3 A FORMALIZATION OF CK DESIGN THEORY BASED ON INTUITIONIST LOGIC

Akın O. Kazakçı

CGS - Mines ParisTech, Paris, France, Tel: +331 4051 9208, Fax: +331 4051 9065.
E-mail: akin.kazakci@ensmp.fr

The paper introduces a formalization of Concept-Knowledge (CK) theory of design reasoning based on Intuitionist Logic and Kripke type semantics. The concept space is defined as a tree of formulae containing free variables and knowledge space corresponds to an incomplete theory. A set of operations is defined to model the progressive elaboration of the concept space, the expansion of the knowledge and the interaction of concepts and knowledge.

Keywords: Design, logic, CK design theory, concept, knowledge, design reasoning.

1. INTRODUCTION

CK design theory, introduced by Hatchuel and Weil, describes creative design reasoning based on the interaction of a concept space and knowledge of a designer.[1−4] Knowledge represents all the available knowledge to a designer. Concept space contains concepts, i.e., propositions that are undecidable with respect to the knowledge a designer has. Knowledge is used to build such undecidable propositions in the conceptual space. Design ends when an undecidable proposition becomes known which is only possible if new knowledge is acquired and knowledge is expanded.

Although the literature suggests there are deep relationships between CK theory and other formal theories (e.g. set theory), few formal work exist on CK theory itself. Nonetheless, in a recent work, some similarities between the CK theory and the Forcing technique in modern Set Theory (see next section for more details) has been reported.[4] The authors claim that this is an important result that deepens the foundations of CK theory and a significant step for design theory in general.

The introduction of Forcing technique by Paul Cohen is a significant event for mathematics[5,6] but also for logic (see e.g. Ref. 15). In a seminal work, Saul Kripke has given a semantic model for Intuitionist Logic and he has shown that the same semantics can be used to capture the idea of Forcing.[8] In this work, based on the conceptual insight provided by this connection, we give a logical formalization of CK theory based on Intuitionist Logic. One of the implications of our formal model is that design reasoning, as described by CK theory, is more sophisticated than the one described by the Intuitionist Logic.

Section 2 presents briefly CK theory and related formal work. Section 3 discusses similarities and differences between Intuitionist Logic and CK theory. Section 4 presents the formal definitions of the key notions of CK theory. Section 5 summarizes.

2. A CONCEPT-KNOWLEDGE THEORY OF DESIGN REASONING

2.1. An Overview of CK Theory

Design is about building a definition for an object that is yet to exist. The construction of such a definition requires a particular type of reasoning process which is different from usual ones such as search, optimization or theorem proving. Hatchuel and Weil[1−4] propose a theory of design reasoning

which captures some of the fundamental properties of this process based on the distinction and interaction between two spaces;

- **Knowledge space** A knowledge space represents all the knowledge available to a designer (or to a group of designers) at a given time. These are propositions that the designer is capable of declaring as true or false; i.e., propositions whose logical status are known to the designer (e.g., some ships have sails).
- **Concept space** A concept space represents propositions whose logical status are unknown and cannot be determined with respect to a given knowledge space. These are propositions that can be stated as neither true, nor false by the designer at the moment of their creation (e.g., some ships can fly).

Concepts are descriptions of an object of the form "C: there exist an object $\times$ with the properties $p_1,p_2,\ldots, p_n$ such that C is *undecidable* with respect to K".[3,4] A design process begins when such an undecidable formula is created. A designer can then elaborate the initial concept by partitioning C — that is, by adding further properties to C. Current writings about CK theory distinguish two kinds of partitioning:

- **Restrictive partitions** add to a concept a usual property of the object being designed.
- **Expansive partitions** add to a concept novel and unprecedented properties.

It is claimed that creativity and innovation are possible due to expansive partitions: such partitions leads to fundamental revision of the identity (or definition) of objects.[2,3] Since concepts are elaborated by partitioning, the concept space has a treelike structure.

When elaborating a concept space, a designer might use his or her knowledge, either to partition further the concepts, or to attempt a validation of a given concept. This last type of operation is called **K-validation** and it corresponds to the evaluation of a design description using knowledge. The result of a K-validation is positive, if the designer acknowledges that the proposition "there exist an object x with properties $p_1,p_2,\ldots,p_n$" is true[2,3] The result is negative, if the knowledge available to the designer allows him to state that such an object cannot be built.

Often the validation of a concept is not readily possible. Due to the expansions of the concept space, the idea of objects that the designer knows nothing about the existence has been created. In order to validate concepts, new knowledge warranting the existence conditions of such an object should be acquired. In terms of the theory, knowledge should be expanded. The expansion of knowledge space is called **K-expansion**. Thus, "design is the interaction and dual expansions concepts and knowledge".[1,2]

2.2. Current Work on Formalization of CK Theory

In, Ref. 3 Hatchuel and Weil attempt to bring the theory on more formal grounds. Although a complete formal treatment is not given, some of the notions of the theory are desribed using the terminology and ideas from the axiomatic set theory of Zermelo-Fraenkel (ZF). It is emphasized that the knowledge space K and the concept space C are built on different axiomatics and have different logics. In particular, if ZF is used to model the concept space, the axiom of choice cannot be used. This claim is based on the idea that during design it is not possible to choose an object having the desired properties, otherwise, this would mean that the design is already finished.

More recently,[4] presented similarities between CK theory and Forcing. Forcing is a method used in set theory invented by Paul Cohen for the creation of new sets[5,6] Using Forcing, Cohen showed that, starting with a model M verifying the axioms of the set theory and a proposition P that is undecidable in M (i.e., it is impossible to state the truth or falsity of P in M), it is possible to construct progressively a new model N (containing M) in which P (or its negation) is true. According to, Ref. 4. Forcing can be seen as a special case of CK design theory where set theory is taken as a *model of K space*. The proposition P is a (partial) description of an object whose existence cannot be confirmed or rejected in the beginning of the design process. M is the initial knowledge space. The conditions introduced successively correspond to properties introduced in the C space by partitioning. Design ends when a new collection of sets N is constructed where P becomes true for N.

3. INTUITIONIST LOGIC AND CK THEORY: SIMILARITIES AND DIFFERENCES

In traditional Tarskian account of logic, the truth of a proposition is determined by a model. A model consists in a domain D of objects and an interpretation function I, assigning each predicate P a set of objects (i.e., determining whether a relation holds between a set of objects). Given a proposition P, a model M is said to *satisfy* or to *force* P, denoted as M $\models$ P, if the proposition is true in M.

Based on these notions, one can be tempted to straightforwardly associate the notion of a model with that of a knowledge space (a model states which propositions are true and which propositions are false). In which case, a concept might be defined as a proposition P such that neither M $\models$ P, nor M $\models$ g¬P. However, the classical notion of model for first-order logic does not allow such indeterminacy, dictating that, for every proposition P, either P, or its negation ¬P must be true. This principle called *logical omniscience* does not leave place for design reasoning according to CK theory: since everything is already decided and known, no concept can be formulated.

Furthermore, in the classical model theory the domain of objects (although it can be infinite) is considered fixed and unchanged. By contrast, in design, at least one new object appears at the end of the process the *properties of which are not known but are referred to* during the process. This implies that, in order to give a logical formalization of CK theory, we need a logic that allows *partial knowledge* of the world and a model of knowledge where the domain of objects and the extension of predicates can grow in time.

Intuitionist Logic provides such a logic. In Intuitionist Logic and Intuitionist philosophy in general, truth or falsity of propositions are discovered by a subject through his mental activity over the time.[16,17] Thus, the subject can learn about the existence of new objects and relationships during his reasoning activity. Propositions that the subject cannot state as true or false at some point in time, can be learnt to be true (or false) in later stages. This is a particularly suitable characteristic to model the apparition of new design objects and the expansion of knowledge space.

Yet, another relevant characteristic of Intuitionist Logic for our purposes is the interpretation of truth and falsity. In this interpretation, "true" is the equivalent of "constructible" or "has a proof". A proposition is "false" if there is a proof that "every attempt to construct a proof for it, leads to a contradiction" (i.e. there is a proof that such an object cannot be built). This idea corresponds well to the idea of K-validation in CK theory (see above).

The above characteristics show that Intuitionist Logic can be used appropriately to formalize the evolution of knowledge space and validation of concepts. Yet, despite its acknowledgement that there can be undecided propositions during the evolution of knowledge, Intuitionist Logic does not present explicitly such propositions and their evolution over time. This is obviously a major difference between the kind of reasoning described by CK theory and Intuitionist Logic — a very significant one: It shows that design reasoning, as described by CK theory, is a more sophisticated type of reasoning than the logic proposed by the Intuitionist approach. Not only CK theory describes the expansion of knowledge over the time, but it asserts that this expansion is far from being purposeless and that it is governed by a concept space — which, itself, depends on the evolution of the knowledge space! Nonetheless, as we shall see, the more precise formalism provided by the Intuitionist Logic will allow us to formalize the idea of concept space and various operations described by CK theory.

4. A FORMALIZATION OF CK DESIGN THEORY BASED ON INTUITIONIST LOGIC

4.1. Language and Semantics of Intuitionist Logic

We assume a standard first-order language L without function and equality symbols. The terms of L are constants, denoted by d,d_1,d_2,…g and variables, denoted by x,y,z,… possibly with subscripts and superscripts. Predicate symbols of L are denoted by p,q,r,uuu possibly with subscripts or superscripts. We have the following logical connectives ˆ, ?, ∃, ∀, → and ⊥ where $\neg\phi$ is taken to mean ϕg → ⊥

(ϕ leads to absurdity). Complex formulas of L are built up from the atomic predicate symbols using the logical connectives. The definition of a Kripke model for L is as follows.

Definition 1 (Kripke frame) *A Kripke frame is a quadruple $< K_0, \mathbf{K}, \leq, d >$ where* **K** *is a set of nodes, K_0 is a distinguished element of K, $\leq$ is a partial order over K and d is a function assigning a domain D_i to each node K_i. The following condition is satisfied by d:*

$$\text{if } K_i \leq K_j \text{ then } d(K_i) \subseteq d(K_j)$$

The nodes in a frame correspond to information stages and K_0 represents the initial state of knowledge. The condition implies the domain of objects grows monotonically; new objects may appear when moved from a node K to another node K' $\geq$ K.

Definition 2 (Kripke model) *A Kripke model is a pen-tuple $< K_0, \mathbf{K}, \leq, d, I >$ where $< K0, \mathbf{K}, \leq d >$ is a Kripke frame and I is a interpretation function for the predicates, such that:*

- $I(p_n,K) \subseteq d(K)^n$
- *if* $K_i \leq K_j$ *then* $I(p_n, K_i) \subseteq I(p_n, K_j)$

where p_n is an n-place predicate.

These conditions imply that the extension of a predicate grows monotonically as well. Assignments are mappings g from the set of variables to $\cup_{K \in \mathrm{k}}$ d(K). A variable assignment for g a formula α is denoted α[g]. Let p an n-place predicate symbol, a variable assignment, ϕ and ψ arbitrary complex formulae. The satisfaction relation $\models$ between an information stage K and any arbitrary formula is defined by induction on the length of that formula as follows:

- K $\models$ p(x_1,..., x_n)*[g] iff* $< g(x_1), \ldots, g(x_n) >$ $\in I(p,K)$
- K $\models$ $\neg\phi$ iff iK' such that K $\leq$ K', K' $\not\models$ ϕ
- K $\models$ $\phi \hat{} \psi$ iff K $\models$ ϕ and K $\models$ ψ
- K $\models$ $\phi ? \psi$ iff K $\models$ ϕ or K $\models$ ψ
- K $\models$ $\phi \rightarrow \psi$ iff iK' such that K $\leq$ K', if K' $\models$ ϕ then K'g ψ
- $\exists$x. ϕ[g] iff for some d $\in$ d(K) such that K $\models$ $\phi[g_x^d]$
- $\forall$x. ϕ[g] iff iK such that K $\leq$ K', id $\in$ δ(K'), K' $\models$ $\phi[g_x^d]$

The inductive definition of truth given above may look similar to its counterpart for classical logic. However, there are significant differences in the treatment of the logical connectives and the notion of truth. First, truth (or falsity) are now defined based on future information as well (hence, the totality of the model is concerned not only the current knowledge). Second, truth and false are not symmetric notions anymore. In order to verify ϕs we need a proof for ϕ.gowever, in order to verify $\neg\phi$s we need to make sure that there cannot be a stage of information where we can find a proof of ϕ. Said in other terms, once the negation of a formula ϕ is established (i.e. there is a proof that no proof of ϕ can be constructed) at some point K, there cannot be an information stage K'g $\geq$ K where ϕ is true. As pointed out before, this interpretation is based on truth as proof and knowledge as construction.

It can be seen that, these definitions allow the apparition of new objects and the gain of new knowledge over time. Although at some point K_i the information an individual has might be incomplete, at some later stage K_j, he or she can have more information ($K_j \geq K_i$) even though K_j might still be not complete. Another way to put this is that this formalism allows expressing not only what is known but also what is not known.[9]

4.2. Concepts and Knowledge Space

As we have seen in the previous paragraph, an information stage K is a specification of what is known to be true or false at a given point in time in a way that allows incomplete knowledge. This is in perfect accordance with the definition of a knowledge space in CK theory.

Definition 3 (Knowledge space) *A knowledge space is an information stage (or a node) K in a Kripke model.*

In the following, we will sometimes confound the formal definition of a knowledge space with a corresponding database and denote both by K by abuse of notation. We suggest the following definition of concept for the purposes of the current work:

Definition 4 (Concept) *For a given knowledge space K, a concept is a formula c = p_1* suuus*p_n (where p_1,*uuu*, p_n are atomic predicates) such that:*

- *At least one of the predicates p_i occurring in c contains one or more free variables.*
- *Every couple $(p_i, p_j)_{(i \neq j)}$ of predicates occurring in c are related such that there exist a sequence of terms t_1,…t_n such that t_1 occurs in p_i and t_n in p_j and for all t_i, t_{i+1} in the sequence there is a predicate R in c such that t_i and t_{i+1} occur in R.*
- *If the sequence of free variables occurring in c is $\boldsymbol{x}$= x_1,…x_k then, for the existentially quantified formula* k$\boldsymbol{x}$*c, neither K* $\models$ k$\boldsymbol{x}$*.c, nor K* $\models$ gk$\boldsymbol{x}$*.c.*

In CK theory, the definition of a concept is expressed as a formula of the type "there exists an object x, p1, p2,…,pn" We believe that the use of the existential quantifier is suggestive of some kind of claim or hypothesis, which is not the intention behind the definition of concepts.[1] For this reason, the first condition requires concepts to be formulae containing at least one free variable. The second condition requires that the abstract idea expressed by a concept to be about a *whole*. Any property introduced into the definition will be somehow related to the rest of the described object (no independent sub-definitions). The third condition incorporates the essence of the original definition of a concept in CK theory: within the current state of knowledge K, it is not possible to decide whether or not some set of objects may exist such that k**x**.c is true (or that, this can never be true).

As in the original theory, the definition of a concept depends on a given K. Thus, for two different designers with respective knowledge spaces K_1 and K_2, the status of a formula c may be such that $K_1 \models$ c (or, $K_1 \models \neg$c) but neither $K_2 \models$ c, nor, $K_2 \models \neg$c.

Example. *Consider the following knowledge space K : K={ ship(ship_11), flies(bird_23) }. Hence, there are only two objects known at this stage: ship_11, bird_23. In this case, ship(x)^flies(x) is a concept since* k*x.(ship(x) ^flies(x)) (nor its negation) is not satisfied in K: no object is known to be a ship and to fly at the same time. Note that ship(ship_11)^*k*x.flies(x), ship(x)^flies(y) are not concepts because of the conditions 1) and 2) respectively.*

4.3. Operations of a Design Process

4.3.1. Partitioning and Concept Space

In CK theory, a concept space is a treelike structure that can be generated starting from an initial concept. The operation that allows the generation of the tree of concepts is called partitioning (see section 2). In the simplest case, partitioning can be seen as the conjunction of a concept c with a predicate q or its negation ¬q. While the former allows introducing additional properties for an object (e.g. mobile apartment), the latter prevents the use of some properties (e.g. bagless vacuum cleaner). In the more general case, complex formulas may be used in partitioning. For simplicity, we will only consider partitioning by atomic predicates without negation.

Definition 5 (Partitioning) *Partitioning is an operation ρ that takes as argument a concept c=p_1 ^. . . ^p_n and a predicate q such that K* $\models$ *q[g] for some assignment g and it returns the conjunction ρ(c,q)=c^q. The partition is said to be restrictive if there exist some assignment g' such that (p_i ^q)[g'] is true for some p_i occurring in c, otherwise, the partition is expansive.*

According to the definition, the property q should be true for some set of objects in K. Thus, as with the CK theory, the definition considers only those predicates that are known to be true for some set of objects.[1] The partitioning is a restrictive one if there are objects in the domain that are known to have both the introduced property q and (at least) one of the properties occurring in the description of the

concept. If no known object verifies at least one of the properties p_i occurring in the concept and the property q, the partition is said to be expansive.

Example. *Consider the following K space: K={ship(ship_11), flies(bird_17), transparent(glass_34), floats(ship_11)}. Starting with the concept c_0=ship(x)^flies(x), we can apply partitioning to get $\rho(c_0$, float(x)) = ship(x)^flies(x)^floats(x) which is a restrictive partition (there are objects that are floating objects known to be ships). Or, we can have $\rho(c_0$, transparent(x)) = ship(x)^flies(x)^transparent(x) which corresponds to an expansive partition (there is no known transparent object that is a ship or that flies).*

It is possible to be more precise on partitions that can be operated. The previous definition allows some partitions that we do not necessarily want. We introduce some conditions on partitions.

Definition 6 (Proper partitions) *A partition ρ(c,q) is said to be proper if the following conditions hold:*

- *Relatedness. At least one of the terms occurring in q appears in c.*
- *Non-redundancy. The predicate q must be different than $p_i$occurring in c or have at least one different term.*
- *Consistency. There should be no contradiction; i.e. it should not be the case that $\rho(c,q) \rightarrow g\bot$.*

In the following, we will only consider proper partitions but we will refer to them simply as partitions.

Example. *Let K space be like the following: K={ship(ship_11), flies(bird_17), bird(bird_17), wing(wing_28), has(bird_17, wing_28), propulsion_system(ps_29)}.*

Let c = ship(x)^flies(x) be the initial concept. The following are not proper partitions: c^propulsion_system(y), c^flies(x) and c^¬flies(x). The first partition is not related to c, the second introduces redundant information and the third leads to inconsistency. On the other hand, c'=c^has(x,y) is a proper partition. Remark that we can properly partition c' to get c'^wing(y) and c'^propulsion_system(y).

Considering the above definitions, we can define a concept space as a tree of concepts generated by partitioning an initial concept c. The initial concept is the root of the tree and it is called the generative concept.

Definition 7 (Concept space) *A concept space is a treelike structure of concepts given by a set of concepts C with a distinguished element c (the generative concept) and where, for each c_i belonging to C either $c_i = c$ or there exists q such that $K \models q[g]$ for some assignment g and c_j belonging to C such that $c_i = \rho(c_j, q)$.*

We assume that all the partitions are not immediately generated and that the designer will progressively build its C space. This corresponds well to the idea of *expansible rationality*.[12] Let us also underline that, according to the way we conceived the concept space, it is not possible to *choose* in C space. C space is a hierarchy of ideal definitions that makes no reference to a particular domain of objects (at least not completely; see the definition); if any selection will be applied, it will be done so in K space.

4.3.2. Query and Expansion of the K Space and Validation of Concepts

We have seen that partitioning allows forming new concepts and generating a conceptual tree. Once a concept is formulated, a designer might try querying the knowledge space to retrieve relevant knowledge for further partitioning or validating the concept.

Definition 8 (K-query) *This is an operation that takes as argument a concept c, a predicate p_i occurring in c and that returns a predicate q such that $K \models k\boldsymbol{x}.(p_i \hat{} q)$ or, NIL, if no such q exists. From any concept of the C space, the designer can query the K space in order to retrieve related knowledge in the K space.*

K-query returns the value NIL when no knowledge exist about the query expression. Otherwise, it returns an additional property q that might be used for further partitioning. Remark that if a partition is operated using the returned predicate, this would be a restrictive partition. It is possible (and, in design, it is often the case) that K space cannot answer such a query in a satisfactory way. In such situations K space may be expanded by new knowledge.

Definition 9 (K-expansion) *This is an operation that corresponds to a (forward) move from a knowledge space K to a (next) knowledge space K' such that* $K \leq K'$.

K-expansion operation is the addition of knowledge to the available knowledge. One may wonder where such knowledge comes from. If we assume logical omniscience (that is, all the consequences of what is known is also known), introduction of such knowledge implies that the K space should interact with an environment space E.[13] The interaction of an agent (and particularly, a design agent) with its environment may cover many different processes such as perception, observation, communication with other agents, actions. For example, in design practice, one particular way of interacting with an environment is by building prototypes and observing their properties.

Definition 10 (K-validation) *The K-validation of a concept c is an operation that takes a knowledge space K and a concept c as argument and returns true, if* $K \models \mathrm{k}\boldsymbol{x}.c$ *or* $K \models \neg\mathrm{k}\boldsymbol{x}.c$ *and false, if not.*

A validation operation checks whether there are known objects that have the properties of a given concept c. A concept is validated if K-validation return true. This is not possible at the beginning of the process since, by definition, neither $K \models \mathrm{k}\mathbf{x}.c$ or $K \models \neg\mathrm{k}\mathbf{x}.c$ for the generating concept. In order to validate a concept, K should be extended by adding new knowledge (e.g. new objects and their properties or new properties for existing objects).

Just like for the concept space, we consider that validations are not immediate. A designer may decide to try to validate any given concept of the C space, independently of the order of partitions or the state of the K space. When a design process reaches a semantic conjunction (a concept that has been validated), necessarily we have more knowledge than beginning ($K' \geq K_0$) and there are other concepts to be investigated (if they are created during the reasoning).

Example. *Consider the following K space: K={flies(bird_17), bird(bird_17), has(bird_17, wing_23)), wing(wing_23), ship(ship_11), allow_transport(ship_11), has(bird_17, wing_22)) }. Starting again with the concept flies(x)^ship(x), some of the partitions that can be generated is depicted in Figure 1. In this example, some partitions are operated based on the available knowledge then, new knowledge is sought out to further partition. Design ends when a set of objects (o1, w2, ps2) appears which allows validating the last formulated concept.*

Some of the partitions depicted in Figure 1, such as fly(x)^ship(x)^bird(x) or fly(x)^ship(x)^wing(x), may not have any meaning for the reader. The point that should be underlined here is that *notions such as meaning, sense and semantics belong to K space*. These are all based on evaluations with respect to a knowledge space and the whole point of CK theory is to stress that such seemingly non-sense propositions are catalyzers for creating new objects and acquiring new knowledge.

5. SUMMARY

We have presented a formalization of CK design theory based on Intuitionist Logic. This logic reflects a particular view on reasoning and knowledge: an individual can learn new knowledge that was inaccessible to him before. This conception is well suited for modeling the evolution of K space in CK theory. However, as we have seen, in order to formalize the reasoning described by CK theory, we introduced a C space representing concepts that govern the acquisition of new knowledge.

Our work provides precise formal definitions for key ideas of CK theory and thus, enables us to formally analyze the properties of the reasoning process described by it. This will be addressed in future works.

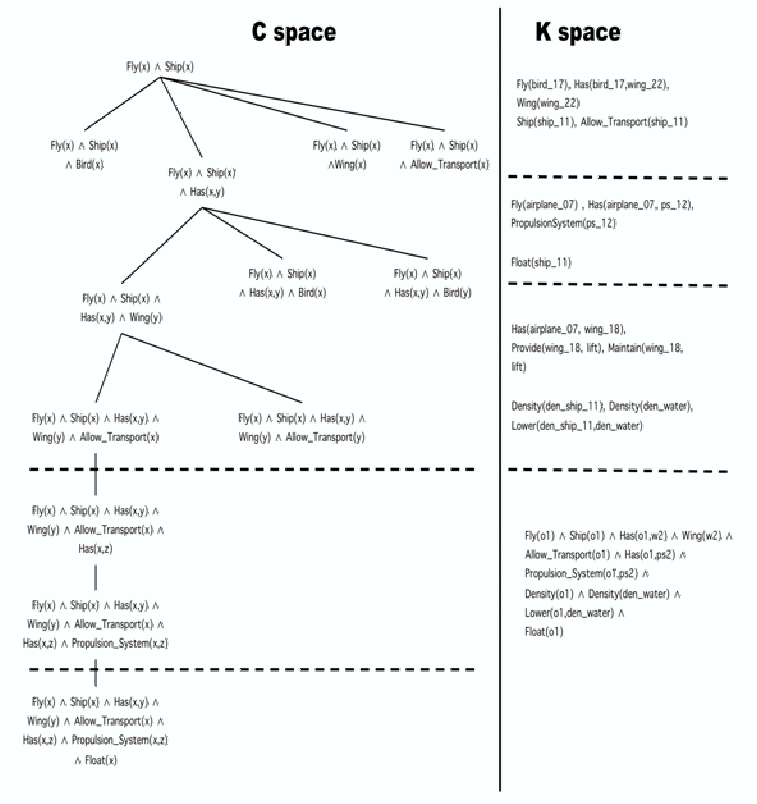

Figure 1. An example of dual expansion for concept and knowledge spaces. Dotted lines represent respective expansions of C and K spaces.

ACKNOWLEDGEMENTS

The author would like to thank to Armand Hatchuel, Joel Uckelman, Yann Chevaleyre, Nicolas Maudet and Lex Hendriks for comments on earlier drafts and enlightening discussions.

REFERENCES

[1] Hatchuel, A. and B. Weil (1999). Pour une th_orie unifi_e de la conception, Axiomatiques et processus collectifs. CGS Ecole des Mines, GIS cognition-CNRS.
[2] Hatchuel, A. and B. Weil (2002). La th_orie C-K: Fondaments et usages d'une th_orie unifi_e de la conception. Colloque Sciences de la Conception, Lyon.
[3] Hatchuel, A. and B. Weil (2003). A new approach of innovative design : an introduction to C-K design theory. ICED_03, Stockholm, Sweden.
[4] Hatchuel, A. and B. Weil (2007). Design as Forcing:Deepening the foundations of Ck theory. 16th International Conference on Engineering Design — ICED 2007, Knowledge, Innovation and Sustainability, Paris, France.
[5] Cohen, J. P. (1963). The independance of the continuum hypothesis I. Procedings of the National Academy of Science, USA.
[6] Cohen, J. P. (1964). The independance of the continuum hypothesis II. Procedings of the National Academy of Science, USA.
[7] Kazakci, A., Hatchuel, A. and B. Weil. (2008). A model of C-K design theory based on a term logic: a formal background for a class of design assistants. Marjanovic D., Storga M., Pavkovic N., Bojcetic N. International Design Conference 2008, Dubrovnik, Croatia.
[8] Kripke, S. (1965). Semantical analysis of intuitionist logic I, Formal Systems and Recursive Functions, 1965.
[9] Landman, F. (1991). Structures for semantics. Springer, 388p.
[10] Lewis, D., (1986) On the Plurality of Worlds. Oxford, New York: Basil Blackwell.

[11] Loux, M. J. [ed.] (1979) The Possible and the Actual Ithaca, London: Cornell University Press.
[12] Hatchuel, A. (2002). Towards Design Theory and Expandable Rationality: The unfinished program of Herbert Simon. Journal of Management and Gouvernance **5**(3), 260–273.
[13] Kazakci, A. and A. Tsouki_s (2005). Extending the C-K design theory: A theoretical background for personal design assistants. Journal of Engineering Design **16**(4), 399–411.
[14] Gardenfors, P. (1992) Belief Revision. Cambridge University Press New York, NY, USA.
[15] Fitting, M. C.(1969). Intuitionist Logic, Model Theory and Forcing. North Holland, Amsterdam. 191p.
[16] van Dalen, D., Ed. (1981). Brouwer's Cambridge Lectures on Intuitionism. Cambridge, Cambridge University Press.
[17] Brouwer, L. E. J. (1948). Consciousness, Philosophy and Mathematics. Proceedings of the 10th International Congress of Philosophy, Amsterdam.