INTERNATIONAL DESIGN CONFERENCE - DESIGN 2014
Dubrovnik - Croatia, May 19 - 22, 2014.

CONCEPTIVE ARTIFICIAL INTELLIGENCE: INSIGHTS FROM DESIGN THEORY

A. O. Kazakçı

Keywords: artificial intelligence, design, design creativity, artificial creativity

1. Introduction – how can design research contribute to AI?

Artificial intelligence is a vast field of research whose output is shaping information and communication technologies that affect our economic and social life on a daily basis. On the rather rich list of topics relevant to research in artificial intelligence (AI), it is surprising to note that creativity, a hallmark of human intelligence, is at the bottom of the list and not a priority in the mainstream research on AI. For a field whose ambition is to produce human-level performance in tasks that require intelligence, this absence is curious. A major reason for this apparent lack of interest in studying creativity formally and implementing systems that aspire to be creative is the absence of clear, rigorous definitions of creativity. This lack of clarity is in contrast, for instance, to decision-making and learning, two processes for which more than half a century of efforts have led to precise formal frameworks enabling systematic and fruitful studies of these notions and the development of high-performance systems.

In contrast to its low priority in AI, creativity has been studied intensively in design research, especially using empirical approaches. Starting with early work by Eastman [1969], [1970], Akin [1978] and others, a major approach that has been used in the study of the thinking processes of designers is protocol-based analysis. Several contributions have been produced (see, e.g., Cross [2001] for a review). One overall striking feature of most of this research is the quasi-total reliance on the problem-solving paradigm in the interpretation of the results. There is now a growing consensus (see, e.g., Dorst [2006] or Hatchuel [2002]) claiming that problem-solving (even in a broad sense of the notion) is too restrictive as a "lens" through which to interpret design – which is a cause for concern in the interpretation of these results. Moreover, adopting the problem-solving paradigm as their conceptual framework for analysis, these studies do not contribute to the modeling of design – which is a cause for concern if design research is to find and develop its own authentic models.

In a paper called *Natural intelligence of design*, Cross [1999] argues that design research should also contribute to AI and not only the other way around. The current paper adopts and expands that position: design involves possibly the richest forms of reasoning, thereby providing a privileged context for the study of human cognition. We believe design reasoning is substantially different from ordinary reasoning situations because it involves the construction of previously non-existent objects. Rather than reducing design to other cognitive phenomena (such as problem-solving or incubation), design research should build richer models of design and creativity that would also be useful for other disciplines. If we simply take formal models developed for AI and use them to describe design, we are restrained and forced to reduce design theories to the reasoning paradigm underlying decision-making and learning – eliminating the possibility of studying design creativity. In contrast, if we produce models specific to design, thus representing the specificity of design reasoning, we may be able to offer insight in return to other disciplines.

1.1 Autonomous artificial systems require conceptive intelligence

A most important chapter in current AI research is to build systems that are autonomous. In most cases, if not all, what is meant by autonomous is the ability of an agent to behave within set limits in a manner designated by the system-builder without the necessity of human intervention. This type of design is enforced by an omnipresent engineering concern: the reliability of a system, which requires, in turn, predictable performance. Typically, on projects such as the Curiosity rover [Goth 2012], which entail substantial investments, the reliability of the system has priority over most other possible criteria. In such contexts, a system that has the ability to *surprise* its builders would rarely be a particularly desirable feature. Thus, most of the efforts in AI have concentrated on planned and predictable behavior, independent of the technique used to implement intelligent functions. From this seemingly natural characteristic of AI philosophy and objectives, it can be observed that all incentives and aspirations for systems that can surprise their builders in a desirable manner have also been eliminated. This insistence on the construction of plans of action (whether off-line or on-line), voluntarily restricting the domain and optimizing algorithms or programs for a specific, pre-defined and fixed set of tasks has naturally prevented AI researchers from developing systems capable of building new tasks in a creative manner that exploits old experiences to respond to novel situations.

1.2 Beyond solving given tasks: conceptive systems capable of designing original tasks

The present paper claims that truly autonomous systems require a specific form of reasoning that we call *conceptive intelligence*. By conceptive intelligence, we mean the capacity for an agent to design new concepts with respect to what it has observed (outside the scope of what is learned, e.g., by induction, over the observed objects) and to take necessary actions to build, realize or implement those concepts. Such systems would thus be able to formulate new tasks continuously and attempt to solve them. There are numerous engineering and theoretical challenges for building such a system. These issues have been discussed in AI under the theme artificial life or open-ended evolution [Bedau 2003]. The perspectives offered are based on traditional paradigms of AI, such as learning, interaction and randomness. In this work, we present an alternative view called imaginative constructivism, originating from design research [Kazakci 2013]. Based on the previously described creative reasoning process [Brouwer 1907, 1908, 1948], [Heyting 1975], [van Dalen 1981], [Niekus 2010], this view suggests that design is a process by which the construction of objects proceeds toward conceivable and imagined properties. This is a dual constructivist process in which creativity can occur both at the level of the top-down generation of new definitions and the bottom-up generation of methods for building objects.

1.3 Brouwer machines: a model for conceptive systems

Given the above orientation, this paper defends the thesis that classical and foundational models in AI and related fields, such as decision and learning models, are implicitly based on certain premises that we call *the-world-as-it-is* paradigm. We discuss basic formal models of decision-making and learning to explain the differences of these models compared with design (Section 2). This discussion allows us to introduce an alternative worldview, namely, *the-world-as-it-can-be* view, based on the notion of design as imaginative constructivism (Section 3). This framework allows us to discuss and analyze the traditional notion of search, which is omnipresent both in AI and design literature. We analyze several search processes and discuss their limits for describing design. In particular, we argue that a combinatorial search can be observed as a construction process – *although it has hardly ever been applied in the dual constructivist perspective described by the notion of imaginative constructivism*. Building on these analyses, we sketch a model, called a Brouwer machine, for a system that would incorporate a form of conceptive intelligence and discuss certain issues related to its implementation. Finally, we discuss the type of creativity that can be achieved using genetic algorithms. By interpreting this approach through our framework, we show that these models are construction machines rather than conceptive systems.

2. Revealing hidden limitations of traditional formalisms for conceptive reasoning: “the world as it is” as a paradigm

The traditional formal basis of models of decision-making and learning are implicitly based on a paradigm that analyzes the world as it exists. In the decision-making paradigm, decisions are made about objects that exist or that are known to be feasible. In machine learning, the aim is to learn categories in a bottom-up fashion for objects that exist. Consequently, these formal approaches are not adapted for the creation of new objects. Let us discuss the properties of those models based on the underlying formalisms.

2.1 The decision paradigm as an evaluation of known objects

Tsoukiàs [2008] defines a generic evaluation model into which many of the existing decision-aiding models and methods can be fit. His model is an n-tuple:
$M =< A, D, E, H, U, R >$, where

- A is a set of objects (alternatives, solutions) to which the model will apply,
- D is a set of dimensions (attributes) under which the elements of A are observed and measured,
- E is a set of measurement scales associated with each element of D,
- H is a set of criteria under which each element of A is evaluated,
- U is a set of uncertainty measures associated with D and/or H and
- R is a set of operators enabling synthetic information about

the elements of A or $A \times A$, namely, aggregation operators (acting on preferences, measures, uncertainties, etc.), to be obtained.

The distribution of the various parameters in the model irrevocably delineates where the majority of the efforts have been concentrated in the decision literature. With the exception of A, all the elements of the model are intended to measure and compare the properties of the alternatives. Objects from A are described on a certain scale along different dimensions. Whether uncertainty measures are present, an aggregation procedure compiles this various information, where some information is usually lost, but an overall evaluation is performed.

Tsoukiàs [2008] claims that this model can accommodate almost all major formal decision techniques and approaches. It is instructive to note that this model can only function if all the model parameters are supplied; otherwise, there can be no evaluation. Compared with decision processes, in design situations, most of this information does not exist and cannot be collected simply by asking participants about the process. In particular, the set A is empty prior to the process [Hatchuel and Weil 2002]. It is precisely the aim of the design process to construct the objects called alternatives. Innovative design is such an important process in current economic processes because we do not have alternatives to many challenging problems. Most of the writings about the decision paradigm consider that the inability to make (correct) decisions will lead to a crisis. However, major crises occur when no alternatives exist when an action is necessary.

It should be remarked that the decision paradigm does not consider the question of generating objects. Some work exists, mainly in engineering design literature, using the terms “generation of alternatives” by means of evolutionary computation. In these works, what is being generated is a discrete and finite set of alternatives, typically on the Pareto frontier given by some set of criteria, from among an infinity of solutions (called the feasible solution set) that are assumed to already exist and be feasible. Thus, no new object is being created, and objects that are unfeasible at the beginning of the process are not even considered.

In decision and evaluation models, objects are defined at the beginning of a process and assumed to be feasible; the only knowledge that is being derived from the process is the preferential information from what is already known about the objects. However, design focuses on the construction of new objects.

Another way to characterize the difference between decision and design processes is to consider the notion of “states of nature” that are omnipresent in decision models. In decision, the main source of uncertainty and ambiguity is often conceptualized as an uncertainty measure (e.g., a probability distribution or a possibility relation) over the states of nature. Most approaches addressing such structures either attempt to reduce the uncertainty or make decisions in such a way that some estimated outcome will be optimized (e.g., minmax regret). Note that it is never the question of changing the world to provoke the creation of new states of nature; rather, it is considered that the world can be changed using the actions of an agent (thus moving forward to some alternative state of nature), but no new worlds are created. Design models should extend beyond this framework because the passage from one system (states and transitions) to the next and, ideally, toward a more fruitful system, is of interest. A related question that we shall attempt to investigate later is "how does an artificial program provoke new states of nature?"

Bouyssou et al. [2013] argue that evaluation tools "are a consequence of the decision aiding process" and should not be selected before the problem has been formulated or the evaluation model constructed. Based on the discussion thus far, we argue that evaluation models (and tools) are a consequence of a design process, especially in the case where there are no feasible alternatives at the beginning of the process and not the other way around.

A simple model that explains the evaluation process is as follows:

$$X, f \rightarrow Y, \tag{1}$$

where X is the set of alternatives and Y = f(X) is the evaluation, and the task is to determine D, E, H, U and R to build f and apply it on X. No new objects are created; at best, information about already feasible alternatives is discovered and compiled using formal processes to obtain additional information, called evaluation. Evaluation models operate within a world-as-it-is paradigm without changing the world through the creation of new types of objects.

2.2 Learning algorithms as eliciting consequences of known objects

Machine-learning approaches have also been designed to operate on known objects. Two major paradigms are inductive and deductive learning. In deductive learning approaches, a database of knowledge is used to produce new knowledge under a given closure operator. For example, in propositional logic, modus ponens allows the consequences of known facts to be discovered. Given a theory $T = (P, P \rightarrow Q)$ and an operator of deduction $\vdash$, we can derive $T \vdash Q$. Levesque calls T explicit knowledge, whereas Q, the logical consequence of T, is implicit knowledge [Levesque 1984]. In such a learning mechanism, any notion of novelty would be deceptive. When Q is revealed through deduction, some new facts are indeed learned, but no new objects have been created. At the very best, in more general cases such as first-order logic deduction, new knowledge about already-existing objects will be produced.

In inductive learning approaches, the aim is to extrapolate relationships from a set of observed objects, called a training set, such that accurate predictions about future examples can be made. More formally:

- $(x_i)_{i=1..n}$ are observations (objects), where for all i, $x_i \in X \subseteq R^p$;
- $X = (x_{i,j})_{i=1..n, j=1..p}$, the matrix of observations; and
- $Y = (y_i)_{i=1..n}$, the observed output.

In a classification problem, the output corresponds to C classes, where each individual object must be assigned (e.g., good, medium or poor grades for students). In a linear regression problem, the output will be a continuous variable, $Y \in R^n$.

Independent of the specific algorithm used or the number of the output classes, learning problems formulated in this fashion aim to build a function f representing the mapping that exists between the observed objects and the corresponding outputs as much as possible. More formally, assuming that $(x_i,y_i)_{i=1..n}$ are realizations of certain random variables $(X_i, Y_i)_{i=1..n}$ from an unknown distribution over existing objects, the aim of the learning algorithm is to approximate a function f such that $Y^{-} = f(X)$ that is close to Y given a distance metric l. The function f is called a predictor, and the metric l is called a loss function. Much of the machine-learning literature revolves around this notion of the loss function: algorithms are optimized for specific cases of learning problems to address problems related to loss functions and the accuracy of the predictions for future observations in terms of recall and precision. A basic model that explains this approximative (or predictive) stance is thus

$$X,Y \rightarrow f, \quad (2)$$

where X, Y and f are as defined above. With respect to design, we can see that what is created here is not new objects nor new classes of objects (the outputs) but, rather, a function that is hoped to be a good representation of the mapping between objects and their corresponding classes. What is sought is not the creation of new classes of objects, for which corresponding objects are built, but a fit between the available and already-existing objects and their known categories. Learning models are not designed to create new objects or use existing available data to hypothesize about the possibility of objects that may not even be contained in the available data. These models do not consider what cannot be observed in the current data but, rather, what could be interesting to know. These models thus neglect the fact that sometimes, it is simply more important to decide what to look for than to find what is already there. As such, although different from the evaluation models, learning algorithms also operate assuming implicitly that the world exists as it is, and by applying a series of operations, we can obtain new knowledge from that world by generalizing existing objects or relationships.

3. Design reasoning as "the world as it can be"

3.1 Design as the evolution of object definitions

In contrast to decision and learning paradigms, design is the creation of some new object. Design theories and models attempt to capture in various ways these basic and fundamental observations. Most theories of design can be described as the evolution of some description D of the designed artifact (i.e., $D_1 \rightarrow D_2 \rightarrow \ldots \rightarrow D_m$). For instance, in Schön and Wiggins's description [Schön and Wiggins 1992], a transition from one description to another occurs with what is called a "design move." In topological spaces [Braha and Reich 2003], the descriptions are 2-uples of the form $<F_i, D_j>$, where transitions may either change F_i (e.g., functional descriptions) or D_j (e.g., structural descriptions) as a result of operators called "closure" (e.g., deductive closure). Most such design models, if not all, do not attempt to describe the creative mechanism driving such transitions. An exception is C-K design theory [Hatchuel and Weil 2003]. Compared with most other formal design process models, C-K theory describes the evolution of design definitions by placing knowledge at the heart of the transitions. In other words, knowledge is a necessary resource for the generation of new descriptions: object descriptions are created and evolve using knowledge, taken in the form of logical propositions with a truth-value.

3.2 Design as imaginative constructivism

Going further, Kazakci [2013] raises new perspectives concerning the elaboration of definitions in design. He argues that constructivism is a foundational issue in design research, and he studies forms of constructivism in the elaboration of object definitions in design. Several such forms exist in the design literature. First, we can find a social constructivist approach, for instance in Bucciarelli [1988], where object definitions are constructed collectively over time. Second, we can identify an interactive constructivism, for instance in the work of Schön and Wiggins [1992], where a designer interacts with some medium to progressively construct a design. In addition to these traditional forms, Kazakci contends that there is a third form, called imaginative constructivism. Imaginative constructivism argues for a worldview in which, in innovative design, new *types* of objects are imagined while methods for building or implementing these objects are sought. This is a dual constructivist process, where the construction of the object definition interacts with the construction of the method that would allow for the building or implementation of that object. For both of these processes, i.e., in the construction of the type or the method, it is possible – and often required – to introduce novelty.

Foundations for the notion of imaginative constructivism arise from the study of a particular design domain, namely, mathematics. Kazakci [2013] studies Brouwer's intuitionism, one of the major constructivist approaches to mathematics that captures several fundamental properties of design reasoning. First, this approach explains mathematical activity as a reasoning process performed over time. Second, this approach places emphasis on the *constructability* of objects rather than the truth of their existence. Third, this approach acknowledges the incompleteness of knowledge and the possibility of constructing new objects. Fourth, the construction of unprecedented and unpredictable objects is considered using the notion of the creativity of the mathematician, i.e., her *free choices*. From a design theory standpoint, this is an important feature of intuitionist mathematics. Allowing an act of free choice at any moment and the possibility of breaking away from any fully determined (lawlike) object allows for the consideration of *partially determined objects* with *novel properties.* This conception recognizes the creative nature of the mathematical activity. There is always the possibility to continue defining an object in a way that distinguishes it from all the others that are known thus far, thereby creating a novel object [van Dalen 2005].

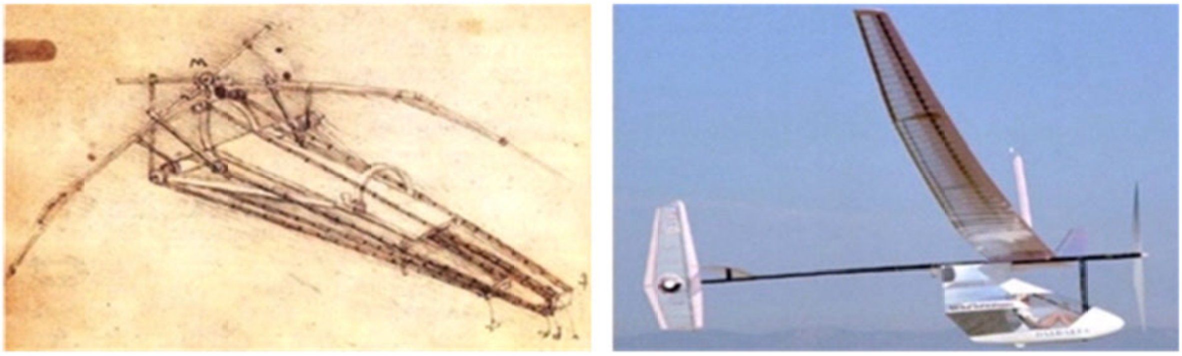

Figure 1. On the left, Leonardo da Vinci's human flying machine concept is shown. On the right, the Daedalus human flying machine is shown. Da Vinci *imagined what* a human-powered flying machine would look like centuries before a working prototype was *built* – the Daedalus – by a group of NASA researchers and MIT engineers [Eris 2006]

3.3 The clash of imagination vs. knowledge

Kazakci [2010] defines imagination as the ability of the mind to create thoughts that are not and have never been experienced (such as a horse with wings, a flying chair, a mobile phone preventing heart attacks). It is suggested that this definition is fundamental to understanding human creativity. Strikingly, this definition has been ignored, or at least not directly accounted for, in models of creativity. However, human beings are programmed to be sense-making machines creating meaning out of their experiences [Bartlett 1932], [Clancey 1999]. Throughout our education and everyday experiences, we are often required to reject, ignore or dismiss implausible or absurd ideas (*normally*, chairs do not fly, horses do not have wings and phones do not prevent heart attacks).

Based on these premises, we can see the difficulty in the creation and evaluation of fundamentally new ideas: our tendency to look for meaning and possibility may prevent us from pushing our imagination toward new ideas. New ideas may appear strange or meaningless by their very nature, preventing the designer from recognizing any value or investigating further. Nevertheless, it is the capacity of the designer that allows her to look past the domain of known objects with known properties to formulate interesting combinations of properties for not-yet-existing objects and to initiate genuinely innovative design processes through which new classes of objects can be bred.

3.4 The "what" and "how" of an object definition

Mathematics is a particular design domain where new objects with interesting properties are sought [Kazakci and Hatchuel 2009], [Kazakci 2013]. A fundamental issue that causes strong opposition among mathematicians directly concerns design theory, namely, the definability of objects. Constructivist mathematics defend the argument that an object can only be defined and be made to exist if it can be constructed explicitly using a method. This view on the existence of objects can be called *existence as constructability*. A method is a set of ordered operations that transforms an input into an output. In this sense, we can equivocally use a method, an algorithm or a proof. In contrast, non-constructivist mathematics accepts object definitions whose existence can be proved by logical means (i.e., without a method or an algorithm of construction, proven solely based on given axioms using logical deduction). This view on the existence of objects can be called *existence as truth*. This distinction describes the fundamental dichotomy between constructivism and non-constructivism: either the world exists and should be studied as such or the only way to guarantee the existence of the world is to construct the said world (nothing that we cannot construct exists).

In design, these two extreme positions are overly strong and may easily become false depending on the context. In the world, various objects already exist about which some particular actor might know or observe certain properties without knowing how to construct or reproduce an object with those properties. These objects are nevertheless useful, and we can make use of them, select them over other objects or learn about their properties without knowing how to construct them (be it individually or as a society). Thus, both modes of thinking are used: *what* the object will be and *how* the object will be built is constructed together and interdependently [Kazakci 2010].

Such dynamics are reminiscent of the distinction between the formulation of a theorem and its demonstration by a proof. It is known in mathematics that in some cases, such as Fermat's last theorem, several centuries have passed between the two processes. In design literature, it is possible to find cases that highlight similar dynamics. For instance, Eris [2006] describes an example of the human flying machine conceived of by Leonardo da Vinci that inspired, centuries later, the Daedalus built by NASA engineers (Figure 1). Examples such as this one are indicative that there may be various processes of construction in design processes (e.g., the construction of a definition vs. the construction of an actual object). Recent experimental data [Eris 2004], [Edelman 2012] also supports the idea that designers think and act differently when thinking about *what the object should be* or *how the object can be built* [Kazakci 2010]. The imaginative constructivist dynamics thus enable the revelation of a dual constructivism in design processes. This issue has been under-investigated in the design literature, often collapsing both notions into a single concept. Although theories of co-evolution (e.g., problems-solutions, concepts-knowledge, functions-structures) exist in the design literature [Maimon and Braha 1996], [Braha and Reich 2003], [Hatchuel and Weil 2009], either they do not explain the constructive aspects or they do not consider the free choices of the designer.

4. Collapse to mono-space search: the implicit elimination of a duality in AI

Given the world-as-it-can-be paradigm described in the previous section, we shall analyze in this section some particular techniques from the AI literature to better understand their relationship with design. A fundamental notion in traditional AI is *search*. When the cardinality of the set of solutions to be considered is large or even potentially infinite, search procedures are used to find solutions with desirable properties. Although the notion of search is omnipresent in AI and has been generally accepted as being inevitable, what the search *creates* has rarely been discussed. The type of objects manipulated during search and why has been made abstract, and the focus instead remains on optimizing the search performance (i.e., search time). We shall discuss two common examples of search to discuss and distinguish two contrasting purposes that usually go unnoticed. Let us remember that search-related models (e.g., problem-solving) have long been considered adequate models for design (see e.g., Cross [2001] for examples and Dorst [2006] for a discussion).

4.1 Linear programming search: search without construction

An archetype of search problems is the search for optimal solutions to mathematical linear programming problems. For maximization, a general form for a linear program is the following:

$$\text{Max } \mathbf{z.x}, \text{ s.t. } \mathbf{A.x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \tag{3}$$

where $x = (x_1,...,x_n)$ are the real-valued variables and $z = (z_1,...,z_n)$ are unit profits that define the (economic) objective of the search. The constants **A**, **b** and **0** circumscribe an acceptable domain of variation for the variables. The aim of the search algorithm is to find, among all feasible objects respecting the constraints, a particular object that optimizes the economic function (the objective).

In this formal problem, we do not have an explicit list of the individual objects. Instead, we have knowledge about certain properties that all the objects satisfy (the constraints that give what is called a feasible region) and a means for measuring their quality (the objective). Let us remark that given the form of the feasible region, a solution always exists, and all the represented objects are assumed to be feasible. Several search algorithms that guarantee an optimal solution are known (e.g., the simplex method or interior point algorithms).

We do not know in advance what the optimal solution will be; however, we know for certain what type of solution it will be. The solution is already characterized by the constraints and even more so by the dimensions $D = (d_1, \ldots d_n)$ on which all of the objects considered in the problem formulation have been defined before the algorithm is applied. The "what" of the objects has been determined, and the aim of the algorithm is not to explore alternative definition types. In other words, the dimensions and constraints do not change during the search. Compared with a design process as discussed in the previous section, neither a new definition nor an object fitting that definition is constructed.

Let us also note that in traditional sensitivity analysis or robustness analysis, the dimensions D do not change either. Hence, any variability in the search parameters does not provoke a new type, only a different optimal solution. Aside from the stability of object definitions, another major point of interest for us is that the optimal solution is guaranteed to be from among the feasible solutions. The purpose of the search is not to seek solutions that may be unfeasible for the moment but interesting nonetheless. At least, in this context, the search is not aimed at finding imaginary (or fictive) solutions that are yet to be made feasible.

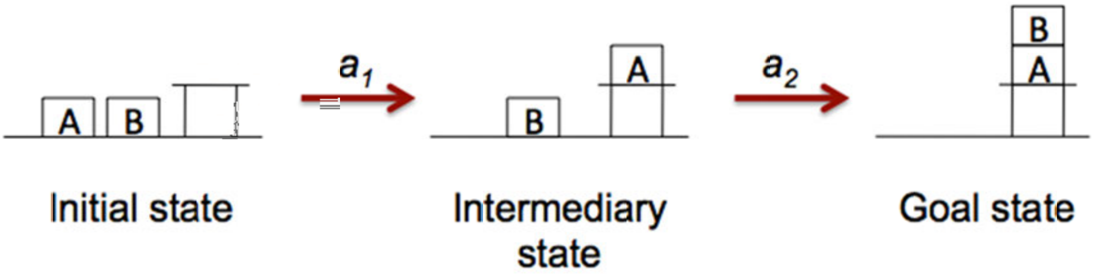

Figure 2. A simple blockworld diagram. The initial state is transformed to the goal state by applying actions a_1 and then a_2

4.2 Combinatorial search: search with construction…of?

Search formalisms have also been extensively applied in planning tasks [Fikes and Nisson 1971], [Bonet and Geffner 2001]. To obtain a better understanding of what type of objects are elaborated during AI planning, we shall consider one of the most conventional examples used in the development of planning programs, namely, the blockworlds (see Figure 2) [Nagata et al. 1973], [Russell and Norvig 1995]. Blockworld is an abstract, closed world for experimenting with AI systems. This conceptualization was particularly useful for building early systems for planning and robotic navigation. This world consists of a flat surface, a set of blocks and a robot (or a robotic arm) that is able to move the blocks around by applying certain actions (e.g., pick up () or stack ()). States are descriptions of the world using predicates (e.g., on (A, B) or on Table (A)).

Starting with an initial configuration (or state), the program searches a combination of actions to reach a final, desired configuration, called the *goal state*. Often, there is no trivial solution for the combinatorial search, and the program might get stuck (e.g., in the above example, if B is put on top of the table before A), in which case backtracking is necessary. The search starts over from a previously explored configuration that is evaluated to be likely to lead to a solution. The program stops if the available computational resources are exceeded or a sequence of actions leading to the goal state has been found.

This model has been used extensively as a valid model for studying human reasoning skills in cognitive psychology and design research [Cross 2001]. In a design context, the goal state has been interpreted as the design to be reached at, i.e., a set of requirements. A designer must take certain actions to manipulate objects (or their representations, e.g., sketches) around her to reach the goal state. This view of design raises a set of legitimate questions (see, e.g., Dorst [2006]). In our context, the main question is what type of object definitions are being created and manipulated during reasoning.

Figure 3. The transformation of an input material by a CNC machine via a set of planned actions

Two things are created by this process. First, a plan of actions – with a clearly defined input and output set – is created. In terms of the discussion of the previous section, this plan corresponds to a *method* of construction (or to a proof [Kautz and Selman 1998]). The search creates a set of instructions (as an algorithm) that *transforms* an input to an output. In design- and manufacturing-related fields, a standard application of this type of search is the use of early CNC machines; see Figure 3. Second, the output configuration that corresponds to the goal state will be created by the execution of the plan. In the example shown in Figure 3, this output is a sculpture of a female body. In the most general case, this output would be a re-arrangement of the initial input objects.

Where is design? The answers to this question are ambiguous in the literature precisely because the interpretative model used to define design is the same as the model that is being interpreted! Our overview of the model of combinatorial search based on our imaginative constructivism framework provides insight into what is really being designed (and to what extent). The entity that appears and that allows for the creation of the same artifact as many times as it is applied is a method of production. The "sculpture" in this example is not the design; it is the first of many output objects that can now be produced. The design is the method. Thus, when combinatorial search is applied to a set of actions, what is being designed is a constructive proof of existence, producing a first example of the new type of object (assuming the system has not previously produced a plan for the same goal state).

Hence, the interesting question for us is where does the *type* (definition) come from? Is it new? The answer to this question is now straightforward: the design of the "type" of new object to be constructed has been completed before the combinatorial search process has started; it was the goal state given to the program as a parameter. Whether the design was new cannot be determined based solely on the input-output pair of the search process.

We must be able to define a reference library of types that is specific to each designer and that contains the old objects and their types to be able to determine whether a particular type of sculpture is new. Therefore, there is no creative design of types nor creative design of a method (only known actions are combined) but only a novel sequencing of actions to create a new method for constructing an object with a stable definition (of what that object is).

Unless the search process can change the type definition of objects on the run, there is no possibility of imagining new types of objects (in this setting, new goal states). In the previous example of body sculptures, new goal states are simply new representations of the human body (Figure 4). In traditional combinatorial search, this aspect does not exist because the combination is only applied to individual *actions* conducted to build methods. As we observed in Section 3, the imaginative constructivist nature of design presupposes that the creative design act requires the combinative creation of both type descriptions and methods. In AI programs, one or the other of these two dimensions is not treated. *Search is applied only on one dimension of the object definition, thus collapsing the creative dynamics* we described previously to a mono-space search. This result is unfortunate insofar as the real richness of a design reasoning occurs at the interaction of these two types of definition construction processes [Kazakci 2013].

Another shortcoming of this type of search process, in terms of its adequacy in incorporating conceptive intelligence, is its inability to create goals that might even be unachievable by the current set of actions. This shortcoming assumes not only the ability to formulate goal states but also goal states that cannot be reached using the current set of actions. In this case, new actions must be learned. Systems that are able to learn new types of actions, thus enabling the realization of new types of tasks, exist in AI and robotics literature (see, e.g., Konidaris et al. [2010]). However, a dual constructive search mechanism, allowing the system to build new and unprecedented tasks while acquiring new types of actions using interaction with the environment, does not exist. This mechanism, however, is the *sine aqua non* of conceptive intelligence for autonomous creativity.

Figure 4. Different definitions of "human body" – the creation of such "types" is not considered by a standard combinatorial search over action sequences

5. Brouwer machine: a conceptual model for systems with conceptive intelligence

According to the imaginative constructivism framework, design implies a dual constructivism on the definition of types of objects and the methods by which they are produced. This type of process involves the articulation of top-down and bottom-up processes. Both the construction of definitions and the construction of methods may be changed greatly during the activity as a result of the free choices of the designer. Based on these premises, the first feature of the model we introduce is a language with two components.

5.1 Dimensions of definition

The overall idea behind our notion of a Brouwer machine is that there are two fundamental dimensions through which a class of objects might be specified:

- The type definition – part of the definition of an object stating *what* the object is through the specification of its properties;
- The method definition – part of the definition of an object stating *how* the object can be built through the specification of a sequence of actions.

Let us consider a set of properties $P^T = (p_1, p_2, \ldots, p_n)$ accessible to the system at time T. Let $L_P{}^T = (c_1, c_2, \ldots, c_N)$ be the library of conceptual descriptions at time T. Each c_i is a non-empty subset from P^T; i.e., $\forall i, c_i \in [(\Pi P^T)\backslash \emptyset]$, where ΠX represents partitions of set X.
Let us consider a set of actions $A^T = (a_1, a_2, \ldots, a_m)$ available to the system (i.e., through its effectors) at time T. Let $L_A{}^T = (\pi_1, \pi_2, \ldots, \pi_M)$ be the library of procedural descriptions at time T. Each π_k is a non-empty subset from A_T; i.e., $\forall k, \pi_k \in [(\Pi AT)\backslash \emptyset]$. Note that L_P allows the definition of goal states, whereas L_A allows the definition of plans of action to reach these goal states.

5.2 Free choices by novelty-driven search on definitions

The aim of the system is to mimic a type of design reasoning by continually alternating between the following:

- the generation of new type descriptions c_{new} such that $c_{new} \in [(\Pi P^T)\backslash \emptyset]$ and c_{new} □ $L_P{}^T$ and

- the generation of new plans of action π_{new} such that $\pi_{new} \in [(\Pi A^T) \backslash \emptyset]$ and $\pi_{new} \square L_A^T$ and allowing the construction of c_{new}.

There are several points to be considered. First, the system requires a mechanism with which to find new c_{new} and π_{new}. One way of generating these entities is through novelty-driven search (NDS), which might, for instance, be a genetic algorithm pushing for novelty rather than fitness with respect to predetermined criteria [Lehman and Stanley 2008]. In each iteration, privileged solutions are only those that are newest (i.e., farthest apart) from the existing set of solutions. Taking advantage of the crossover and mutation operators, genetic algorithms are indeed able to build a myriad of combinations of existing entities, some of which will not exist in the current libraries. Other solutions are conceivable but will not be discussed in the current work.

Second, for any c_{new} generated, it is likely that the system will not have an existing π_k able to build c_{new}. Thus, the imagination of a new enunciation will trigger the necessity to imagine new plans of action. Let us also note that there is an injective mapping from P^T and A^T in the sense that for each c_i, there may be multiple plans π_k building it. The inverse is not true.

Third, the system will not be able to generate new plans and concepts indefinitely. The combinatorial search between L_P and L_A will eventually exhaust all possible new combinations, at which point, no further imagination is possible for the system (with respect to the definition given in Section 3). To prevent this exhaustion and to progress toward truly autonomous agents, the system requires another operator that allows it to communicate with its environment and to be able to add new elements to P or to A to enrich its design languages.

5.3 Example and discussion: mazes and escape artists

An example Brouwer machine might be set up in a hypothetical domain consisting of the design of mazes. Mazes are typical examples for traditional search programs in AI. In our case, rather than having a solver for a particular maze, a Brouwer machine would be an escape artist, not only solving mazes but aspiring to create new ones in a continuous manner such that every new maze would be more interesting to solve in some sense (i.e., more challenging) with respect to the currently known mazes. In such an application, the system would generate mazes, always targeting newer mazes that are different. For each maze, the system would search for a plan of escape (e.g., evolving plans of action until a plan that solves the maze is found). At first, this approach might appear to be a standard co-evolutionary process. However, there are several important differences. First, we are interested in finding a repertoire of mazes and escape plans rather than individuals overcoming each other by mutual co-evolution. It is not the maze versus the escape artist that is being evolved; rather, it is a designing entity, the Brouwer machine, that explores the creation of new mazes and plans to solve them rapidly. In contrast to typical co-evolution logic, a newly generated maze will not be a maze that the plan from the previous iteration would not be able to solve (hence, not necessarily a better maze, according to some criteria). Instead, it would simply be a newer maze with respect to the previously explored mazes. Each plan will solve only one maze; it will be a method for that instance only.

Even with this simple example, some limits for implementation are already apparent. First, the maze domain is too limited in the vocabulary it allows for L_P and L_A. For instance, whatever new maze the system imagines, it suffices to use the same set of navigation actions to solve it. In this example domain, there will be no opportunity to eventually extend L_P and L_A. In contrast, if we consider a Brouwer machine set up on a digital image or physical prototyping domain, the scope is considerably broadened because the possibilities are endless.

Second, the iterative alternation between the maze definitions and solver definitions are rather simplistic. More interesting would be a mechanism by which interaction with previous mazes and their solvers can be achieved to foster the dual construction process. There are at least two types of interactions we can consider:

- How can old solutions help a particular maze?
- How can old solutions and mazes help produce new mazes (e.g., mazes that are different and that exhibit particularly interesting behavior, such as an increased level of difficulty without any explicit objectives)?

In realistic design situations, these interactions immediately become complex. These interactions are not considered in the current framework; however, they constitute a necessary step to progress toward conceptive intelligence.

5.4 The central problem of conceptive intelligence

The previous remarks allow us to finally state and discuss what we deem to be the central problem of conceptive intelligence. In what we called an imaginative constructivist process, a dual constructivism with free choices can occur for both the properties and methods defining objects. The central ability of a Brouwer machine, or any designer thereof, is the ability to select which novelty will be pursued and elaborated. As noted above, in a realistic design setting, an expert designer would generate more than one novel object definition that might be explored next. The true mark of an expert is to better judge which of the currently considered novelties is worth exploring given the available resources. In other words, it would be a decision mechanism not for selecting the best among existing objects but, rather, the most interesting to explore among a set of novel definitions. Work initiated by Hendriks and Kazakci [2011] offers some perspectives on the logic of this issue. As signaled in Kazakci [2013], this is a decision theory specific to design processes and that is yet to be formulated.

6. Discussion: are genetic algorithms creative?

6.1 Genetic algorithms as a means of scientific discovery

Genetic algorithms offer a powerful approach for combinatorial search. Based on the metaphor of natural evolution, a genetic algorithm maintains a population of candidate solutions for the problem at hand and causes this population to evolve by iteratively applying a set of stochastic operators. At each iteration, a subset of the population survives and is given the opportunity to produce offspring. The survival of candidate objects depends on evaluation criteria, called fitness. Genetic algorithms have often been associated with creative processes [Koza 1999], [Koza et al. 1999], [Renner and Ekárt 2003] because, given a problem formulation, the solution space can be explored conveniently by evolving candidate solutions in various directions.

A recent application of genetic algorithms that offers promising perspectives concerns the discovery of scientific laws from experimental data. Schmidt and Lipson [2009] have been able to *distill* free-form natural laws from motion-tracking data captured from various dynamic physical systems ranging from a double pendulum to harmonic oscillators using genetic algorithms. Starting with symbolic expressions (e.g., +, /, sin(), ω, θ, etc.), the algorithm was able to generate increasingly complex sentences by combination. The representation of a symbolic equation in computer memory is a list of successive mathematical operations. The construction of such symbolic expressions that fit a given dataset is a traditional application in data-mining called symbolic regression.

The particular insight Schmidt and Lipson propose is a principle for identifying non-triviality. This insight is based on the observation that to claim a fit between the generated symbolic expression and the data observed, the partial derivatives of both the symbolic expression and the numerical data should vary in the same manner. Using this principle as the fitness measure and the genetic algorithm over symbolic expressions, these authors were able to generate complex invariants such as Hamiltonians, Langrangians and equations of motion for systems of various complexity. The type of law discovered depends on the type of variables provided to the system on a given run. Symbolic expressions obtained from simpler systems have been observed to be effective in bootstrapping the search for more complex systems.

Without prior knowledge about physics, kinematics or geometry, Schmidt and Lipson's [2009] system detected complex relationships such as nonlinear energy conservation laws, Newtonian force laws, geometric invariants and system manifolds in various synthetic and physically implemented systems. In addition, it is claimed that many applications exist for this approach in fields ranging from systems biology to cosmology, where theoretical gaps exist despite an abundance of data, and that scientists may use such approaches to focus on interesting phenomena more rapidly and to interpret their meanings.

Figure 5. On the left, the picture of Mona Lisa as a "type" representation; on the right, "an instance" approximating the type using a combination of polygons – pictures produced using Watchmaker framework

6.1.1 The intelligence of genetic algorithms – conceptive or not?

Schmidt and Lipson's system is a powerful discovery engine. Because discovery is a notion that we can relate to intelligence, the question that is interesting for us is, is it intelligent? Is it creative, considering the fact that the algorithm discovers complicated laws of nature without any notions of physics or domain knowledge? There is indeed a tendency to assume that genetic algorithms are, in a sense, intelligent. For instance, Reynès [2007] considers that genetic algorithms incorporate intelligence at the level of the selection of survivors and that this is the only intelligent step. In other words, it is claimed that the intelligence of genetic algorithms is in the selection mechanism. Jacques Monod, Nobel Laureate in Biology, defends this very view: "Many distinguished minds appear not to be able to accept, nor to comprehend that, from a source of noise, the *selection* was able to, by itself alone, pull all the music in the biosphere."

The selection step in a genetic algorithm is based on a metric representing the membership (or the distance from) a type definition (Figure 5). As we have stressed previously, creating a type and building an instance of that type are two separate matters. As can be observed in the Mona Lisa example in Figure 5, the only thing selections do is evaluate how far away an instance is from the type defining its category. The genetic algorithm in this context simply *constructs* an instance of a type given the type. If any intelligent act occurred in this context, it is not during the selection but, rather, during the type creation – by Da Vinci himself, in this example. In contrast to what Monod claims, it is not the selection that pulls the music from a continuum of noise; rather, it is a design effort that determined, before the process of construction had even started, *what is a nice noise*.

6.1.2 Interpreting Schmidt and Lipson's system through Brouwer machines

Now that we have qualified genetic algorithms as a construction machine (similarly to the combinatorial search process discussed in Section 4.2), we can discuss Schmidt and Lipson's system with respect to the notion of a Brouwer machine. A point that agrees immediately between these concepts is the symbolic expressions of equations, which can be easily mistaken as types. However, when we consider that equations are stored in memory as a list of successive operations, it is easier to see that this list of operations can be seen as plans of action, which, when applied, reproduce an instance similar to what has been observed. A_T are all the elementary operations and constants (e.g., ω, θ, sin(), +, etc.). L_A^T is then all the sentences that have been formulated by a combination at iteration T (e.g., $\sin^2(x_1) + \cos^2(x_1)$ or $x_1 + 4.56 - x_2x_1/x_2$). Note that the instances (streams of motion data captured by a camera) are not perfectly generalized by the system. Thus, the application of a plan discovered along the way will not produce the signal that was observed. Note also that the methods formulated by the system are not created *per se*; rather, they are generalizations over observations using a convenient construction machine, which also implies that the system will not exhibit any free selection on the methods created because it does not target any act of creativity or even intelligence, it only aims to accurately reproduce what has been observed.

Concerning the set P_T, we can see that there are no properties, or any combination thereof, that are considered by the system. Consequently, there is neither type creation nor any attempt to create new types by free choices. The finding would have been different if the system had the ability to think of new types of natural laws and then take action and provoke a change in the world to test its set of methods. This result takes us back to the central question of conceptive intelligence: how would the system know what novel type to create and to attempt to build? We see that Schmidt and Lipson's system, despite its power and accuracy for the task domain with respect to which it was built, is not a system with which we can hope to create an autonomous explorer that would conceive any new scientific concept. In its current form, rather than distilling laws, Schmidt and Lipson's system distills only regularities within the observed data, which would be recognized as theoretical concepts in physics by an expert on the matter who was already familiar with that concept. Without such knowledge and the ability to formulate preferences on unknown types obtained by free choices, it is not possible to devise systems with conceptive intelligence.

7. Conclusions

The current paper offers a perspective on what we term conceptive intelligence – the capacity of an agent to continuously think of new object definitions (tasks, problems, physical systems, etc.) and to look for methods to realize them. We call this framework a Brouwer machine, and it is inspired by research in design theory and modeling, with its roots in the constructivist mathematics of intuitionism. The dual constructivist perspective we describe offers the possibility to create novelty both in terms of the types of objects and the methods for constructing objects. More generally, the theoretical work on which Brouwer machines are based is called imaginative constructivism. Based on the framework and the theory, we discuss many paradigms and techniques omnipresent in AI research and their merits and shortcomings for modeling aspects of design, as described by imaginative constructivism. To demonstrate and explain the type of creative process expressed by the notion of a Brouwer machine, we compare this concept it with a system using genetic algorithms for scientific law discovery.

References

Bartlett, F. C., "Remembering: a study in experimental and social psychology", 1932.

Bedau, M. A., "Artificial life: organization, adaptation and complexity from the bottom up", TRENDS in Cognitive Sciences 7(11), 2003, pp. 505-512.

Bonet, B., Geffner, H., "Planning as heuristic search", Applied Intelligence, 129, 2001, pp. 5-33.

Bouyssou, D., Dubois, D., Prade, H., Pirlot, M., "Decision Making Process: Concepts and Methods", Wiley-ISTE, 2013.

Braha, D. Reich, Y., "Topological structures for modeling engineering design processes", Research in Engineering Design, 14, 2003, 185-199.

Brouwer, L. E. J., "Consciousness, Philosophy and Mathematics", Proceedings of the 10th International Congress of Philosophy, Conference Location, 1948, pp. 1235–1249.

Brouwer, L. E. J., "De Onbetrouwbaarheid der logische principes", Tijdschrift voor Wijsbegeerte 2, 1908, pp. 152-158.

Brouwer, L. E. J., "Over de Grondslagen der Wiskunde [On the foundations of Mathematics]", 1907, 183p.

Bucciarelli, L., "An ethnographic perspective on engineering design", Design Studies 9(3), 1988, pp. 159-168.
Clancey, W. J., "Conceptual coordination: how the mind orders experience in time", L. Erlbaum Associates, 1999
Cross, N., "Design Cognition: Results From Protocol And Other Empirical Studies Of Design Activity", Design Knowing and Learning: Cognition in Design Education,. W. C. Newstetter, C. Eastman and W. M. McCracken, Elsevier Science, 2001, pp. 79-124.
Dorst, K., "Design problems and design paradoxes" Design Issues 22(3), 2006, pp. 4-19.
Edelman, J. A., "Understanding radikal breaks through media", Phd Thesis Stanford University, 2012
Eris, O., "Effective Inquiry for Innovative Engineering Design", Boston, Kluwer, 2004
Eris, O., "Insisting on Truth at the Expense of Conceptualization: Can Engineering Portfolios Help?", International Journal of Engineering Education 22(3), 2006, pp. 551-559.
Fikes, R. E., Nisson, N., "STRIPS: A New Approach to the Application of. Theorem Proving to Problem Solving'.", Artificial Intelligence 2, 1971, pp. 189-208.
Goth, G., "software on mars.", Communications of the ACM 55(11), 2012, pp. 13-15.
Hatchuel, A., Weil, B., "C-K design theory: an advanced formulation", Research in Engineering Design 19(4), 2009, pp. 181-192.
Hendriks, L., Kazakci, A., "A method for design reasoning using logic: from semantic tableaux to design tableaux", ICED, Proceedings of the 18th International Conference on Engineering Design (ICED11), , Conference, S. J. H., Culley, B. J., McAloone, T. C., Howard, T. J., Reich, Y., Location, Vol. 2, 2011, pp. 275-286.
Heyting, A., Ed. "L.E.J. Brouwer. Collected Works 1", Philosophy and Foundations of Mathematics. Amsterdam, Heyting, A. (Ed). North Holland, 1975.
Kautz, H., Selman, B., "BLACKBOX: A new approach to the application of theorem proving to problem solving. AIPS98 Workshop on Planning as Combinatorial Search", Conference Location, 1998, pp. 58-60.
Kazakci, A., "On the imaginative constructivist nature of design: a theoretical approach", Research in Engineering Design, April 2013, Volume 24, Issue 2, 2013, pp. 127-145.
Konidaris, G., Kuindersma, S., Grupen, R., Barreto, A. S., "Constructing skill trees for reinforcement learning agents from demonstration trajectories", Advances in neural information processing systems, Conference Location, 2010, pp. 1162-1170.
Koza, J. R., "Human-competitive machine intelligence using genetic algorithms", Festschrift in Honor of John H. Holland, 1999, pp. 15-22.
Koza, J. R., Bennett III, F. H., Stiffelman, O., "Genetic programming as a Darwinian invention machine", Springer, 1999.
Lehman, J., Stanley, K. O., "Exploiting Open-Endedness to Solve Problems Through the Search for Novelty", ALIFE, Conference Location, 2008, pp. 329-336.
Levesque, H. J., "A logic of implicit and explicit belief", AAAI, Conference Location, 1984, pp. 198-202.
Maimon, O., Braha, D., "A Mathematical Theory of Design", International Journal of General Systems 27(4-5), 1996, pp. 275-318.
Nagata, T., Yamazaki, M., Tsukamoto, M., "Robot planning system based on problem solvers", Proceedings of the 3rd international joint conference on Artificial intelligence, Conference Location, Morgan Kaufmann Publishers Inc., 1973, pp. 388-395.
Niekus, J., "Brouwer's Incomplete Objects", History and Philosophy of Logic 31, 2010, pp. 31–46.
Renner, G., Ekárt, A., "Genetic algorithms in computer aided design", Computer-aided Design 35(8), 2003, pp. 709-726.
Reynès, C., "Etude des Algorithmes génétiques et application aux données de protéomique", Université Montpellier I., 2007.
Russell, S., Norvig, P., "Artificial Intelligence: A modern approach", 1995.
Schmidt, M., Lipson, H., "Distilling free-form natural laws from experimental data", Science 324(5923), 2009, pp. 81-85.
Schön, D. A., Wiggins, G., "Type of seeing and their functions in designing", Design Studies 13(2), 1992, pp. 135-156.
van Dalen, D., Ed., "Brouwer's Cambridge Lectures on Intuitionism", Cambridge, Cambridge University Press, 1981.

Dr. Akın Osman Kazakçı, Assistant Professor
Mines Paristech, CGS.
60, Bd. Saint-Michel, 7527, Paris Cedex 06
Telephone: +33 1 40 51 93 12
Telefax: +33 1 40 51 90 65
Email: akin.kazakci@mines-paristech.fr
URL: http://www.cgs-mines-paristech.fr/equipe/kazakci/