Supporting Creativity in Symbolic Computation

Boi Faltings
Artificial Intelligence Laboratory (LIA)
Swiss Federal Institute of Technology (EPFL)
IN-Ecublens, 1015 Lausanne, Switzerland
FAX: +41-21-693-5225, e-mail: faltings@lia.di.epfl.ch

November 15, 1995

Abstract

For many people, creativity and computers are contradictory concepts: how can novel concepts arise in a machine which is designed to only do what it is programmed to do? In this paper, I give a computational model of at least one form of creativity and describe how the processes underlying it can be implemented in symbolic computation. Finally, I discuss the lessons that can be learned from the model for building CAD systems that support a designer's creativity.

1 Introduction

The word "creative" implies not only the creation of objects, but also of new categories or laws to describe their behavior. This is seemingly in inherent contradiction with symbolic computation, whose essential quality is that it predictably follows a set of preprogrammed instructions. Many designers therefore believe that using CAD systems prohibits creative design. This criticism applies especially to knowledge-based CAD systems, which provide preprogrammed functionalities to support the designer.

It follows directly from Gödel's theorem that no matter how flexible the modelling language of a CAD system is, one can always find examples which cannot be represented and manipulated. However, if we apply the idea that human thought is based on some form of representation, the same limitations also apply to human creativity. Creativity must then be limited to what is attainable within a single representation language, and could in principle be adequately supported by computers.

In this paper, I present a model in which creativity arises from the different generative power of the designers analysis and synthesis knowledge. The model explains how creative solutions can be obtained entirely within the knowledge-based system itself, without reference to outside influences. Even if the model would only explain certain forms of creativity, the fact that these kinds of creativity can in principle be supported by computer tools makes them interesting. The point of view I take in this paper is thus very different form that taken by philosophers like Margaret Boden ([BO91]) who conclude that creativity is in principle beyond the power of computers. In contrast, I propose that a precise definition of this notion will show that it can indeed be supported by computer.

For practical applications, it is unlikely that computers can be provided with knowledge bases large enough to actually automate the creative process. In the immediate future, it is more promising to use the model to decide what functionalities a CAD system must provide in order to support the designer's creativity. I will conclude this paper with a preliminary discussion of this issue.

2 An example of a knowledge-based design system

Design knowledge can be classified into synthesis and analysis knowledge. *Synthesis* knowledge is usually defined by prototypes ([GE90]) or some equivalent structure and maps:

Function $\Rightarrow$ Artifact

Design knowledge is applied in an obvious way: starting with a specification of desired functions, *deduce* artifacts that achieve them. On the other hand, *analysis* knowledge maps in the opposite direction, namely:

Artifact $\Rightarrow$ Behavior $\Rightarrow$ Function

Note that if

1. the space of possible artifacts can be discretized, and
2. analysis knowledge can be formulated in a finite set of rules on this discretization,

then analysis knowledge can be used *abductively* to map functions to artifacts. We call analysis knowledge which can be specified in this way *domain knowledge.* Other forms of analysis knowledge, such as numerical analysis procedures, only make properties *emerge* and cannot be used abductively. Since such knowledge usually consists of a set of principles which make the functions emerge, I denote it by the term *principled knowledge.*

As an example, consider a knowledge-based system which designs stable arrangements of two-dimensional polygons influenced by gravity, as illustrated by Figure 1.

An example of *design knowledge* in such a system is the general prototype shown in Figure 2. The prototype rule can be used to generate solutions in cases where it can be unified with

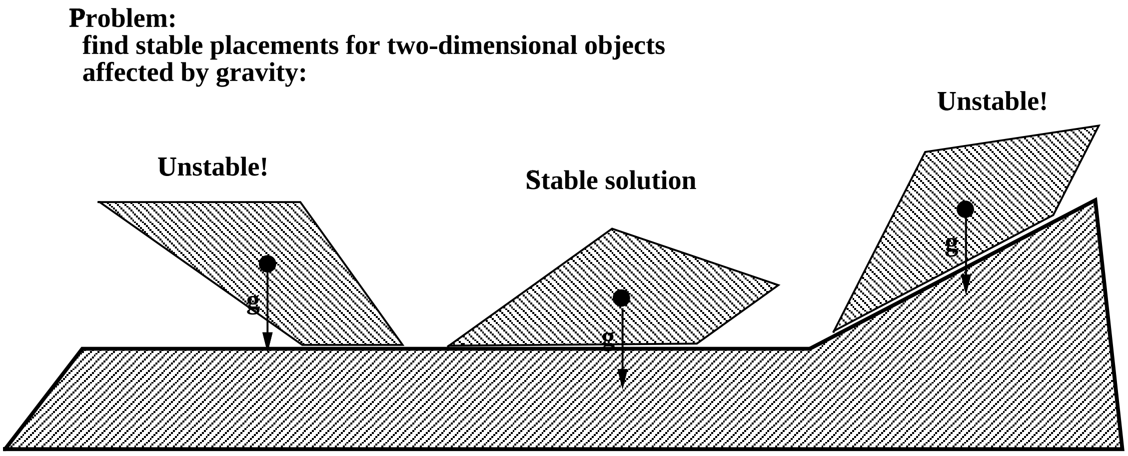


Figure 1: *The domain of the example system: find stable placements for a polygon in an environment.*

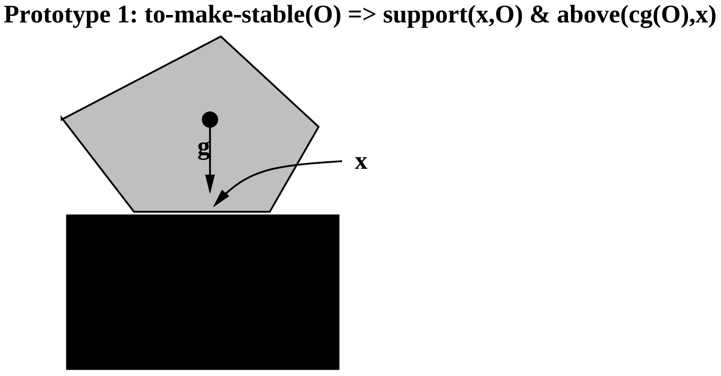
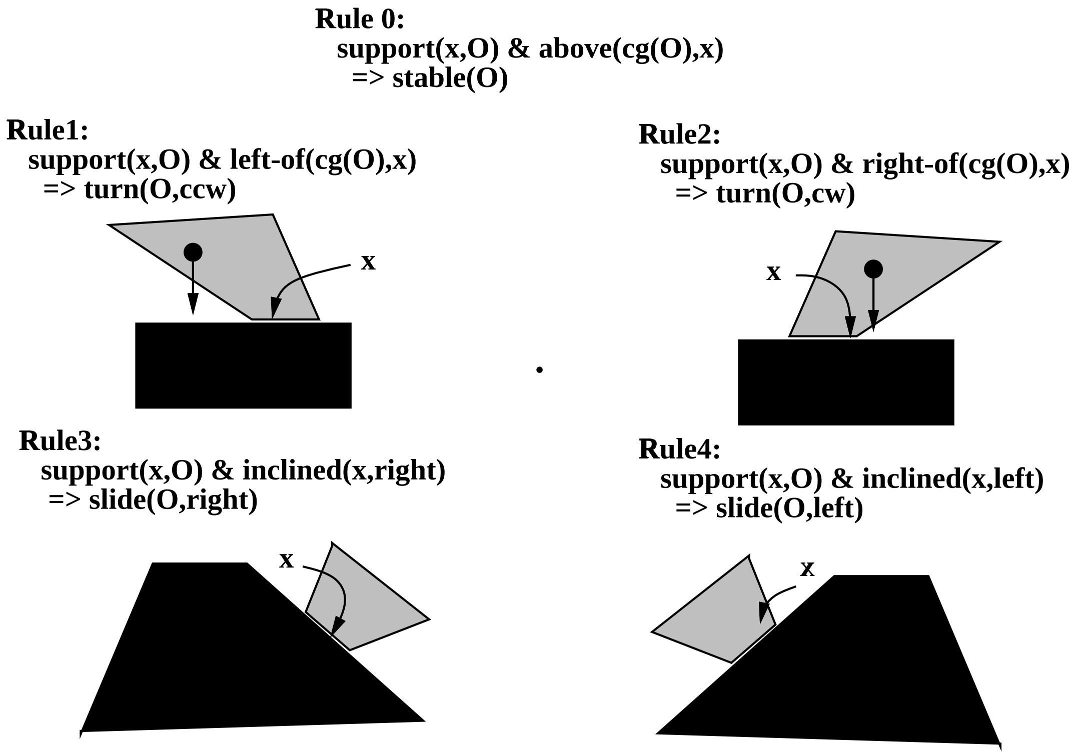


Figure 2: *An example of design knowledge: a polygon can be stably placed by supporting it on an edge x such that its center of gravity falls within x.*



Rule 0:
support(x,O) & above(cg(O),x)
=> stable(O)

Rule1:
support(x,O) & left-of(cg(O),x)
=> turn(O,ccw)

Rule2:
support(x,O) & right-of(cg(O),x)
=> turn(O,cw)

Rule3:
support(x,O) & inclined(x,right)
=> slide(O,right)

Rule4:
support(x,O) & inclined(x,left)
=> slide(O,left)

Figure 3: *Four domain rules which allow prediction of movements, and could also be used abductively if the goal was to find a placement in which an object would move.*

the elements of the problem description. I assume that initially, the systems design knowledge contains only this single prototype.

The *domain knowledge* of our example system is shown in Figure 3. Note that the design prototype in Figure 2 can be inverted to become an additional rule of domain knowledge. Note also that in practical systems, domain as well as design knowledge may be structured in many levels of abstraction and decomposition, involving more than direct rules mapping between function (or behavior) and structure.

Finally, an element of principled knowledge useful to the example system is the theory of stability shown in Figure 4, due to ([NIE88, SHO85]). Such a theory can be used to predict the stability and certain qualitative motions for any arrangement of polygons. However, it would be difficult to use abductively, as it makes stability the result of evaluating an unbounded number of regions[1]. In general, the characteristic feature of principled knowledge is that the analysis *emerges* through a computation which is difficult to invert. The prime example of principled knowledge in existing CAD systems are numerical tools such as finite element analysis.

A useful criterion for classifying knowledge in a design system is the set of artifacts which can be handled by it. For the three types of knowledge identified above, the division of the space is illustrated by Figure 5. The space of artifacts which can be synthesized using design knowledge is a subset of those which can be analyzed using domain knowledge, since by pattern matching (unification) design prototypes can always be mapped back to their functions. Likewise, the space of artifacts covered by domain knowledge is contained in that which can be analyzed by principled knowledge, since all domain knowledge can be used as principled knowledge also.

3 A working definition of creativity

Because the word "creativity" has many meanings, I shall now give a working definition of the aspect of creativity which I address in this paper. The fundamental characteristic of creativity is that something novel is generated which was not known before. In terms of the knowledge structure outlined above, this can be made more precise:

> Creativity is the act of creating a problem solution which is not computable from the agent's design or domain knowledge.

I call this a working definition because I do not argue that it covers all aspects of creativity. It defines that aspect of creativity which is addressed in this paper.

As an aside, note that the classification of artifacts with respect to the knowledge that handles them also provides clean definitions for routine design and innovative design. In routine design, the problem-solver instantiates prototypes from its base of *design* knowledge. In innovative design,

---

[1] Note, however, that the discrete nature of the formulation makes techniques such as Williams ([WI90]) applicable.

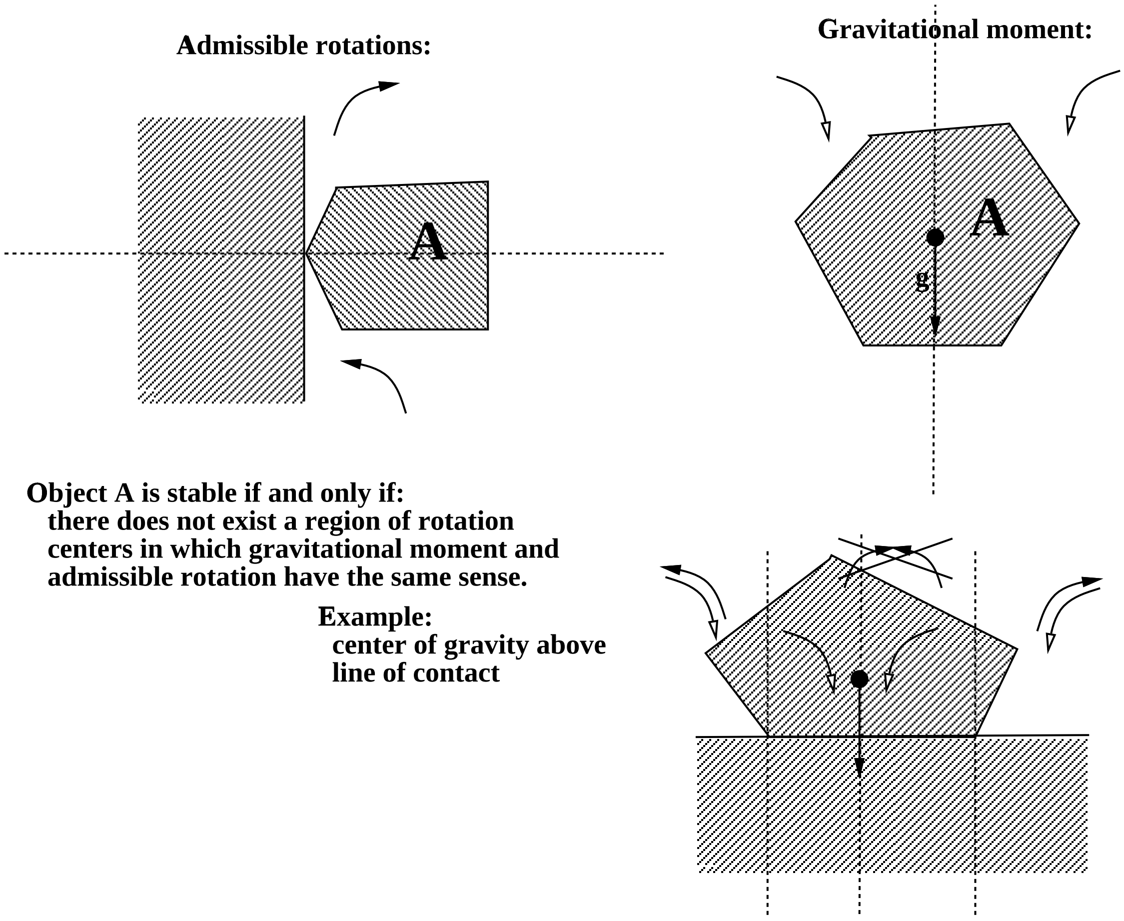


Figure 4: *The possible motions of a 2-dimensional object can be characterized by the directions of admissible rotations for different locations of the rotation center. In particular, contact with a surface means that only clockwise rotation is permitted for rotation centers to the left of the outgoing surface normal, and only counterclockwise rotation for all positions in the other half plane. For determining stability, the gravitational moment divides the plane into one half plane where rotation centers have a clockwise moment, and another half where the moments act counterclockwise. An object is stable if and only if there is no region of rotation centers where the direction of freedom of motion and the gravitational moment coincide, for example when it is supported by an edge and the center of gravity lies above it. In this case, rotation centers to the left would only admit counterclockwise motion, but the moment is clockwise. Rotation centers above the line of contact admit no rotation at all, as the two contact points at the ends admit rotations in opposite directions. On the right side, direction of motion and gravitational moment are again opposed.*

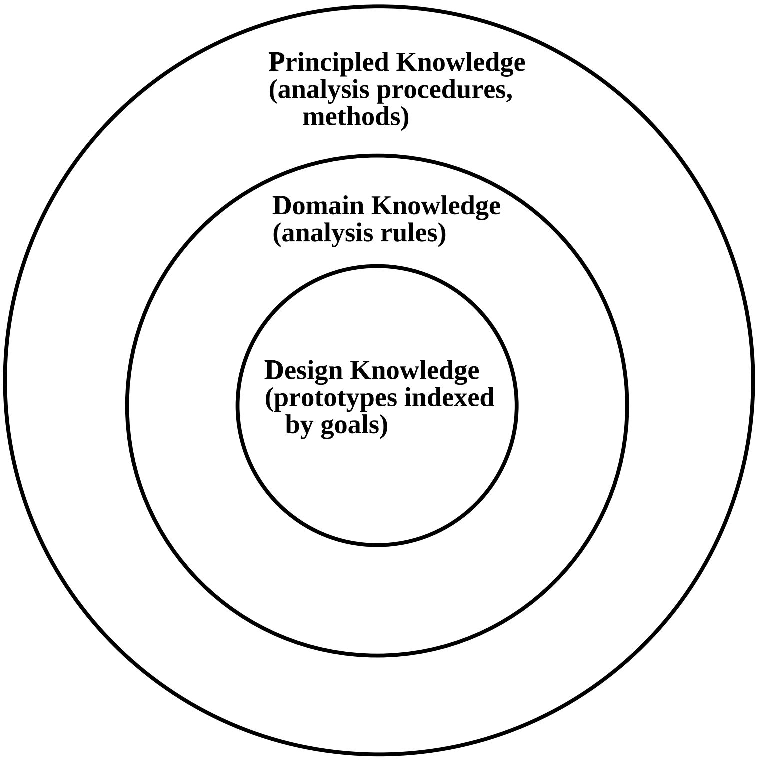


Figure 5: *Three different forms of knowledge in a design system, and the space of artifacts they can address.*

it uses abductive reasoning on its *domain* knowledge to generate novel prototypes. This process goes well beyond routine design, since the space of possible solutions is usually infinite.

The advantage of this definition over others found explicitly or implicitly in the literature is twofold. First, it makes the act of creativity independent of the environment: the fact that somebody had already invented but not publicized the same solution earlier does not change the nature of the creative act. However, it avoids the pitfall of making implicit assumptions about the computational process underlying creativity, but defines it entirely at the knowledge level. The interesting fact is that there are many designs which are creative by this definition and which at the same time *can* be handled by principled knowledge. In retrospect, this is not surprising: numerical analysis programs have often been used as a basis for creative designs. For example, turbine designers use numerical simulation to find patterns of turbulence which emerge in certain designs. These patterns are previously unknown to the designers, and by coupling it with optimization the numerical analysis program can generate solutions which are novel and should probably be called creative.

In this paper, I show that *emergent* phenomena which arise, for example, in numerical computation can be used as a basis for creativity. The computational *process* of creative design would involve several steps:

1. *generation* of candidates which might allow a creative solution.
2. *discovery* of a novel phenomenon as a basis for a creative design.
3. *generalization* of the new phenomenon so that it can be used to extend domain and ultimately design knowledge.
4. *instantiation* of a particular creative solution.

In the following sections, I will first address each of the steps for the small example problem used above, and then discuss how this process could be supported in a realistic knowledge-based CAD system to support and enhance a designer's creativity.

4 An example of a creative design

In order to illustrate the implications of this definition, consider again the earlier example of an agent which is arranging polygons in a two-dimensional space with gravity. Figure 6 shows an example of a problem which requires a creative solution: stably support the polygon shown in grey using the surfaces formed by the black obstacles.

4.1 Generation of candidates for creative solutions

Once the system exhausts its knowledge in searching for a routine or innovative solution to the problem, it is known that a solution to the problem must lie outside of the scope of its current

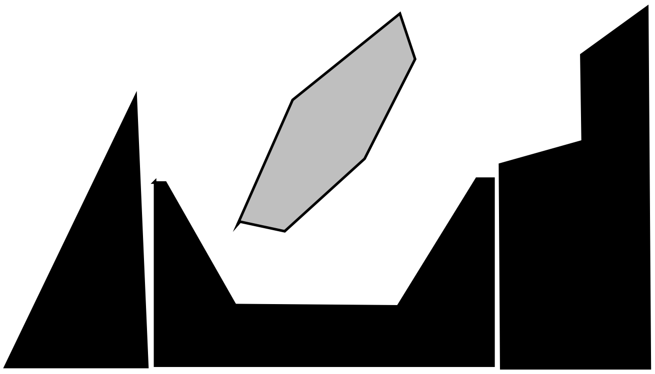


Figure 6: *Example of a problem which requires a creative solution: the shaded object cannot be placed in any stable position which can be generated using the design and domain knowledge present in the system.*

knowledge. In order to obtain criteria for finding creative solutions, it is interesting to see what predictions would the current domain knowledge make on such a solution. Since it does not cover the new solution, its predictions could either be

- contradictory, as several elements of domain knowledge apply in incompatible ways. Domain knowledge has to be refined with additional conditions to make it applicable.
- incomplete or missing, since no knowledge can be applied to the situation. Additional elements of domain knowledge have to be added.

These two possible relations between a creative solution and existing knowledge can be used to systematically generate creative solutions. Figure 7 shows an example of generating a creative solution based on contradictory predictions. In the example, the system finds by unification that the example allows it to produce a placement where there are two surface contacts, for which domain knowledge generates contradictory predictions. The existence of this example proves that domain knowledge is incomplete. Extending it to correctly cover this case will introduce new categories which make creative solutions possible. In general, one can formulate schemas of contradictions in domain knowledge which can be used to systematically search for such candidates.

Using the second process is much more difficult, since it involves structures for which no representation exists in domain knowledge. An example of such a candidate would be to support the object by contacts of corners, as shown in Figure 8. Such solutions could be generated in several ways. The first are overly general heuristics which propose novel structures, a technique which has been explored by Lenat ([LE84]). In this case, the heuristic might be to explore an intersection of two different contact regions. A second technique which would have a similar effect would be analogies from other domains. In temporal domains, most events have a duration,

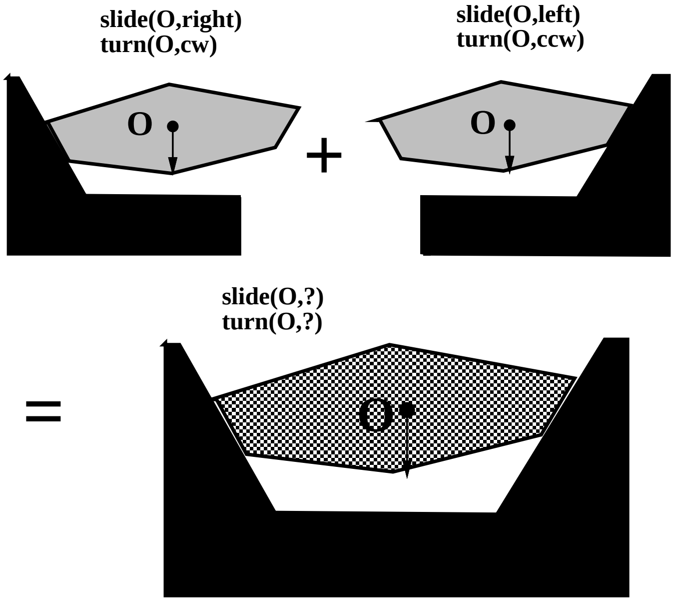


Figure 7: *Example of how a candidate for a creative solution can be obtained by looking for contradictions in domain knowledge.*

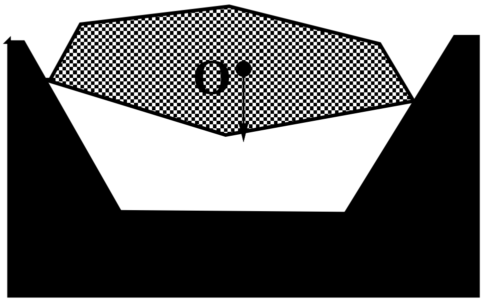


Figure 8: *A solution which is creative by the fact that domain knowledge does not apply: suspending a polygon by its corners.*

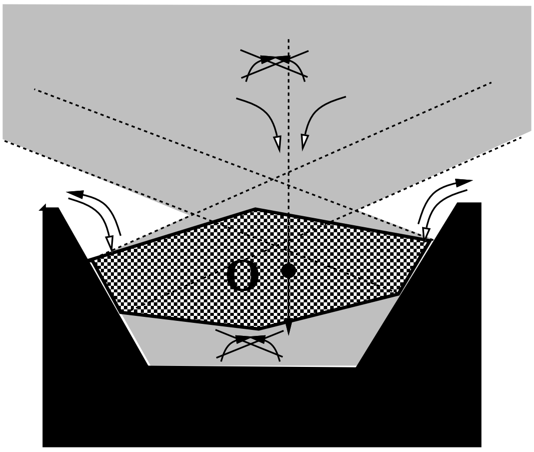


Figure 9: *Using principled knowledge to determine that a placement with two surface contacts is in fact stable. Only two regions allow any motions of the object, and in both the gravitational moment is opposed to the direction of allowed motion.*

but there are also instantaneous events which would correspond to the corner contacts. A third technique is discovery by numerical simulation: simulating random motion of a polygon in its environment finds examples of corner contacts. The examples and their properties can be generalized by unsupervised learning algorithms and produce new categories as a basis for creative solutions. Even though this solution might seem far-fetched, it is already heavily used by engineers who use numerical simulation to "get a feel" for the behavior of complex systems in fluid mechanics and elsewhere.

4.2 Recognition of novel phenomena

Once a candidate for a creative solution has been generated, the fact that it is indeed a novel solution to the problem must be recognized. For both examples of candidates shown earlier, this can be done using the general principled knowledge about stability outlined in the introduction to the example.

For the solution based on a pair of surface contacts, the analysis is shown in Figure 9. The result of the analysis is that the particular candidate proposed is in fact stable: there are only two regions of admissible rotation centers and in both regions, the gravitational moment is opposed to the admissible direction of motion. Since the principled knowledge is based on contact points, it also covers the analysis of corner contacts in a similar way.

In cases where the available principled knowledge is insufficient to evaluate a candidate for a creative solution, physical prototypes must be built and tested to extend the frontiers of analysis knowledge. I consider such scientific discovery to be outside of the scope of creative *design*.

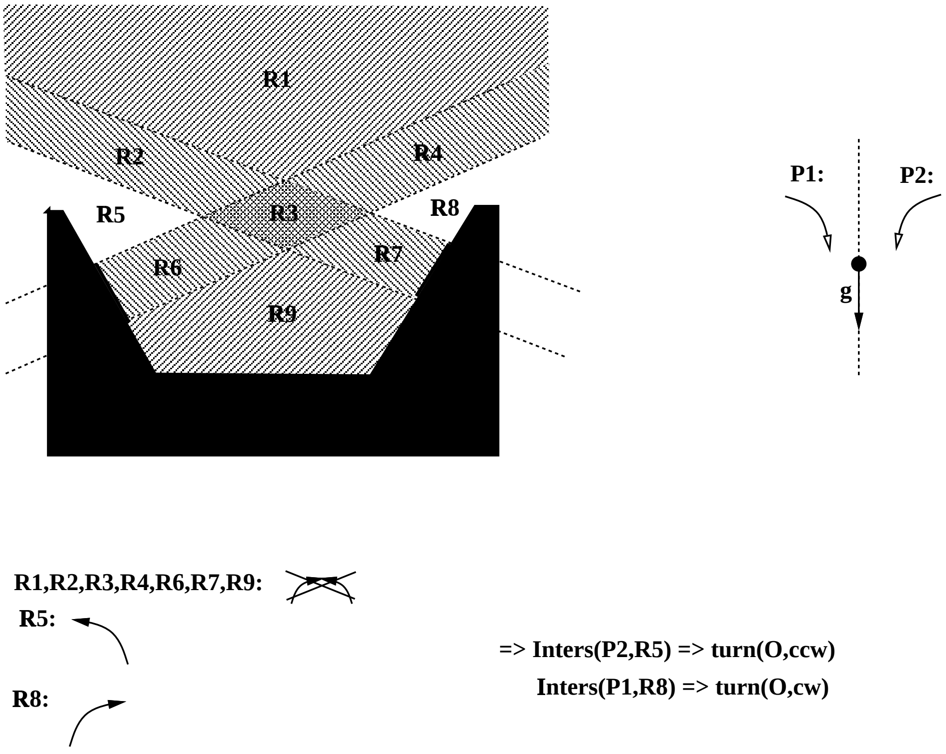


Figure 10: *Explanation of why the creative solution is stable.*

4.3 Generalization of novel phenomena

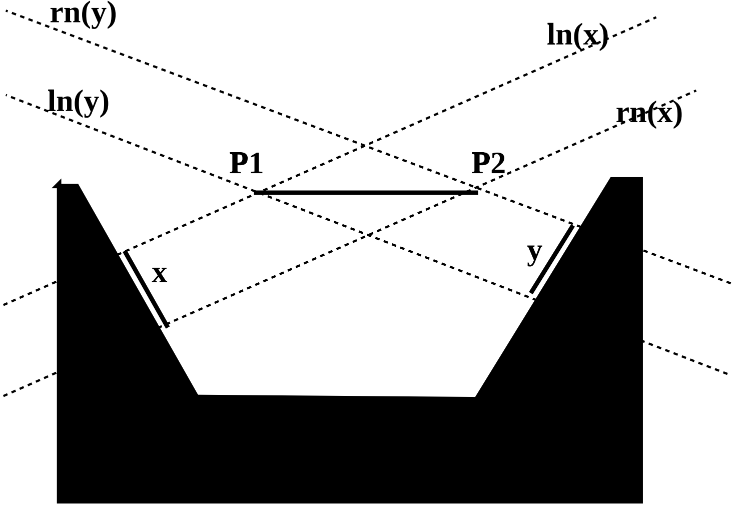
For design, it is in general insufficient to discover just one instance of a novel phenomenon, since this would make it impossible to integrate it with other design requirements. A novel phenomenon should therefore be *generalized* and added to domain knowledge before it is used in a creative design solution.

Generalization is the central topic of machine learning research, and there are two main techniques suggested for it. In *inductive* learning, large numbers of examples are used to find a common description by syntactic means. In *explanation-based* learning, a single example is generalized by using an explanation to determine its characteristic features and abstracting away all other information.

In our example, principled knowledge provides explanations which can be used for explanation-based learning. As shown in Figure 10, the prediction of stability rests on 9 different regions of potential rotation centers, defined by properties of the two involved contacts. The explanation of stable behavior is valid as long as (i) the region structure remains the same, and (ii) the gravitational moments are opposed to the admissible rotation directions in regions R5 and R8.

Based on the explanation, one can extend domain knowledge as shown in Figure 11. The

L1:
support(x,o) & support(y,o) =>
space-inters(line(inters(rn(x),ln(y)), inters(ln(x),rn(y))),x,y)



Rule5: support(x,O) & support(y,O)
&inclined(x,right) & inclined(y,left)
& space-inters(z,x,y) & within(cg,z)
=> stable(O)

Rule6: support(x,O) & support(y,O)
&inclined(x,right) & inclined(y,left)
& space-inters(z,x,y) & left-of(cg,z)
=> turns(O,ccw)

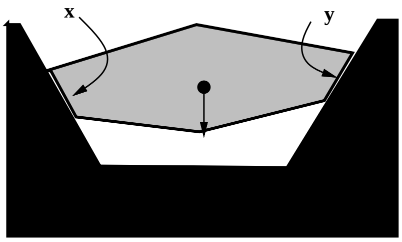
Rule7: support(x,O) & support(y,O)
&inclined(x,right) & inclined(y,left)
& space-inters(z,x,y) & right-of(cg,z)
=> turns(O,cw)

Figure 11: *Definition of a predicate space − inters and three rules which extend domain knowledge to pairs of contact surfaces.*

predicate *space − inters* defines the boundaries of regions R5 and R8 in Figure 10, and the line which satisfies it must be found by geometric computation based on the contact surfaces. It defines the limits within which the center of the object's gravity must fall to ensure a stable position, as shown by the new rule 5. As a by-product, note that domain knowledge will also be extended by rules 6 and 7, showing what happens when the center of gravity is not within the required interval. Note that a similar analysis and generalization can be made for corner contacts and combinations of corner and surface contacts.

Explanation-based learning is in general much more efficient than inductive learning, and should therefore be the preferred method for evaluating creative solutions. However, in many practical domains principled knowledge takes the form of numerical simulations, which do not furnish explanations useful for learning. In such cases, generalization by induction on many examples is the only way to generalize the phenomenon. This approach has been demonstrated by Cagan in the 1st Prince program ([CA88]), which is capable of inducing a general formula from a progression of instances and subsequently use it to produce an optimal design. However, the success of inductive learning methods depends crucially on the representation of the problem: only if suitable features appear can inductive learning construct meaningful descriptions. Further research is needed to determine how sufficiently general languages of features can be developed.

Prototype2: to-make-stable(O) =>
support(x,O) & support(y,O)
&inclined(x,right) & inclined(y,left)
& space-inters(z,x,y) & within(cg,z)
=> stable(O)



L1:
support(x,o) & support(y,o) =>
space-inters(line(inters(rn(x),ln(y)), inters(ln(x),rn(y))),x,y)

Figure 12: *A new prototype which encapsulates the creative solution.*

4.4 Instantiation of creative solutions

After domain knowledge has been extended to cover the new solution, the system can finally define the prototype which corresponds to the creative solution it is developing. Because domain knowledge has been extended by generalizing a single case to its limits, the same knowledge can be formulated as a prototype in which the limits of applicability are clearly defined. In our example, the new discovery would result in the new prototype shown in Figure 12.

5 How can computers support creativity?

Even though the preceeding example has shown that it is possible to build creative computer programs in very simple domains, the size of the models that would have to be provided to enable creativity in realistic domains would be unmanageably large. In practice, it is more important to build intelligent CAD tools which *support* human creativity by allowing extensions to their qualitative models of the design being produced.

More precisely, we are considering the context of a knowledge-based CAD system where users do not act on the geometric model of the artifact directly, but through the intermediary of a qualitative, abstract model which generates the parts according to his specifications. This model is being followed by several commercial intelligent CAD systems, such as ICAD or WISDOM CAD, in order to capture design rationale and constraints. However, such a system evidently hinders creativity in that the user cannot design devices which are not expressible in the qualitative model. How can this paradigm be enhanced to support creative design? In this section, I examine the different processes underlying creativity to examine how knowledge-based design systems should be constructed to support them.

Generating Candidates I have discussed two different ways of generating candidates for creative solutions. The first involves searching for situations to which domain knowledge applies but results in contradictory predictions. In an intelligent CAD system, this process can be readily supported and even automated. If the KB system operates purely deductively by prototype instantiation, it must provide a facility for *prototype combination*. If it is also capable of abductive reasoning using domain knowledge, then abductive schemata for constructing conflicting explanations must be provided. Although both requirements appear feasible using current technology, existing intelligent CAD systems do not make any provisions for them yet.

If candidate generation involves completely novel elements, such as the corner contacts discussed in the examples, pure combination is not enough. Since the candidate falls outside of the representation handled by the system's knowledge base, it can only be represented at the elementary geometric level. A number of ways can be imagined for automating the generation process, ranging from heuristic mutation of geometry to Monte-Carlo search methods. Creative solutions provided by the user can be entered using traditional drawing interfaces.

Analysis of creative solutions As long as creative solutions are covered by principled knowledge within the system, it can be analyzed automatically. However, if the solution is completely novel and falls outside the scope of the system's knowledge, there should be a way for the user to feed evaluations obtained by separate means back into the system. This is especially important if the creative solution is only a part of a larger design which has to take its performance into account. So far, knowledge-based design systems rarely provide such means, even though question-answering is common in knowledge-based technology. The possibility to communicate the behavior of components outside of the system's prototypical knowledge is an important feature which should be part of future ICAD systems.

Generalization Generalization of a creative design requires a representation of the limits up to which the solution remains valid. Such a representation in turn requires a language of features which is suitable for expressing it. It is not clear whether creation of new features could be automated, although researchers on self-organizing systems are hopeful to be able to do so ([FO90]).

In cases where principled knowledge allows a causal analysis of the device, it automatically defines features useful for generalization. In our example, the causal analysis traces stability back to features of point contacts, which define the necessary features for introducing corner contacts. A causal theory allows explanation-based generalization, which never requires adding new features. Work on theory revision (for example, [RDF85, RAJ90]) shows examples of how causal theories can be automatically extended to cover new phenomena.

This highlights the importance of general causal theories such as those developed in qualitative physics ([DKW90]) for the development of design systems that support creativity. Qualitative analysis can not only help the designer understand devices, but also defines the features required for reasoning about them.

In general, the ability to support creative design in a knowledge-based system depends very heavily on the availability of principled knowledge in *causal* form. The example used to illustrate the concepts in this paper crucially depends on the causal nature of the theory of stability in all stages of the creative process. There are many domains where the development of causal theories has been abandoned in favor of efficient numerical simulations; fluid dynamics, structural analysis and mechanism kinematics are examples of this. For supporting creativity, progress in fields which develop causal theories, such as qualitative physics, is an important prerequisite.

6 What creativity is not covered by this model?

The model I have presented in this paper is geared towards understanding the questions of how any form of creativity could be supported on computers. It has focussed on aspects of creativity which one can hope to support on computers. These are surprisingly broad: for example, many forms of artistic creativity, in which new interactions of elements are explored, are fully covered by it.

A first important group of creative solutions left out by this model are those which rely on discovery of entirely new phenomena in the real world, such as nuclear fission. In theory, it is possible to support such creativity as well by providing the computer with the possibility to experiment in the real world, but this does not seem to be a promising avenue for design systems.

Another aspect which has been ignored in this paper is that of extending the usable knowledge using analogies across different domains. People seem to make heavy use of such analogies, but supporting them on a computer would require a uniform representation which is sufficiently general to cover many domains - a task which the human brain seems to achieve but which so far has eluded computer and cognitive scientists. Further development in this area will enable design systems to support many other forms of creative design.

7 Conclusion

The starting point of this paper was the question of how a computer could support creativity. I have given a definition of creative solutions in the context of a knowledge-based design system, and shown the processes that could support their generation on a computer. The result, which should be surprising to many readers, is that by exploiting the different scope of analysis and synthesis knowledge a computer can in fact create novel solutions without any reference to outside stimulus. This amplifies the early results of Lenat ([LE84]) on heuristic discovery systems which could be called creative.

The model developed in this paper shows a direction how knowledge-based CAD systems can be developed into tools which support creative design. The most important directions for this development are generalization and learning techniques, and causal models of the world which permit their application. Such techniques are already well-developed in certain areas, but much effort remains to be done to make the sufficiently general and robust for integration with practical knowledge-based CAD systems.

In the entire discussion, I have carefully avoided the issue of *evaluation*: when is a solution to be considered interesting and worthy of further investigation? Besides the fact that this issue is exceedingly difficult to handle in a formal way, I believe that users of design systems will always want to maintain control of the design process and intermediate evaluations. Interactive design systems are an elegant way of bypassing the issue of evaluating creative designs.

References

[BO91] **M. A. Boden**: "The Creative Mind: Myths & mechanisms," George Weidenfeld and Nicolson Ltd. (Great Britain), 1990.

[CA88] **J. Cagan, A.M. Agogino**: "Inducing constraint activity in innovative design," *Artificial Intelligence in Engineering Design, Analysis and Manufacturing,* 1988

[DKW90] **J. DeKleer, D. Weld**(eds.): "Readings in Qualitative Reasoning about Physical Systems," Morgan-Kaufmann, 1990

[FO90] **S. Forrest**(ed).: "Emergent Computation," MIT Press, 1990

[GE90] **J. Gero**: "Design Prototypes: a knowledge representation schema for design," *AI Magazine*, 1990

[GEM91] **J. Gero, M.-L. Maher**: "Mutation and Analogy to Support Creativity in Computer-Aided Design," in *CAAD futures '91*, Vieweg 1991

[LE84] **D. Lenat**: "The Role of Heuristics in Learning by Discovery: Three Case Studies," in *Machine Learning: An Artificial Intelligence Approach*, Springer 1984

[NIE88] **P. Nielsen**: "A Qualitative Approach to Mechanical Constraint," *Proceedings of the 7th Conference of the AAAI*, 1988

[RAJ90] **S.A. Rajamoney**: "Towards a Computational Model of Theory Revision," in *Computational Models of Discovery and Theory Formation*, Lawrence Earlbaum Associates, 1990

[RDF85] **S.A. Rajamoney, G. DeJong, B. Faltings**: "Towards a Model of Conceptual Knowledge Acquisition Through Directed Experimentation," *Proceedings of IJCAI-85*, Los Angeles, 1985

[SHO85] **Y. Shoham**: "Naive Kinematics: one aspect of shape," *Proceedings of the 9th IJCAI*, 1985

[SYN91] **K. Sycara, D. Navinchandra**: "Index Transformation Techniques for Facilitating Creative Use of Multiple Cases," *Proceedings of the 10th IJCAI*, 1991

[WI90] **Brian Williams**: "Interaction-based Invention: Designing novel Devices from first Principles," *Proceedings of the 8th Conference of the AAAI*, 1990