A critical review of General Design Theory

Yoram Reich[1]
Department of Solid Mechanics, Materials and Structures
Faculty of Engineering
Tel Aviv University
Tel Aviv 69978
Israel

In *Research in Engineering Design*, 7(1):1-18, 1995

Keywords: design theory, mathematical theory, topology, model of design, computer-aided design, CAD, modeling

Running head: A critical review of General Design Theory

Abstract:

This study is a critical review of General Design Theory (GDT), a mathematical theory of design. The study reviews the assumptions (axioms) and predictions (theorems) of GDT with respect to design and illustrate them with a simple example. The scope of GDT with respect to design, the guidelines it provides for building computer-aided design (CAD) systems, and the possibility of implementing these guidelines are examined. GDT assumptions are too restrictive to apply directly to design, and several potential avenues for modifying the theory to attempt to broaden its scope are discussed. Nevertheless, these modifications may not lead to proving strong predictions about design. Treating GDT as a model, rather than as an accurate reflection of design, allows treating the guidelines as hypotheses to be tested empirically. The paper discusses these guidelines and some experimental implementation that embody some of them.

[1]Part of this work was performed while the author was with the Engineering Design Research Center, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213

1 Introduction

Over the last few decades, significant amount of research has been devoted to developing design theories (Hubka and Eder, 1988; Suh, 1984; Warfield, 1990; Yoshikawa, 1981). These theories have been classified into various conceptual categories such as empirical/descriptive or prescriptive (Finger and Dixon, 1989) as well as into their geographic origin (Arciszewski, 1990; Eder, 1990; Hundal, 1990; Tomiyama, 1990). A notable exception in the set of design theories is General Design Theory (GDT) (Yoshikawa, 1981; Tomiyama and Yoshikawa, 1986; Tomiyama et al, 1989; Tomiyama, 1994) in that it is based on mathematical foundations.

Since its presentation in English over 13 years ago (Yoshikawa, 1981), GDT has been hardly used or even referred to by researchers other than its developers. Those who referred to GDT's concepts never used them to guide their work in a serious manner. Many explanations of this situation can be offered.

One possible explanation is tied to the goals of GDT as stated by Yoshikawa (1981, p. 35). GDT

> ultimately aims at clarifying the human ability of designing in a scientific way, and at the same time, producing the practical knowledge about [...] design methodology. It has been also revealed that the general theory of design is useful, or essential, to construct CAD system in a scientific manner. The theory indicates what CAD systems should be, concerning the tasks to be imposed to computers, data structures, the role of a designer, design sequence, the interface of man and computer, etc.

Clearly GDT does not achieve all these goals and it is not apparent that it could. Combining this with the use of the qualifier "General" in the name of the theory, may have caused researchers to disbelieve its usefulness. Another plausible explanation is tied to different "trends" in design research. When design research focused mainly on prescriptive methods, GDT was ignored (and therefore misunderstood) due to the perceived complexity of its mathematical formulation. Such misunderstandings are manifested in the discussion at the end of Yoshikawa's 1981 paper. While the mathematics of GDT is not straight forward, it is not too complex either. Recently, when descriptive methods have received significant legitimacy, GDT may have been perceived to be too formal to be relevant to design. Such criticism, however, applies to any theory of design: no theory can capture all of design (Konda et al, 1992); rather, each theory provides one *perspective*, broad or limited, that may improve design understanding and practice.

Consequently, even though GDT does not achieve all its stated goals it can still be perceived as a *model* of design: an approximation of design (or what design is perceived to be) created to serve some goals. While GDT does not clarify the human ability to design, it provides guidelines for building computer-aided design (CAD) systems.

This paper is based on a careful study of GDT's definitions, axioms, theorems and their proofs. Through such study, particular choices in interpreting abstract terms into mathematical concepts were uncovered (e.g., in Theorem 1), and the proofs were checked and indeed verified. This alone is a critical examination of the mathematical underpinning of GDT. The paper also analyzes the way GDT models (or approximates) design and the nature of the guidelines for building CAD systems it proposes and their usefulness. Thus the usefulness of GDT as a theory of design is assessed.

It is important to introduce the interpretation of the term "design" used in this study. Design is defined as a process: Given a description of a desired function and constraints, called *specification*, provide a description of an artifact that produces the function and satisfies the constraints. This description is called *artifact description* or *artifact structure*.

Organization. The remainder of the paper is organized as follows. Section 2 describes the domain of chairs which is used to illustrate the ideas discussed in this paper. Section 3 reviews the concepts of **GDT**, emphasizing the important assumptions and results of the theory. Section 4 discusses the theory, the insight it provides about design, and its implications for building CAD systems. Section 5 summarizes the paper.

2 The domain of chairs

Figure 1 depicts eight chairs that are used to explain the concepts discussed in this study. They will be referred to as the chairs domain. Even though this domain is too simple to illustrate the full complexity of design, it does not reflect the scope of GDT which is not limited in the representation of artifacts (e.g., to attribute-value pairs), or to a small number of attributes of limited types.

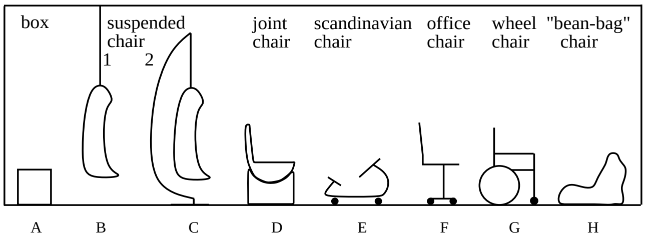


Figure 1: The chairs domain

Each chair in the figure is denoted with a letter. The chairs provide some *functionality* that is summarized in Table 1. Each row describes a different function of a chair. The "+" in the table denotes that a chair provides the corresponding function, and a "−" denotes its lack thereof. Additional functions that chairs may have but that are not mentioned include *provides comfortable seating*, *provides access to ceiling*, *provides resistance to fire*, etc. In addition to providing functions, each chair has properties that can be observed and therefore describe the *attributes* or the *structure* of the artifact; some of these are summarized in Table 2. Additional observable properties that chairs may have include *color*, *material texture*, *type of upholstery*, *dimensions*, etc. We will limit the discussions to the properties in Tables 1 and 2.

Table 1: Functional properties of chairs

		chair							
	function	A	B	C	D	E	F	G	H
1	*seats*—prevents a downward movement of the body	+	+	+	+	+	+	+	+
2	*supports back*—supports an upright posture	+	+	+	+	+	+	+	+
3	*revolves*—revolves around a vertical axis	+	+	+	+	+	+	+	+
4	*movable*—can be easily moved	+	—	—	—	+	+	+	+
5	*constrains back*—constrains backward movement of back	—	+	+	—	—	+	+	—
6	*easy to manufacture*—has a simple design with standard components	+	—	—	—	—	+	—	+
7	*aesthetic*	—	+	+	+	+	—	—	—

Table 2: Observable properties of chairs

		chair							
	structure	A	B	C	D	E	F	G	H
1	*has a seat*	+	+	+	+	—	+	+	—
2	*has a back support*	—	+	+	+	—	+	+	—
3	*has legs*	—	—	+	—	—	+	+	—
4	*has wheels*	—	—	—	—	+	+	+	—
5	*has a vertical rotational dof*	—	+	+	+	—	+	—	—
6	*is lightweight*	+	+	—	—	+	+	+	+
7	*has a hanger*	—	+	+	—	—	—	—	—
8	*has a brake*	—	—	—	—	—	—	+	—

Naturally, there are functions that are directly derived from the structure of a chair. For example, a chair that *has wheels* is *movable* or a chair that *has a vertical rotational degree of freedom (dof)* can *revolve*. Note that this structure-function relation may be imprecise; for example, chair C with a *rotational dof* does not allow for 360° rotation. Other functions may be more complex and could not be inferred from one observable property. To illustrate, a chair can *support back* although it does not have a *back support*. For instance, chair A provides back support due to its location near a wall; its function is context dependent. Also, chair E provides back support due to its structure although is does not have a *physical* back support. Some functions may qualify other functions. For example, the function *constrains back* qualifies the function *support back*. This function is quite complex to assess: Chairs F and G constrain back movement due to their structure, while chairs B and C constrain it due to static considerations; chairs D and H do not constrain back movement, while chairs A and E do not even have a physical support.

The previous examples concentrated on inferring potential functionality from artifact structure. This is useful in *analysis*. In contrast, designing is mainly concerned with *synthesis*: the generation of artifact structure that will satisfy a desired function. For example, the specification of a chair that will be *movable* and *constrain back* leads to two potential designs: F and G. These designs can be generated in two ways. The first way starts with $\{A, E, F, G, H\}$ as the *movable* designs and refines them with the *constrain back* property. The second way starts with $\{B, C, F, G\}$ as the *constrain back* designs and refines them with the *movable* property. The most concise description of the solution is the chairs that have *physical back support* and *wheels*. Another description, that does not seem relevant but is nonetheless correct, is the chairs that *have legs* and are *lightweight*. Note that the refinement process was made easier by the use of the eight representative chairs as mediators between the specification and the design description. In the absence of these chairs, the process might have been more difficult.

As another example, assume that in addition to the previous specification, it is also required that the chair will be *aesthetic*. There is no chair that satisfies the three functions. A *redesign* process can be invoked by taking the current candidate designs $\{F, G\}$ and retracting either the *movable* or the *constrain back* specification properties and then trying the new specification. Since we are working with extensional descriptions of candidate designs, there will still not be any candidate that satisfies all three specification properties. Nevertheless, we will have three sets of *nearly* good candidates: (1) *constrain back* and *aesthetic* $\{B, C\}$, (2) *movable* and *aesthetic* $\{E\}$, and (3) *movable* and *constrain back* $\{F, G\}$. If the set of designs was not confined to the eight chairs, D could have been made to *constrain back* by adding a brake to its joint (i.e., circular) seat-base connection. In contrast, the first group cannot be easily made movable since B is attached to the ceiling and C would have to receive wheels and a brake.

The chair domain consists of eight chairs only, and each described by the few properties in Tables 1 and 2. This simple domain clearly does not reflect the complexity of design. For example, it is not detailed enough to illustrate the selection of material and production techniques nor the sizing and proportions of various chair parts, both have substantial influence on the aesthetic and stylistic value, as well as the cost, of chairs. The incorporation of additional properties (both functional and observable), such as those dealing with comfort analysis using a detailed model that better approximates real chairs (Eastman, 1991), can elaborate the domain. The addition of these properties must be accompanied by the addition of many chairs that will extend the set of possible chairs that can be designed. While it would be nice to have these properties in our examples, it would complicate the example to a point that it will lose its usefulness in this review. it is important to emphasize again that GDT is not limited to the present chairs domain; it applies equally to a chairs domain that includes as many chairs and properties as we wish. Of course, such applicability has its costly memory and computational complexity properties as mentioned in the examples of Definitions 7 and 8 and in Section 4.

3 General Design Theory

Yoshikawa's General Design Theory (GDT) is a formal mathematical theory of design. It attempts to cast design in the framework of set theory. GDT starts with assumptions about the nature of objects and uses them to prove theorems about the nature of design. GDT makes interesting claims about design. As such, it has a flavor of a descriptive theory of design, but more importantly, GDT provides a prescription for the development of CAD systems.

This section reviews GDT's terminology, definitions, assumptions, and theorems while using the chairs domain as a working example throughout. While reviewing GDT, the section points to its important assumptions, how they determine the success of the theory (measured as the ability to prove theorems about design), and how they can be relaxed.

3.1 Preview

The purpose of this preview is to expound some of GDT's mathematical concepts. Since the theory of topology can be viewed as a generalization of the concept of continuity (Sutherland, 1975), it may help to introduce some of the concepts in topology through an example from the area of continuous real mappings. For the purpose of simplicity, assume that in a certain design domain, a specification as well as the artifact are described by real numbers. A design process is a mapping from the desired function to the artifact description. Figure 2 illustrates an example domain. A mapping from the artifact description to the function (i.e., analysis) is modeled as $1 + \sin\frac{1}{x}$ from 0 to a, and by an arbitrary mapping that is asymptotic to $-\infty$ at e, from a to e. The mapping from the function to the artifact description (i.e., synthesis) is the "inverse" of this mapping. Qualitatively, such a mapping might describe, for example, the mapping between the observable properties and the *easy to manufacture* property. The value of the property can be determined by various methods (e.g., (Boothroyd and Dewhurst, 1989)).

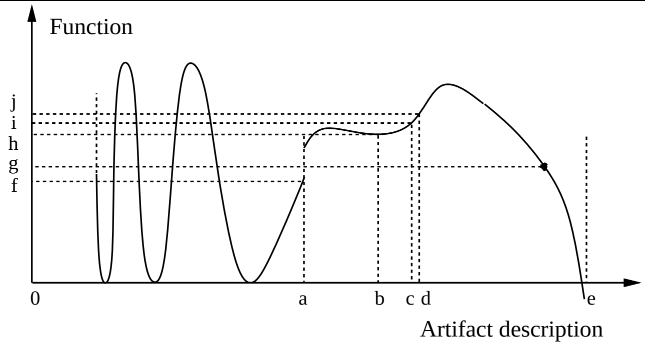


Figure 2: A simplified design domain

There are several important properties of continuous mappings that are of interest in various design tasks such as synthesis, analysis, or redesign: *distance*, *continuity*, *convergence*, and *transformation*. The *distance* between two functions or between two artifact descriptions is a useful concept. For example, distance can be used in evaluation. It can provide answers to questions such as: how close are two functionalities, or how far is the functionality of the candidate from the desired functionality? In our example, both functions and artifact descriptions are described by real numbers, therefore, distances can be calculated by subtraction.

Continuity is a process-oriented concept. It guarantees that a small change in the design description will result in a small change in the artifact functionality and vice versa. Therefore, if the current candidate's functionality differs slightly from the required function, a small modification to the structure may be sufficient. For example, a required move from functionality h to i can be achieved by a small move from b to c. In contrast, an artifact whose description is close to 0 cannot be analyzed since the mapping is discontinuous at 0; therefore, an artifact cannot be synthesized to have a description close to 0. Further, a required move from f to g around point a, which is a discontinuity point, cannot be achieved by a small change in the artifact description.

Convergence is also a process-oriented concept; it provides a different perspective of continuity. Convergence guarantees that a sequence of incremental refinement changes (e.g., from b to c to d) will cause only small incremental changes to functionality (e.g., leading from h to i to j). A negative example are the small changes when moving from a to 0, or from d to e, that do not converge to a specific or finite artifact functionality, respectively.

A *transformation* that conserves the continuity or convergence properties is useful in design because it allows creating different viewpoints of the desired functionality and the partial design description that may simplify or direct future design steps. In topology, such a transformation is called *homeomorphism*.

These concepts are generalized in the theory of topology. As such, a topological structure of objects provides an interesting perspective of viewing design. With this background we turn to reviewing General Design Theory.

3.2 Preliminary definitions

DEFINITION 1: An *entity* is a real object that existed, exists presently, or that will exist in the future.

EXAMPLE : Any chair that existed since the invention of the first chair and that will exist is an entity. For the purpose of simplification, assume that Figure 1 contains all these entities, even though, the set may be infinite.

DEFINITION 2: The set of all objects is called the *entity set*.

EXAMPLE : The chairs domain can be viewed as an entity set.

DEFINITION 3: An *attribute* is a physical, chemical, mechanical, or any other property that can be observed or measured, potentially through the use of an instrument. Each entity has *values* for its attributes.

The terms property and value are not defined by GDT but can be given mathematical meaning (as suggested by T. Yagiu in the discussion following Yoshikawa's (1981) paper). Note that usually, property and attribute are synonymous. However, in **GDT**, the term attribute has a special meaning as defined above.

EXAMPLE : The properties listed in Table 2 are all attributes of chairs. They all can be observed or measured by some instruments. The table specifies the property-value pairs for each chair.

DEFINITION 4: When an entity is subjected to a situation, it displays a behavior which is called a *functional property*. The collection of functions observed in different situations is the *functional description* of the entity.

EXAMPLE : The properties listed in Table 1 are all functional properties of chairs. The table specifies the functional behavior manifested by each chair. For example, *constrain back* is a behavior manifested when a human leans at the back, e.g., chair D will swing backward, therefore, it does not constrain back. For chair D, it is clear that this function is a direct consequence of the joint seat-base connection; in most cases however, the function-structure relationships is less clear—an unfortunate situation for design.

DEFINITION 5: The representation of an object is called *concept of entity*.

We will refer to the concept of entity as *entity* as well since from now on, only representations are discussed, not real objects.

EXAMPLE : The representation of a chair using the function and structure properties from Tables 1 and 2 is a concept of entity. Also, a representation of a chair using its picture in Figure 1 as an icon is a concept of entity.

Until now, the discussion has concentrated on single entities and their representation. The important concept of *classification* is now introduced.

DEFINITION 6: A classification over the entity set is a division of the entities into several classes. Each class is called an *abstract concept*. The set of *all* abstract concepts is denoted by $\mathcal{T}$.

EXAMPLE : Chairs A and B can form a class and the remaining chairs can form another class. A more meaningful classification can be obtained by classifying the chairs based on their properties. For example, the property *has legs* divides the set of chairs into two classes: chairs with legs $\{C, F, G\}$ and chairs without legs $\{A, B, D, E, H\}$. A classification can be based on several properties. For example, the properties *has legs* and *has wheels* divide the set into two classes: $\{A, B, C, D, E, H\}$ and $\{F, G\}$.

DEFINITION 7: The set of all functions, called the *function space*, is the set of *all* the classes from all the classifications of the functions. It is denoted by $\mathcal{T}_1$.

EXAMPLE: In principle, the function space for the chairs domain can contain 2^8 functions (i.e., the size of the power set of the set of chairs),[1] each function being a different classification over the set of chairs. For example, the functions *support back*, *movable*, and *aesthetic* define a classification that singles out chair E from the whole set of chairs. The number of functions, however, is not accurate since some of the potential classifications (e.g., *easy to manufacture* and *aesthetic* or *is movable*, *aesthetic* and *constrain back*) do not contain any chairs.

DEFINITION 8: The set of all artifact descriptions, called the *attribute space*, is the set of *all* classes of all the classifications of attributes. It is denoted by $\mathcal{T}_0$.

EXAMPLE: The attribute space for the chairs domain can contain 2^8 potential artifact descriptions. For example, the attributes *has legs* and is *lightweight*, designate the artifact descriptions of chairs F and G. Some of the classifications may be empty; however,

[1]This size (i.e., 2^8) may pose problems for a computational implementation of a realistic chair domain.

often it may be easy to construct a new chair that will have the classification attributes. For example, the attributes *has wheels* and *is hanging* designate an description of an nonexistent chair; nevertheless, is is easy to hang a chair on wheels sliding on a rail attached to the ceiling or modify chair C by adding wheels to its base.

3.3 GDT's axioms

GDT's axioms convey the assumptions of the theory about the nature of objects and their potential manipulation to achieve a desired functionality. These axioms are the foundations of the theorems discussed later. The importance of the axioms lie in their coherent formulation. This allows an assessment of their relevance to design.

AXIOM 1 (**Axiom of recognition**): Any entity can be *recognized* or *described* by its attributes and/or other abstract concepts.

The meaning of this axiom is that GDT supports the extensional description of concepts.

EXAMPLE: Each of the chairs in Figure 1 can be easily singled out from the set of chairs by using one or more of its artifact description attributes. For example, chair A can be recognized as the only chair that *has no vertical rotational dof* and *no wheels* and chair C is the only chair that is *not lightweight* and *has a leg*.

The axiom fails if two chairs have the same description. In this case, the two chairs cannot be differentiated. To illustrate, in reality, we often face decisions about which product to buy and our description (which reflects our knowledge) is not sufficient to differentiate between the products. The solution to this problem lies in adding attributes that differentiate between the candidate artifacts. This entails having a flexible representation of entities, but it may require that an entity be described by a large, possibly infinite, number of attributes to enable its recognition. While this is not critical for GDT, it may require having large storage capacity and processing speed which may burden a computational implementation of the theory. These issues are further discussed in Section 4.2.

AXIOM 2 (**Axiom of correspondence**): The entity set and its representation have *one-to-one* correspondence.

EXAMPLE: If each of the chairs in Figure 1 is perceived as a real object (i.e., entity) and its description given in Tables 1 and 2 as the concept of entity, the axiom says that there is a one-to-one mapping between them.

The discussion on the previous axiom applies to the present axiom, therefore, to the representation of entities. An appropriate representation may require the use of an infinitely long property-value list. Of course, this is practically impossible. Nevertheless, the theory assumes the availability of resources that overcome this difficulty.

> **Axiom 3 (Axiom of operation):** The set of all abstract concepts is a *topology* of the entity set.

A topology $(S, \mathcal{T})$ is a mathematical entity consisting of a set S and the set $\mathcal{T}$ of subsets of S that satisfies the following properties:[2]

(1) $\phi \in \mathcal{T}$ and $S \in \mathcal{T}$,
(2) for every s_1, $s_2 \in \mathcal{T}$, $s_1 \cap s_2 \in \mathcal{T}$, and
(3) for $s_i \in \mathcal{T}$, $i \in \Lambda$, Λ a countable set, $\cup_i s_i \in \mathcal{T}$.

The fact that the set of all abstract concepts is a topology, influences both its structure (through properties (1), (2), and (3) of topology) and the possible operations on it (properties (2) and (3)). Axioms 1 and 3 demand that all entities be treated equally, thus, excluding the enforcement of a hierarchy on objects (Kurumatani and Yoshikawa, 1987; Veth, 1987).

> **Example:** The simplest topology over the set S of chairs is $\mathcal{T} = \{\phi, S\}$. Another obvious topology is the power set of S (having $2^8 = 256$ elements), also called the discrete topology. Both of these topologies are not too interesting. Another topology $\mathcal{T}$ can be constructed such that $\{\phi, \{A, H\}, \{B, C, D\}, \{E, F, G\}, S\} \subseteq \mathcal{T}$. The inclusion of these subsets of S may be caused by the need to *differentiate* between the entities in these different classes. To complete the topology, $\mathcal{T}$ must satisfy the three properties listed above. Therefore, $\mathcal{T} = \{\phi, \{A, H\}, \{B, C, D\}, \{E, F, G\}, \{A, B, C, D, H\}, \{B, C, D, E, F, G\}, \{A, E, F, G, H\}, S\}$ Some of these abstract concepts can be labelled; for example, $\{A, H\}$ can be labelled as the set of simplest chairs, whereas $\{B, C, D\}$ is the set of *immovable* chairs.

3.4 Ideal knowledge

> **Definition 9:** *Ideal knowledge* is the one that knows *all* the entities and can describe each of them by abstract concepts without ambiguity.

This definition adds another constraint to the axioms: the recognition of entities should be without ambiguity. In the chairs domain, this assumes that one can recognize that two chairs are different through observing their attributes, which is clearly correct for the chair domain. The term "without ambiguity" can be interpreted as one of the instances of the topological concept of *separation* which is formalized in the axiom of separation.

[2] $(S, \mathcal{T})$ is the mathematical notation of a space, we often denote it by $\mathcal{T}$ when it is clear what the set S is.

Axiom 4 (**Axiom of Separation,†[3]**): There exists a hierarchy of separation/recognition ability:

T_0: For each pair $a \neq b$ in S there is $U \in \mathcal{T}$ such that $a \in U$ and $b \notin U$ or vice versa.

T_1: For each pair $a \neq b$ in S there is $U, V \in \mathcal{T}$ such that $a \in U$ and $b \notin U$ and $b \in V$ and $a \notin V$.

T_2: (Hausdorff). Similar to T_1, but $U \cap V = \phi$.

T_3: (A regular T_1 space). Satisfies T_1 and for every closed[4] set $A \subset \overline{S}$ (i.e., closure of S) and for every $b \in S - A$ there exists a pair of disjoint open sets U, V such that $b \in U$ and $A \subset V$. (T_3 is a generalization of T_2 where A is a set instead of a single entity.)

T_4: (A normal T_1 or a compact Hausdorff space). Satisfies T_1 and for every pair of disjoint closed sets $A, B \subset \overline{S}$ there exists a pair of disjoint open sets $U, V \in \mathcal{T}$ such that $A \subset U$ and $B \subset V$.

T_5: (A completely normal T_1 space). Satisfies T_1 and for every pair of closed sets $A, B \subset \overline{S}$ with $\overline{A} \cap B = A \cap \overline{B} = \phi$, there exists a pair of disjoint open sets $U, V \in \mathcal{T}$ such that $A \subset U$ and $B \subset V$.

Metric space: There exists a metric on the space.

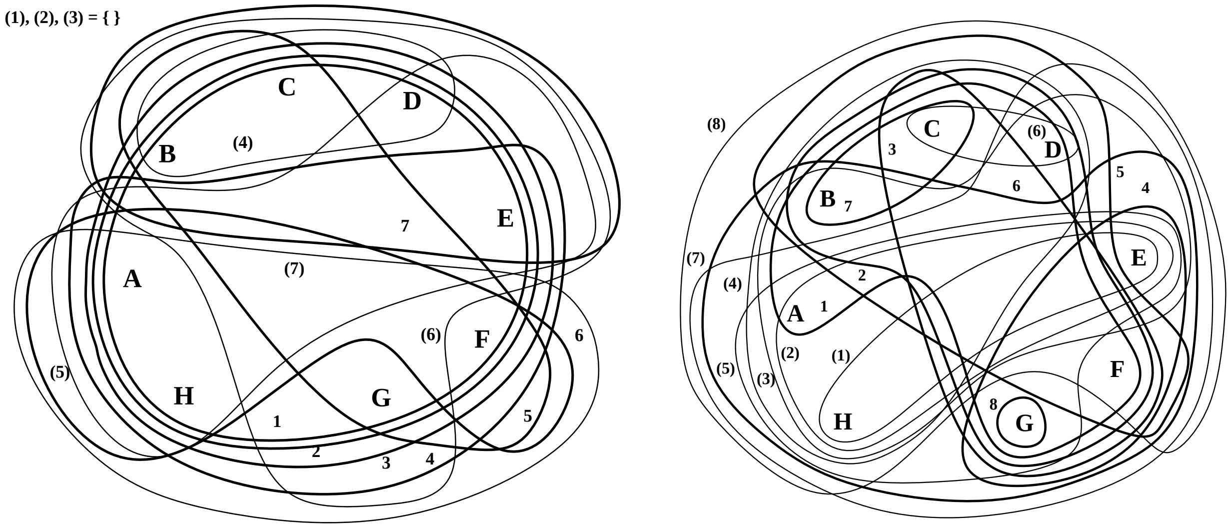
It can be shown that *metric* $\Rightarrow T_5 \Rightarrow T_4 \Rightarrow T_3 \Rightarrow T_2 \Rightarrow T_1 \Rightarrow T_0$, and that none of these implications is reversible. Therefore, the type of separation defines an order on topological spaces.

Example: To facilitate the recognition without ambiguity of each chair, $\mathcal{T}_0$ must include each chair. The second and third properties of topology immediately imply $\mathcal{T}_0 = 2^S$. However, to be useful, we want the topology to have some meaning with respect to the domain. Therefore, it is preferable that the abstract concepts be derived from properties describing chairs. Figure 3 illustrates the classes created from the properties in Table 2. We will use these classes as the "topologies" over the set of chairs even though they do not satisfy the properties of topologies. Each line in the figure circles the chairs that satisfy the property number that is written along the line. A number in parenthesis denotes that the circled chairs have "$-$" as their value for this property. We see that each chair can be recognized through its attribute description since in Figure 3(b) each chair is enclosed by lines. Figure 3(a) shows two pairs of chairs, $\{B, C\}$ and $\{A, H\}$, each of which contains chairs that cannot be differentiated based on their functionality since they are not separated by a line. This can be remedied by adding the function *allows for easy floor cleaning* whose values will be "+" for all the

[3] Whenever this symbol is used hereafter, it denotes that an axiom, theorem or definition is not a part of GDT.

[4] Terms that were not defined in the text are defined in the Appendix.

movable chairs and chair B and "$-$" for chair C, and the function *shapeless* whose value is "$+$" for chair H and "$-$" for the remaining chairs. The inability to differentiate between entities based on function indicates that given a specification, there may be several candidate designs that satisfy it.



(a) Function "topology" (b) Attribute "topology"

Figure 3: The function and attribute "topologies"

THEOREM 1: The ideal knowledge is a Hausdorff's space.

This theorem almost follows from the definition of ideal knowledge. Although the definition only warrants a T_1-type separation and not necessarily T_2-type, the proof of the theorem makes use of the latter and stronger interpretation of the term "without ambiguity." It remains to be determined in future work whether a weaker interpretation can still lead to proving the same or similar theorems about design that GDT proves.

EXAMPLE: Observing the chairs domain in Figure 3(b) and considering as abstract concepts *only those drawn in the figure*, entities A and F cannot be distinguished in the Hausdorff sense. They can, however, be distinguished in the T_1 sense. For example, $\{A, H, G, E\}$, the abstract concept *does not have vertical rotational dof*—denoted by (5), and $\{C, G, F\}$, the abstract concept *has legs*—denoted by 3, differentiate between A and F but these abstract concepts are not mutually exclusive. On the other hand, entities C and G can be easily differentiated in the Hausdorff sense. Note that if we allow using the second property of topology, then A and F could be distinguished in the Hausdorff sense as well. In particular, the abstract concepts *has a seat* and *does not* *have a back support*, $\{A\}$, and *has wheels* and *has a vertical dof*, $\{F\}$, clearly separate between A and F. The fact that one cannot separate between entities in the Hausdorff sense means that given available knowledge about chairs it may be unclear what are the differences between candidate designs.

The next definition formalizes the intuitive definition of design specification given in the introduction. This definition is in correspondence with Definition 7.

DEFINITION 10: The design specification, T_s, designates the function of the required entity by using abstract concepts.

This definition implies that $T_s \in \mathcal{T}$. Later, we see that a solution to a design problem is $s \in T_s \in \mathcal{T}$.

THEOREM 2: The specification can be described by the intersection of abstract concepts.

EXAMPLE: It is natural to describe the specification of an object by the intersection of abstract concepts since the specification describes functions that the desired chair must fulfill. Therefore, a specification that a chair must *revolve* and be *movable* is described easily (using Table 1) by $\{A, B, C, D, E, F, G, H\} \cap \{A, E, F, G, H\} = \{A, E, F, G, H\}$. A solution to this design problem is any $s \in \{A, E, F, G, H\}$. If the specification would insist on an *easily manufactured* design $\{A, F, H\}$, the result would be $\{A, E, F, G, H\} \cap \{A, F, H\} = \{A, F, H\}$. We see how the second property of topology is used to identify the set of candidate designs for each specification.

THEOREM 3: The set of design specifications is a filter.

A filter allows access to an entity through a sequence of increasingly refined abstract concepts. Note that a filter is not a topology; in particular, it does not contain ϕ.

EXAMPLE: If the set of design specifications contains $\{A\}$. Then from the definition of a filter, it contains all the supersets of $\{A\}$ appearing in Figure 3(a): $\{A, F, H\}$, $\{A, F, G, H\}$, $\{A, D, E, H\}$, $\{A, E, F, G, H\}$, and S. The design A can be approached gradually by traversing a subset of this sequence starting from S. Note that in the chairs domain as depicted in Figure 3, the design specification does not include $\{A\}$. This will prevent obtaining it as a single candidate design.

THEOREM 4: $\mathcal{T} \supset \mathcal{T}_0$ and $\mathcal{T} \supset \mathcal{T}_1$.

This theorem establishes the relationships between the concepts of function and the ideal knowledge and the concepts of attributes and ideal knowledge.

> Definition 11: A design solution is an entity s that is *included in* its specification and contains its necessary manufacturing information.

The notion of "contains its necessary manufacturing information" is not defined but can be interpreted. To illustrate the meaning, recall that ideal knowledge contain all the chairs that were and will be manufactured. There will be similar chairs that will differ in the technology by which they are manufactured or in the tolerances on their dimensions. In order to differentiate between these chairs by some properties (i.e., abstract concepts), thereby satisfying Definition 9, the description of chairs (for a realistic chairs domain, not the one we use) must contain all their dimensions, tolerances, and manufacturing techniques. Consequently, the description contains the necessary manufacturing information.

> Theorem 5: The entity concept in the ideal knowledge is a design solution.

This theorem states that *every* entity is a design solution because it satisfies some requirements and contains the necessary manufacturing information. An intuitive explanation was given after Definition 11. The formal reason why each entity contains all its manufacturing knowledge lies in the mathematical concept of neighborhood. A neighborhood of an entity is an abstract concept that contains the entity. Many neighborhoods can be identified for each entity. Some of these neighborhoods (or abstract concepts) represent manufacturing processes as well as tolerances.

> Example: In the specification *revolve* and be *movable* and *easy to manufacture*, described before, A, F, and H are the design solutions. The neighborhoods of F that contain the manufacturing knowledge are those elements in the attribute topology that contain F (see Figure 3(b)): $\{A, B, C, D, F, G\}$, $\{B, C, D, F, G\}$, $\{C, F, G\}$, $\{E, F, G\}$, $\{B, C, D, F\}$, $\{A, B, E, F, G, H\}$, $\{A, D, E, F, G, H\}$, and $\{A, B, C, D, E, F, H\}$. Of course, these neighborhoods correspond to the attributes of chair F that appear in Table 2. The translation of these attributes to manufacturing information is simple: the chair will be manufactured with a seat, back support, legs, wheels, etc.
>
> Note that if we accept $\{A, F, H\}$ as the solution, we would be able to locate only partial information related to the neighborhoods that contain the final candidate set, $\{A, F, H\}$. These neighborhoods are $\{A, B, E, F, G, H\}$, $\{A, D, E, F, G, H\}$, and $\{A, B, C, D, E, F, H\}$, corresponding to the attributes *is lightweight*, *does not have a hanger*, and *does not have a brake*, respectively.

> Theorem 6: The entity concept in the attribute space $\mathcal{T}_0$ is a design solution. Each of the attributes can be perceived as manufacturing information.

The attribute space is the one that is created by the observable properties that are needed to manufacture an entity. Therefore an entity in the attribute space is a design solution. The next definition formalizes the notion of design solution described in the introduction.

THEOREM 7: The design solution is represented by the intersection of classes of S that belong to the attribute space $\mathcal{T}_0$.

EXAMPLE: From Figure 3(b) it is clear that all the chairs are design solutions since they are divided by lines.

The next definition formalizes the intuitive definition of design given in the introduction. Its two subsequent theorems establish the nature of design in the state of ideal knowledge.

DEFINITION 12: Design process is the designation of a domain in the attribute space $\mathcal{T}_0$ which corresponds to a domain specifying the specification in $\mathcal{T}$.

THEOREM 8: If the function space is a subspace of the attribute space, then design is complete when the specification is provided.

The design process that corresponds to this situation is *choosing a design from a catalogue.*

EXAMPLE: Let a specification be to design a chair that *constrains back* and that is *movable*. We can illustrate this design by going through Table 1 or Figure 3. Specification 1 (a chair that *constrains back*) is $\{B, C, F, G\}$. The addition of specification 2 (*a movable* chair), $\{A, E, F, G, H\}$, generates the specification $\{F, G\}$. The only way that a design solution exists is that there is a class in the attribute space that is a subset of $\{F, G\}$. In the chairs domain, G clearly satisfies this property since it is a singleton set identified by the attribute *has a brake*. The neighborhoods of this design include all the manufacturing information in the form of abstract concepts representing the attributes in Table 2. F is also a design solution even though its description requires a conjunction of attributes.

THEOREM 9: In the ideal knowledge, design is completed immediately when the specification is described.

EXAMPLE: Let a specification be to design a chair that *constrains back*, is *movable*, and is *easy to manufacture*. Specification 1 (a chair that *constrains the back*) is $\{B, C, F, G\}$. The addition of specification 2 (*a movable* chair), $\{A, E, F, G, H\}$, generates the specification $\{F, G\}$. The further addition of specification 3 (*easy to manufacture*), $\{A, F, H\}$, leads to the combined specification $\{F\}$. At this stage all the specification properties have been satisfied and the resulting design solution is $\{F\}$ (provided that we use conjunctions of properties).

The next definition refines Definition 12.

> **Definition 13**: Design is a mapping between the function space to the attribute space.

This means that design is an algorithmic process, which includes the catalogue design as a special case. Recalling that the function and attribute spaces are topologies over the same set of entities, this mapping is implicitly built into these topological structures.

> **Theorem 10**: The identity mapping between the attribute space and the function space is continuous.

The interpretation of the theorem is that $\mathcal{T}_0 \supset \mathcal{T}_1$. Intuitively it suggests that any class in the function space is equal or contain several classes from the attribute space; therefore, if a specification is determined there will always be at least one candidate solution.

> **Theorem 11**: If two design solutions can be discriminated functionally, then $\mathcal{T}_0 \subset \mathcal{T}_1$.

This theorem holds only if both topologies are equal since from Theorem 10, $\mathcal{T}_0 \supset \mathcal{T}_1$. Since this equally is unlikely, one often cannot discriminate between the functionality of several candidate designs. This is where subjective judgment come into play in design.

Summary of ideal knowledge. The state of ideal knowledge is characterized by the ability to separate between any two entities. This separation may require the use of infinitely long descriptions of entities. The separation between entities and the requirement that the knowledge structure be a topology guarantee that design would terminate with a solution after a specification is given. The operation on large descriptions of entities by means of set operations that manipulate the topology (i.e., properties 2 and 3 of topology), or by other means for handling extensional descriptions of entities, may require infinite memory capacity and processing speed.

GDT-Ideal, denoting the state of ideal knowledge, restricts the nature of knowledge to have two perfect properties and through them guarantees the termination of design. Design is limited to the selection from a catalogue or the utilization of an algorithmic mapping between functions to attributes. Real design never has these properties and therefore, its termination cannot be guaranteed. Nevertheless, several additional design strategies, and assumptions can guarantee design in a less ideal state. These issues are discussed next.

3.5 Real knowledge

Real knowledge cannot be structured as a perfect topology and cannot be manipulated by infinite resources. These discrepancies from the state of ideal knowledge require two important modifications to the theory. First, only finite descriptions of entities can be manipulated. Second, instead of assuring the successful termination of design, alternative models of design must be devised to allow for solving design problems. To illustrate the need for these modifications, consider designing a chair that will be *aesthetic* and *movable*, and will *seat* and *constrain back*. In the present chairs domain this is impossible. The intersection of the classes corresponding to these desired properties does not contain any chair although the existence of such a chair is guaranteed by GDT-Ideal, if the domain had a topological structure. The design can be accomplished by the addition of a *brake* to chair D to make it *constrain back*. As we said before, if we elaborated the domain with additional chairs and properties, this particular problem would have been solved. However, in general, we cannot locate and fix such imperfections *a priori*. In the real knowledge case, when such void is found, it could be fixed by adding entities, or properties (e.g., abstract concepts). These additions can be easily accommodated by GDT but, the *findings* of these entities or properties are creative acts and are not discussed in GDT.

The remainder of this section reviews the definitions, assumptions, and theories of the extension of GDT to real knowledge denoted by GDT-Real. Since most of the effort in this part of GDT goes into proving similar theories about design as in the state of ideal knowledge, examples will be limited to the distinctions between GDT-Ideal and GDT-Real.

Definition 14: A *physical law* is a description about the relationship between object properties and its environment.

Example: Physical laws include gravity which establishes the *lightweight* property. The remaining artifact description properties in the chairs domain can be determined by observation. These are associated with optics laws which explains how observable attributes are sensed by people.

Definition 15: An *attribute* is a physical quantity which is identifiable using a set of *finite* number of physical laws.

This definition is in correspondence with Definition 3 except for the addition of the *finite* adjective. Insisting on a finite number of laws has an important implication that is later given a precise meaning in Hypothesis 1 — one of the core assumptions of GDT-Real.

Definition 16: A concept of physical law is an abstract concept. It is formed if entities are classified based on physical manifestations of physical laws.

If each physical law corresponds to one and only one observable property, the topology of physical law is equal to that of the attribute space, otherwise it may be slightly different.

GDT now makes a distinction between S, the set of entity concepts, and $\tilde{S}$ which is the set of entities that do not contradict physical laws. This distinction is not too important since S is the representation of existing entities via one-to-one mapping. Therefore, S contains only feasible objects. The distinction between the two sets is ignored hereafter.

Definition 17: A feasible object is an object that does not contradict physical laws.

A similar definition is that an object whose attributes are manifestations of physical laws is feasible.

Example: An object that can be realized is by definition a feasible object. Since all the chairs are examples of existing designs, they are all feasible.

Theorem 12: The topology of physical law, $\mathcal{T}_p$, on $\tilde{S}$ is a *subspace* of $\mathcal{T}_0$.

The fact that $\mathcal{T}_p$ is a subspace of $\mathcal{T}_0$ guarantees that anything whose attributes can be associated with physical laws will be realizable as an artifact. Therefore, if specifications can be expressed by such attributes, design is possible. The definition of real knowledge is now formalized:

Hypothesis 1: The real knowledge is the set of feasible entity concepts which are made compact by coverings selected from the physical law topology.

Any finite space (e.g., the chairs domain) or a space with a finite number of open sets is compact. Formally, insisting on a compact space is a very restrictive hypothesis; it demands the existence of a finite subcover for each cover of the space. From the formal perspective, this hypothesis is as important foundation of GDT as the assumption that the structure of knowledge is a topology (Axiom 3). It, in addition to Definition 15, immediately allows to derive the following theorems including one of the major conclusions of GDT (Theorem 29). Conceptually, this hypothesis says that the description of entities is restricted to a finite number of properties. Note that this does not remove the memory and computational complexity properties as mentioned in the examples of Definitions 7 and 8.

To illustrate the importance of this hypothesis, compactness guarantees the feasibility of a specification (i.e., the intersection of the classes representing the specification properties is not empty) if it can be represented as closed sets and if the specification is feasible for any finite collection of requirements. This is formalized in the following theorem (Christenson and Voxman (1977); p. 67):

Theorem 13 (†): A space S is compact if and only if for each collection $\mathcal{C} = \{C_\lambda | \lambda \in \Lambda\}$ of closed subsets of S with the finite intersection property, $\cap C_{\lambda, \lambda \in \Lambda} \neq \phi$

Another immediate result is that the compactness property with the Hausdorff space property of real knowledge determine the metrizability of real knowledge. This implication is further discussed later.

The following four theorems are stated for completeness but are not discussed further.

THEOREM 14: The topology of $\mathcal{T}_0$ on the set of feasible concept $\tilde{S}$ is a compact Hausdorff space (T_4 in Axiom 4).

THEOREM 15: The real knowledge is second countable (i.e., has a countable basis).

THEOREM 16: The real knowledge is a closed subset of the (ideal) set of entity concept S.

THEOREM 17: If a continuous function $f : \tilde{S} \to R$ exists in the real knowledge, this function has maximum and minimum values.

This last theorem says that continuous functions have finite values ranging between two limits. This property is important when dealing with functions that represent properties of physical objects.

The next theorem is stated for completeness since it is used in proving subsequent theorems.

THEOREM 18: The real knowledge is a Lindelöf space.

Note that a Lindelöf space is a less restrictive notion of a compact Hausdorff space.

DEFINITION 18: Let $\mathcal{T}$ be the topology of abstract concept and Λ a countable set. *Feasible design specification*, $T = \cap_{\lambda \in \Lambda} T_\lambda$ where $T_\lambda \in \mathcal{T}$ and $T_\lambda \neq \phi$), is defined by the condition $\cap T_\lambda \neq \phi$.

This definition adds to Theorem 2 the restriction on a *countable* intersection of abstract concepts. This guarantees that the specification will be feasible (i.e., will contain candidate designs).

THEOREM 19: Any feasible specification in the real knowledge has a cluster point.

This conclusion is equivalent to Hypothesis 1 (see Schubert, 1968, p. 69). The set of candidate designs for a particular specification will be a set of entities that also contains a cluster point. The next theorem shows how a converging subsequence can be built that will arrive at a single solution. The proof postulates the existence of a mapping that can select one entity from a set. The existence of such mapping is guaranteed by the Axiom of selection. In design, it corresponds to a designer's style that imposes preferences over candidate designs.

> **Theorem 20:** In the real knowledge, it is possible to make a converging subsequence from any design specification and to find out the design solution for the specification.

The constructive proof of the theorem shows how to create this sequence. This theorem restates Theorem 9 for the state of real knowledge. This proof can serve as a template for a design process.

The following definitions and theorems deal with the important concept of models. The key idea is that models are described by finite number of attributes. This is in correspondence with Hypothesis 1. Intuitively, models are some abstraction of entities that focus on few properties. In many cases, models are then subjected to some analysis based on a theory to find behaviors that are attributed to the original entity.

> **Definition 19:** A metamodel M_Λ is defined as $\cap_{\lambda \in \Lambda} M_\lambda$, where $M_\lambda \in \mathcal{T}_0$ and Λ is finite.

> **Example:** Modeling the chairs as two dimensional entities and the application of statics laws lead to estimating the behavior of chairs A, D, E, and H as *constraining back* and chairs B, C, F, and G as *unconstraining back.*

> **Definition 20:** The *metamodel set*, $\mathcal{M}$, is the set of metamodels that are formed by finite intersections of abstract concepts from the attribute topology.

> **Theorem 21:** The metamodel set is a topology of the real knowledge.

> **Theorem 22:** In real knowledge, the metamodel set is a topology *weaker* than the attribute topology.

The term *weaker* means that the metamodel set is not as fine as the attribute topology and thus, can only approximate it.

> **Theorem 23:** In real knowledge, the *necessary* condition for designing is that the topology of the metamodel set is stronger than the topology of the function space.

Formally, since $\mathcal{T}_0 \supset \mathcal{M}$ (Theorem 22), then if $\mathcal{M} \supset \mathcal{T}_1$ we obtain $\mathcal{T}_0 \supset \mathcal{M} \supset \mathcal{T}_1$. This implies $\mathcal{T}_0 \supset \mathcal{T}_1$ which is the prerequisite for designability since any specification contains several concepts of entity or design solutions. Conceptually, this theorem says that a condition for designing is that for any complex functional requirement, there will be a model that will allow differentiating the designs that potentially satisfy this requirement from other designs.

> Example: Consider designing an *immovable* chair. According to Table 1 or Figure 3(a), which describe the function topology, there are three chairs that satisfy this requirement: B, C, and D. From Figure 3(b), we see that the attribute topology is finer at this region than the function topology because we have two separate subsets of $\{B, C, D\}$: $\{B, C\}$ — the chairs that have a *hanger*, and $\{C, D\}$ — the chairs that are *not lightweight*. A solution exists because we can create a model based on kinematics that will determine that $\{B, C\}$ cannot be moved away because they are hanged. Or under a different interpretation of the term "immovable," we could use a mechanics model that will determine that $\{C, D\}$ cannot be pushed by a human thus are immovable. The fact that we had these models at our disposal and that they were finer than the function topology allowed us to do the mapping between the function and the attribute topologies and focus on the candidates that fit our interpretation of the requirements.

> Theorem 24: If designing is possible, then the identity mapping from $\mathcal{T}_0$ to $\mathcal{M}$ is continuous.

This theorem is similar to Theorem 10 in GDT-Ideal.

> Theorem 25: If the identity mapping from $\mathcal{M}$ to $\mathcal{T}_1$ is continuous, the identity mapping from $\mathcal{T}_0$ to $\mathcal{T}_1$ is continuous.

> Theorem 26: If we evolve a metamodel by way of intersection of abstract concepts, we get an entity concept as the limit of the evolution.

Note that the limit here is an element of $\mathcal{M}$. Therefore, it may only be an approximation of the design solution. This corresponds to real design where products are described by a finite number of properties thus implicitly represent an infinite number of possible designs.

> Definition 21: A *function* of an entity is a physical phenomenon caused by the physical laws governing the situation.

This is the intuitive notion of a behavior of an object when exposed to a physical situation, for example, pushing a chair and observing whether it is stable. This definition slightly differs with Definition 4. This definition is directed towards defining functions by attributes, and a concept by a finite number of measurable or observable attributes (e.g., length, weight, or color).

DEFINITION 22: A *function element* is a metamodel $\cap_{\lambda \in \Lambda} M_\lambda$, where $M_\lambda \in \mathcal{T}_0$ and Λ is a finite set, such that $\forall M_{\lambda \in \Lambda}$, $M_\lambda \in \mathcal{T}_p$.

The next theorem corresponds to Theorem 9 in **GDT-IDEAL**. It does not guarantee arriving at a solution when the specification is given, but guarantees finding an approximate design solution.

THEOREM 27: If we choose function elements as the metamodel, design specification is described by the topology of the metamodel, and there exists a design solution that is an element of this metamodel.

THEOREM 28: The real knowledge is a normal space.

This theorem says that the topology of real knowledge is finer than was originally stated through Axioms 1, 2, and 3, namely, it is a T_4-type space instead of a T_2-type space. Of course, this is a direct result of Hypothesis 1 that restricts the Hausdorff space to be compact. Subsequently, it is proved that this topology is even finer than T_4-type (i.e., it is a metric space).

THEOREM 29: In the real knowledge there exists a distance between two different entities.

The topology of the real knowledge is "fine" enough to allow for the calculation of a metric, that is, one can calculate a distance between any two entities in the space. This is one of the most important results of **GDT**. A simple example of the benefits from properties derived from a metric such as continuity, convergence, incremental refinement, etc. was discussed in Section 3.1. The ability to create a metric, however, can be derived from less restrictive assumptions. The least committing theorem on metrizability is:

THEOREM 30 **(Nagata (1950) and Smirnov (1953),†)**: A space S is metrizable if and only if S is T_3 and has a σ-locally finite basis.

Since all previous results are the consequence of adding Hypothesis 1 to Axioms 1, 2, and 3, Hypothesis 1 may be relaxed. We briefly discuss this in Section 4.1.

The next theorem restates that if a space is metric then the distance measure can be used to assign values to each of the attributes describing an entity such as those presented in Tables 1 and 2.

THEOREM 31: In the real knowledge, an attribute has a value.

Definition 23: When a design solution is identified and realized, it may have behaviors different from the specification. These behaviors are called *unexpected functions.*

Theorem 32: In the real knowledge, the design solution has unexpected functions.

This definition and theorem state that since specifications are defined by a finite set of functional an other properties, they cannot fully determine all the attributes or functional behaviors of the set of candidate designs. Therefore, designs will have behaviors that were not dealt with in the design process and thus may be unexpected.

Summary of real knowledge. The state of real knowledge is an adaptation of the concept of ideal knowledge to the real world. Entities are described by a finite number of attributes that can be observed or measured by instruments. Therefore, these descriptions can be manipulated in finite time and require finite storage capacity. Note that even this finiteness can be very expensive computationally unless we introduced the stepwise refinement process via metamodels. The main assumption about the topological nature of knowledge remains intact.

Design in the state of real knowledge requires the ability to continually model the designed artifact until it evolves into a set of candidates that satisfy the specification. A model is an abstraction of the actual entity that serves as a focus for the particular design stage. The use of models guarantees arriving at an approximate solution in the state of real knowledge. This model of design is different than the catalogue (Theorem 8) or algorithmic (Definition 13) design discussed in GDT-Ideal.

4 Contribution of GDT towards understanding design and CAD

The first step of understanding the contribution that GDT makes towards building CAD systems involves an analysis of its scope, that is, which types of domains satisfy GDT's axioms so that GDT's theorems apply and guarantee certain properties of design. This analysis leads to formulating some guidelines for building design systems and to analyzing the possibility of actually implementing a system that follows the guidelines.

4.1 The scope of GDT

General Design Theory proves strong theorems about design (e.g., Theorems 9 in GDT-Ideal and 26 and 27 in GDT-Real). It arrives at these theorems by imposing restrictions on knowledge about designed artifacts. By limiting the description of objects to a finite set of properties (Hypothesis 1), the assumptions about infinite processing speed and memory capacity employed in **GDT-Ideal** are relaxed in **GDT-Real** to finite resources but that can still be computationally prohibitive. Subsequently, **GDT-Real** introduces designing mediated via metamodels (Theorems 26 and 27). Such a stepwise refinement process can eliminate the resources problem because only a limited number of properties are addressed at each step. Note that this refinement process requires the use of many models reflecting enough understanding of the domain such that the description of the artifact could be analyzed carefully at each step according to Theorem 23. Since finite descriptions of entities are close approximations of design solutions and can be sufficiently detailed to allow manufacturing,[5] the only remaining restriction is the assumption about the representation of design knowledge.

The assumption about the topological knowledge representation emerges from the axioms of **GDT**. Some of these axioms can be relaxed. First, in **GDT-Ideal**, the definition of ideal knowledge can be interpreted as a less restrictive notion of the Axiom of separation. This will slightly improve the applicability of GDT but will reduce the strength of its predictions. Second, in **GDT-Real**, the hypothesis that real knowledge is a compact space is also restrictive. In particular, since some of the important results of **GDT-Real** are proved using the Lindelöf space property of real knowledge (i.e., Theorem 18), there is no need to assume compactness. This is also a minor change, it may increase the scope of GDT and perhaps without reducing its predictive power. These two relaxations are more cosmetic than fundamental. In addition to them we are interested in identifying considerably less restrictive assumptions that can support more realistic and useful predictions about designability.

In summary, the scope of **GDT** might be broadened by insisting on less restrictive assumptions. The extent to which GDT's assumptions can be relaxed depends on the ability to use them effectively in proving some useful results.

Before any change is made to the assumptions, GDT applies to domains with topological structure but, no realistic domain satisfy such requirement. Nevertheless, we may hypothesized that the predictions of GDT will be more applicable to established design domains that have evolved an elaborate categorizations of objects and that have accumulated enough knowledge about available metamodels that can mediate between the function and the attribute categorizations. Designing in these domains may range from simple selection from a catalogue, to routinely composing systems from available components, or to a stepwise refinement process via metamodels.

So far we implicitly assumed that design starts with a specification, but notice that Definition 12 does not state when is the specification given nor does Theorem 9. The latter merely says that the design terminates *when* the specification is described. In contrast,

[5] While our example domain do not demonstrate this, it could be extended to include as many properties as needed to support this.

Definition 13 explicitly states that design is a mapping from the function to the attribute spaces. However, this definition can be eliminated from GDT without loss.

In reality, design most often starts with a specification that is incomplete, contradictory, or infeasible. Significant part of design is invested in elaborating the specification through a design process that not only moves from the function to the attribute categorization but also progresses in the opposite direction (Tomiyama, 1994). Thus, GDT predictions about design do not hold for real design processes.

At this point we can simply terminate the study. GDT discusses an ideal design process that utilizes an ideal kind of knowledge. We can state that since both do not exist, GDT cannot guide the building of CAD systems. Nevertheless, we can do better than that. We can treat GDT as a *model* of design whose usefulness aught to be determined by the potential influence it might have on the building of CAD systems and their empirical success.

4.2 Guidelines for building design systems

GDT offers guidelines for building design support systems. The contribution of the theory towards this end can be assessed by its influence on the development of CAD systems and the success of these systems. We focus on the guidelines related to the representation of knowledge and design processes.

Representation of design knowledge.

Artifact representation.

Two potential artifact representations exist for representing objects: extensional and intensional. In the extensional representation, an attribute is expressed as the set of objects having this attribute. In this representation, all objects have the same status, no predefined hierarchy is imposed upon them. For example,
has a seat = $\{A, B, C, D, F, G\}$ or
has a seat (A), has a seat (B), . . ., has a seat (G)
are two extensional representations of the attribute *has a seat*. This attribute could also be described as a relation between *seat* and *chair* as two objects. For example, suppose *a, b, . . ., g* are seats, then the relation could be
seat(a), has (a, A), seat(b), has (b, B), . . ., seat(g), has (g, G)
Graphically, Figure 3 represents the chairs domain extensionally. In the intensional representation objects are described by the set of attributes characterizing them or the components from which they are built. For example, a chair might be described as
chair(seat, back support, legs, wheels, vertical rotational dof, weight, hanger, brake)
Tables 1 and 2 can be thought of as intensional descriptions of chairs.

Each of the two representations have advantages and disadvantages that can be summarized as follows (Tomiyama and Yoshikawa, 1986; Tomiyama and Ten Hagen, 1990). (1) Extensional descriptions can be easily modified by the addition of new entities, properties, or their new categorizations; thus, they can support innovations although not introduce them themselves. (2) They supports recognizing the similarity among almost identical objects; thus, can support designing from precedences. (3) Extensional representations may be hard to understand since they are not concise. In contrast, intensional descriptions of objects (1) are hard to modify; (2) do not support inferring the similarity between objects; but (3) their descriptions are more concise.

The first two properties are *conceptual* and fundamental to supporting (incremental) design. The third property describes the ability to understand the *representations* of objects, but not necessarily the objects themselves; it impacts on the computational efficiency of operations. Thus, property (3) deals with implementation issues that are important for CAD systems but less relevant to GDT as a theoretical entity. It is no surprise, therefore, that GDT axioms are built upon extensional representations of entities (Tomiyama and Yoshikawa, 1986). The recognition of objects by abstract concepts described in Axioms 1 requires the use of extensional representations, and Axiom 3 and the progression of design process (i.e., by the intersection of abstract concepts) demand the use of extensional representations of abstract concepts.

The conceptual superiority of extensional over intensional representations, and the theoretical predictions of GDT suggest the use of extensional representations in CAD systems. Nevertheless, most CAD systems, including commercial computer graphics software, favor intensional representations due to their ease of computer implementation and their computational efficiency in answering some types of queries. This also is no surprise because commercial CAD tools are *drafting*, rather than, *design* tools.

For the purpose of implementing computationally efficient CAD system, extensional descriptions work well when the description is not too detailed; their ability to support the recognition of similarity between objects and their incremental nature make them useful for exploration which is important in the conceptual or preliminary design stages. In the detail design phase, objects are detailed with many additional attributes, thereby preferring an intensional representation that is more concise and computationally efficient. This is in agreement with Yoshikawa and Tomiyama (1985) suggestion and Kurumatani and Yoshikawa (1987) proposal; even though, it differs with GDT favoring of extensional descriptions.[6]

Knowledge organization.

The second aspect of knowledge representation is the overall organization of concepts and

[6]Kurumatani and Yoshikawa (1987) propose the same solution but say that (p. 724) "Axiom 1 guarantees the possibility of recognizing an entity by its connotation instead of its denotation"(i.e., intensional instead of extensional). I interpret this statement as arguing that Axiom 1 does not preclude the use of intensional representations in addition to extensional representations.

entities. **GDT** is based on a topological structure of the universe of entities. All entities have the same status. Nevertheless, it is recognized that in real design perfect topological structures do not exist. In response, Yoshikawa (1981) postulates that human designers use hierarchical knowledge structure, therefore, it may be suitable for design support systems. However, hierarchical knowledge structure is insufficient to prove any of the interesting theorems about design that GDT proves.

Graph structures are more reminiscent of topologies than are hierarchies. Thus employing such knowledge structures may result in better support of design. This again is a hypothesis to be tested by empirical investigations. Graph structures need not be build *a priori*. They can be built incrementally on top a the universe of entities by taking advantage of the flexibility provided for by extensional representations.

Design process.

GDT implicitly assume that designing is a mapping from the function to the attribute topology under certain constraints. **GDT-Real** presents the concept of models as mediating from the function to the attribute topologies. Models that are sufficiently detailed (to satisfy Theorem 23) must be used for supporting the incremental modification of the design towards a solution. Theorems 26 and 27 guarantee arriving at an approximate solution when using this design process with topological knowledge structure. It is hypothesized that such a process, when operating on design knowledge in forms closer to topology (e.g., graph structures) and when managing extensional information at the early stages of design and intensional information in later stages may improve CAD systems.

Figure 4 illustrates the concept of design as a converging process using a detailed version of the chairs domain. An arrow emanating from the metamodel denotes a particular model mediating between two consecutive design stages. The function addressed in the transition between design and the physical phenomena used in creating the model are written on the arrow. The first two transitions can be represented extensionally by explicitly intersecting sets. The third stage involves adding a brake to chair E. This involves copying E into a new entity, E', and refining it extensionally by adding a brake, for example, in either of the following ways:
has a brake (E')
or
brake(b), has (b,E').
Subsequent refinements that involve sizing parts of the chair may be better dealt with intensionally. This will involve creating the intensional representation of entities and the assignment of values to these representations. For example, the seat may be described as:
seat(thickness, firmness, width, length, ...)
and the "slots" in this "scheme" or "frame" could be filled by the actual values. After the detailing of the intensional descriptions, the process terminates with a feasible solution.

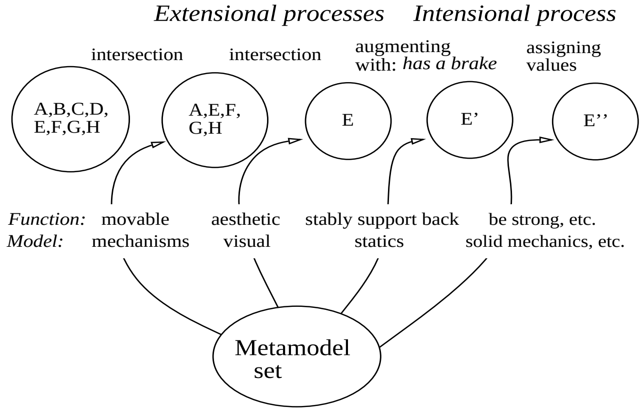


Figure 4: Design as a converging process

4.3 Implementations related to GDT

Several system implementations embody some of the guidelines discussed before. A complete implementation must follow as many guidelines in a consistent manner as possible. Some implementations discussed the creation of a schema for representing design knowledge about artifacts (e.g., (Kurumatani and Yoshikawa, 1987)) and some discussed the organization of concepts (e.g., (Taura et al, 1989)). Here we concentrate on another implementation that is far more developed than the others, and briefly mention implementations primarily relevant to the organization of concepts.

Implementations directly related to GDT.

IDDL is an Integrated Data Description Language for coding design knowledge (Tomiyama and Ten Hagen, 1987a; Tomiyama and Ten Hagen, 1987b; Veth, 1987) which was highly influenced by GDT. IDDL was implemented twice in two different systems (Tomiyama et al, 1991; Veerkamp et al, 1991). The specification of IDDL was based on the study of three components that contribute to CAD: theory of knowledge, theory of design, and theory of designed objects. The first two are directly related to GDT. Also, IDDL makes a commitment to adopt logical formalisms.

The interleaved extensional and intensional representation discussed before was embodied in IDDL. Extensional descriptions are used to represent entities and their relationships by using *predicates*, while intensional descriptions are used to describe the attributes of entities by using *functions*. GDT says that design is a stepwise refinement process mediated by metamodels. The nature of design is such that information may be unknown or uncertain. Therefore, IDDL is equipped with three-valued logic, modal logic, and inheritance in addition to mechanisms for representing design processes and their control. The third component of CAD — the theory of designed objects — deals with the metamodels of the particular domain of interest. For this, IDDL is augmented with facilities to represent qualitative physics models. The consistency between, and management of, metamodels and their operation is handled by an ATMS (assumption-based truth maintenance system).[7]

IDDL can be used to manually represent and structure knowledge about design objects, processes, and behaviors. The control of the operation is maintained by the designer using the system but many operations can be executed automatically by an inference engine. While IDDL was developed as an infrastructure for CAD, its principles were use in the design of several innovative machines that embody intelligent control for self maintenance. A prototype of a self-maintained copier was patented (Tomiyama, 1994).

Implementations not-directly related to GDT.

Taura et al. (1989) discussed the creation of a classification over a set of design entities by defining a metric over the space of entities and using it for clustering. Theorem 29 guarantees the possibility of creating a topology, and not just a classification. There are ways to construct a knowledge organization that is closer to a topology, for example, the creation of hierarchical classification of entities. **Ecobweb** is a program that provides support for incrementally creating hierarchical structures of entities (Reich and Fenves, 1992). The classes in the hierarchical structure can be described by both intensional and extensional descriptions that can be used in synthesis. **Ecobweb** has several synthesis methods that can use the hierarchical structure and others can be easily incorporated due to the flexibility of the extensional representation. The principles that are embedded in **Ecobweb** can be used to construct a more flexible entity structure: a flexible graph rather than a hierarchical structure, thus approximating better topological structures. Research on this subject is underway and was initiated only following the hypothesis that a better approximation of topological structures will yield better performance of **Ecobweb**.

Knowledge structures need not be created automatically; they can emerge from design given appropriate facilities such as the support of a "flat" space of objects (i.e., all objects have equal status) and facilities for creating complex categorizations and embedding modeling tools for analyzing objects as implemented in the *n*-dim program (Levy et al, 1993; Subrahmanian et al, 1993).

The future empirical success of these and other systems that embed some of the guidelines the GDT propose for building CAD systems can be perceived as a support for the applicability of the guidelines. This, in turn, will strengthen the applicability of GDT as a *model* of design.

[7]See the references for more details.

5 Summary

This paper critically reviewed **GDT**. It analyzed the assumptions made by **GDT** and the theorems it proved. The core concepts of **GDT** were highlighted and analyzed. It was found that GDT cannot be an adequate description of real design. In response to the restricted scope of **GDT**, the review suggested relaxing some theoretical assumptions that may maintain similar but more realistic predictions about the convergence of design processes. Independent of whether this path of research will be fruitful, GDT is useful as a model of design because as such, it can offer some guidelines for building CAD systems. We have briefly mentioned systems that were implemented directly in relation to the guidelines of GDT and others that embed some of the guidelines but were developed independently.

Skeptics may claim that GDT presents a far too ideal model of design, thus, it is unclear what is its central contribution to building CAD. Nevertheless, we think that the guidelines that GDT offers are tangible. The support for this claim remains as a future work through the building of CAD systems that embed these guidelines and their testing in real design tasks. It is hoped that this review presented GDT's concepts in a clear way that may lead researchers to benefit from the insight GDT provides and perhaps undertake this challenge.

Acknowledgments

I would like to thank the three reviewers for their thoughtful remarks on drafts of this paper.

References

Arciszewski, T. (1990). "Design theory and methodology in Eastern Europe." In *Design Theory and Methodology–DTM'90 (Chicago, Il)*, pages 209–218, New York, NY, The American Society of Mechanical Engineers.

Boothroyd, G. and Dewhurst, P. (1989). *Product Design for Assembly*, Boothroyd Dewhurst, Wakefield, RI, 3rd edition.

Christenson, C. O. and Voxman, W. L. (1977). *Aspects of Topology*, Marcel Dekker, New York.

Eastman, C. M. (1991). "Use of data modeling in the conceptual structuring of design problems." In Schmitt, G. N., editor, *CAAD Futures '91: Proceedings*, pages 207–223, Zürich, Department of Architecture, Swiss Federal institute of Technology.

Eder, W. E. (1990). "Engineering design - a perspective on U.K. and Swiss developments." In *Design Theory and Methodology–DTM'90 (Chicago, IL)*, pages 225–234, New York, NY, The American Society of Mechanical Engineers.

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(1):51–67.

Hubka, V. and Eder, W. E. (1988). *Theory of Technical Systems: A Total Concept Theory For Engineering Design*, Springer-Verlag, Berlin.

Hundal, M. S. (1990). "Research in design theory and methodology in West Germany." In *Design Theory and Methodology–DTM'90 (Chicago, IL)*, pages 235–238, New York, NY, The American Society of Mechanical Engineers.

Konda, S., Monarch, I., Sargent, P., and Subrahmanian, E. (1992). "Shared memory in design: A unifying theme for research and practice." *Research in Engineering Design*, 4(1):23–42.

Kurumatani, K. and Yoshikawa, H. (1987). "Representation of design knowledge based on general design theory." In Eder, W. E., editor, *Proceedings of The 1987 International Conference on Engineering Design, ICED-87*, pages 723–730, Boston, MA, American Society of Mechanical Engineers.

Levy, S., Subrahmanian, E., Konda, S. L., Coyne, R. F., Westerberg, A. W., and Reich, Y. (1993). "An overview of the *n*-dim environment." Technical Report EDRC-05-65-93, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Nagata, J. (1950). "On a necessary and sufficient condition of metrizability." *Journal Institute of Polytechnics*, 1:93–100.

Reich, Y. and Fenves, S. J. (1992). "Inductive learning of synthesis knowledge." *International Journal of Expert Systems: Research and Applications*, 5(4):275–297.

Schubert, H. (1968). *Topology*, MacDonald Technical & Scientific, London.

Smirnov, Y. M. (1953). "On metrization of topological spaces." *American Mathematical Society Translation, Series 1*, 8:62–77.

Subrahmanian, E., Konda, S. L., Levy, S. N., Reich, Y., Westerberg, A. W., and Monarch, I. A. (1993). "Equations aren't enough: Informal modeling in design." *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, 7(4):257–274.

Suh, N. P. (1984). "Development of the science base for the manufacturing field through the axiomatic approach." *Robotics & Computer-Integrated Manufacturing*, 1(3/4):397–415.

Sutherland, W. A. (1975). *Introduction to Metric and Topological Spaces*, Oxford University Press, Oxford, UK.

Taura, T., Kubo, S., and Yoshikawa, H. (1989). "Generation of design solutions by a metric space." In *Proceedings of The Third IFIP WG 5.2 Workshop On Intelligent CAD, Osaka, Japan.*

Tomiyama, T. and Ten Hagen, P. J. W. (1987). "The concept of intelligent integrated interactive CAD systems." Technical Report CS-R8717, Centre For Mathematics and Computer Science, Amsterdam.

Tomiyama, T. and Ten Hagen, P. J. W. (1987). "Organization of design knowledge in an intelligent CAD environment." In Gero, J. S., editor, *Expert Systems in Computer-Aided Design*, pages 119–152, North-Holland, Amsterdam.

Tomiyama, T. and Ten Hagen, P. J. W. (1990). "Representing knowledge in two distinct descriptions: Extensional vs. intensional." *Artificial Intelligence in Engineering*, 5(1):23–32.

Tomiyama, T. and Yoshikawa, H. (1985). "Knowledge engineering and CAD." In Yoshikawa, H., editor, *Design and Synthesis, Proceedings of The International Symposium on Design and Synthesis*, pages 3–8, North-Holland, Tokyo, Japan.

Tomiyama, T. and Yoshikawa, H. (1986). "Extended general design theory." Technical Report CS-R8604, Centre For Mathematics and Computer Science, Amsterdam.

Tomiyama, T., Kiriyama, T., Takeda, H., Xue, D., and Yoshikawa, H. (1989). "Metamodel: A key to intelligent CAD systems." *Research in Engineering Design*, 1(1):19–34.

Tomiyama, T., Xue, D., and Ishida, Y. (1991). "An experience with developing a design knowledge representation language." In ten Hagen, P. J. W. and Veerkamp, P. J., editors, *Intelligent CAD Systems III: Practical Experience and Evaluation*, pages 131–154, Berlin, Springer-Verlag.

Tomiyama, T. (1990). "Engineering design research in Japan." In *Design Theory and Methodology–DTM'90 (Chicago, IL)*, pages 219–224, New York, NY, The American Society of Mechanical Engineers.

Tomiyama, T. (1994). "From General Design Theory to knowledge intensive engineering." *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 8(4):319–333.

Čech, E. (1966). *Topological Spaces*, John Wiley & Sons, London.

Veerkamp, P., Kwiers, P. R., and ten Hagen, P. J. W. (1991). "Design process representation in ADDL." In ten Hagen, P. J. W. and Veerkamp, P. J., editors, *Intelligent CAD Systems III: Practical Experience and Evaluation*, pages 155–167, Berlin, Springer-Verlag.

Veth, B. (1987). "An integrated data description language for coding design knowledge." In *Intelligent CAD Systems I*, pages 295–313, Berlin, Springer-Verlag.

Warfield, J. N. (1990). *A Science of Generic Design*, Intersystems Publications, Salinas, CA.

Yoshikawa, H. (1981). "General Design Theory and a CAD system." In Sata, T. and Warman, E., editors, *Man-Machine Communication in CAD/CAM, Proceedings of The IFIP WG5.2-5.3 Working Conference 1980 (Tokyo)*, pages 35–57, North-Holland, Amsterdam.

Appendix. A glossary of mathematical concepts

This appendix defines the set theory and topological terms used in this study. The definitions listed were compiled from (Čech, 1966; Christenson and Voxman, 1977; Schubert, 1968; Sutherland, 1975) and appear in alphabetical, rather than dependency, order. The terminology used in this study is at the introductory level of topology.

Axiom of selection/choice (simultaneous selection): If there exists a collection of disjoint sets, then there exists a set that has precisely one element from each of the sets in the collection.

Basis: If $(S, \mathcal{T})$ is a topological space, then a basis for $\mathcal{T}$ is a sub-collection $\mathcal{B}$ of $\mathcal{T}$ such that if $s \in S$ and U is an open set containing s, then there is a set $V \in \mathcal{B}$ such that $x \in V \subset U$.

Cauchy sequence: A sequence (s_n) in a metric space S with a metric d is a *Cauchy sequence* if given $\epsilon > 0$, there exists N such that $d(s_n, s_m) < \epsilon$ for all $n, m \geq N$.

Closed set: A subset of a topological space is closed if its complement is open.

Closure: Let $(S, \mathcal{T})$ be a topological space and $A \subset S$. The closure of A in S is defined as $\cap\{C \mid C \text{ is closed in } S \text{ and } A \subset C\}$ and denoted by $\overline{A}$.

Cluster point: A point $x \in X$ belonging to the closure of $X - \{x\}$ is a cluster point. Intuitively, a cluster point is a point where many other points accumulate and converge to that point.

Compact: A topological space is compact if every open cover has a finite (not just countable) cover.

Continuous mapping: Let $(A_1, \mathcal{T}_1)$, $(A_2, \mathcal{T}_2)$ be two topological spaces, a mapping $f\colon A_1 \rightarrow A_2$ is continuous if for all $U \in \mathcal{T}_2 \Rightarrow f^{-1}(U) \in \mathcal{T}_1$.

Convergence of a sequence: A sequence (s_n) in a topological space S *converges* to a point $s \in S$ if and only if for every neighborhood U of s there is a positive integer N_U such that $s_i \in U$ whenever $i > N_U$.

Countable basis: A basis is countable if it has a countable number of sets.

Cover: A cover of a set S is a collection of sets whose union is a superset of S.

Directed set: A set S with a partial order $\preceq$ is called a *directed set* if for each $s_i, s_j \in S$, there is $s_k \in S$ such that $s_i \preceq s_k$ and $s_j \preceq s_k$. A *directed sequence* is the same except that now also $i \leq k$ and $j \leq k$.

Family: A family $\mathcal{A}$ of elements of S is a set Λ, a mapping $\sigma : \Lambda \rightarrow S$, and the subset $\sigma(\Lambda)$ of S.

Filter: A filter of S is a collection $\mathcal{F}$ of subsets of S that has the following properties: (1) $\phi \notin \mathcal{F}$, (2) if $A \in \mathcal{F}$, and $A \subset B \subset S$, then $B \in \mathcal{F}$, and (3) if $A, B \in \mathcal{F}$ then $A \cap B \in \mathcal{F}$.

Finite intersection property: A collection of sets $\mathcal{C} = \{C_\lambda | \lambda \in \Lambda\}$ has the *finite intersection property* if and only if for each nonempty finite subset $N \subset \Lambda$, $\cap C_{n \in N} \neq \phi$.

Hausdorff space: A Hausdorff space is a space S that satisfies the condition: any two distinct entities in S can be surrounded by disjoint neighborhoods.

Homeomorphism: If $(S_1, \mathcal{T}_1)$ and $(S_2, \mathcal{T}_2)$ are topological spaces, then a mapping $f : S_1 \rightarrow S_2$ is called a *homeomorphism* if and only if f is invertible and both f and f^{-1} are continuous.

Lindelöf space: A Lindelöf space is a space that satisfies the condition that every open covering of the space has a countable sub-covering.

Locally finite: A family $\mathcal{A} = \{A_\lambda\}_{\lambda \in \Lambda}$ of subsets of a topological space S, is *locally finite* if for every $s \in S$ there is a neighborhood $U(s)$ such that $U(s) \cap A_\lambda \neq \phi$ for at most a finite number of λ.

σ-locally finite: A cover $\mathcal{U}$ of a space is σ-locally finite if and only if $\mathcal{U}$ can be expressed as the union of a countable collection of families, each of which is locally finite.

Metric space: A set S is called a *metric space* if with every pair of points $x, y \in S$, there exists a non-negative real number $d(x, y)$ that satisfies:

(1) If $d(x, y) = 0$ then $x = y$ and $d(x, x) = 0$ always holds.
(2) For any pair of points x, y, $d(x, y) = d(y, x)$.
(3) For any three points x, y, and z, $d(x, z) \leq d(x, y) + d(y, z)$.

Neighborhood: If $(S, \mathcal{T})$ is a topological space then the neighborhood of $s \in S$ is any of the sets $U \in \mathcal{T}$ such that $s \in U$.

Normal: A topological space $(S, \mathcal{T})$ is normal if for every pair of disjoint closed sets $A, B \subset \overline{S}$ there exists a pair of disjoint open sets $U, V \in \mathcal{T}$ such that $A \subset U$ and $B \subset V$.

Open set: If $(S, \mathcal{T})$ is a topological space then all the subsets of $\mathcal{T}$ are called open sets.

Second countable: A topological space is second countable if it has a countable basis.

Subspace: A space $\mathcal{T}_1$ is a subspace of $\mathcal{T}_2$ if $\mathcal{T}_1 \subset \mathcal{T}_2$.

Weaker topology: If $(S, \mathcal{T}_1)$ and $(S, \mathcal{T}_2)$ are two topological spaces. $\mathcal{T}_1$ is said to be weaker than $\mathcal{T}_2$ if $\mathcal{T}_1 \subset \mathcal{T}_2$.