# A COMPUTATIONAL MODEL FOR PROBLEM-DECOMPOSING STRATEGY

CHUN-HENG HO
*Graduate School of Applied Arts, National Chiao Tung University, Hsinchu,*
*Taiwan*

**Abstract.** Conventional computational models such as Soar, Act*, and Mental Models, solve problems by pattern matching. However, according to other cognitive psychology-related studies, the searching strategies employed by experts and novices in well-structured problems closely resemble each other. Restated, problem-decomposing strategies allow expert designers to perform more effectively than novices. In this study, we construct a rule-based floor-planning CAD system in Lisp to closely examine the relationship between problem-decomposing strategies and design behavior in computation. Execution results demonstrate that the larger the number of elements that the system considers implies more efficient problem-decomposing strategies.

## 1. Introduction

The interaction between cognitive psychology and artificial intelligence has received increasing interest (Simon 1981; Minsky 1985; Akin 1996; Liu 1996a). Simon’s (1981) search model, while contending that problem-solving is a symbolic process through the problem space with a series of alternative searches, has becoming an important theory in artificial intelligence (Liu 1996b).

Searching strategies employed by experts and novices in well-structured problems closely resemble each other, as evidenced by cognitive psychology-related studies (Akin 1988; Ho 1997). Furthermore, prior to idea sketches, expert designers tend to hire domain specific knowledge to decompose design problems into sub-problems (Fig. 1). The tree structure of design problem assists experts in efficiently handling different design knowledge that become entangled in various design phases (Ho 1997).

The computational model can assist / facilitate experts in representing or accounting for some phenomenon observed in the cognitive experiment (Liu 1996a). In addition, such a model can facilitate a systematic study of

CAADRIA '98 : Proceedings of The Third Conference on Computer
Aided Architectural Design Research in Asia.
eds. T. Sasada, S. Yamaguchi, M. Morozumi, A. Kaga, and R. Homma
April 22-24, 1998. Osaka University, Osaka, Japan. Pp. 415-424

empirical results by simulating the designers' mental insight mechanism (Akin 1996). In this study, we examine Ho's (1997) cognitive model of problem-decomposing strategy via simulation.

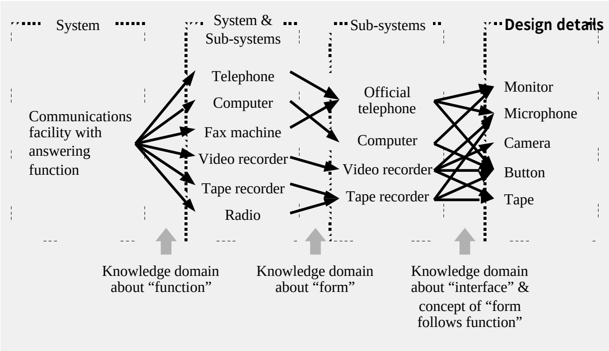


*Figure 1.* Problem-decomposing tree structure (after Ho 1997)

## 2. Problem statement and objective

In problem-solving, a previous study has indicated that (a) the more experience the designers gain implies a better ability to handle the structure of problems and (b) problem-decomposing strategies allow experts to perform more efficiently than novices (Ho 1997). Artificial intelligence related studies confer that compounding the structure of a computer program is better than other structures (Carley 1996). According to other studies, multi-agent systems significantly increase the computer systems' ability to solve problems (Aiba and Terano 1996; Pinson et al. 1997). In addition, the effectiveness of multi-agent systems lies in how to decompose the problem (Fox 1981). Meanwhile, hierarchy is an important feature of decomposing a problem's structure (Pinson et al. 1997; Steeb et al. 1988).

According to the above theory, we can infer that problem-decomposing strategies heavily influence the ability of computer systems. As anticipated, adopting problem-decomposing strategies to represent problem in a hierarchical structure would increase a system's ability to resolve a problem.

More closely examining the role in which problem-decomposing strategy plays in design procedure is highly desired. Two processes are available to construct a CAD system with the mechanism discovered in the cognitive model. Initially, exactly how problem-decomposing strategies influence the CAD system must be clarified. Second, the mechanism to control problem-decomposing strategies must be explored. This study addresses the first task.

## 3. Background

Design problems are ill-structured. Previous studies confer that experts and novices tend to resolve design problems via a working backward searching strategy (Anderson 1990; Ho 1997) (Fig. 2).

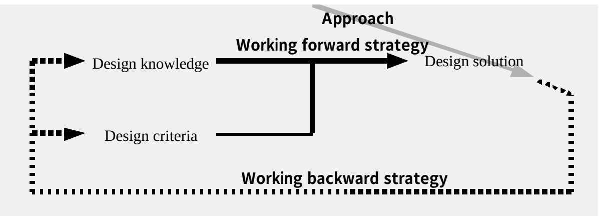


*Figure 2.* The model of searching strategy (after Ho 1997)

Furthermore, expert designers tend to decompose design problem and, then, move from the problem's initial state to goal state. Those designers also tend to search for what is still unknown in the goal state and, then, adopt the working backward strategy to solve it. The design solution occasionally creates new problems. When these new problems can be decomposed further, designers attempt to decompose them again. However, if these new problems can not be solved further, designers adopt the working backward strategy to backtrack to previous situation (Fig. 3).

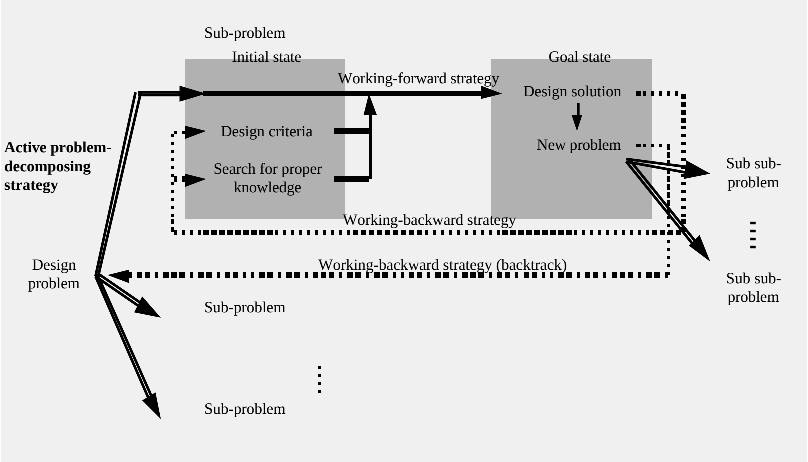


*Figure 3.* The flowchart of expert designers' problem-solving strategy (after Ho 1997)

## 4. Expert designer's problem-solving behavior

Several floor-planning methods are available, and the strategy of using *support* and *infill* system to design variant planning (Habraken 1976) is adequate to expert designer's problem-solving behavior (Ho 1997) mentioned above. Both of these studies contend that design has hierarchical nature with different levels of abstraction. Hence, in this study, we construct a CAD system which corresponds to the design behavior that integrates *support* and *infill* system (Habraken 1976), which include *zoning analysis*, *section analysis*, *basic variant*, and *sub variant*, into a cognitive study (Ho 1997). Those results indicate the following:

1. Design experts tend to analyze design task first. Consider floor planning as an example, in which the initial state of the problem includes the requirement of rooms and the support of the house;

2. Following analysis, designers apply domain specific knowledge to represent the design task. Those designers also tend to adopt problem-decomposing strategies to decompose the task to construct the tree structure of the task. In floor planning, the knowledge of *section analysis* is necessary to generate each room's minimum size. Moreover, designers perform *zoning analysis* to identify the proper position of each room in different bays and, then, employ *basic variant* to generate all possible variants of planing. Hence, the floor-planning problem is decomposed into several sub-rooms' (e.g. living room, kitchen) inner planning (Fig. 4);

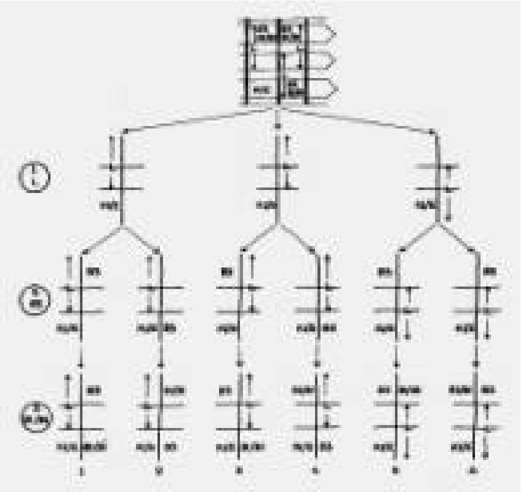


*Figure 4.* The tree structure of *basic variant* (after Habraken 1976)

3. Designers follow the tree structure of the task to resolve each isolated sub-problem. In floor planning, designers begin to identify all possible positions of furniture (e.g. table, chest); and

4. After each sub-problem is solved, designers accumulate all sub-problems together to verify the practical nature of the results and modify them if deemed necessary. In floor planning, this step belongs to *sub variant*. Designers begin considering service rooms (e.g. toilet, bathroom) to ensure the practical nature of *basic variant*.

## 5. The mechanism of CAD system

Herein, the CAD system's mechanism is divided into two phases to accomplish the design task. Phase one focuses on *basic variant*, which consists of the first and second steps mentioned in section 4. The second phase concentrates on each sub-room's inner planning, which comprises of the third and fourth steps mentioned in section 4. The floor-planning task performed herein is a dwelling unit with a depth of 1080cm and a width of 720cm (Fig. 5). The pipeline and the entrance are located in the lower-left ┘\ bay. The requirement of rooms includes the kitchen (K1), living room (L), master bedroom (B3), and two single bedrooms (B1).

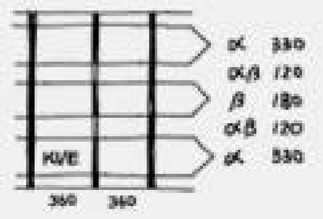


*Figure 5.* Support of dwelling unit (┘\ is the bays having direct light form the outside, ┘] is the bays do not have direct light, and ┘\┘] is the border bays)

Table 1 presents the requirement of each room's depth of the bays. Consider the living room as an example, in which it always strides by more than one bay and ends in a border bay. Hence, the living room's depth of bays is ┘\+┘\┘]+┘].

TABLE 1. The requirement of each room's depth of the bays

| Room | The depth of the bays | Possible location |
|---|---|---|
| Kitchen (K1) | ┘\(pipe)+┘\┘] | 1 |
| Living room (L) | ┘\+┘\┘]+┘] | 4 |
| Master bedroom (B3) | ┘\ | 4 |
| Single (B1) | 1/2┘\ | 8 |

Results obtained from the human cognitive study (Ho 1997) indicate that adopting the problem-decomposing strategies increases the design efficiency; meanwhile, the study does not indicate how to search efficiently. Herein, we address two different rules to generate *basic variant.* One rule is initially generated from the room having the most possibilities: two B1 could have $C_2^8$ possible locations. Two different situations are obtained: two B1 could be located in the same ┘\ bay or in different ┘\ bays. If two B1 are not in the same bay, B3 could have $C_1^2$ possible locations, and L could have $C_1^1$ possible location. However, under such a circumstance, K1 would not have adequate space. If two B1 are in the same bay, B3 could have $C_1^3$ possible locations, and L could have $C_1^2$ possible locations. In addition, K1 could have $C_1^1$ possible location. Restated, the system must search 448 nodes to search for all possible plans:

$$C_2^8 * (C_1^2 * C_1^1 + C_1^3 * C_1^2 * C_1^1) = 448$$

The other rule is initially generated from the room having the fewest possibilities: K1 could have $C_1^1$ possible location, L could have $C_1^3$ possible locations, and B3 could have $C_1^2$ possible locations; in addition, two B1 could only have $C_1^1$ possible location. Hence, the system would only need to search twelve nodes to search for all possible plans:

$$C_1^1 * C_1^3 * C_1^2 * C_1^1 = 12$$

Based on the above equation, we can conclude that generating from the room having the fewest possibilities would yield a better efficiency. Hence, in this study, the search procedure was fixed in the following order: K1, L, B3, and double B1.

This study does not address the mechanism for planning design procedure, accounting for why the procedure of problem-solving is fixed as default in the system. The initial state of CAD system’s phase one includes the requirement of rooms, support system, and measurements of the dwelling unit. The goal state of CAD system’s phase one attempts to generate all feasible floor plans. The results of this phase are each room’s location and opening. In addition, these dates are later regarded as the initial state of CAD system’s phase two. Notably, the initial state of phase two also includes rules to generate all possible positions of furniture. The goal state of this phase attempts to generate all possible inner planning of each room. Figure 6 depicts the structure of the CAD system.

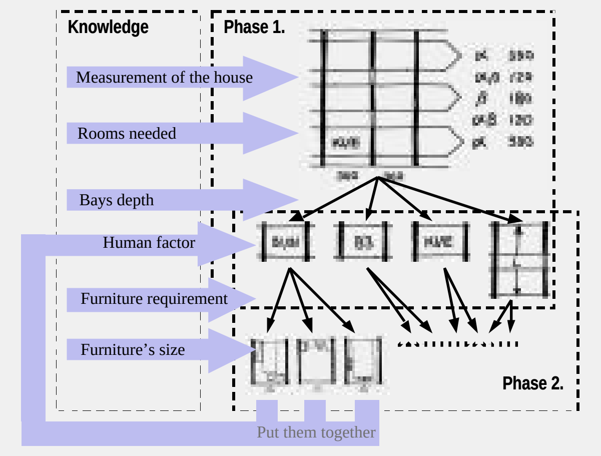
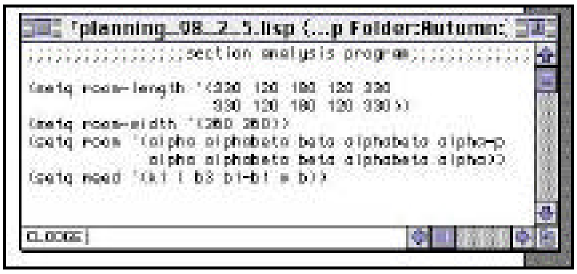


*Figure 6.* The structure of the CAD system

## 6. Execution of the CAD system

Figure 7 displays the initial state of phase one in the CAD system. The definition of rooms' requirement was also indicating the sequence of search.



```
;;;;;;;;;;;;;;;;section analysis program;;;;;;;;;;;;

(setq room-length '(330 120 180 120 330
                    330 120 180 120 330))
(setq room-width '(360 360))
(setq room '(alpha alphabeta beta alphabeta alpha-p
             alpha alphabeta beta alphabeta alpha))
(setq need '(k1 l b3 b1-b1 w b))
```

*Figure 7.* The initial state of phase one

Herein, the CAD system generates six lists indicating six variant floor plans at the end of phase one. Each room has its own data in a list. Consider the kitchen as an example, in which its data generated in the first list is "(K1 0 0 240 450 LEFT REVERSE)." From the data, we can infer that the kitchen's lower-left coordinate is (0, 0), and the upper-right coordinate is (240,450). The opening of the room is on the left hand side of the back wall when facing the windows. The default opening of the room is located in the lower wall, and the remaining data "REVERSE" indicate that after completing its inner planning in phase two, it should rotate 180 degrees.

Results obtained from phase one are put into phase two's initial state. Next, the CAD system attempts to solve each room's inner planning. Consider B1 as an example, in which the inner planning of the room could be further decomposed into furnishing the bed, table, and chest. The procedure of problem-solving proposed herein is the same as in phase one. Initially, the CAD system considers all possible locations of each furniture separately. After considering all of the furniture, the system adopts the knowledge of human factor to generate all feasible inner planning of B1 (Fig. 8).

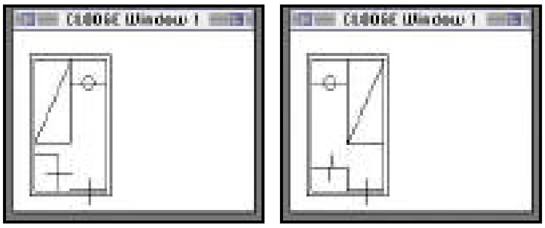
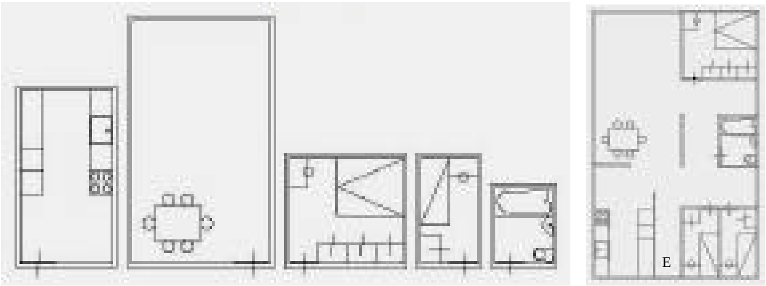


*Figure 8.* Two inner planning results of B1

Figure 9a summarizes the results obtained from the end of phase two. According to the coordinates of each room, those coordinates can be integrated into a feasible floor plan, as illustrated in Fig. 9b.



(a) (b) (processed image)

*Figure 9.* (a) The result of phase two (b) One example of floor-planning

## 7. Conclusion

Previous studies indicate that floor planning is essentially hierarchical (Alexander 1964; Akin 1978; Habraken 1994; Theo et al. 1997). In this study, we construct a rule-based floor-planning CAD system in Lisp to closely examine the relationship between problem-decomposing strategies and design behavior. Theo et al. (1997) contended that the floor-planning problem includes aspects of special system and social system. However, this study only addresses a special system having distinct rules and a two-dimensional relationship.

A CAD system with a cognitive mechanism can be constructed in two ways, providing further insight into the role in which the problem-decomposing strategy plays in design procedure. However, this study only addresses the relationship between problem-decomposing strategies and design efficiency in computation. According to our results, problem-decomposing strategies appear closely related to design efficiency. Consider B1, in which there are two possible locations for the bed, six ones for the table, and sixteen ones for the chest. If the system considers all of the furniture once, the depth of the search would be three, and the lowest level would have one hundred and ninety two ($C_1^2 * C_1^6 * C_1^{16} = 192$) nodes. While the system considers each furniture separately, the depth of search would be two, and the lowest level would have twenty four ($C_1^2 + C_1^6 + C_1^{16} = 24$) nodes. Notably, the larger the number of elements that the system should consider implies more efficient problem-decomposing strategies.

Herein, the CAD system's problem-decomposing strategies are imitated directly from the human design experts' design process, as shown in Ho's (1997) cognitive experiment. Experiments involving problem-decomposing strategies on both cognition and computation reveal the same phenomenon. However, two interesting phenomena are observed:

1. As Fig. 3 depicts, if sub-problems or sub-sub-problems are related to each other, the larger the number of possibilities that one element has implies that it should be considered at a later time; and

2. Similar to human designers (Fig. 2), the CAD system should be able to identify what a problem's goal state is by stating its own problem-decomposing strategies. Restated, the system should initially generate the tree structure of the problem to define the restriction of goal state, thereby eliminating some possible solutions.

In a future study, above phenomenon could be important mechanisms in controlling problem-decomposing strategies. While considering these mechanisms, the CAD system could more effectively resolve floor-planning problems. Hence, the CAD system could resolve such problems by its own unique problem-solving strategy and, in doing so, increase the computer's creativity (Simon 1966).

## References

Aiba, H., and T. Terano, 1996. A Computational Model for Distributed Knowledge Systems with Learning Mechanisms. In *Expert Systems With Applications*, Vol. 10, No. 3/4, pp. 417-427.

Akin, Ö. 1978. How do architects design? In *Artificial intelligence and pattern recognition in computer-aided design,* ed. J. C. Latombe, 65-104. New York: North Holland.

_____. 1988. Expertise of the architect. In *expert systems for engineering design,* ed. M. D. Rychener, 171-196. New York: Academic Press.

_____. 1996. Frames of reference in architectural design: analysing the hyperacclamation (A-h-a-!). In *Design Studies*. Vol. 17, 341-361.

Anderson, J. R. 1990. *Cognitive psychology and its implications.* 3d ed. New York: W.H. Freeman.

Alexander, C. 1964. *Notes on the synthesis of form.* Cambridge, MA: Harvard University Press.

Carley, K. M., 1996. A comparison of artificial and human organizations. In *Journal of Economic Behavior & Organization.* Vol. 31, pp. 175-191.

Fox, M. 1981. An organizational view of distributed systems. In *IEEE Transaction on Systems, Man, and Cybernetics,* 11(1), 70-80.

Habarken, N. J.1976. Variation: The Systematic Design of Supports. MIT Laboratory of Architecture and Planning.

_____. 1994. Support and infill: design and implementation. In *Issues and New Directions in Urban Housing: Proceedings of Urban Housing Towards the 21 Century,* Ming H. Wang, ed., Tainan, Taiwan, pp. 3-32.

Liu, Y. T. 1996a. Restructuring shapes in terms of emergent subshapes: a computational and cognitive model. In *Environment and planning B,* Vol. 23, 313-328.

_____. 1996b. Is design one search or two searches? A model of design thinking involving symbolism and connectionism. *Design Studies* 17, no 3.

Minsky, M. 1985. *The society of the mind.* New York: Simon & Schuster.

Pinson, S. D., J. A. Louçã, and P. Moraitis, 1997. A distributed decision support system for strategic planning. In *Decision Support Systems,* Vol. 20, pp. 35-51.

Simon, H. A. 1966. Scientific descovery and the psychology of problem solving. In *Mind and cosmos,* ed. R. Colodny, 22-40. Pittsburgh, PA: university of Pittsburgh Press.

_____. 1981. *The science of the artificial.* 2d ed. Cambridge, MA: MIT Press.

Steeb, R., S. Cammarata, F. A. Hayes-Roth, P. W. Thorndyke and R. B. Wesson, 1988. Architectures for Distributed Air-Traffic Control. In *Readings in Distributed Artificial Intelligence.* A. H. Bond and L. Gasser (Eds.). (Morgan Kaufmann)

Theo, J. M. van der Voordt, Vrielink Dick and Herman B R van Wegen, 1997. Comparative floorplan-analysis in programming and architectural design. In *Design Studies.* Vol. 18, 67-88.

Ho, C. H. 1997. A cognitive model of searching strategy For design thinking: the difference between novice and expert. *Technical report at National Chaio-Tung University.*