management, *Journal of Engineering Design*, 9(2), 107-119.
43. Khoo L.P., Chen C.H. and Yan W. (2002) AN investigation on a prototype customer-oriented information system for product concept development, *Computers in Industry*, 49, 157-174.
44. Dahan E. and Hauser J.R. (2002) The Virtual Customer, *The Journal of Product Innovation Management*, 19, 332-353.
45. Chen C.H., Khoo L.P. and Yan W. (2002) A Strategy for acquiring customer requirements patterns using laddering technique and ART2 neural network, *Advanced Engineering Informatics*, 16, 229-240.
46. Naes T. and Nyvold T. E. (2004) Creative design – an efficient tool for product development, *Food Quality and Preference*, 15, 97-104.
47. Braha D. and Peich Yoich (2003) Topological structures for modeling engineering design processes, *Research in Engineering Design*, 14, 185-199.
48. Cross N. (2000) Engineering Design Methods: Strategies for Product Design. 3rd ed., England: Wiley.
49. Moulianitis V.C., Aspragathos N.A. and Dentsoras A.J. (2004) A model for concept evaluation in design – an application to mechatronics design of robot grippers, *Mechatronics*, 14, 599-622.
50. Goel V. (1995) Sketches of thought. Cambridge MA: MIT Press.

References

51. Roy U., Pramanik N., Sudarsan R., Sriram R.D., Lyons K.W. (2001) Function-to-form mapping: model, representation and applications in design synthesis, *Computer-Aided Design*, 33, 699-719.
52. Nellore R., Soderquist K. and Eriksson K-A. (1999) A Specification Model for Product Development, *European Management Journal*, 17(1), 50-63.
53. Chakrabarti A. and Bligh T. P. (1996) An approach to functional synthesis of design concepts: Theory, application, and emerging research issues. *AI EDAM*, 10, 313-331.
54. Zhang W.Y., Tor S. B. and Britton G. A. (2001) A Prototype Knowledge-Based System for Conceptual Synthesis of the Design Process, *The International Journal of Advance Manufacturing Technology*, 17, 549-557.
55. Pugh S, (1991) Total Design: Integrated Methods for successful product engineering. Wokingham, England: Addison Wesley.
56. Fricke G. (1996) Successful individual approaches in engineering design, *Research in Engineering Design*, 8, 151-165.
57. Gero J.S. and Sudweeks F. eds. (1998) Artificial Intelligence in Design '98, Dordrecht: Kluwer Academic.
58. Eagan R.J. (2001) Approached to improve Engineering Design, Washington, D.C: The National Academies Press.
59. Parametric Technology Corporation [On-line] Available from <http://www.ptc.com> [Accessed 18th May 2004].
60. Sun S. and Lin Z. (2003) Models and techniques of computer supported cooperative conceptual design, *Integrated Manufacturing Systems*, 14(6), 530-536.
61. Li W.D., Lu W.F., Fuh J.Y.H. and Wong Y.S. (2004), Collaborative computer-aided design – research and development status, *Computer-aided Design*, 37, 931-940.

References

62. Nishioka Y., Nakasuka S. and Hori K. (1996) PICCSS: Problem Interactive Clarification and Concurrent Solving System, *Knowledge Intensive CAD Vol. 2*, P39-58, Chapman & Hall.
63. Elsas P.A. and Vergeest J.S.M. (1998) New functionality for computer-aided conceptual design: the displacement feature, *Design Studies*, 19, 81-102.
64. Montoya-Weiss M. M. and O'Driscoll T.M. (2000) From Experience: Applying Performace Support Technology in the Fuzzy Front End, *Journal of Product Innovation Management*, 17, 143-161.
65. Szuba J., Schurr A. and Borkowski A. (2002) GraCAD – Graph-Based Tool for Conceptual Design. In: Corradini A. (Eds.): ICGT 2002, LNCS 2505, pp.363-377. Berlin Heidelberg: Springer-Verlag.
66. Gero J. S. Kannengiesser U. (2004), The situated function-behaviour-structure framework, *Design Studies*, 25, Issue 4, 373-391.
67. McKay A., Pennington A.D. and Baxter J. (2001) Requirements management: a representation scheme for product specifications, *Computer-Aided Design*, 33, 511-520.
68. Umeda Y. and Tomiyama T. (1997) Functional reasoning in design, *IEEE Expert*, 12 (2), 42-48.
69. Ratchev S., Urwin E., Muler D., Pawar K.S. and Moulek I. (2003) Knowledge based requirement engineering for one-of-a-kind complex system, *Knowledge-based System*, 16,1-5.
70. Kim J.K., Han H.H., Choi S.H. and Kim S.H. (1998) A knowledge-based approach to the Quality Function Deployment, *Computers and Industrial Engineering*, 35(1-2), 233-236.
71. Han Y. and Lee K. (1999) Virtual Function Generators: Representing and Reusing Underlying Design Concepts in Conceptual Synthesis of Mechanisms for Function

References

- Generation. In: K.-D. Althoff, Bergmann R., Branting L.K. (eds.), ICCBR-99, LNAI 1650, pp. 453-467, Berlin Heidelberg: Springer-Verlag.
72. Zavbi R. and Duhovnik J. (2001) Conceptual design chains with basic schematics based on an algorithm of conceptual design, *Journal of Engineering Design*, 12(2), 131-145.
73. Sun J., Kalenchuk D.K., Xue D. and Gu P. (2000) Design candidate identification using neural network-based fuzzy reasoning, *Robotics and Computer Integrated Manufacturing*, 16, 383-396.
74. Qiu S.L., Fok S.C., Chen C.H. and Xu S. (2002) Conceptual Design Using Evolution Strategy, *International Journal of Advanced Manufacturing Technology*, 20, 683-691.
75. Lin J.J. (2003) Constructing an intelligent conceptual design system using genetic algorithm, *Journal of Material Processing Technology*, 140, 95-99.
76. Goldberg D.E. (1991), Genetic algorithm as a computational theory of conceptual design, *Application of Artificial Intelligence in Engineering*, VI, 3-16.
77. Yan W., Chen C.H. and Khoo L.P. (2002) An integrated approach to the elicitation of customer requirements for engineering design using picture sorts and fuzzy evaluation, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 16, 59-71.
78. Wang J. (2002) Improved engineering design concept selection using fuzzy sets, *Int. J. Computer Integrated Manufacturing*, 15(1), 18-27.
79. Wang J. (1997) A fuzzy outranking method for conceptual design evaluation, *International Journal of Production Research*, 35(4), 995-1010.
80. Fung R.Y.K., Tang J., Tu Y. and Wang D. (2002) Product design resources optimization using a non-linear fuzzy quality function deployment model, *International Journal of Production Research*, 40(3), 585-599.

References

81. Tauber E. M. (1972) HIT: Hueristic Ideation Technique - A Systematic Procedure For New Product Search, *Journal of Marketing*, 36, 58-70.
82. Li C. L., Tan S. T. and Chan K. W., A qualitative and heuristic approach to the conceptual design of mechanism, *Engineering Applications of Artificial Intelligence*, 9 (1), 17-31.
83. Bobrow D. G. (1975) Dimensions of Representation, New York: Academic Press.
84. Tomiyama T., Umeda Y., Ishii M., Yoshioka M. and Kiriya T. (1995) Knowledge systemization for a knowledge intensive engineering framework, *Knowledge Intensive CAD Vol. 1*, pp34-53, Chapman and Hall.
85. Friedman K. (2003) Theory construction in design research: criteria: approaches, and methods, *Design Studies*, 24, 507-522.
86. Eysenck M.W. (1997) A Handbook of Cognitive Psychology, London: Lawrence Erlbaum Associates.
87. Luger G.F. (2002) Artificial Intelligence: Structures and Strategies for Complex Problem Solving, London: Addison-Wesley.
88. Davidson J.E. and Sternberg R.J. eds. (2003) The Psychology of Problem Solving, New York: Cambridge University Press.
89. Mantyla M. (1995) Knowledge Intensive CAD: Introduction and a Research Agenda, *Knowledge Intensive CAD Vol. 1*, pp3-12, London: Chapman and Hall.
90. Dougherty E.R. and Giardina C.R. (1988) Mathematical Methods for Artificial Intelligence Autonomous Systems, Englewood Cliffs, NJ: Prentice-Hall.
91. Pearl J. (1984) Intelligent Search Strategies for Computer Problem Solving, Massachusetts: Addison-Wesley.
92. Cagan J. and Agogino A.M. (1988), 1stPRINCE: Innovative Design from First Principles, 7th National Conference on Artificial Intelligence, AAAI-88, Minneapolis, MN, August 21-26, 1988.

References

93. Haugeland J. eds. (1981) *Mind Design: Philosophy, Psychology, and Artificial Intelligence*, Cambridge, MA: MIT Press.
94. Farreny H. (1999) Completeness and Admissibility for General Heuristic Search Algorithms – A Theoretical Study: Basic Concepts and Proofs, *Journal of Heuristics*, 5, 353-376.
95. Bolc L. and Cytowshi J. (1992) *Search Methods for Artificial Intelligence*, London: Academic Press.
96. Webster R. W. (1991) Useful AI Tools: A Review of Heuristic Search Methods, *IEEE Potentials*, 10 (3), 51-54.
97. Simon H.A. (1969), *The sciences of artificial*, MIT Press, Cambridge MA.
98. Al-Ayyonb A.-E. and Masoud F.A. (2000) Heuristic Search Revisited, *The Journal of Systems and Software*, 55, 103-113.
99. Feldman R. S. (1993) *Understanding Psychology*, 3rd ed., New York: McGraw-Hill.
100. Graham B. and McGowan K. (2004), *Build your own all-terrain robot*, New York: McGraw Hill.
101. Hornback P. (1998) The Wheel Versus Track Dilemma, *Armor*, March-April 1998, 33-34.
102. Slater G. (not dated) Mountain Biking: V-Brake Vs Disc Brake. [On-line] Available from <http://mountainbike.about.com/library/weekly/aa020601a.htm> [Accessed 18th September 2004].
103. Chiang C.P. and Tseng C.H. (1999) Development of Conceptual Design for Brake Shoes of a Bicycle by Using TRIZ. [On-line] Available from the archives of *The Triz Journal* (Online), October 1999, <http://www.triz-journal.com> [Accessed 18th September 2004].
104. Eder W. E. (1995) *Systematic Conceptualizing – With Computational Tools*, Knowledge Intensive CAD Vol. 1, pp.206-224, Chapman and Hall.

References

105. Saaty T.L. (1990) How to make a decision: the analytic hierarchy process, *European Journal of Operational Research*, 48 (1), 9-26.
106. Yoshioka M., Oosaki M. and Tomiyama T. (1995) An application of quality function deployment to functional modeling in a knowledge intensive design environment, *Knowledge Intensive CAD Vol. 1*, pp.300-314, Chapman and Hall.
107. Goker M.H., Darmstadt T. H. (1997) The effects of experience during design problem solving, *Design Studies*, 18, 405-426.
108. Bell D., Raiffa H. and Tversky A. (1998), Descriptive, normative, and prescriptive interactions in decision making, In: *Descriptive, normative, and prescriptive interactions*, eds. Bell D., Raiffa H. and Tversky A., Cambridge University Press, 357-358.
109. Olewnik A.T. and Lewis K. E. (2003), On validating design decision methodologies, *Proceedings of ASME 2003 Design Engineering Technical Conferences, Design Theory and Methodology conference*, Chicago, IL, Sep 2-6, 1-10.
110. Horvath I. (2004), A treatise on order in engineering design research, *Research in Engineering Design*, 15, 155-181.
111. Nilsson N.J. (1980) *Principles of Artificial Intelligence*, Palo Alto, CA: Tioga.

Appendix

Appendix

name	Safety	Comfort	Efficiency	Easy Use		Brake	Presence	Weather	Modulate
superclass	N	N	N	N		N	N	N	N
class	C	C	C	C		P	P	C	C
id	1	2	3	4		8	9	5	6
contribution_fraction	1.0	0.0	0.0	0.0		1.0	0.0	0.3	0.4
solve status	0	0	0	0		0	0	0	0
description		Excluded in case	Excluded in case	Excluded in case		Quality braking system	Excluded in case	Robust in rain and muddy track	Good control

name	Light	Brake Force	Wet resistance	Dirt	Modulation	Density	Volume	Force ratio	Weather
superclass	N	N	N	N	N	N	N	N	N
class	C	C	P	P	P	P	P	P	C
id	7	8	10	11	12	13	14	15	9
contribution_fraction	0.1	0.2	0.6	0.4	1.0	0.5	0.5	1.0	0.3
solve status	0	0	0	0	0	0	0	0	0
description	Minimise weight	Little force required to brake	High coefficient of friction in rain	Prevent dirt between pad and rim	Linear relationship between applied force and braking force	Low overall density	Low overall volume	Low applied vs braking force ratio	Robust in rain and muddy track

name	Easy to modulate	Light	Brake Force	Wet resistance	Dirt	Modulation	Density	Volume	Force ratio
superclass	N	N	N	N	N	N	N	N	N
class	C	C	C	P	P	P	P	P	P
id	10	11	12	16	17	18	19	20	21
contribution_fraction	0.4	0.1	0.2	0.6	0.4	1.0	0.5	0.5	1.0
solve status	0	0	0	0	0	0	0	0	0
description	Good control	Minimise weight	Little force required to brake	High coefficient of friction in rain	Prevent dirt between pad and disc	Linear relationship between applied force and braking force	Low overall density	Low overall volume	Low applied vs braking force ratio

Table A-1 Object Class C and P in G'

Appendix

name	Stallion		Rim Brake	Disc Brake	Ai Alloy Rim	Dirt Scraper	Hydraulic	Cable	Ai material
superclass	N		N	N	N	N	N	N	N
class	S		S	S	S	S	S	S	S
id	0		15	15	16	17	18	18	19
l id	0		1	2	1	1	1	2	1
contribution_fraction	1.0		1.0	1.0	1.0	1.0	1.0	1.0	1.0
description			Adequate brake power; Light; Non-weather resistance	Low braking power; Heavy; Good brake force modulation	Ai alloy stop bicycle 4 times faster than steel	A scraper on pad mounted on spring. More parts.	Good modulation. But high maintenance	Bad modulation when dirty.	light and high strength
MR	0.5		0.465	0.5	1.0	0.2	0.4	0.2	1.0
heuristics h	1.0		0.7	0.5	1.0	0.2	0.4	0.2	1.0
solve status	0		0	0	0	0	0	0	1
F id	1		12	13	14	15	16	17	18
B id	1		12	13	14	15	16	17	18
A id	1		12	13	14	15	16	17	18

name	Minimise vol.	V-system	Ai Disc	Clearance	Hydraulic	Cable	Material	Minimise vol.	Disc Geometry
superclass	N	N	N	N	N	N	N	N	N
class	S	S	S	S	S	S	S	S	S
id	20	21	22	23	24	24	25	26	27
l id	1	1	1	1	1	2	1	1	1
contribution_fraction	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
description	Minimal part volume	V-brake system provides leverage for force	Ai material disc	Disc is far higher than ground lever avoiding mud	Disc inherently give relatively good modulation	Disc inherently good modulation. Cable catch dirt.	Mix of forged Ai and Steel in system. Overall relatively heavy.	Minimise volume, but Disc is unavoidably high volume	Special disc geometry, / drilled holes to up coefficient
MR	1.0	1.0	1.0	1.2	1.0	0.8	0.7	0.5	1.0
heuristics h	1.0	1.0	1.0	1.2	1.0	0.8	0.7	0.5	1.0
solve status	0	0	0	0	0	0	0	0	0
F id	19	20	21	22	23	24	25	26	27
B id	19	20	21	22	23	24	25	26	27
A id	19	20	21	22	23	24	25	26	27

Table A-2 Object Class S in G'

[illegible]

	E	E	E	E	E	E	E	E	E	E	E	E	E
superclass													
class	W	W	U	U	U	U	U	U	U	V	V	V	V
parent id	15	15	5	5	6	7	8	9	10	11	12	12	12
child id	7	8	10	11	12	13	14	15	16	17	18	18	18
l id	NA	NA	NA	NA	NA	NA	NA	NA	1	1	1	1	2

[illegible][illegible]Table A-3 Relational Object Class U , V and W in G''

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
1) Put the start node s_0^0 on both OPEN and G_0 .	The seed of the given design problem, s_0^0 , is created in G' (See G'' in Table A-2 and Figure 5-8).	OPEN= $\{s_0^0\}$ CLOSED= $\{\emptyset\}$ $G_0=\{s_0^0\}$	Created s_0^0 , that is, the solution of the highest abstraction, <i>Stallion</i> .
2) Using f_2 , select a node n^* (if initial, just s_0^0). Remove n^* from OPEN and place it on CLOSED.	$n^*=s_0^0$ is assigned	OPEN= $\{\emptyset\}$ CLOSED= $\{s_0^0\}$ $G_0=\{s_0^0\}$	No Action
3) Expand node n^* - generating its immediate <i>Context</i> successor by using data from the UI and/or database. Add all generated nodes, as well as their relational pointers (leading back to n^*) to the search graph G' . Place all leaf nodes on OPEN.	<p>I. n^* generates four children (c_1, c_2, c_3 and c_4) with their respective contribution ratio. Relational pointers objects (w_1, w_2, w_3 and w_4) created in G'.</p> <p>II. Each C node generates their respective children ($p_1, \dots, p_8, p_9$). Relational pointers objects ($\dots, u_{1,8}, u_{1,9} \dots$) are created in G'. One of the product requirements, p_8, is selected as the focus of this case study.</p> <p>III. p_8 generates two solution options: s_{15}^1 and s_{15}^2. Relational pointers objects ($\dots, v_{8,15}^1, v_{8,15}^2$) are created in G'.</p> <p>IV. The two solutions are placed on OPEN.</p>	<p>OPEN=$\{s_{15}^1, s_{15}^2\}$</p> <p>CLOSED=$\{s_0^0\}$</p> <p>$G_0=\{s_0^0\}$</p>	<p>I. Four customer requirements (CR) and their respective contributive ratio are identified from customer group X.</p> <p>II. Designers identified the product requirements (PR) for the respective CRs. One of the PR of 'safety' (a CR) is 'Quality Brake System'.</p> <p>III. Two solutions (Rim Brake System and Disc Brake System) are raised as alternatives for the PR 'Quality Brake System'.</p>

Table A-4 A Braking System Conceptualization Case (1 of 7)

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
4) For each new leaf node s'_k , from UI, acquire heuristic information h that characterizes the set of solution graphs rooted at s'_k . Let new leave node s'_k 's $solve_status(s'_k) = I$, if instructed by UI.	I. $h(s^1_{15}) = 0.7$ and $h(s^2_{15}) = 0.5$ are acquired from users. The basis of these heuristic values are represented as $description(s^1_{15})$ and $description(s^2_{15})$, as shown in solution frames (Table A-2). II. Neither s^1_{15} nor s^2_{15} are declared "solved".	OPEN= $\{s^1_{15}, s^2_{15}\}$ CLOSED= $\{s^0_0\}$ $G_0 = \{s^0_0\}$	I. For the Rim Brake and Disc Brake System, users arrived at heuristic value of 0.7 and 0.5 respectively. The h values and their basis are input into the system. II. The users wish to explore into more detailed conceptualization within the brake system.

Table A-4 A Braking System Conceptualization Case (2 of 7)

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
5) If any of the new ...	No Action	$OPEN = \{s_{15}^1, s_{15}^2\}$ $CLOSED = \{s_0^0\}$ $G_0 = \{s_0^0\}$	No Action
6) Base on the explicit search graph G' constructed so far, compute the most promising solution base graph G_0 using function f_l and the heuristics h provided in Step 4. Refresh G_0 .	<p>I. Using Eqn 4.3, calculating MR from bottoms up,</p> <p>Since s_{15}^1 and s_{15}^2 are in OPEN,</p> $MR(s_{15}^1) = h(s_{15}^1) = 0.7$ $MR(s_{15}^2) = h(s_{15}^2) = 0.5$ <p>Since s_0^0 is in CLOSED,</p> $MR(s_0^0) = h(s_0^0) \sum_i \Lambda(c_i) \sum_j \Lambda(p_j) \sum_k \max[\Lambda(s_k^l) MR(s_k^l)]$ $= h(s_0^0) \times \Lambda(c_1) \times \Lambda(p_8) \times \Lambda(s_{15}^1) \times MR(s_{15}^1)$ $= 1.0 \times 1.0 \times 1.0 \times 1.0 \times 0.7$ $= 0.7$ <p>II. Solution s_{15}^1 is included in G_0, instead of s_{15}^2, as chosen by Eqn 4.4 in the computation of G_0's merit ratio, or $MR(s_0^0)$.</p>	$OPEN = \{s_{15}^1, s_{15}^2\}$ $CLOSED = \{s_0^0\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^1, w_l, u_{1,8}, v_{8,15}^1\}$	Users are presented with the resultant G_0 .

Table A-4 A Braking System Conceptualization Case (3 of 7)

Appendix

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
7) If $solve_status(s_0^0) = I$, exit with G_0 as the solution graph. Else if $MR(s_0^0) = -\infty$, exit with failure. Else go to Step 2.	No Action	$OPEN = \{s_{15}^1, s_{15}^2\}$ $CLOSED = \{s_0^0\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^1, w_l, u_{1,8}, v_{8,15}^1\}$	No Action
2) Using f_2 , select a node n^* (if initial, just s_0^0). Remove n^* from OPEN and place it on CLOSED.	$f_2(s_k^l \in G_0 \cap OPEN)$ $= \min_k [h(s_k^l \in G_0 \cap OPEN)]$ $= s_{15}^1$ $n^* = s_{15}^1$ is assigned	$OPEN = \{s_{15}^2\}$ $CLOSED = \{s_0^0, s_{15}^1\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^1, w_l, u_{1,8}, v_{8,15}^1\}$	Users are presented the next logical node for expansion, n^* .
3) Expand node n^* - generating its immediate <i>Context</i> successor by using data from the UI and/or database. Add all generated nodes, as well as their relational pointers (leading back to n^*) to the search graph G' . Place all leaf nodes on OPEN.	I. n^* generates CR objects (c_5, c_6, c_7 and c_8) and their relational pointers objects. (as recorded in Table A-1 and Table A-3) II. Each C node generates their respective children ($p_{10}, \dots, p_{15}$) and their relational pointers objects. III. Each P node generates their respective nodes ($s_{16}^1, \dots, s_{21}^1$) and their relational pointers objects. IV. The seven solutions are placed on OPEN.	$OPEN = \{s_{15}^2, s_{16}^1, s_{17}^1, s_{18}^1, s_{19}^2, s_{20}^1, s_{21}^1\}$ $CLOSED = \{s_0^0, s_{15}^1\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^1, w_l, u_{1,8}, v_{8,15}^1\}$	Guided by the system, users provide entries to system prompts.

Table A-4 A Braking System Conceptualization Case (4 of 7)

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
<p>4) For each new leaf node s'_k, from UI, acquire heuristic information h that characterizes the set of solution graphs rooted at s'_k. Let new leaf node s'_k's $solve_status(s'_k) = I$, if instructed by UI.</p>	<p>I. Heuristic values and their basis are acquired from the users, and recorded in graph G' (Table A-2).</p> <p>II. Users declared s'_{19} “solved”.</p> <p>$solved_status(s'_{19}) = 1$ is assigned</p>	<p>OPEN = $\{s_{15}^2, s_{16}^1, s_{17}^1, s_{18}^1, s_{18}^2, s_{19}^1, s_{20}^1, s_{21}^1\}$</p> <p>CLOSED = $\{s_0^0, s_{15}^1\}$</p> <p>$G_0 = \{s_0^0, c_l, p_8, s_{15}^1, w_l, u_{1,8}, v_{8,15}^1\}$</p>	<p>I. Guided by the system, users provide entries to system prompts for h values and their basis.</p> <p>II. Users deemed PR of ‘light weight’ is adequately solved by solution ‘forged aluminum’, and no further details are needed in this conceptualization phase.</p>
<p>5) If any of the new leaf node s'_k on OPEN has been marked $solve_status(s'_k) = I$ in Step 4, invoke Solved-labeling function with s'_k as the operand. Else go to next step.</p>	<p>Invoking Solve-labeling function, with s'_{19} as operand.</p> <p>I. s'_{19} is removed from OPEN</p> <p>II. The parent, $solved_status(p_{13}) = 1$ is assigned.</p> <p>III. Since not all siblings along dimension X (i.e p_{14}) is solved, exit function.</p>	<p>OPEN = $\{s_{15}^2, s_{16}^1, s_{17}^1, s_{18}^1, s_{18}^2, s_{20}^1, s_{21}^1\}$</p> <p>CLOSED = $\{s_0^0, s_{15}^1, s_{19}^1\}$</p> <p>$G_0 = \{s_0^0, c_l, p_8, s_{15}^2, w_l, u_{1,8}, v_{8,15}^2\}$</p>	<p>Users are informed of the solved status of p_{13} (by propagations)</p>

Table A-4 A Braking System Conceptualization Case (5 of 7)

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
6) Base on the explicit search graph G' , constructed so far, compute the most promising solution base graph G_0 using function f_i and the heuristics h provided in Step 4. Refresh G_0 .	<p>Using Eqn 4.3, calculating MR from bottoms up,</p> <p>I. Since s_{15}^1 is in CLOSED,</p> $MR(s_{15}^1) = h(s_{15}^1) \sum_i \Lambda(c_i) \sum_j \Lambda(p_j) \sum_k \max[\Lambda(s_k^1) MR(s_k^1)]$ $= 0.7 \{0.3[0.6(1.0) + 0.4(0.2)] + 0.4[0.4(1.0)] + 0.1[0.5(1.0) + 0.5(1.0)] + 0.2[1.0(1.0)]\}$ $= 0.465$ <p>II. Since $MR(s_{15}^1) < MR(s_{15}^2)$ and $\Lambda(s_{15}^1) = \Lambda(s_{15}^2)$, in computation of G_0's merit ratio, or $MR(s_0^0)$ below, $MR(s_{15}^2)$ is used.</p> <p>III. Since s_0^0 is in CLOSED,</p> $MR(s_0^0) = h(s_0^0) \sum_i \Lambda(c_i) \sum_j \Lambda(p_j) \sum_k \max[\Lambda(s_k^1) MR(s_k^1)]$ $= h(s_0^0) \times \Lambda(c_1) \times \Lambda(p_8) \times \Lambda(s_{15}^2) \times MR(s_{15}^2)$ $= 1.0 \times 1.0 \times 1.0 \times 1.0 \times 0.5$ $= 0.5$ <p>IV. Solution s_{15}^2 is included in G_0, instead of s_{15}^1 and its descendents.</p>	<p>OPEN = $\{s_{15}^2, s_{16}^1, s_{17}^1, s_{18}^2, s_{19}^1, s_{20}^1, s_{21}^1\}$</p> <p>CLOSED = $\{s_0^0, s_{15}^1, s_{19}^1\}$</p> <p>$G_0 = \{s_0^0, c_{I_1}, p_8, s_{15}^2, w_{I_1}, u_{1,8}, v_{8,15}^2\}$</p>	Users are presented with the resultant G_0 , shown graphically as a tree representation.

Table A-4 A Braking System Conceptualization Case (6 of 7)

Appendix

CDH Algorithm	Design Processes	OPEN; CLOSED; G_0	User Actions
7) If $solve_status(s_0^0) = I$, exit with G_0 as the solution graph. Else if $MR(s_0^0) = -\infty$, exit with failure. Else go to Step 2.	No Action	$OPEN = \{s_{15}^2, s_{16}^1, s_{17}^1, s_{18}^1, s_{20}^2, s_{21}^1\}$ $CLOSED = \{s_0^0, s_{15}^1, s_{19}^1\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^2, w_l, u_{1,8}, v_{8,15}^2\}$	No Action
2) Using f_2 , select a node n^* (if initial, just s_0^0). Remove n^* from OPEN and place it on CLOSED.	$f_2(s_k' \in G_0 \cap OPEN)$ $= \min_k [h(s_k' \in G_0 \cap OPEN)]$ $= s_{15}^2$ $n^* = s_{15}^2$ is assigned	$OPEN = \{s_{16}^1, s_{17}^1, s_{18}^1, s_{20}^2, s_{21}^1\}$ $CLOSED = \{s_0^0, s_{15}^1, s_{19}^1, s_{15}^2\}$ $G_0 = \{s_0^0, c_l, p_8, s_{15}^2, w_l, u_{1,8}, v_{8,15}^2\}$	Users are presented the next logical node for expansion, n^* .
....			

Table A-4 A Braking System Conceptualization Case (7 of 7)