# DESIGN CONCEPT LEARNING USING INDUCTIVE LEARNING TECHNIQUES IN AN INTEGRATED DESIGN SUPPORT SYSTEM

MING XI TANG, JOHN HAMILTON FRAZER, HONG LIU
*The School of Design, The Hong Kong Polytechnic University*

**Abstract.** Design and learning are closely related activities. Conceptual design is characterized by the uncertainties of the issues related to the design problem, design requirements, design constraints, and design solutions. Organizing design information to define an area of design problem within which these issues are gradually structured involves an inductive learning process. This paper tackles this learning process during conceptual design by utilizing inductive learning and concept formation techniques. The paper presents the architecture and the implementation of a Design Concept Learning System (DCLS) within a multi-agent architecture. It then reports on how the DCLS can be applied to conceptual design tasks in different domains.

## 1. Introduction

At the conceptual stage of the design process, a way to understand a design problem and its solution is to study known design examples, previously recorded design cases or design plans to form an abstract knowledge structure that can be further explored. This knowledge structure can be a generalized model of past design solutions, or a conceptual description in which design requirements and factors are initially organized.

Defining a knowledge structure of a design problem and its solutions involves an *inductive learning process.* In this inductive learning process a structural and characteristic description of a class of solutions to similar design problems gets gradually organized by analyzing previous design examples using the knowledge and information already available.

The capability of using inductive learning methods in design is an integral part of the intelligent behavior demonstrated by human designers. This capability should be part of any intelligent computer-based systems that support design activities. However, existing CAD systems are unable to identify the role of inductive learning in design process. As a consequence they offer limited support to systematic utilization of the available inductive learning techniques.

This paper attempts to establish an understanding of inductive learning and its relevance to intelligent design support. Two fundamental issues about inductive learning and intelligent design support are addressed in order to develop an advanced design support system that integrates inductive learning techniques with other AI-based design techniques:

1. The role of inductive learning in design process; and
2. How inductive learning algorithms can be developed and integrated into knowledge-based design support systems.

The paper starts with a review and a discussion on the relationship between design and inductive learning. This discussion is focused on the role of inductive learning in various stages of the design process. Based on this discussion an inductive learning model of design is described. This inductive learning model provides a framework upon which inductive learning techniques can be integrated with other AI-based design techniques to support conceptual design. The Design Concept Learning System (DCLS) is intended to be a generic learning component within a multi-agent design system architecture. In the paper, the current implementation and application of the Design Concept Learning System in product design are presented.

## 2. Inductive Learning Techniques

Inductive learning techniques are divided into supervised learning and unsupervised learning. Discovering regularities or similarities from examples without prior knowledge of the categorizations of the examples is termed *unsupervised learning*. That is, in an unsupervised learning system, there are no explicit examples of desired input/output associations. There is only a general notion about the way in which input examples should be mapped to output class definitions. In most cases this general notion specifies how the similarities of the input examples are to be measured. While the task for a supervised learning system is testing hypotheses, clustering methods are intended for generating hypotheses.

### 2.1 Cluster Analysis

The task of unsupervised learning is to discover how some data are divided into similar classes, and what these classes are. Many unsupervised learning procedures closely resemble statistical clustering procedures based on *numerical taxonomy* [Everitt 1981]. Learning algorithms in this paradigm deal with the problem of grouping or clustering similar instances into the same category or segregating dissimilar instances into distinct categories.

Clustering must involve the production of a class hierarchy using a numerical measure of similarity defined over an instance space. In particular,

the instances in this space are typically collections of numerical measures of attributes describing the features of observations. The motivation for clustering analysis is that a cluster provides useful information for further and deeper exploration of data. Because a cluster of similar objects may share some features, or a common cause, this information can be used to build some procedures that infer from the organized data in a more accessible and efficient way than on the original data. In AI, *cluster analysis* is referred to as *Unsupervised Pattern Recognition* [Everitt 1981].

Everitt states the problem of cluster analysis as:

> *Given a collection of n objects, individuals, animals, plants etc., each of which is described by a set of p characteristics or variables, derive a useful division into a number of classes. Both the number of classes and the properties of the classes are to be determined.*

At the beginning of a cluster analysis, both the *number* and the *composition* of the classes are unknown. The solution to a cluster analysis problem normally consists of a pattern of *n* objects (a set of clusters). Each object in the training examples belongs to one cluster only and the complete cluster, i.e., the structure of the data set, contains all the objects. In cluster analysis the measurement of distance is an important consideration as the recognition of clusters involves the assignment of relative distances between points. A clustering system must therefore be concerned with the following issues: representation of data; similarity measurement of clusters; control of the clustering process; and evaluation and interpretation of results.

## 2.2 Hierarchical Clustering Techniques

In a hierarchical approach to clustering, the original data set is not partitioned into a number of clusters in one single step. Instead the classification process consists of a series of partitions which run from a single cluster containing all individuals, to *n* clusters each containing a single individual, resulting in a hierarchical tree structure in which each node represents a cluster. There are *hierarchical clustering* methods that differ in the direction in which data are processed. These are *agglomerative approach*es (bottom-up) and *divisive approaches* (top-down). The former starts the process at the bottom of the tree by successively clustering the *n* individuals into a series of groups, while the latter starts it from the top by separating the n individuals successively into finer groupings. Hierarchical classifications can be represented by a two-dimensional diagram known as a *dendrogram* which illustrates the fusion or divisions made at each successive stage of the analysis [Fisher *et al* 1985].

### 2.3 Quantitative Similarity Measures

The major computation task of agglomerative methods is to determine the similarities among categories, so that the 'most similar' can be merged into a new category. There are a number of well-known measures for determining similarity between items of data:

- Single linkage;
- Complete linkage; and
- Group-average linkage.

As shown in Figure 1, *Single linkage* is one of the simplest agglomerative hierarchical clustering methods, known as the *nearest neighbour* strategy. The distance between groups is defined as that of the closest pair of individuals, where only pairs consisting of one individual from each group are considered. The single linkage distance is calculated using the following formula (1):

$$\sum_{k=1}^{n} [(x_{ik} - x_{jk})^2]^{1/2}, \qquad (1)$$

Where $x_{ik}$ and $x_{jk}$ are the values of attribute k for observations i and j, respectively. The closer the values of individual attribute the smaller the distance, and the greater the similarity.

The *complete linkage* or *furthest neighbor* clustering method is the opposite of single linkage in the sense that the distance between groups is defined as that of the most distant pair of individuals, one from each group.

The distance between two clusters in the *group average linkage* is defined as the average of the distances between all pairs of individuals that are made up of one individual from each group. Empirical studies have shown average and complete linkage as the most useful in practice [Everitt 1981].

### 2.4 The Divisive Approach

The divisive method works top-down, repeatedly trying to break down instances into smaller groups [Quinlan 1979]. A major concern of this method is to identify the most decisive factors of the observations to construct a simple tree. Divisive clustering techniques are further divided into two types, the *monothetic* technique which divides data on the basis of the possession or otherwise of a single specified attribute, and the *polythetic* technique which divides observations based on the values of all the attributes.

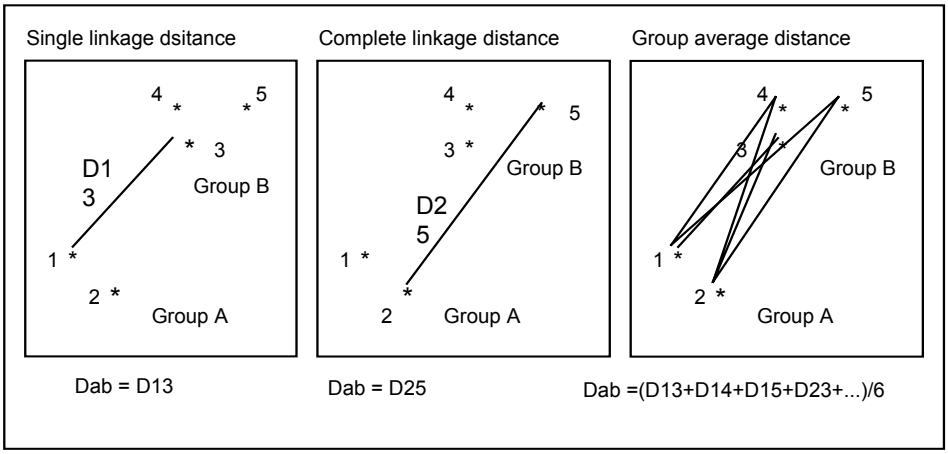


*Figure 1*: Similarity Measures

The divisive method measures *dissimilarity* during the course of splitting the instances into groups. In this method, a so-called *splinter* group is accumulated by sequential addition of the individual whose total dissimilarity with the remainder, less its total dissimilarity with the splinter group, is a maximum. When this difference becomes negative the process is repeated on the two sub-groups. The usual measure of dissimilarity in the divisive approach is the *average Euclidean distance* between each individual and the other individuals in the group.

### 2.5 Concept Formation Systems

Numerical taxonomy based procedures in cluster analysis divide data solely on the basis of the attributes of the instances themselves. Therefore there is no easy way of taking into account the semantic relationships between instance attributes, or global concepts that are relevant to the classification being developed. Everitt points out, *sometimes the assumption that useful clusters will arise from a series of local, pair-wise comparisons may be invalid* [Everitt 1981]. In addition hierarchical methods cannot repair what has been done in previous steps. Since all agglomerative hierarchical techniques ultimately reduce data to a single cluster containing all the individuals, and the divisive techniques will finally split the entire set of data into *n* groups each containing a single individual, the user wishing to have a solution with an 'optimal' number of clusters has to decide on which particular stage to stop at.

Duffy and Kerr observed this problem when applying a clustering method in their investigation of customized perspectives [Duffy *et al* 1993]. Furthermore, most similarity-measures are best suited to numerical data. Techniques are needed for dealing with nominal data (e.g., the proportion of identically valued

attributes in two observations). In AI applications there are two additional goals: to facilitate the application in AI domains where nominal data are frequently found, and to incorporate some of the human analyst's search strategies into the clustering program itself.

Concept clustering is an approach that attempts to address these problems. A concept clustering system uses additional heuristic knowledge to guide the clustering process so that the structure of instances derived represents some context dependent concepts. The role of concept clustering is to discover appropriate classes as well as concept descriptions for each class. There is an additional intention that each node in the classification should be a concept that can somehow be evaluated in terms of compactness, naturalness of interpretation by humans, or the level at which a concept can be easily recognized, retrieved, or distinguished, etc. It is the explicit coupling of *characterization* and *clustering* that makes *concept clustering* a different approach from numeric taxonomy.

Fisher and Langely [Fisher *et al* 1985] view conceptual clustering as composed of two sub-tasks: clustering, which determines useful subsets of an object set; and characterization, which identifies a concept for each *extensionally* defined subset discovered by the clustering process.

Unsupervised systems that employ non-incremental learning strategies require all training instances at the outset. However, a system should be capable of adjusting concepts when new information becomes available. Concept formation has essentially the same goals as that of conceptual clustering. The main difference between the two is that in concept formation there is an additional constraint on the way in which the concept is induced. That is, the process of learning is *incremental.* A learning system is said to be incremental if it accepts instances one at a time, and if it does not extensively reprocess previously encountered instances when encountering new ones.

Given a sequential presentation of instances and their associated descriptions, a concept formation system must find out: *clusters* that group those instances in categories; an *intentional definition* (a concept) for each category that summarizes its instances; and a *hierarchical organization* for those categories. The essence of concept formation is to embody external heuristic knowledge into the learning process to form useful concepts rather than simple clusters of data.

The main features of a concept formation system are the hierarchical organization of concepts, top-down classifications, and an unsupervised, incremental and hill-climbing approach to learning. The presence of a concept hierarchy suggests that classification begins at the most general (top) node and sorts the instance down through the hierarchy. However, in concept formation an instance can be sorted down more than one branch, indicating that some

concepts may overlap. In addition, the sorting process may stop at a node higher than a terminal node in the hierarchy.

In a concept formation system, a description, a definition or an image in the concept hierarchy that represents a concept that matches a subset of the instances is stored in a node of the concept tree. By doing so the system *remembers* what has been learned and is therefore able to compare a new instance with one contributed to a current concept definition. In such a system, classification is based on some similarity measures but the external knowledge of the training instances is also used to control the process of growing the tree, thus influencing the number of classes to be clustered which in turn affects the formulated concepts.

## 3. A Design Concept Learning System in a Multi-agent Architecture

While many design systems utilized inductive learning techniques in various design tasks and demonstrated the importance of machine learning techniques for intelligent design support, few of them has addressed the issue of integrating inductive learning techniques with a knowledge-based design support system framework as a general computational component [Tang 1996].

Learning in design is not purely a matter of induction. A design system using inductive learning techniques needs not only to abstract class relationships from design examples, but also needs to make room for modifications by utilizing domain specific knowledge. In design, the purpose of learning is to capture knowledge that can be used to create new ideas [Reich *et al* 1993]. An important issue is how to make use of both domain knowledge and design heuristics in the learning process.

In our research, the integration of inductive learning techniques with a multi-agent design system is a major focus. This is to ensure that a learning system can provide wholesome solutions to complex design problems.

### 3.1 The Structure of DCLS

The Design Concept Learning System is a multi-agent computer-aided design environment that provides support for a group of software agents and designers to co-operatively solve design problems and then gradually improve its design ability via concept learning in distributed environment.

The structure of DCLS is shown as Figure 2.

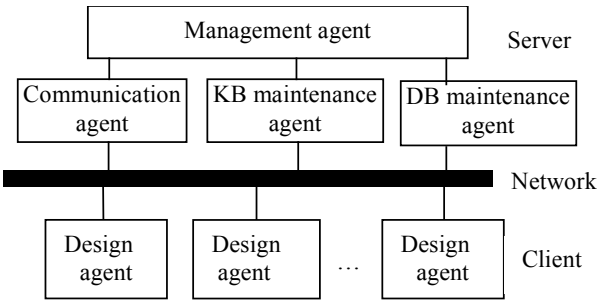


*Figure 2.* Structure of DCLS

In which, all agents are divided into three classes and located on the different layers. The top layer is a management agent that usually shows the function of the decision and inquiry for the design problem, control and supervision for lower layer agents. It grasps the features of all lower layer agents, including the ability, operation, interface and so on. When an agent is added to or deleted from the group, the change of all relative information in management agent is followed.

The second layer is a kind of tool agents. They help management agent to complete a certain system task, such as communication management, task decomposition, Knowledge Base (KB) and Data Base (DB) maintenance. These agents construct system management group together with the management agent.

The third layer is a group of design agents. These are agents with specific design functions in a special domain.

The multiple design agents are connected via a network and they cooperative work with the help of communication agent, KB maintenance agent, DB maintenance agent and so on. Management agent supervises the work of all the agents in the system and controls the design process together with the human designers and engineers [Liu *et al* 1997].

### 3.2 The Storage of Design Knowledge

There are two kinds of design knowledge in DCLS, long term one and short term one. The knowledge of long term is organized as files of four types:

- Set of product files consisting of a number of past designs described by corresponding design information and product model tree.
- Set of component files consisting of the component name, class, function, material, relation list, design agent ID and updated date.
- Set of feature files consisting of the feature name, type, location, orientation, parameter list and corresponding datum file names.
- Set of datum files.

The knowledge of short team is organized as temporary files of four types:

- New design files consisting of descriptive information of the new design which have not matched with any past designs.
- Selected component file consisting of components' names and pointers which points to set of component files.
- Selected feature file consisting of features' names and pointers, which points to set of feature files.
- Design state file consisting of a record of design agent's working state, that is, what work the agent has done and what step has been going along.

### 3.3 The Process of Design Concept Learning

There are three phases in design concept learning. The first phase is inductive in nature. In this phase, previous design examples as input data are classified and characterized to form a *design concept tree.* Domain knowledge as the *background knowledge* specifies the levels of concept to be learned whilst the design examples are used to actually build the design concept tree along these concept levels. From this design concept tree the structure of design problem and its solutions can be retrieved and explained.

The second phase is of a more deductive nature compared with the first phase. The induced design concept tree is further analyzed by making plausible changes to parts of it in an attempt to explore desired behavior (or properties) unseen in the previous design examples and that is required to meet a stated new design requirement description. In this phase, different assumptions/changes on key design variables can be made in order to obtain multiple design results.

The third phase is the evaluation that involves matching the design solutions against the design requirement description to find out whether a particular design solution is acceptable. If a design solution is acceptable, then it is transferred to the design knowledge base as a new example for future use. The design decision process that has led to the specification of this solution is also recorded as a design history.

In DCLS, design knowledge is stored on different KBs, The public KB (located on server) and the local KB. Each design agent has its own KB. Design agent helps human designer to do design and get the design requirement via interaction with the human designers.

The design datum from different agents are passed to public KB and are classified to form a *design concept tree.* This process is done by the KB maintenance agent and a divisive approach mentioned in 2.4 is performed.

During a learning process, a design agent seeks for past design knowledge according to the specified design requirement. This involves matching the design solutions against the design requirement description to find out whether an acceptable design solution can be found. If a design solution is found, then it is transferred to the designer.

The search begins from a local KB. If the local search fails, the requests and replies take place between the design agent and public KB maintenance agent. The KB maintenance agent looks for the acceptable solution from the *design concept tree* in public KB.

If the search is successful, the copy of component design knowledge in public KB is passed to the requesting agent, and is placed on temporary KB (see Figure 3).

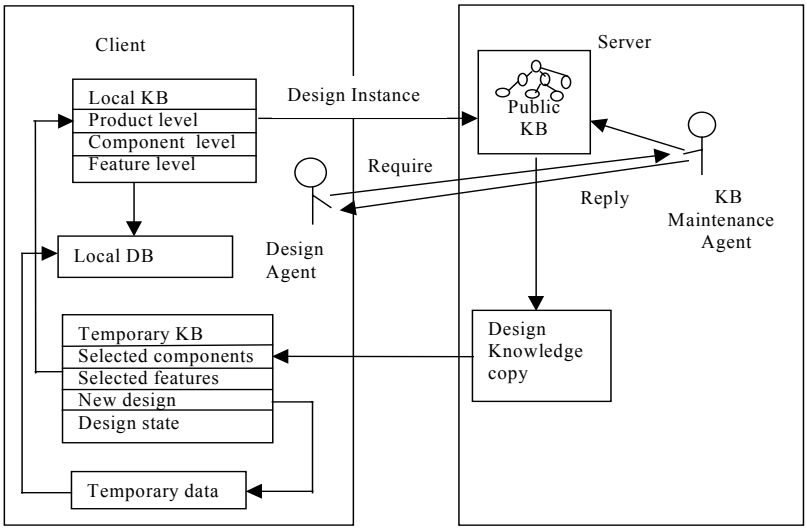


*Figure 3:* Design Concept Learning Process

Whenever a design modification phase is over, the short-term design knowledge will be updated. After all designs are fulfilled and are confirmed by the design engineers, the short-term design knowledge will be used to update the long team design knowledge.

When a design agent's long team design knowledge is updated, the corresponding information is passed to public KB maintenance agent via communication. Public KB maintenance agent uses them as instances to classify and update KB as mentioned in the third phase.

## 4. Applications

In this section, we introduce a mobile phone design as an example to show the design and learning process in DCLS.

**Step 1**. When a product design task is introduced, the design agent helps a design engineer to select the class of design product from a list. If he/she can find a suitable one (similar match), the corresponding product model tree (see

Figure 4) is shown on the screen. If no one is found, the designer selects New from the menu, and follows the guide to new classes and the product model tree.

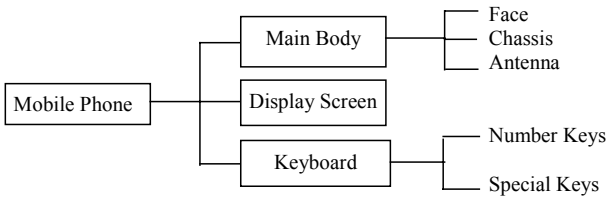


*Figure 4*: A Product Model Tree

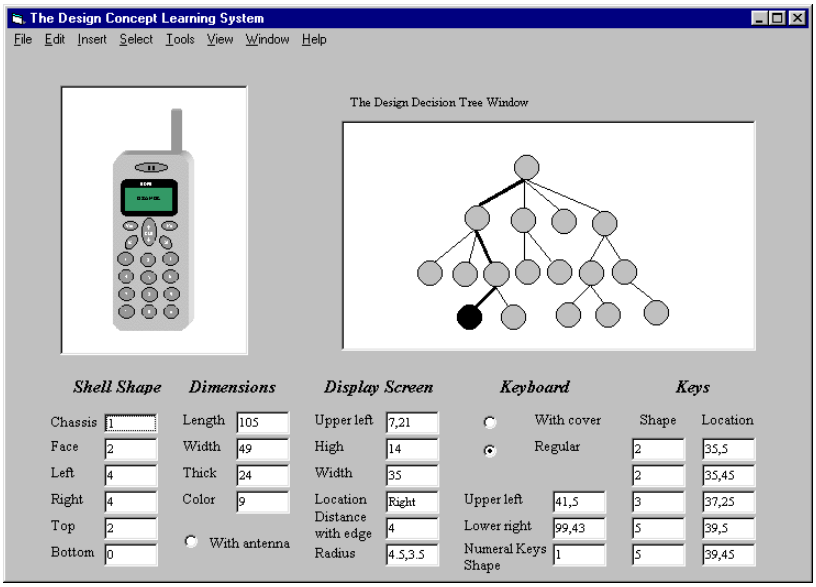


*Figure 5:* A concept tree and design product

**Step 2**. After Step1, the design task is decomposed and a product model tree is formed. The designer can add or delete components from the product model tree until he/she is pleased with it. She/he can then select one component and go to step 3. If this step is repeated until all components in the tree are complemented, then go to step 4.

**Step 3**. The design agent searches for the component according to the design requirement as introduced in section 3.3. If an acceptable component is found, it will be shown on the display screen, and the system will let the designer decide whether it will be selected. If it is accepted, then go to step 2, else the search repeats or new component design begins.

**Step 4.** The designer assembles the components according to their relations in the product model tree. Designer will decide to accept it or not. If it is accepted, the related new components will be saved to local KB and be passed to KB maintenance agent as a new example, then a new classifying process starts and the design decision tree is modified.

Figure 5 shows a design concept tree that is classified according to the divisive arithmetic and a design product with related parameters.

## 5. Conclusions

In conclusion, design is a complex problem-solution and stepwise refinement process. Learning can improve the abilities of designer and computing system based on experiential knowledge. The paper put forward a inductive learning model of design first. Then it presents the architecture and the implementation of a Design Concept Learning System DCLS within a multi-agent architecture. The integration of inductive learning techniques with multi-agent design system structure ensures that the system can provide wholesome solutions to complex design problems. It is therefore important to establish a theoretical model for such an integration as well as to develop a design concept learning system on the basis of this model

## Acknowledgements

This project is founded by the Hong Kong Polytechnic University (Project Reference No.S901) and a CERG project from the HK UGC (Project Reference No. PolyU5015/97H).

## References

[Duffy *et al* 1993] Duffy, A. H. B. and Kerr, S. M., 1993, "Customised Perspectives of Past Designs from Automated Group Rationalisations", Special Issue on Machine Learning and Design, International Journal of Artificial Intelligence in Engineering, Vol. 8, 1993.

[Everitt 1981] Everitt, B., "Cluster Analysis", London: Heinemann, 1981.

[Fisher *et al* 1985] Fisher, D. H. and Langley, P., 1985, "An Approach to Concept Clustering", Proceedings of Ninth International Joint Conference on Artificial Intelligence (pp. 691-697), Los Angeles, CA: Morgan Kaufmann, 1985.

[Liu *et al* 1997] Liu H., Zeng G. Zh. & Lin Z.K. An agent-based approach to cooperative design. In: Proc. Of Workshop on CSCW in Design' 97, Bangkok, International Academic Publishers, 1997;191-195.

[Quinlan 1979] Quinlan, J. R., 1979, "Discovering Rules by Induction from Large Collections of Examples", *Introductory Readings in Expert Systems*, D. Michie (Ed.), Gordon and Breach, London, 1979.

[Reich *et al* 1993] Reich, Y., Konda, S. L., Levy, S. N., Monarch, I. A., and Subrahmanian, E., 1993, "New Roles for Machine Learning in Design", Special Issue on Machine Learning and Design, International Journal of Artificial Intelligence in Engineering, Vol. 8, 1993.

[Stepp *et al* 1986] Stepp, R. E., Michalski, R. S., 1986, "Conceptual Clustering of Structured Objects: A Goal-Oriented Approach", *Artificial Intelligence*, 28, 43-69.

[Tang 1996] Tang, M., 1996, "Knowledge-based Design Support and Inductive Learning", PhD Thesis, Department of Artificial Intelligence, University of Edinburgh, Scotland, 1996.