# Design Simplification by Analogical Reasoning

Marton E. Balazs & David C. Brown
*University of Cambridge, United Kingdom*
*Worcester Polytechnic Institute, USA*

**Key words:** Design complexity, design simplification, model-based reasoning, analogical reasoning

**Abstract:** This paper presents some results of our research on design simplification by analogical reasoning. It first defines what we mean by a simplification problem. Then it describes our approach to solving design simplification problems by analogical reasoning and presents an implementation of this approach. One of the important contributions of the research presented in the paper is the use of relevance for guiding the analogical reasoning process. The benefits of using relevance are analyzed through a set of experiments presented in the paper. Finally, the paper draws a set of conclusions and proposes directions for future research.

## 1. INTRODUCTION

Ever since artifacts have been produced, improving them has been a common human activity. Improving an artifact refers to modifying it such that it will be either easier to produce, or easier to use, or easier to fix, or easier to maintain, and so on. In all of these cases, "easier" means less *resources* are required for those processes. While 'resources' is a general measure, which can ultimately be expressed by some measure of *cost* (such as time or money), we believe that at the core of many improvements is the notion of *reduction of complexity*, or in other words, *simplification*. For instance, the less complicated an artifact is, as measured by the number of parts it consists of, the easier it will be to manufacture. It is clearly the case that the cost of the actual manufacturing process will depend on the technological sophistication of the manufacturer, experience and skill of the workers and so on. However, as opposed to cost, the complexity of an artifact gives an objective characterization of the difficulty of its manufacturing.

Studying simplification is a very important direction of research since it targets the understanding and simulation of a basic human (cognitive) activity. This can lead to important results from both theoretical and applicative points of view. On the other hand, the study of simplification, may set a context for studying human creativity as a by-product of goal-directed reasoning processes.

This paper presents some results of our research on performing simplification of designs by analogical reasoning. The paper is structured as follows: Section 2 defines the design simplification and uses and example to illustrate our approach to solving design simplification problems. Section 3 presents our approach to solving design simplification problems using goal-driven analogical reasoning and describes the implementation of a design simplification system. Section 4 analyzes some experimental results produced by the system. Finally the paper discusses related work, draws some conclusions and outlines the most important plans for our future research.

## 2. SIMPLIFYING DESIGNS

Simplifying a design means to reduce its complexity. We view complexity as a way of characterizing designs from a given *point of view*, that is *context*, *aspect* and *measure*. A *context* for characterizing a design by its complexity refers to a process that can be performed on the design (e.g., describing it, producing it, using it and so on). For a given context, an *aspect* is the collection of those elements of the design which play a role in their characterization in the context considered (e.g, its structure, its behavior or its function). Finally, for a given context and aspect, a *measure* is a function that assigns to a design a numeric value that characterizes the complexity of the design in the given context and aspect. For example, counting the components of a design can be a measure of complexity defined for the context of manufacturing, in the aspect of structure: it characterizes the number of components that have to be manufactured before the design can be completed. In the following, when talking about the complexity of a design we will assume that a point of view has been chosen.

We say that a design *A* is *simpler* than another design *B* if the complexity of *A* is lower than the complexity of *B*. In this case we also say that the designs *A* and *B* are in the *simpler-then* relation. We will call an instance of the simpler-than relation a *simplification*.

A simplification can be discovered in one of the following two ways: a) by measuring and then comparing the complexities of two given designs, or b) solving a design simplification problem. In our research we are interested in solving design simplification problems.

Given a design, and a point of view (context, aspect and measure) the *design simplification problem* is the problem of finding another design that has the same function as the original design and its complexity is less than that of the original design. We call the process of solving a simplification problem *simplification process* (when no ambiguity can occur we will used the term *simplification* to denote a simplification process).

The question to which our research is proposing an answer is “How can design simplification problems be solved in an effective and efficient way?”.

Clearly simplification can be done in different ways. For instance one could apply simplification rules acquired from designers. Alternatively one could use previously performed simplifications as model for simplifying a design. Reusing known simplifications by adapting them to new problems allows the solution of new simplification problems. If the reuse is based on generalization and mapping to a different domain, new and previously unsolved simplification problems could be solved. More than that, the generalization would support the search for general simplification principles. These observations are the basis for our choice to use analogical reasoning to perform simplification. The approach to solving simplification problems presented in this paper is based on what we call “goal-directed” analogical reasoning.

In the following subsection we illustrate our approach to solving simplification problems using an example.

### 2.1 A Design Simplification Example

Consider the design of a personal fax machine presented in Figure 1a (Petroski 1996). The functions of this design are to allow the sending and receiving of faxes. The complexity of this design in the context of manufacturing, with respect to the aspect of structure as measured by the number of its components is 6.

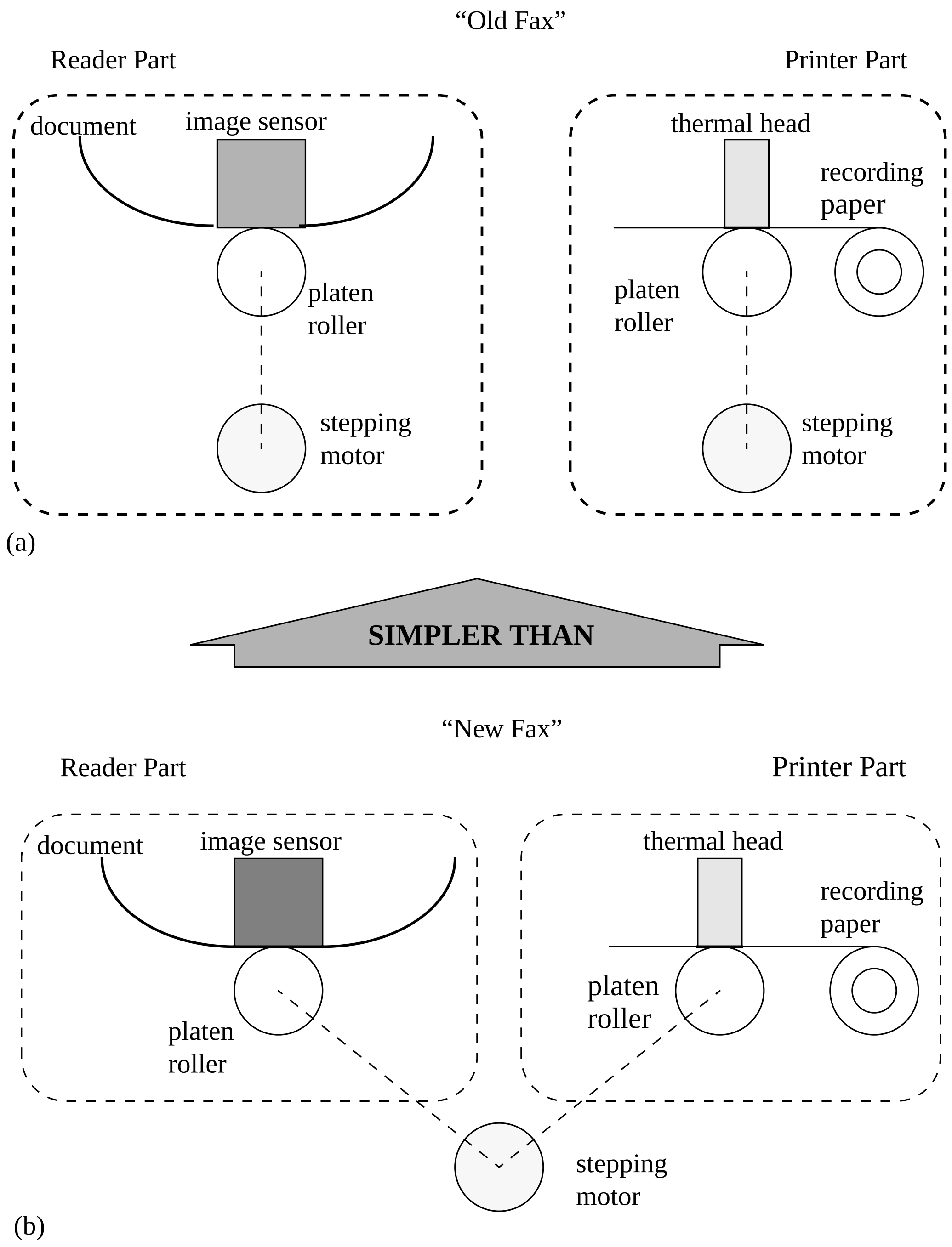

*Figure 1.* Simplification of a personal fax machine. Adapted from (Petroski 1996)

A solution to the simplification problem specified by this design and the point of view considered is presented in Figure 1b (Petroski 1996). This new design has the same functions as the original one and its complexity (measured for the same point of view) is 5, that is less than the complexity of the original design.

We propose that the simplification process that produces the “New Fax” design from the “Old Fax” design consists of the following steps:

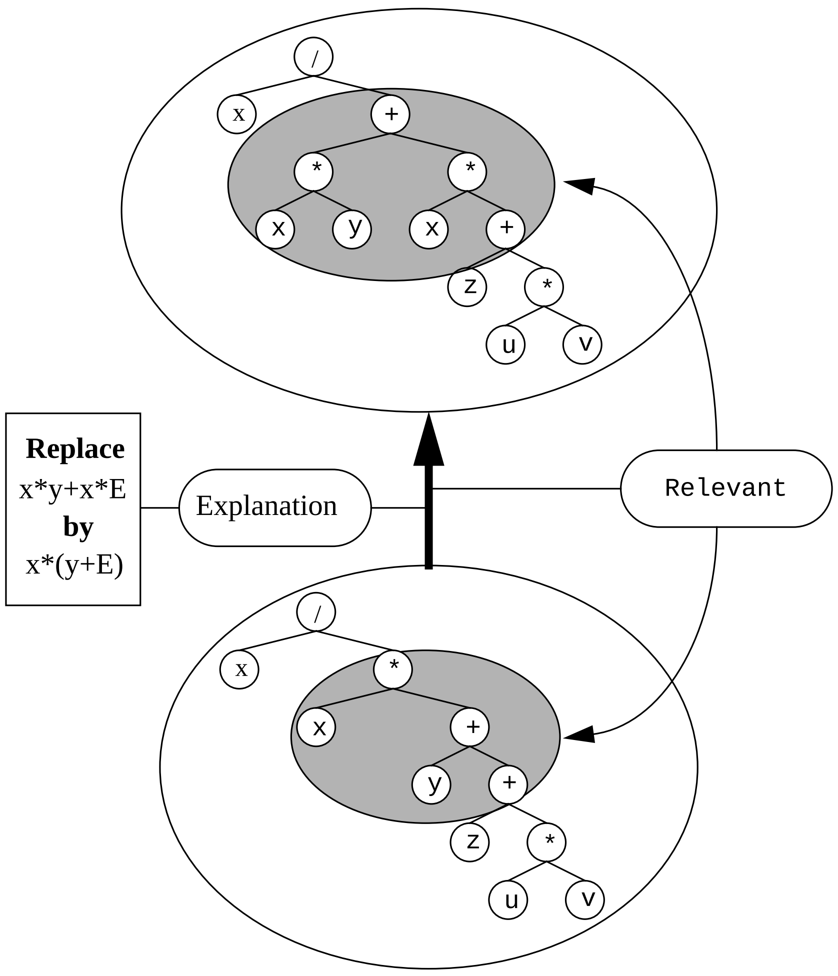

*Figure 2.* Source analog retrieved for the fax design simplification problem

1. A database of known simplifications is searched for simplifications that could be used as model for solving the problem. This search is based on measuring the degree of similarity between the “Old Fax” and designs that are the “more complex” members of known simplifications. The measure used has to favor designs that have a structure (i.e. system of relations) similar to that of the “Old Fax”. It is important to note here that it is possible that individual simplifications may refer to only some parts of designs. We say that those parts are *relevant* to the corresponding simplification. This observation can be exploited in searching the database of known simplifications as well as in subsequent phases of the simplification process.
   If we use a database of simplifications that contains examples of simplification for arithmetic expressions the simplification shown in Figure 2. may be retrieved as the best model for solving the simplification problem at hand. The simplification retrieved is called *source analog*.

2. Next the simplification knowledge represented in the source analog has to be transferred to the problem domain and applied to the simplification problem. Both the transfer and the application may require adaptations such that the result is valid in the problem domain.

For the example considered the simplification process transferred from the source simplification can be described by

REPLACE:
*a motor* **driving** *a roller* **and** a *motor* **driving** *a roller*
BY:
*a motor* **driving** (*a roller* **and** *a roller*)

Applying this process to the "Old Fax" will result in the simpler "New Fax" presented in Figure 1b.

Note that the actual processing is much more complicated than described above. In order to be able to preform it the following questions must be answered (Bhatta and Goel 1994):

- What should be the content and representation of source analogs?
- How is the target problem specified?
- Given a target problem, how might the retrieval of the source analog occur?
- Once a source analog has been retrieved, how can it be mapped onto the target problem and how will this mapping be used to transfer the problem solving knowledge?
- How can the solution to the target problem be completed?
- How will a solution to the target problem be evaluated?
- How can it be decided whether a useful generalization over the source problem and the target problem can be built. How can such a generalization be built?
- How can it be decided whether the target problem and its solution are different (novel) enough to be worth storing for later use?
- How can the generalization and/or the target problem be stored into the database of problems for later use?

Our research proposes answers to all these questions in the context of design simplification by analogical reasoning taking into consideration the possibility of using the simplification goal to improve the performance of the processing.

## 3. DESIGN SIMPLIFICATION BY ANALOGICAL REASONING

In this section we describe the model of the goal-driven analogical reasoning process we are proposing for solving design simplification problems. This was derived from a quite general model of analogical reasoning. The analogical reasoned process consists of the following phases:

*Retrieval of candidate source analogs*: this phase selects from the set of known simplifications those that have the same point of view as the problem, and which are "similar" to the problem. Similarity is measured in terms of the number and kind of elements (e.g., components, relations and attributes) they share.

*Selection of the source analog*: each candidate analog retrieved has associated with it a score which measures its similarity to the object to be simplified. This score is used to select the simplification that is closest to the problem.

*Mapping of the source analog onto the problem*: this phase will produce several "global mappings" that are consistent sets of correspondences between relevant elements in the source analog, and elements in the problem.

*Selection of the best global mapping*: each of the global mappings obtained will be evaluated for quality by combining the scores of the member correspondences (e.g., correspondences between relations will assigned higher scores than correspondences between attributes for analogical reasoning). The scores of the member correspondences are assigned at the time of retrieval. The global mapping with the highest score will be selected to be used for transferring the simplification knowledge.

*Transfer of simplification knowledge*: the best global mapping will be used to produce several candidate simplifications by associating the unmapped elements in the source analog with elements in the problem.

*Application and evaluation of candidate simplifications*: all of the candidate simplifications are applied to the simplification problem, producing new objects. The objects produced will be evaluated against the problem constraints and for the simplification condition. If an object produced does not satisfy the constraint or is not simpler than the object specified in the object, it is dropped.

*Selecting the solution*: the object that has the minimal complexity from among those which satisfy the constraint and are simpler than the object to be simplified, will be reported as solution to the simplification problem.

*Generalization and learning*: if the simplification that was applied is significantly different than the source analog it has been derived from, it will be added to the database of known simplifications. Also, if a useful generalization over the new simplification and the source simplification can be built, it will also be added to the database.

Figure 3 illustrates the intermediate results of the different phases in the processing.

## 3.1 Representing and Organizing Design Simplifications

To be able to perform analogical reasoning on simplifications we need to define how simplifications will be represented as well as how those representations will be organized to support the reasoning process.

We will represent a simplification by a binary relation (called *simpler-than relation*) that connects two objects, a simpler one and a more complicated one (Figure 4). A simpler-then relation has two attributes: an *explanation* of the simplification it represents and a description of the aspects of the two objects that are *relevant* to the simplification.

The explanation is used for two purposes. One one hand it is the basis for determining which elements of the objects involved in the simplification are relevant. On the other hand it can be used to build abstractions over simplifications, with the purpose of organizing simplifications into hierarchies. Such hierarchies are useful for the analogical transfer of simplification knowledge, as well as for generating general simplification rules and/or principles.

The relevant elements are central to this research because they are used to focus the processing in all the phases of the analogical reasoning process to only those portions of objects that are involved in some simplification.

### *Explaining a Simplification*

The explanation of a simplification can be given in either of the following two ways: a) specifying the difference between the two objects involved in the simplification, or b) specifying the process by which the simplification was achieved.

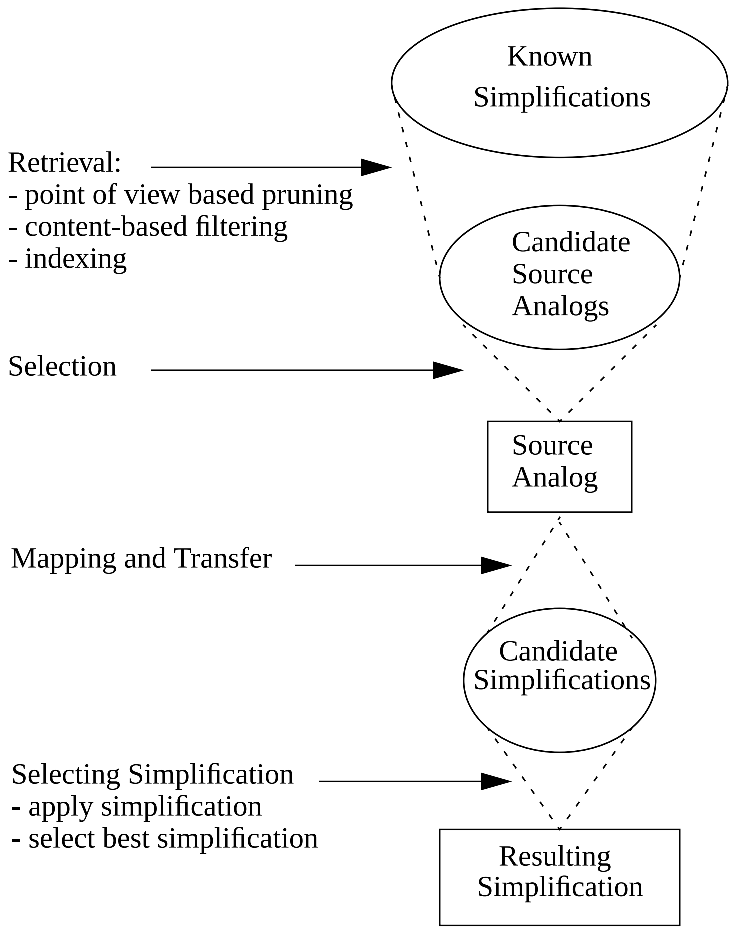

*Figure 3.* The process of producing a simplification. The ovals represent sets of simplifications, with larger ones containing more simplifications, the rectangle represents one simplification.

Specifying the difference is needed when the fact that an object is simpler than another one was "discovered", but no simplification process is known. How the difference can be specified depends on the ontology used for representing the objects. For instance if the objects are represented using an objects, components, relations and attributes ontology, the difference can be represented by two sets: a set of elements (components, relations and attributes) that are part of the more complicated object, but not part of the simpler one (elements removed), and a set of elements that are part of the simpler object, but not of the more complicated one (elements added).

When the process by which the simplification was achieved is known, the description of this process can be added to the simpler relation as an explanation. A simplification process will be represented as a sequence of transformations. Each transformation involves two objects, the transformation operation applied, and the precondition which had to be satisfied in order for the operator to be applicable.

## *Elements Relevant to a Simplification*

Elements relevant to a simplification (or simply, *relevant elements*) are elements of the designs involved in the simplification that play some role in discovering or

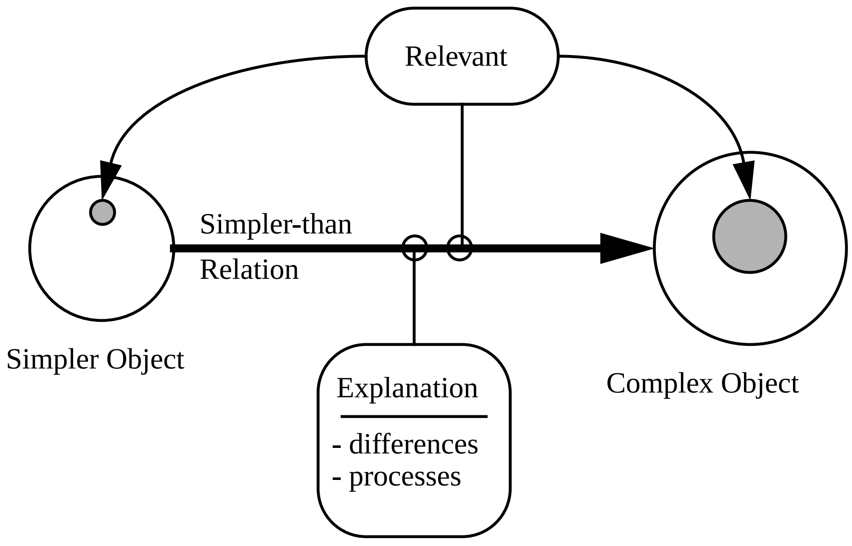

*Figure 4.* The structure of a simplification.

explaining the simplification. For example elements referred to in the explanation of a simplification are relevant (to that simplification). Relevant elements are useful for two purposes. On one hand, they allow building *abstractions* of the objects involved in simplifications. These abstractions will not contain those portions of the objects that are irrelevant to the simplification. On the other hand, the relevant elements can be used as a basis for building indexing schemes over the set of objects involved in known simplifications.

The set of relevant elements corresponding to a given simplification can be computed automatically from the two objects involved in the simplification and the explanation of the simplification. We call the process of finding the relevant elements *relevance calculation*. Relevant elements of a simplification can be computed at the time the simplification is created and can be used thereafter whenever needed.

The relevance calculation can be decomposed into two phases: a) c*ollecting the elements that are not absolutely irrelevant* (with respect to the explanation) and b) *propagating relevance* along relations in the objects.

Elements not absolutely irrelevant (Levy 1994) with respect to an explanation are elements that are explicitly mentioned in the explanation of the simplification. They may occur in the description of differences, in the case that the explanation is given in terms of differences, or in partial descriptions of objects, specifications of preconditions and arguments of operators, if the explanation is given as a process. These elements are said to be not absolutely irrelevant because, while mentioned in the explanation, they may not be absolutely needed. However there may not be any basis for discarding them as irrelevant.

Note that not absolutely irrelevant elements are elements explicitly present in the explanation of a simplification. These elements may be related to other elements in the design which were not explicitly present in the explanation, but which may bear some relevance to the simplification. This could happen for instance when the explanation of the simplification is given by the difference between the two objects involved. In this case only the removed and added elements are specified, without any reference to relations between them. We conclude that relevance may also need to be propagated inside the more complex objects involved in the simplification, along the decompositions, relations and attributes. This propagation can be done either downwards (i.e., from component to subcomponents, from relations to arguments, or components to attributes), or upwards (i.e., from subcomponents to their “parent” component, from components to relations they are arguments of, and from components to relations of which attributes they are).

## 3.2 The Analogical Reasoning Process

In this subsection we describe the analogical reasoning process we are proposing for solving simplification problems emphasizing the use of relevance to guide the process.

### *Retrieving*

Retrieving is the first phase of the analogical reasoning process. Its purpose is to find a simplification that corresponds to an object “similar” to the object that needs to be simplified.

For the purpose of retrieval, simplifications are organized first into classes of simplifications corresponding to points of view and, second by an indexing scheme over the relevant elements of the objects involved in the simplification. Consequently retrieving similar simplifications will also work in two stages: a) *pruning*, that is restricting the search to only the class of simplification with the same point of view as the one specified in the simplification problem, and b) *indexing*, that is search using the indexing schemes.

The first stage is trivial and will be implemented by marking each object involved in some simplification with the corresponding point of view. Note, that if we decide to organize points of view into a hierarchy, a more efficient data structure should be used.

The second stage in retrieving a similar simplification is to *use the indexes* built over the relevant elements of the simplifications, in the class under consideration.

The result of retrieving is a set of candidate analogs. Each candidate analog consists of a set of *match hypotheses* on which its selection as a candidate was based. Each match hypothesis has associated with it a *score* computed during the retrieval process.

The candidate analog with the highest score is selected to be used in the next phase of the analogical reasoning process. In the rest of this subsection we will assume that a candidate analog has already been selected. As usual, we will refer to the selected candidate analog as the “source” and to the simplification problem as the “target”.

### *Mapping*

*Mapping* is the second phase of the analogical reasoning process. It builds maximal sets of consistent correspondences (*matches*) between relevant elements of the source and elements in the target, called *global mappings* (or *gmap*s, as the are called in the Structure Mapping Engine (SME) literature). For mapping we propose to use a modified version of Falkenheiner’s SME (Falkenheiner et al., 1993).

SME takes as input two descriptions, one of the source and one of the target, and produces as output a set of gmaps of the source onto the target. Each gmap contains a maximal set of matches. Here ‘maximal’ means that adding any match to it would violate the consistency of the gmap. SME also attaches to each gmap a *structural evaluation score* which provides an indication of the quality of the mapping.

In our approach to solving the simplification problem by analogical reasoning, the retrieving process associates with each candidate source analog a set of correspondences between relevant elements of the source and elements in the target. Each of these correspondences has assigned to it a score that is an indication of the quality of that correspondence. Our implementation of SME uses these correspondences as initial match hypotheses

The mapping process produces a set of global mappings which can be the basis for different simplifications that are likely to be applicable to the simplification problem to be solved. Our purpose is to select the best of these global mappings to increase the chances of generating a simplification in the target. For evaluating global mappings we use a structural evaluation method that assigns higher scores to deeper structures.

At the end of the mapping phase the global mapping with the highest evaluation score will be selected for further consideration in the analogical reasoning process.

#### *Transferring Simplification Knowledge*

Once a global mapping has been selected as the best candidate for analogical transfer, it will be used to compute *candidate simplifications*. A candidate simplifications is a simplification in the source which can be hypothesized to be applicable in the target as a result of the correspondences of the global mapping.

A candidate simplification is computed by finding elements in the source which are consistent with the global mapping's correspondences, but are not in fact included in them. We will call these elements *unbound elements*. Unbound elements are searched for in the set of relevant elements of the source since those are the only ones that play some role in that simplification.

Once the unbound elements are found the existence of corresponding elements in the target can be hypothesized. Building these hypotheses is performed by the simplification knowledge transfer process.

How exactly the simplification knowledge will be transferred depends on whether the explanation for the simplification is given by a difference or by a simplification process.

#### *Evaluating the Result of the Simplification*

Each of the simplified targets resulting from transferring the simplification knowledge and adapting the result has to be evaluated for the: a) *requirements* on the object, b) *constraints* of the simplification and c) *complexity.*

After the best simpler object was selected the corresponding simplification can be generated. The target and the new object will be respectively the 'more complicated' and 'simpler' objects involved in the simplification. The explanation will be computed as the difference between the two objects, and the relevance calculation will be applied.

### 3.3 A Design Simplification System

To demonstrate our model of design simplification by goal-directed analogical reasoning we implemented a design simplification system. The system was implemented in the CLIPS language (CLIPS 1993). The architecture of the system is shown in Figure 4. The system consists of a *database of known simplifications*, an *interface* module, a *data management* module, a *simplifier* and a *simplification abstraction* module.

## 4. ANALYZING THE EFFECT OF USING RELEVANCE

Applying the relevance of object elements (e.g., subexpressions of arithmetic expressions, or components of designs) to simplifications throughout the analogical reasoning process is one of the important contributions of this paper. It is the way by which the

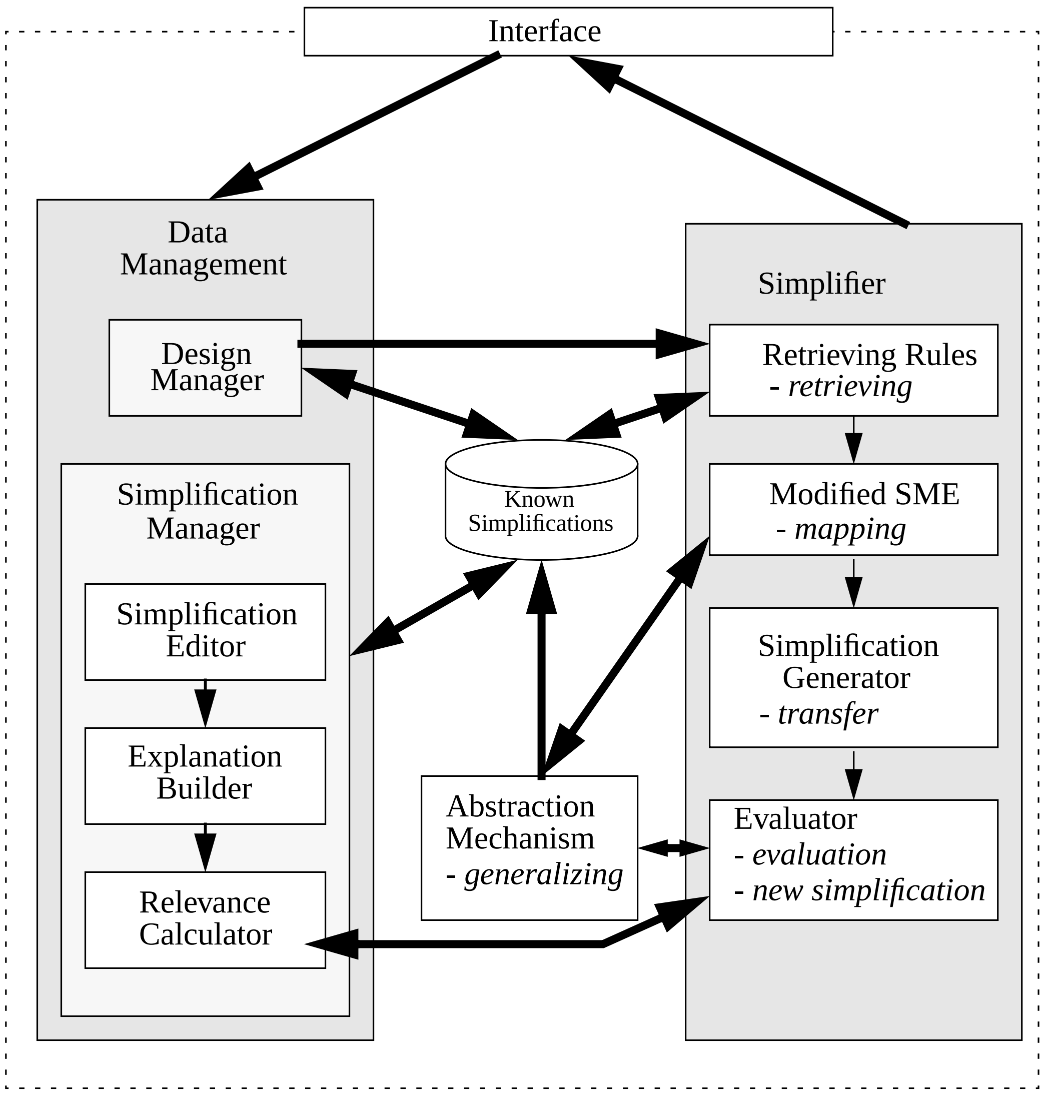

*Figure 5.* Architecture of the simplification system

model of analogical reasoning proposed takes into account the problem solving goal (i.e., simplification), producing goal-directed analogical reasoning.

To demonstrate that using relevance to guide the analogical reasoning process improves the performance of our system, we had to measure its effect. For this we needed:

1. to hypothesize which phases of the analogical reasoning process could be affected by the use of relevance,
2. to define a measure of performance for each of those phases,
3. to design and perform a set of experiments to collect statistics about the performance measures of the different phases.

As it is the practice in analyzing algorithm complexity we propose that the time performance of a process should be measured by the number of *specific operations* performed. Unfortunately there is no unique specific operation that can be counted to measure the system's overall performance. For this reason we chose to identify specific operations for each of the phases of the analogical reasoning process that are affected by the use of relevance.

In addition to simply measuring the effects of using relevance on the performance of the system, in this set of experiments we also studied the effect of different kinds of relevance propagation methods (e.g., downward propagation, limited propagation, no propagation) on the operation of the system (Note here, that in the current implementation of our system we do not support upwards propagation, which is the reason why this propagation method was not tested).

The following subsections describe the setup for the experiments performed to measure the effect of using relevance to guide the analogical reasoning process, and present and discuss the results obtained.

## *Setting up the Experiments*

To measure the effect of using relevance on the operation of the system, we first hypothesized that the phases affected will be: a) the retrieval of source analogs, and b) the mapping of the source analog retrieved onto the target. For both of these phases we defined a measure of performance in terms of a specific operation. The measures used were:

- the *number of a match hypotheses created* for the retrieving phase,
- the *number of global mappings* (called *gmaps*) *generated* for the mapping phase.

To perform the experiments the system was loaded with a simplification database consisting of simplifications of arithmetic expressions. The database was generated in a previous session, using the system's simplification management capability and contained simplifications explained by difference, as well as simplifications explained by a simplification process description. We performed the following experiments:

1. We turned off the relevance checking in the system. Then, for each of the versions of the simplification database (corresponding to different numbers of simplifications) we presented the system with all those examples used in the first set of experiments which produced correct simplifications. We repeated the same set of experiments with the relevance checking turned on. For both of the cases we collected the measures defined above and compared the results.
2. We also wanted to study whether considering relevant elements to be only the ones explicitly referred to in the explanation of simplifications would make a difference in the performance of the system. For this purpose we regenerated the databases used for our experiments such that the "relevance propagation" is limited to only one level (i.e., only to the elements explicitly referred to). We reran our experiments described in point 1. above with the newly generated simplification databases and performed the same measurements and comparisons.

### *Results and Discussion*

The results of the experiments for measuring the effect of using relevance are given in

| | **Relevance OFF** | | **Relevance ON** | | **One-Level Relevance** | |
|---|---|---|---|---|---|---|
| **Database Size** | **Match Hypotheses Generate** | **Gmaps Generated** | **Match Hypotheses Generated** | **Gmaps Generated** | **Match Hypotheses Generated** | **Gmaps Generated** |
| 18 | 18 | 7 | 11 | 4 | 14 | 6 |
| 40 | 37 | 13 | 27 | 9 | 32 | 11 |
| 50 | 49 | 17 | 36 | 12 | 41 | 17 |
| 75 | 65 | 24 | 41 | 15 | 52 | 21 |
| 85 | 67 | 27 | 56 | 21 | 61 | 33 |

*TABLE 1.* Summary of Experimental Results for Measuring the Effect of Using Relevance

Table 1. The three main columns in the table correspond to performing the experiments without taking into account the relevance, with the relevance fully propagated down, and with the relevance only propagated down one level in the structure of the design matched, respectively. In all the cases the system produced the expected simplification.

The results show that when taking relevance into account the system generates much fewer match hypotheses and, as a consequence of this, much fewer global mappings. This leads to an improvement of the performance of the system. Just by looking at the results in Table 1, we could claim that applying (fully propagated) relevance produced an improvement of about 50, in terms of both the number of match hypotheses generated and number of global mappings generated. This is however a result that cannot be claimed to hold in every situation because the actual results depend on the size of the database, the complexity of designs involved in the simplifications and the number of elements in each simplification that are relevant to it (e.g., for a given simplification it is possible that only one element is relevant, but it is also possible that all the elements of the “simpler” design involved are relevant).

Our experiments with the one-level propagation of relevance show that, in this case, the system generates slightly more match hypotheses and gmaps than when using full propagation, but less than when not using propagation at all. The reason for this is that one-level propagation restricts the elements that can be matched, but not to the extent by which full propagation does. It appears that using full propagation would always be the best choice. However this may not always be the case. For simplifications for which the explanation is given by difference a full propagation may be needed because there is no way of knowing under what conditions the difference is applicable. On the other hand, for simplifications for which the explanation is given by a simplification process description, no propagation should needed, because, ideally, all the relevant elements should be referenced in the process description (in a condition, a transformation, or a state description). As a consequence, the issue of relevance propagation needs further studying.

The conclusion we draw from these experiments is that using relevance to guide the analogical reasoning process can significantly improve the performance of the system.

## 5. RELATED WORK

We have no knowledge of any ongoing research in the area of “goal-based analogical reasoning for design simplification”. However, there is certainly a rich body of research results in the relevant domains — model-based analogical reasoning about designs and design optimization.

Model-based analogical reasoning refers to using mental models of the underlying domain in the analogical reasoning process (Gentner 1983). Most of the work on analogical design (Qian & Gero 1992) (Bhatta & Goel 1994) (Goel 1997) relies on mental models of designs.

Analogical reasoning theory postulates that goals help determine both what gets matched and how the match gets evaluated (Gentner 1993). This idea is incorporated in some of the research on analogical reasoning (Holyoak and Thagard 1989) (Forbus and Oblinger 1990). Our approach is related to both Holyoak & Thagard’s and Forbus & Oblinger’s work. Similar to Forbus & Oblinger, we propose goal-based filtering. However, in our approach the filtering doesn’t only refer to local matches considered, but to designs and design parts based on relevance of their components, attributes and relations.

Goal-directed analogical reasoning is not to be mistaken for *purpose-directed analogical reasoning* (Kedar-Cabelli 1988). Goal-directed analogical reasoning refers to using the problem-solving goal to guide the analogical reasoning process. On the other hand purpose-directed analogical reasoning, as used by Kedar-Cabelli refers to using the purpose of using (the function of) an artifact to guide the analogical reasoning about its structure.

Design simplification, and simplification (as a cognitive activity) in general, is a less researched area. One of the most clear formulations of (what we may interpret as) design simplification principles are Stoll’s (1991) design rules for efficient design for manufacture. The only general approach to design simplification we know of is Suh’s (1990)(1999) “Reduction of the Information Content of a Product”. This work gives a formal definition of the *information content* of a design, which we may interpret as a ‘measure of complexity’. In their work on methodologies for estimation of design projects Bashir and Thomson (1999), suggest that the accuracy of estimating the time required by a design project depends on the accuracy of effort estimation. They propose a way to measure complexity of a project in the context of designing it and from the aspect of its function. Bothroyd and Dewhurst (1991) developed a set of principles for reducing the manufacturing and assembly cost of a product. Their work is in the domain of Design for Manufacture (DFM) and Design for Assembly (DFA). They view simplification as the reduction of a complexity measure in the context of manufacturing. As a general principle for simplification they propose the reduction of the number of parts.

## 6. CONCLUSIONS & FUTURE WORK

The research presented in this paper had two major objectives: a) to define the simplification problem, and b) to propose a way to solve simplification problems. In pursuing these objectives this research has produced the following contributions:

- It gave an *operational definition of simplification and of a simplification problem*.
- It proposed a model for *solving simplification problems by using analogical reasoning*.
- It proposed a *model-based analogical reasoning approach to design simplification* problems.

- It *produced a working system* that implements the model proposed for solving simplification problems.
- Through a series of experiments, this research *demonstrated that the system is operational.*
- It demonstrated that the use of relevance in generating match hypotheses, retrieving source analogs and building mappings improves the performance of the analogical reasoning mechanism.

As a general conclusion the research presented in this paper proposed new research directions, presented original definitions, proposed new applications and approaches to existing problems, and implemented and experimentally studied a new system.

### *Future Work*

In addition to the results proposed, the research presented in this paper opened up new research directions and raised a series of theoretical and practical questions that need to be studied. This gives us several opportunities to extend our research on simplification, in general, and on design simplification, in particular.

We will perform further experiments with the system with the purpose of testing it for various examples, especially with real life problems, drawn from the area of design.

Currently the system has been tested for structural simplifications, but in our future research we will perform experiments with both behavioral and functional simplification.

Adding new application domains to the ones currently accepted by the system will extend the area of possible applications as well as increase the capability of the system to use cross-domain similarities to produce interesting, and hopefully novel simplifications.

The simplification propagation problem (i.e., how simplification in one aspect is propagated to other aspects) raises questions such as: What representation can adequately support the propagation of simplifications? How can the propagation of simplification be performed? What are the possible consequences of simplification propagation and how can those consequences be evaluated and anticipated? In our future research we plan to address the problem of simplification propagation.

Finally we expect that due to the use of analogical reasoning, our approach to solving simplification problems may come up with creative simplifications. For example, “importing” a simplification idea from a different domain may suggest a completely novel way of simplifying. We are interested in studying under what conditions our goal-directed analogical reasoning simplification process will be able to produce creative results.

## REFERENCES

Balazs, M.E. (1999). *Design Simplification by Analogical Reasoning,* Ph.D. Dissertation, Worcester Polytechnic Institute.

Balazs, M.E.  and Brown D.C. (1998). A Preliminary Investigation of Design Simplification by Analogy, In Gero, J. and Sudweeks, F. (eds.), *Proceedings of Artificial Intelligence in Design '98,* Kluwer.

Balazs, M.E.  and Brown, D.C. (1998). Structural, Behavioral and Functional Simplification of Designs, In *Proceedings of the Functional Modeling and Teleological Reasoning Workshop, AAAI-98*.

Bashir, H. A. and Thomson, V. (1999). A Quantitative Estimation Methodology for Design Projects, In *Journal of Engineering Design* (submitted).

Bhatta, S.R., Goel, A.K. and Prabhakar, S. (1994). Innovation in analogical design: A model-based approach, In Gero, J. and Sudweeks, F. (eds.), *Proceedings of Artificial Intelligence in Design '94,* Kluwer, pp. 57-74.

Boothroyd, G. and Dewhurst, P. (1991). Product Design for Manufacture and Assembly, In Corbett, Dooner, Meleka and Pym (eds.) *Design For Manufacture*, Addison-Wesley, pp. 165-173.

CLIPS, (1993). *C Language Integrated Production System,* Version 6.0, Lyndon B. Johnson Space Center, Software Technology Branch.

Falkenheiner, B., Forbus K. and Gentner, D. (1993) . The structure-mapping engine: Algorithm and examples, In: Buchanan and Wilkins (eds.) *Readings in Knowledge Acquisition and Learning, Morgan* Kaufmann Publishers, pp. 695-726.

Forbus, K. and Gentner, D. (1989) Structural evaluation of analogies: what counts?, In *Proceedings of the Cognitive Science Society*

Forbus, K.  and Oblinger, D. (1990).  Making SME greedy and pragmatic. *Proceedings of the Cognitive Science Society.*

Gentner, D. (1983) “Structure-Mapping: A Theoretical Framework for Analogy”, Cognitive Science, 7, 1983, pp. 155-170.

Gentner, D. (1993). “The mechanism of analogical learning”, In: *Readings in Knowledge Acquisition and Learning,* (Eds.) Buchanan & Wilkins, Morgan Kaufmann Publishers, San Mateo, CA, 1993, pp. 673-694.

Gentner, D and Forbus, K. (1991)  “MAC/FAC: A model of similarity-based retrieval”, *Proceedings of the Cognitive Science Society.*

Goel, A. (1997). “Design, Analogy and Creativity”, *IEEE Expert,* vol. 12, no. 3, May/June, 1997, pp.62-70.

Holyoak K.J. and Thagard, P. (1989)  “Analogical mapping by constraint satisfaction”, *Cognitive Science*, 7(2).

Levy, A. Y. (1994) Creating Abstractions Using Relevance Reasoning, In *Proceedings of the 12th National Conference on Artificial Intelligence, AAAI'94, Vol. 1*, pp. 588-594

Petroski, H. (1996) *Invention by Design*, Harvard University Press.

Qian, L  and Gero, J.S. (1992).  “A design support systems using analogy”, In: *Artificial Intelligence in Design'92*, (Ed.) J.S. Gero, Kluwer Academic Publishers, 1992, pp.795-813.