Dual-Process Architecture for Reasoning in Design Innovation Problems

Alex Doboli and Simona Doboli

Abstract—This paper presents a dual-process architecture for reasoning in design innovation problems. The architecture includes (i) a uniform mechanism to create various associations among concepts and (ii) a procedure to construct rule-based reasoning flows through abstraction of associated concepts. Three reasoning procedures are discussed based on the architecture. Two case studies show that the dual-process architecture can be utilized for various innovation-related tasks, like searching the solution space search beyond the capabilities of statistical methods and identifying hints to improve the novelty of designs, hence exiting fixation.

I. INTRODUCTION

Creativity and design innovation have been main research topics in psychology, cognitive modeling, computer science, and engineering [1][2]. Creativity is usually characterized by the novelty and usefulness of a product or procedure. Many studies have focused on understanding through experiments how various factors and conditions impact novelty and usefulness [2][3], modeling creativity processes as formal, mathematical descriptions [4][3], and explaining the details of creative reasoning processes [5][6]. In parallel, design innovation in engineering explored important topics like analogies for improving creativity, and design descriptions and strategies for innovation.

In spite of significant results in understanding creativity and design innovation, there are important challenges that must still be tackled. This includes devising models, procedures, and software tools that support sustained design innovation, especially in state-of-the-art technological domains. Many experimental studies consider real-life applications, e.g., product advertising. Others indicate that techniques for systematic innovation, like TRIZ [7], have been used by major companies. However, there is less research on devising general-purpose models and tools that can be employed for representing design knowledge and automating the design process for real-life designs beyond the capabilities of current design automation tools. Specifically, to the best of our knowledge the experimental insight on creativity has not been yet considered to create circuit design tools that can emulate to a certain degree the reasoning and learning processes of human designers working in design innovation.

Devising general-purpose models and software tools for design innovation poses several challenges. A main challenge is the systematic combination of associative and rule-based (causal) reasoning during innovation. Associative reasoning uses the associations among concepts and their features to find alternatives that have been categorized together with the initial concept [8], or to identify other concepts which have occurred in related contexts as the current concept [8]. Associative reasoning is important in diversifying solutions, thus in deciding their amount of novelty. Rule-based reasoning establishes the causal structure of a design, hence validates its correctness and decides its utility up to certain degree.

An overview on dual-process reasoning is presented in [9]. Sloman [8] suggests that associations reflect similarity in structure and relations of temporal contiguity, e.g., regularities, frequencies, and correlations among features. Rules indicate logic operations upon symbolic structures. Many studies show that the two reasoning systems operate in parallel: associative reasoning offers quickly cues that are then validated by rule-based reasoning [10][6][11]. Evans indicates that there is an heuristic biasing before analytical processing based on the relevant features of the concepts [10].

This paper describes a novel architecture for dual-process reasoning in general-purpose innovation problems. While being consistent with existing approaches, the main contributions compared to other architectures are as follows: (i) a uniform mechanism to create associations of various kinds as well as (ii) a procedure to construct rule-based reasoning flows through concept abstraction. Three reasoning procedures are discussed based on the architecture.

More specifically, the traditional approach to define concept associations is based on static rules expressed as frames [12], schema, context-dependent parameterized schema, and action controllers, i.e. Xschema, Petri nets. Specifying the rules, however, is challenging due to the many situations that might occur for a physical system. Instead, the proposed architecture utilizes Pattern Recognition Units (PRUs) as elements of the associative structure to identify invariants in concept features, including the dynamic of their properties in time and space. Kelso [13] explains that dynamic patterns are critical in the self-organizing behavior of the brain, an idea which has also been popularized more recently in books like [14]. This enables a more natural, less cumbersome construction of the association networks through learning. The resulting associations are grouped on five successive layers (see Fig. 1).

Second, the paper explains that abstraction is a critical step in the construction of rule-based reasoning flows starting from the similarities among concepts and their associations. The construction process concludes with a fixed set of causal
patterns, which are also an invariant of the architecture.

The proposed dual-process architecture can be utilized for various innovation-related tasks, such as searching the solution space search beyond the capabilities of statistical methods, identifying hints to exit design fixation, and finding new concept combinations to address novel requirements.

The paper has the following structure. Section II offers an overview of the architecture. The main implementation details are presented in Section III. Section IV discusses reasoning based on the architecture. Two cases studies are presented in Section V. Conclusions end the paper.

II. OVERVIEW OF THE DUAL-PROCESS ARCHITECTURE

The associative network of the architecture has five successive layers distinguished by their roles. Fig. 1 shows the five layers. Each layer includes variables, concepts, attributes, attribute variations as well as descriptions of the learned effects of the attribute variations and the obtained rewards. (Note that each layer can have an arbitrary number of variables, concepts, attributes, attribute variations.) A detailed description of the composing elements is offered in [15]. Variables are features (characteristics) used to define concepts. Every variable is described by a set of physical values (called instances), similar to the model by Barsalou [16]. Concepts are clusters of variables, such that each cluster is well distinguished from another cluster. Attributes are correlations or associations among variables, and attribute variations describe variations of attribute values, e.g., variations in time and space.

The right part in Fig. 1 shows how the five layers emerge through learned connections between the basic building blocks of the architecture. The basic building blocks are called Pattern Recognition Units (PRUs) as their main function is to recognize simple patterns (invariants) in the firing of their inputs. \( I_i \) represent physical instances. The first layer identifies the variables \( (v_i) \) and concepts associated \( (C_i) \) with the current task. The second layer finds the correlations among the variables of the concepts \( (R_i) \). These correlations define the relations among the concept variables of Layer 1, including structural relations between concepts, such as if the concepts are parts of the same, larger entity. Layer 3 presents the properties of the attributes in Layer 2, like the variations of the attributes \( (\text{Var}_i) \). Layer 4 describes the effects \( (E_i) \), i.e., results, outputs, modifications, of attribute variations manifesting themselves for the identified entity and the given context. Finally, Layer 5 relates these effects to the observed rewards \( (\text{Rew}_i) \). More details are discussed in Section III.

A “cross-section” through the layered architecture has the same specification power as frames or schema [12]. Variables and concepts correspond to subjects and their adjectives, attributes and attribute variations represent verbs, objects, and adverbs. Effects and rewards complete the dynamics of the causality chain for a situation. As opposed to frames and schema which are static, the architecture is constructed dynamically based on the identified patterns (invariants).

A. Pattern Recognition Units

Pattern Recognition Units (PRUs) are the main building blocks of the architecture. A PRU recognizes a certain pattern (invariant) in the behavior of its inputs, e.g., patterns in input magnitude and time. More PRUs are networked together to generate more complex pattern recognition activities.

The pattern recognition capability of PRUs is achieved by adjusting (learning) the values of three kinds of parameters: (i) adjustable time sensitivity of its inputs, (ii) adjustable magnitude sensitivity of its inputs, and (iii) adjustable interconnecting of its inputs and outputs. Complex recognition capabilities emerge due to these three kinds of parameters.

As shown in Fig. 2(a), each input of a PRU has an adjustable time sensitivity \( \Delta t_i \) and an adjustable magnitude sensitivity \( \Delta \text{Mag}_i \). A PRU senses its input \( i \) if the magnitude \( M_i \) is within its corresponding range, \( M_i \in \Delta \text{Mag}_i \), and a sensed input is preserved for time \( \Delta t_i \) (hence, the input is locally memorized). If all inputs are sensed within their \( \Delta t_i \) then the PRU output fires (similar to AND operator).

Diverse time-related behaviors are recognized by appropriately adjusting \( \Delta t_i \)’s. Small \( \Delta t_i \)'s describe time synchronization (simultaneous time) of the related inputs. The larger the \( \Delta t_i \) the longer the time interval over which synchronization is sought. Invariant sequences in time of the inputs are recognized as follows: the earlier the input, the longer its corresponding \( \Delta t_i \). Hence, the first input has the largest \( \Delta t_i \) and the input expected to be last has the longest \( \Delta t_i \). Small \( \Delta t_i \)'s also encode inputs continuous in time.
Similarly, different magnitude-related patterns are identified by adjusting $\Delta Mag_i$’s. If all but one input $j$ have very large ranges, e.g. $(0, \infty)$, then the PRU senses the presence or absence of input $j$. Inhibition of an input can be implemented this way too. If all signals $In_i$ are wired to each input of a PRU (Fig. 2(b)) then having at least one input fires the output. This corresponds to OR operator. If the magnitude range $\Delta Mag_i$ is defined based on the current value of input $In_j$, this results in detecting the pattern whether $In_i$ is ‘greater’ than $In_j$. The ‘less’ pattern is encoded similarly. If $In_j$ is constant and $In_i$ has small $\Delta Mag_i$ then PRU implements the conditional invariant that $In_i$ is equal to $In_j$’s constant.

Finally, patterns of the input-related gradients can be found too. For example, the structure in Fig. 2(c) identifies if input $In$ increases (or decreases) in time, thus there is a positive gradient of $In$. For the two inputs, $\Delta t_1 > \Delta t_2$, indicating that $In_1$ arrives first, hence is the initial value. If $In_1$ is used to set $\Delta Mag_2$ then the output fires only if the new value of $In$ at input 2 is greater than the old value. Physical activities, such as an object moving away from the observer can be encoded this way.

B. Construction of the Dual-Process Architecture

The five layers of the associative network emerge as a result of the pattern recognition activities of PRUs at the previous layers. As an example, let’s assume that Layer 1 identifies two concepts, $C_1$ characterized by variables $A_1$ and $A_2$ and concept $C_2$ described by variables $B_1$ and $B_2$. If $A_2$ and $B_1$ are always the same (or in any other invariant relation) then Layer 2 introduces a new PRU to indicate their equivalence. For example, such a situation can describe the structural connection of concepts $C_1$ and $C_2$, in which the output of $C_1$ (variable $A_2$) is the input to $C_2$ (variable $B_1$). Layer 3 identifies PRUs defined by the invariant trends of the PRUs in Level 2, such as variable $A_1$ is decreasing in time while variable $B_1$ is increasing. The trends represent properties of the concepts in Layer 1 related through the associations in Layer 2. Layer 4 indicates the results obtained in time. Finally, Layer 5 defines whether the results in Layer 4 are an invariant benefit or loss in terms of a certain utility.

The second type of pattern recognition creates the rule-based, causal structure of the architecture. This is achieved through abstraction, which is the process of finding patterns across different concepts, attributes, attribute variations, effects, and rewards, hence along all five layers. For example, abstractions result by grouping the common variables of concepts. As some variables are unique but still relevant to the nature of the concept, new, abstract variables ought to be also added to maintain the meaning of the abstraction. In Fig. 2(d), concept $C_1$ originates abstraction $A_1$, attribute $R_1$ is the base for abstraction $AR_1$, and so on. The rule-based path in Fig. 2(d) results through abstraction and includes abstract concepts $A_i$ and abstract relations $AR_j$.

C. Operation of the Dual-Process Architecture

Some variables and concepts are identified directly based on the sensed physical data. However, other associative elements, like correlations, attribute variations, effects and rewards, are more difficult to find as they are not always observed and because of the large number of possible associations that need to be searched. The rule-based component of the architecture helps to reduce the complexity of this search and validate the search by inferring the causal meaning of the selected associative elements. Hence, the rule-based component serves as a skeleton (template) to aid the exploration of the highly connected, associative network. Many paths are created by the associations of the architecture, which would be inefficient to search through exhaustive methods. Instead, the rule-based component acts as an aggregator of the corresponding associations to reduce the complexity of the search.

The dual-process architecture operates as follows. The identified concepts are used to find the related abstractions and the rule-based, causal chain that connects the abstractions. Then, the elements of the causal chain point towards the PRUs of the associative network which are likely to connect the identified concepts. If incorrect predictions are made with regards to the PRUs or causal chains then other alternatives are considered.

The reasoning process using the dual-process architecture creates cycles between the association network and the rule-based, causal chains. Fig. 2(d) shows such a cycle in bold lines. Cycles are equilibrium states (steady states) of the architecture’s operation.

The operation of the dual-process architecture produces continuously equilibrium states that reflect various criteria depending on the tackled application.

III. Detailed Implementation

A. Variables. Concepts. Variables are represented using PRUs similar to the pattern identification units suggested in [14]. Each variable is defined by its set of instances $i$, which represent physically sensed entities. The pattern recognition functionality of the variables identifies the main features of the instances and eliminates any of their imperfections and noise. As shown in Fig. 3(a), firing of a single instance $i$ is sufficient to detect the variable (OR functionality of the PRU).

Variables are clustered to form concepts, like concepts $C_i$ in Fig. 1. A specific concept is identified if a sufficient number of its main variables fire simultaneously. This behavior is described by the representation in Fig. 3(b) (AND function among a certain number of inputs).
Example: Fig. 4(a) presents the associative network for the following example. PRU $A$ identifies instances $a_1$ and $a_2$, thus implements variable $A$. Instances $u$ and $v$ are initially not associated to any variable but become associated to a new variable $X$ after reasoning with the architecture. Instance $r_1$ describes a correlation of instances $a_1$ and $u$, and instance $r_2$ defines the correlation between $a_2$ and $v$. Instances $r_1$ and $r_2$ pertain to concept $R$, which belongs to the level next to the level containing variables $A$ and $X$. The PRU corresponding to $R$ correlates variables $A$ and $X$.

Note that instances $u$ and $v$, which do not correspond yet to a variable, are grouped ad-hoc to form variable $X$. The idea of ad-hoc grouping directed by a goal (e.g., the tackled problem) is also suggested by Barsalou [17]. Similar to central tendency [17], the ad-hoc variable $X$ is characterized by a representative instance, like the new instance $w$ in Fig. 4(a). The representative instance can be the average or the median of the instances $u$ and $v$. Even though instances $a_1$ and $a_2$ are already part of variable $A$, the variable might have a representative instance $a_u$ too.

B. Abstractions. Fig. 4(b) presents the abstraction process to explain how the rule-based, causal chain of the architecture results starting from associations. The associations among the most abstract variables, concepts, and attributes produce the causal connections of the elements to express the problem to be solved. This observation is supported by the following three lemma, which relate instances, variables, concepts, attributes, attribute variations, and goals of an abstraction and its related concepts. For brevity, we did not introduce the proofs.

Lemma: Each variable abstraction reduces the number of instances of its related variables.

Lemma: A sequence of abstractions for a variable converges either to a specific average value for the variable, or to $\phi$, if the instances are dissimilar.

Lemma: Each abstraction has less variables (or concepts) than its corresponding instancing concepts.

Example: Let’s assume that concepts $X$, $A$, and $B$ are related through relations $r_1 : X + \frac{1}{2}$ and $r_2 : X + A \times B$. $C_\phi$ corresponds to the two instances $B$ and $\frac{1}{2}$, which have diverging (unmatched) values. The abstraction represents the relation $r_{abstr} : X + C_{abstr}$, where $C_{abstr}$ corresponds to $A$, $C_\phi$, and $r_\phi : A \times C_{phi}$.

Lemma: Any unmatched (diverging) attributes or attribute variations are not expressed in the abstraction.

Proof: As shown in Fig. 5(c), concepts $C_1$ and $C_2$ and relation $r_1$ are replaced by abstract concept $C_{abstr}$ as they do not have an equivalent in the other structure. If relation $r_1$ or any of the concepts $C_1$ or $C_2$ are introduced in the abstraction then the second structure cannot be derived from the abstraction. Fig. 5(c) presents the general case, in which multiple input concepts to the similar attribute $R$ are abstracted to generate abstractions $C_{abstr,1}$ and $C_{abstr,2}$.

Fig. 5(d) presents the implementation of the abstraction process which involves the wiring of concept $C_{abstr}$ into the network structure so that the dissimilar PRUs are shared.

Lemma: An abstraction has a simpler goal than its related concepts.

C. Attributes and attribute variations. Attributes result based on the correlations between the variables of one or more concepts. Hence, like variables, attributes are a kind of PRUs. Correlations based on a single variable express variations (trends) in space and time, e.g., increasing, decreasing, accelerating, decelerating, and so on.

Time related attributes. Fig. 3(c) compares the variable $v$ at two successive time instances, i.e. $t$ and $t + \delta t$. This is achieved by using the parameterized time sensitivity $\Delta t$ of the related PRU. If the incoming signal encodes magnitude then the attribute indicates the growing or decreasing of the variable. Similarly, Fig. 3(d) checks the existence of the incoming signal, e.g., if the incoming signal exists at times $t$ and $t + \Delta t$ then it is a continuous signal for that time range.

In general, the attributes in Figs. 3(c) and (d) express the behavior of a variable in time. They are approximations of the partial derivates of the variable in time or express time correlations with other variables, like synchronization, delays (separations), and ordering in time. More complex time behavior results by composing such structures.

Causal relations. Causal relations emerge during the abstraction process. Two concepts $C_1$ and $C_2$ are linked in a causal relation, if the firing of $C_1$ always precludes the firing of $C_2$. Fig. 5(c) illustrates the case, where the time sensitivity $\Delta t$ of PRU $R$ with respect to the input from $C_{abstr,1}$ is longer than that from $C_{abstr,2}$, thus indicating the pattern that the occurrence of the first one always precedes the occurrence of the second one too.

Lemma: Causal relations related to the application goals cannot be eliminated during abstraction.

Description of structure. Structure is expressed through PRUs that indicate that certain distance-related variables of two or more concepts are always equal, thus each variable is a substitute of the others. The related PRU identifies that the variables are constantly equal over an unbound,
continuous time range. Similarly, the variables and concepts of a static scene are related through attributes indicating the invariance of their relative distances. The corresponding PRU indicates that the relative sizes of the related magnitude-encoding inputs are constant over a continuous time interval. Thus, PRUs for structure identification find invariants of dimension-related variables based on continuous firing of their inputs.

Examples: Fig. 6 depicts the associative structures for some examples discussed in [12]. The example in Fig. 6(a) represents the fact that the tail and wing of birds are covered by feathers. PRU Sim indicates that concepts TA (tail) and FE (feathers) occur simultaneously, hence the receiving of a firing signal from one concept is always associated to a firing signal from the other concept too. Figs. (b) and (c) present attributes related to physical dimensions used to show how parts are integrated to form aggregated concepts. Fig. (b) indicates that claws (CL) are at the end of feet (FE). PRU Pos describes the fixed position of claws as a part of feet. PRU RelSize states that the relative size of beak (BE) and eyes (EY) is constant. Firing BE or EY also activates the concept for head (HE). There is also a causal attribute between beak (BE) and chirp (Ch). e.g., having a beak produces chirp. Finally, Fig. 6(d) presents a more complex associative structure. Concept wings (WI) has as variables color (CO) and shape (SH), which is described by instances i1 and i2. Color variable is the same for concept feather (FE) too. Finally, PRU Sim between the three concepts describes that feathers cover any part of wings.

IV. REASONING USING ASSOCIATIVE NETWORKS

This paper discusses the using of the dual-process architecture for three reasoning situations: (i) identifying a solution in the associative network, (ii) creating implementations of higher utility, and (iii) producing novel implementations for new goals and functions.

i. Identifying a solution in the associative network. The step finds in a large associative structure the variables, concepts, their correlations and attributes so that they create a solution. The identified information corresponds to a reasoning sequence built based on abstractions of the information.

The reasoning process utilizes simultaneously the associative network and rule-based, causal chains. It includes three situations: First, abstractions of the concepts identified using physical instances are utilized to isolate the most likely causal sequence of rules corresponding to the abstractions. This sequence is then used to search how the variables of the concepts are correlated with each other to create attributes, attribute variations, and effects of these variations that correspond to the considered goals. Second, if the most likely rule chain fails then the other chains are selected in the order of their likelihood. Third, if the previous two steps are unsuccessful then the next higher abstraction level is selected, and then the two previous reasoning steps are repeated. The reasoning process updates the likelihood information of the rule chains to reflect the specifics of the current search.

Fig. 7 illustrates the reasoning process. The variables V1 and V2 identified using physical instances activate the links to concepts C1 and C2, respectively. However, the amount of associations between the two concepts is large, and therefore cannot be searched exhaustively. Instead, as described in the first step, the most likely abstractions Abstr1 of concept C1 and Abstr2 for C2 are identified. They are related through PRU Rel1. The likelihood of this structure is prob1. A less likely structure (of probability prob2) is formed out of abstractions Abstr1 and Abstr2 related through Rel2. Finally, probability probnext is the likelihood of selecting a higher abstraction, if the current abstraction fails in finding the needed elements in the associative structure. Probabilities characterize the effort invested in searching the associative network with a given rule-based pattern (causal chain).

ii. Creating implementations of higher utility. This reasoning case represents situations in which the goal and functions of the current solution remain unchanged but a higher utility is searched, e.g., superior meeting of the existing performance requirements or meeting of new requirements. We denote this case as inward reasoning as the corresponding abstraction hierarchy and reasoning paths remain unchanged but new ways of instancing are being explored.

Specifically, this reasoning case is expressed as searching for new attributes among variables, i.e., variables X and V, to improve utility. The group of tackled variables is identified based on the sensitivity of the found reasoning paths to the variables. There are three different situations:
1. Alternatives. Variables $X$ and $V$ are part of multiple concepts and different relations $r_j$ link the concepts. A new relation $r_j$ is selected from these alternatives. Fig. 8(a) illustrates the situations, its associative structure and activation process. The activation steps are shown in the figure.

2. Analogies. Variables $X$ and $V$ are not part of multiple concepts but instead they are analogous variables in related concepts. The relations among the analogous variables are used to link variables $X$ and $V$. This process corresponds to analogical reasoning [3]. Fig. 8(b) shows the associative structure and the activation steps for this case.

3. Induction. This situation corresponds to induction and is presented in Fig. 9. The relations among variables $X$ and $V$, among analogous variables, or any other useful relation in the associative network are utilized to identify new relations. Also, the existing instances are the starting points to create new instances with the same patterns. No associative structures that are outside these patterns are produced.

The reasoning process is illustrated in Fig. 9(a) and is described by the algorithm in Fig. 9(b). The associative network corresponding to the induction process is called network for induction. The two instances in Fig. 9(a) have relation $sim$ as the common part, which thus becomes part of the network for induction. This creates the middle structure, in which the three $C_{abst,1}$ are placeholders for the inputs and output of relation $sim$. Each $C_{abst,1}$ is handled through a similar process until all common sub-structures of the instances are expressed in the network for induction. For the two instances in Fig. 9(a), $C_{abst,1} = X$, $C_{abst,2} = U$, and $C_{abst,3} = V$. The remaining, uncommon sub-structures are the actual alternatives utilized to create induction instances.

Fig. 10(a) presents the inductive reasoning process depicted as an automaton, where each of the branches corresponds to an alternative sub-structure. The common parts of the instances appear in the automaton, hence they describe features that must be present for any induced instance. The representation shows the recursive nature of induction. The representation is important because it allows to reason about the changes in attributes and their behavior, and how these changes relate to the searched requirements and goals. It also supports reasoning about the number of induction steps that are expected, hence the flexibility of an associative structure to generate more new instances.

The changes of the attributes present in an associative structure and the modifications of the behavior of these attributes is characterized as shown in Fig. 10(b). Each induction alternative $C_{abst,1}$ produces the corresponding $Change_{c}$, which affects the variables in the network, the way these variables are clustered through correlations and attributes, and the nature of these correlations and attributes. The changes due to a sequence of induction steps is summarized as in Fig. 10(c). Region I corresponds to the exploration stage, in which possible induction alternatives are analyzed. Region II is the improvement stage, in which consecutive alternatives generate sustained improvement in satisfying the requirements. Region III is the saturation region, when changes are less important. The flexibility of the associative structure corresponds to the modifications during the three regions.

3. Producing novel implementations for new goals and functions. The reasoning process identifies sufficiently different solutions (expressed as states of equilibrium) by adding extra concepts and attributes to a current solution. The extra attributes and concepts can be due to external hints.

The reasoning process is described in Fig. 2(d). The initial state of equilibrium includes concepts $C_1$ and $C_2$ on the top layer, their abstractions $A_1$ and $A_2$, and concept $R_1$ on the next layer and its abstraction $AR_1$. Two extra attributes (hints), $\Delta C_1$ and $\Delta C_2$ are combined with concepts $C_1$ and $C_2$ respectively to create the new concepts illustrated as oval shapes. Similarly, the two new concepts originate two new abstractions distanced by $\Delta A_1$ and $\Delta A_2$ from the two top abstractions and a third abstraction $AR_2$ that differs by $\Delta AR$ from $\Delta A_1$. The new concept $R_2$ is an instance of $AR_2$.

The new abstraction $AR_2$ must have sufficient flexibility to produce an instance that is at distance $\Delta$ from $R_1$. The abstraction level at which the equation is solved is part of hint selection, as shown in Fig. 2(e). If the search for the unknown concept $C_x$ is at the same level as an existing concept $C$ then the required attributes $\Delta_x$ acts as a separation criterion for concepts $C$ and $C_x$, thus the only insight available is that $C_x$ represents a different variable clustering than $C$. There is obviously a very large number of such clusters. Instead, the complexity of the search is reduced if the search is based on an abstraction $A$ for concept $C$. Combining $A$ with the
unknown hints $H$ produces abstraction $A_x$, for which $C_x$ is an instance. As abstraction $A_x$ incorporates relevant features from concept $C$, the complexity of searching $C_x$ is less than in the previous alternative.

V. TWO CASE STUDIES

This section discusses two case studies for using the dual-process architecture in reasoning.

A. Imaginary Bugs

The first study refers to an example presented in [18]. Let’s consider the five imaginary bugs shown in Fig. 11(a) of type *monek*. Bugs are moneks if most of the following five characteristics are set as follows: horns are long, head is round, body is dotted, number of legs is eight, and tails is short. Thus, the five bugs shown are all moneks.

Fig. 11(b) illustrates a fragment of the associative structure created to represent the five bugs. The variables correspond to the parts forming a bug: $HO$ are horns, $HE$ is head, $BO$ is the body, $TA$ corresponds to tail, and $LE$ is for legs. Each variable has two possible instances as defined by the five bug exemplars ($Sh$ - short, $Lo$ - long, $Ro$ - round, $An$ - angular, and so on). The four correlations $Eq$ indicate that the observed physical position of the related variables is always the same, such as horns and head. The correlations are utilized to express the way bugs are built out of their parts. Each concept $C_i$ corresponds to a specific bug. The inputs to $C_i$ are the related variables, the specific instances of the variables, and correlations $Eq$. Concept $C_i$ represents some of the characteristics of bug 1. Finally, each $C_i$ is an input to concept *Monek*, which pertains to the effect layer of the associative structure.

Fig. 12 presents the abstraction structures that are created for the associative network of the architecture. According to [18], humans are mainly guided by classification features, such as features which allow to distinguish (classify) the five exemplars. Then, bug 4 forms a separate category as it is the only one with angular head. Fig. 12(a) is the abstraction for bug 4 and Fig. 12(b) is the abstraction of the other four bugs. For the latter cluster, the abstraction process continues by separating bug 2 from the set of bugs 1, 3, and 5 as it is the only one with short horns. Fig. 12(c) presents the abstraction for the set of three bugs. Another level of abstraction results by combining the abstractions for bugs 2 and 4. The resulting representation in Fig. 12(d) suggests that a dotted body, eight legs, and short tail represents a monek. The corresponding causal chain states that

\[(PA = Do) \land (Le = 8) \land (TA = Sh) \rightarrow Monek \quad (1)\]

The created network, including the associative structure, its abstractions, and the causal chain is used in reasoning as follows. Let’s assume that a new imaginary bug has the following characteristics: long horns, angular head, dotted body, eight legs, and long tail. Note that this bug is not in the initial set of five bugs. Hence, finding its type is conceptually similar to first exhaustively adding to the architecture all bugs with missing features, followed by applying the first reasoning situation in Section IV.

The main instance values of the bug, e.g., the angular head which acts as the distinguishing feature, activates the abstraction in Fig. 12(a) followed by the abstraction in Fig. 12(d). The causal relation (1) suggests that the new bug is a monek, if the three variables have values as defined by the relation. Even though there is matching with respect to the body pattern and number of legs, the tail variable value (long) does not match the expression (short). Hence, the associative network must be extended to decide the type of the new bug.

The reasoning process drops variable tail as it is the cause of mismatch. In addition, it explores if the remaining, unstudied attributes, horns and head, can be used to decide the kind of the bug. If horns are used as the distinguishing variable then bug 2 represents one cluster and bugs 1, 3, 4, and 5 the second cluster. The procedure considers the second cluster as the horns of the current bug are long. The resulting abstraction is in Fig. 12(e) and suggests that horns must be long. As the new bug has long horns, the conclusion is that it is a monek.

\[(PA = Do) \land (Le = 8) \land (HO = Lo) \rightarrow Monek \quad (2)\]

Similarly, for another bug with round head the procedure reasons that it is a monek too.

\[(PA = Do) \land (Le = 8) \land (HE = Ro) \rightarrow Monek \quad (3)\]

The trace of the reasoning process is summarized in Fig. 12(f). After identifying the causal sequence in expression (1), the considering of the pattern and leg variables creates two more causal sequences in relations (2) and (3).

B. Intelligent Embedded Systems

The second study relates to design innovation in intelligent embedded system design. The task is to devise creative solutions to embedded systems based on a given set of building blocks, like various sensors (i.e. temperature, gas, humidity, camera, etc.), processors, wireless links, and actuators (e.g., robot arm, fan, switch, motor, etc.). Students had to solve this problem for different sets of building blocks. Design creativity was evaluated based on uniqueness and usefulness.
The dual-process architecture corresponding to the solutions includes the sub-structures shown in Fig. 13. The associative network includes PRUs recognizing the following patterns: 1) input variations in time, magnitude and space (e.g., variations of temperature, gas, sound, density, etc.), 2) simultaneous variations of multiple inputs (i.e. temperature and voice), 3) auto-correction of errors for different conditions (like adjusting the input based on the amount of background noise), 4) computation in response to inputs, 5) autonomous (self-adjusting) computation for different conditions, 6) action associated to the computation, and 7) remote operation (e.g., through wireless connections). The structures of the causal chains looks as shown in Fig. 13.

Rating by human raters showed that the creativity of the solutions is mainly correlated to the place in which the solution is used and the purpose of the solution followed by the achieved function.

The dual-process architecture can be used to identify hints that would lead to devising more creativity designs. This corresponds to the third reasoning situation in Section IV. Let’s assume that the current design is considered to have limited creativity as it is a frequently proposed solution. For example, very many students developed home security systems that track intruders based on video camera monitoring and actions like triggering alarms or notifying police. The purpose of hints is to suggest other solutions which tackle new goals or are used in different places while using mostly the same associative network and causal reasoning.

Note that changing only the kind of inputs (video) would not necessarily increase creativity as input types are less correlated to it. Hence, hints must mainly relate to actions and their goals and place of use.

The provided hints introduce new features that can be combined with the features of the concepts present in the causal chain of the current solution. Let’s assume that the causal chain of the typical video-based security system (monitoring through video camera - computation to assess situation - response) is used as a starting point to identify a more creative solution. A new application goal and place would be to create a similar system for improving the comfort at home, e.g., warning if a user is too tired while working at the computer. The hints to be offered must help devising new designs. However, hints can relate to each of the concepts of the causal chain, like the monitoring, computation, and response parts.

As the novel goal is to devise a warning system to suggest if a person is too tired, hints must focus on finding ways of detecting if the person is too tired or not. This relates to the monitoring and computation elements of the causal chain. Following the ideas in Fig. 2(d) and (e), various hints are possible, like monitoring in time the body position, amount of activity, frequency of striking the keyboard, movement of eyes, and so on. Among these alternatives, the hint that suggests to monitor eye movements using a computer’s camera results in a design solution that is unique and is still easy to adapt starting from an existing video-camera based security system.

VI. CONCLUSIONS

This paper presents a dual-process architecture for reasoning in design innovation problems. The architecture includes (i) a uniform mechanism to create various associations among concepts and (ii) a procedure to construct rule-based reasoning flows through abstraction of associated concepts. Three reasoning procedures are discussed based on the architecture. Two case studies show that the dual-process architecture can be utilized for various innovation-related tasks, like searching the solution space search beyond the capabilities of statistical methods and identifying hints to improve the novelty of designs, hence exiting fixation.

REFERENCES