# Collaborative Design using Constraint Solving

**Claudio Lottaz* and Ian Smith****,
*** AI Lab (LIA), Computer Science Department,**
**** Institute of Structural Engineering and Mechanics**
**(ISS-IMAC), Department of Civil Engineering,**
**Swiss Federal Institute of Technology (EPFL),**
**CH-1015 Lausanne, Switzerland**

## Abstract:

This document points out some ideas why collaboration in design can lead to time consuming iterations in the design process and loss of good solutions for the task and why constraint solving may be able to help shorten the time to find a result as well as find better solutions.

# The One-Solution-at-a-Time Approach

Collaboration is the only way to cope with the complexity of large design projects. The task is divided into sub-tasks which are treated separately by several designers. However, these tasks are likely to depend on each other.

In traditional approaches to collaborative design, designers try to find only one solution for the sub-task they are responsible, although there are usually many acceptable solutions. In this process of finding a unique solution much information about alternatives and possible adaptation is lost. Designers must take decisions although they do not have the necessary information at hand or although the decision should be influenced by aspects of other sub-tasks.

After solutions for all sub-tasks have been determined, they are combined together to form a solution for the original complex design-task. Since only one solution to each sub-task is known at this point, many artificial conflicts must be solved although aims of the design may not be in conflict. This solving of conflicts, artificial or real conflicts, is performed through negotiation between the designers about alternative solutions of their sub-tasks. Such a process can iterate for a long time before it eventually ends up with a solutions for the whole task.

To cope with the negotiation in collaboration more effectively, a hierarchical structure is often used to decompose the design-task. Design-tasks on a more general level impose constraints on design-task on lower levels to make sure that an integration leads to no conflicts when combining the solutions to the sub-tasks. This avoids most of the negotiation to solve conflicts, but the design-tasks on high levels tend to constrain the design-tasks on lower levels unnecessarily. Therefore the designer, who is responsible for a task on a high level must solve sub-tasks partially to make sure that a solution for each sub-task can be found and still good solutions may be lost during such a hierarchical decomposition.

Another drawback of traditional methods becomes obvious when parts of the design context that influenced early decisions change. An objection on a political level may force the team of designers to redesign significant parts of the previous work because new conflicts arise and enforce negotiation about solutions.

# Using Design Spaces

An alternative to the above approach is to search for design spaces. Instead of determining a unique solution, a designer has the task to find all acceptable solutions. Decisions are not taken if the necessary information is not available, no arbitrary choices are made and all alternative solutions can be used in subsequent processes. Sub-tasks are solved in a more general way and therefore are more likely to be reusable in other contexts. However, representing a solution space and finding all solutions but only feasible solution might be a difficult task.

As soon as solution spaces for sub-tasks are known, they can be combined to find a solution space for the whole task. Artificial conflicts cannot arise because all acceptable solutions are used in combination. The negotiation of designers for the combination of solutions of the sub-tasks is replaced by an automatic combination of the solution spaces, which avoids the iterative process. After combining solution spaces of sub-tasks, delayed decisions can be made and information can be added or information, which relates sub-tasks can be introduced.

Since no artificial conflicts arise when the combination of solution spaces is done properly no hierarchical decomposition is necessary. Therefore the according drawbacks mentioned in the previous section are not relevant to this approach. It may be possible to find all acceptable solutions for the whole task and eventually the best solution can be chosen.

When the design context changes, the work can be adapted and almost no work is lost. The concerned sub-tasks can modify their solution spaces and since the combination is automated, new solutions for the design-task can be found efficiently although many parts of the design may be involved.

# Constraints to Define Design Spaces

During design-tasks constraints arise in a natural way. In fact, designers solve constraints satisfaction problems (CSPs) when they perform their task. The set of all solutions to a CSP which corresponds to a design task represents also the space of all feasible designs to the design problem. To define all solutions to a designer’s task it is therefore enough to find the complete corresponding CSP. Thus in collaborative design using CSPs each designer would determine the CSP which contains the task’s parameters and all constraints to be maintained without over-constraining the design.

When sub-CSPs which define solution spaces for sub-tasks have been found, relations between these sub-CSPs have to be established. The union of these relations and the sub-CSPs then represent a CSP which defines a solution space for the whole design-task.

Constraint solving algorithms can then determine explicitly this design space and make it available to designers again. These algorithms, however, should return a solution space which is complete and globally consistent, i.e. all solutions contained must not violate any constraint and all such solutions must be contained. Such algorithms are known to be time-consuming. Nevertheless the problem is feasible for CSPs with certain properties. Moreover in the situation of collaborative design it may be acceptable that the results of the combination of solution spaces of sub-tasks is not available immediately but can be explored the day after.

# Exploration of the Design Space

Since the CSP which defines a design-task usually contains continuous variables, its solution space is very likely to contain an infinite number of solutions. It is therefore impossible to enumerate all solutions. Furthermore the representation of the set of solutions as a space is not possible in an intuitive way either because the space of solutions usually has a high dimensionality. However, interactive adaptation seems to be a powerful and intuitive way of exploring a space of solutions.

As soon as the set of solutions of the CSP is found, direct interaction with the user is again possible. Adaptation of certain parameters can be asked by the user within the range of feasible values where the system keeps the constraints satisfied and performs the necessary adaptations on the other parameters involved as soon as necessary.

If the design-task has an obvious geometric representation such as architectural design or any other spatial configuration problem, the interaction on a graphical representation is very intuitive. For more abstract tasks, more abstracts ways of interaction such as faders for parameters can be found.

# Acknowledgements

The ideas presented in this document were developed within a project conducted by Gerhard Schmitt, the head of CAAD at ETH Zurich and funded by the Swiss Priority Programme in Computer Science (SPP-IF).

---

*Claudio Lottaz*
*Tue Apr 29 15:36:55 MET DST 1997*