Designing Optimal Transportation Networks: An Expert Systems Approach

ROBERT SHIENG-I TUNG AND JERRY B. SCHNEIDER

A knowledge-based expert system (KBES) approach can be used to solve the single-mode (automobile), fixed-demand, discrete, multicriteria, equilibrium transportation network design problem. Previous work on this problem revealed that mathematical programming methods perform well on small networks with only one objective. A solution technique is needed that can be used on large networks that have multiple, conflicting criteria with different weights of relative importance. The KBES approach discussed in this paper represents a new way to solve network design problems. The development of an expert system involves three major tasks: knowledge acquisition, knowledge representation, and testing. For knowledge acquisition, a computer-aided network design and evaluation model (UFOS) was developed to explore the design space. This study investigated the problem of designing an optimal transportation network by adding and deleting capacity increments to or from any link in the network. Three weighted criteria were adopted for use in evaluating each design alternative: cost, average volume-to-capacity ratio, and average travel time. The best nondominated design is determined by a multicriteria evaluation technique called concordance analysis. The research started with a design exercise conducted by a group of students who were asked to find a series of link capacity changes that would produce a series of successively better designs. The results were carefully examined and used to generate the facts and rules that make up the knowledge base of the network design expert system (EXPERT-UFOS). It has two phases of analysis. The macrolevel analysis recommends a total budget using trade-off functions for each pair of criteria. The microlevel analysis provides advice about how to add or delete capacity on each link to avoid paradoxes. Test results show that EXPERT-UFOS found, with fewer design cycles, designs that were better than any of the 76 student designs included in the test. EXPERT-UFOS may have enough simplicity to deal with large networks. The results of this study, in which a laboratory-based knowledge acquisition method was employed successfully to generate a functional knowledge base, suggest that the KBES approach is an appropriate method for dealing with the computational complexities of network design problems.

Contemporary transportation network designers face two major problems. The first is computational complexity that has restricted the classical solution method (mathematical programming) to small problems. The second is that traditional single-objective formulations are not well suited to dealing with practical multicriteria problems. A new design process that provides a capability for dealing with a multicriteria evaluation and decision-making process and is computationally feasible for large problems is needed.

R. S.-I. Tung, Department of Public Works, 900 King County Administration Building, Seattle, Wash. 98104. J. B. Schneider, Department of Civil Engineering, University of Washington, Seattle, Wash. 98195.

The limitations of traditional mathematical programming models for dealing with network design problems have been examined thoroughly by many researchers *(1–3)*. Basically, any transportation network design problem encounters a combinatorial explosion because of the discrete nature of the attributes of the transportation network. When searching for a unique optimum solution, intensive calculation is required. Experience has shown that these models can handle only small- to medium-sized networks with currently available computers within a reasonable time. This computing requirement has limited the applicability of such techniques to small problems only.

Approximation methods (i.e., machine-based heuristics or heuristic search and man-machine interaction or interactive problem solving) have shown more promise for dealing with large problems. Heuristic search techniques use empirically derived rules (e.g., add, delete, interchange) to systematically search for near-optimal solutions. In contrast, interactive methods use man's intuitive capabilities and knowledge to guide a search of the combinatorial solution space. Both heuristic search and interactive methods can reduce the size of the search space to some extent. Heuristic search techniques, which usually use a single global heuristic, can often consistently find optimal solutions, but the computational requirements are still prohibitive for dealing with large problems *(2)*. Interactive methods use more heuristics and domain knowledge and usually can find acceptable solutions within reasonable computing times, but these techniques lack consistency because they depend heavily on human knowledge, experience, and perceptual skills.

The need for a multicriteria evaluation component has added to the complexity of the transportation network design task. Only a few researchers have used optimization methods to tackle the multicriteria network design problem *(4)*. The results indicate that such methods are generally not applicable to large problems because of computational difficulties. The results suggest again that approximate methods are likely to be most appropriate for large real-world problems.

Approximate methods that include a multicriteria evaluation component are still under development. The purpose of an approximate method is to derive robust search heuristics that can find high-quality solutions within a reduced solution space. Both heuristic and interactive methods have been developed to achieve this objective. The difference between them is the way in which the search strategy and knowledge (heuristics and facts) are generated and used. Heuristic methods often emulate optimization algorithms. They integrate a search strategy with available knowledge. Heuristics can consistently find solutions that are locally optimal. However, they have no pattern recognition capability and cannot recognize local constraints. Interactive methods have separate simulation and control functions. The control function tends to rely more on human guidance and such inputs can often produce an efficient and effective search. Such search behavior is usually called "intelligent search" and relies on man's powerful pattern recognition capabilities and domain expertise. However, this pattern recognition capability and domain expertise cannot be stored and coded in an explicit form, which makes system performance user dependent. Thus this method lacks consistency, and its reliability cannot be guaranteed.

There are no conflicts between heuristic search and interactive methods. An ideal system would use them both. Heuristics can be created by using interactive methods and then coded as machine-based algorithms. However, not every idea discovered using interactive techniques can be algorithmically defined as required by the traditional heuristic search method. More important, traditional heuristic search methods cannot flexibly handle various rules, facts, and associated domain knowledge while maintaining a user-friendly dialog with the designer. This is why the knowledge-based expert systems approach can be used to tackle this problem.

The knowledge-based expert systems (KBESs) approach has evolved from research in artificial intelligence. In contrast with traditional algorithmic methods, the expert systems approach has separated the control strategy from the knowledge base. It can flexibly handle various heuristics (or rules) and facts. Also, it is interactive and user friendly. The expert systems approach has been found to be useful in many fields *(5)*. However, most applications so far are diagnosis oriented. Only a few applications are in the transportation area *(6)*. No previous research has attempted to deal with the multicriteria equilibrium network design problem using this new approach. It is hypothesized that this new method will be useful for dealing with design problems on large networks.

KNOWLEDGE ACQUISITION

To develop an expert system, normally either a human expert or some written expertise must exist and be available for use. Unfortunately, no human experts exist who can handle the complexity of transportation network design problems. Written expertise does not exist either. However, not all expertise must come from long experience. Design expertise can also be generated by using a simulation model. In dealing with combinatorial design problems, simulation may be the only effective way to generate such expertise. To test this fundamental hypothesis, a computer-aided design and evaluation model (UFOS) was developed. Using the UFOS model, a design exercise was conducted with a group of students. The results of this design exercise provided much valuable design knowledge that was then used to develop the knowledge base for a network design expert system.

Network Simulation Model

UFOS is designed to allow a user to formulate and test a wide variety of ideas about the design of a transportation network. It has the capability of performing both fixed-demand analysis and elastic-demand analysis. In this study, only fixed-demand analyses have been used. UFOS contains four individual modules (Figure 1).

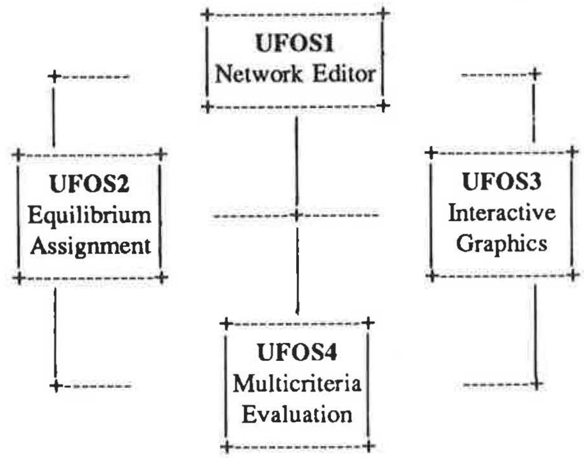

FIGURE 1 Modules of UFOS.

These four modules are linked to provide a user-friendly interactive design and evaluation environment. UFOS has built-in link attribute settings. Five roadway link types are available for conducting network design activities:

Type	*Lanes*	*Capacity (vehicles/hr)*	*Speed (mph)*	*Cost Factor ($/lane-mi)*
1	1	250	35	1
2	2	800	35	1
3	3	1,300	35	1
4	2	2,000	60	10
5	3	3,000	60	10

Types 1–3 represent arterial standards. Types 4 and 5 represent freeway standards. These different link types have different construction costs, and these costs represent one of the performance criteria used in the evaluation process.

Design Exercise

The network for the design exercise is defined by nine nodes that represent nine large zones in the eastern part of the Central Puget Sound region (Figure 2). These nodes are connected by 24 roadway links. This network forms a linear urban shape that usually generates high congestion in its central area. An evening peak origin-destination pattern with a total trip volume of 19,500 vehicles per hour represents the travel demand requirement.

This network was used as the basis for a design problem that was assigned to eight graduate students enrolled in a course on transportation and land use models. The purpose of this exercise was to search for a network design that would produce an efficient loading pattern with minimal congestion, minimal average trip times, and the lowest possible cost. Using the given travel demand pattern, each student was asked to search for an efficient roadway network design by increasing or decreasing the capacity and speed on various links. The travel behavior of each trip maker was assumed to follow the user-optimum principle. As a result, trip makers change their routes

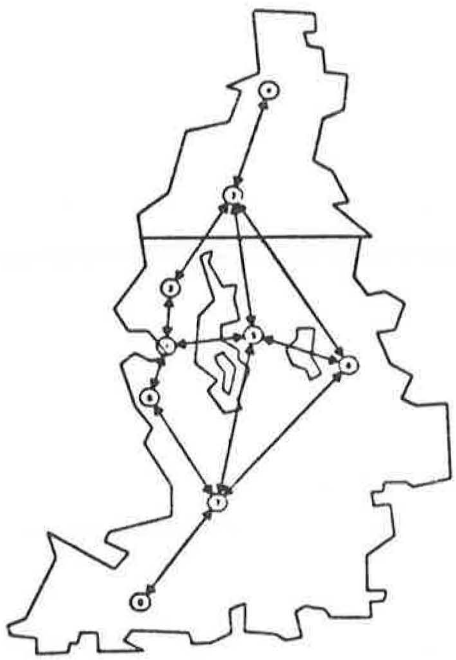

FIGURE 2 Puget Sound test network.

in response to various capacity allocations that produce different congestion patterns. The performance of each design is evaluated using three weighted criteria: total cost, average vehicle-to-capacity (V/C) ratio, and average trip time (ATT).

Theoretically, given a travel demand (origin-destination) pattern and a fixed number of link types, the possible lower-bound and upper-bound values of each design criterion can be calculated by simply setting all links at minimum capacity (Type 1) or at maximum capacity (Type 5). The bounding values for this problem are given in Table 1.

TABLE 1 BOUNDING VALUES

Criterion	Design 999 (maximum capacity)	Design 888 (minimum capacity)	Objective
Cost ($)	8,915.5 (UB)	356.62 (LB)	Less is better
V/C	0.56 (LB)	3.99 (UB)	Less is better
ATT (min)	29.76 (LB)	4795.04 (UB)	Less is better

NOTE: LB = lower bound and UB = upper bound.

The best design for this problem is one of the 5^{24} = 59,604,644,775,380,625 possible alternatives in this design space. Each will have criterion values that lie somewhere between the bounds. Note that the design with a minimum capacity roadway network has the best (lowest) cost but the worst V/C and ATT. On the other hand, the design with all links set at maximum capacity has the best performance in V/C and ATT but is worst (highest) in cost. In reality, there are many conflicts like this among performance criteria. A preferred design is a design that can satisfy most of the objectives well, that is, a best-compromise design.

The exercise started with a network in which all links had been set at minimum capacity. Each student was asked to generate at least six designs to explore the combinatorial design space. The students were asked to record their link-specific design decisions and design performance expectations. It was hoped that they would detect some cause-and-effect relationships between link-type changes and performance criteria that might become rules that could be used to build the knowledge base for an expert system.

The evaluation was a two-stage process. In the first stage, the students evaluated their design against their own design set and the two given bounding designs (Designs 888 and 999). In the second stage, the best designs from each student were aggregated and comparatively evaluated. A single weighting scheme was used throughout: Cost, 0.5; V/C, 0.25; and ATT, 0.25

Design Strategy Analysis

Given such a partially structured problem, the designer wants to devise a strategy that will produce a sequence of successively better designs. Such strategies can be developed from the knowledge and experience of the designer (expert). Usually, the construction of a design strategy involves using both "deep knowledge" or "hard information" (e.g., an explicit model with assumptions, relationships, and constraints) and "surface knowledge" and "soft information" (e.g., intuitive constructs). Moreover, the designer's ability to implement a certain strategy depends on the ability to interpret the hard information in the results and effectively integrate it with the soft information. Usually, for engineering-oriented design problems, a clear understanding of the deep knowledge aspects of the problem is necessary. Thus a domain expert can usually perform better than a novice designer. However, given the right computer-based design aids, many novice designers can reach high performance levels rather quickly. This design exercise allows the student to explore the performance of certain design concepts quickly and easily. By observing their progress, it is possible to learn how effective various design strategies are in dealing with a partially structured problem.

Two basic types of design strategies were used: incremental exploration and logic based. Each strategy has its strengths and weaknesses. Two criteria can be used to evaluate these strategies in terms of their effectiveness and efficiency in producing high-performance designs. First, how consistent was the strategy in finding successively better designs? Second, how efficient was the search strategy used to find an optimum design? The following two examples are used to illustrate these points.

- Incremental exploration design strategies: Most students used an incremental exploration strategy. They simply added some capacity to the links with the highest V/C ratios during each design session. They kept on driving congestion levels down while keeping costs as low as possible. This type of strategy was conservative but did produce better designs easily. The experiences of Student A are typical of this type of design behavior. Table 2 gives his experience for the six designs. The total capacity trends clearly show the incremental changes he made. Only a few of the most congested links were upgraded to the next level of capacity during each design cycle. Table 3 gives the overall design performance of the six designs and Table 4 gives the ranking results produced by a series of multicriteria evaluations. As the ranking results show, this student found a series of designs that were, except for the last, successively better. In addition, his best design (005) was the

TABLE 2 DESIGN EXPERIENCE OF STUDENT A

Link No.	888 Cap	888 V/C	001 Cap	001 V/C	002 Cap	002 V/C	003 Cap	003 V/C	004 Cap	004 V/C	005 Cap	005 V/C	006 Cap	006 V/C
1	250	6.2	800	2.4	800	3.8	800	3.4	1300	1.9	1300	1.6	3000	0.7
2	250	2.4	250	6.7	800	2.4	800	2.4	800	2.2	1300	1.7	2000	1.0
3	250	6.8	800	4.1	800	4.0	800	3.5	1300	2.4	2000	1.8	3000	0.9
4	250	1.6	250	6.6	1300	1.2	800	1.6	800	1.6	800	1.8	800	1.3
5	250	6.7	800	4.5	800	4.1	800	3.8	1300	2.5	2000	2.1	3000	1.5
6	250	2.9	250	4.1	250	4.3	800	0.4	800	0.9	800	1.5	800	1.8
7	250	5.0	250	5.2	800	2.9	800	2.4	800	2.0	800	1.4	800	1.4
8	250	7.7	800	1.8	800	1.5	800	1.3	800	1.2	800	1.5	800	1.2
9	250	3.7	250	5.5	800	2.5	800	2.5	800	2.5	1300	1.5	1300	1.5
10	250	3.7	800	6.9	800	2.5	800	2.5	1300	1.5	1300	1.5	1300	1.5
11	250	5.6	250	2.6	800	3.1	800	3.0	1300	1.7	1300	1.7	3000	0.9
12	250	2.2	250	5.1	250	1.8	800	0.4	800	0.7	800	1.1	800	1.1
13	250	4.6	250	4.1	800	3.2	800	2.8	1300	2.0	1300	1.7	2000	1.0
14	250	0.0	250	2.0	250	0.0	800	0.0	800	0.0	250	0.0	250	0.0
15	250	4.0	250	6.2	800	3.3	800	2.4	800	2.2	1300	1.6	2000	1.1
16	250	1.1	250	1.8	250	1.8	800	0.4	800	0.2	800	0.6	800	0.3
17	250	3.7	250	5.5	250	5.1	800	2.2	800	1.9	800	1.4	800	0.9
18	250	2.0	250	3.8	250	3.1	800	1.0	800	0.8	800	0.4	800	0.3
19	250	1.4	250	2.7	250	4.5	800	1.1	800	1.2	800	0.4	800	0.4
20	250	3.1	250	4.6	250	3.3	800	1.9	800	1.6	800	1.3	800	1.0
21	250	5.1	250	6.3	800	2.7	800	2.8	1300	1.8	1300	1.7	3000	0.9
22	250	4.7	250	4.7	250	5.6	800	2.5	800	2.2	1300	1.6	3000	1.0
23	250	3.7	250	6.0	800	2.5	800	2.5	800	2.5	1300	1.5	1300	1.5
24	250	6.9	800	4.3	800	4.4	800	4.4	1300	2.7	2000	1.8	3000	1.2
Tot. Cap.	6000		9300		14750		19200		23200		27250		39150	
Avg. V/C		3.9		4.5		3.1		2.1		1.7		1.4		1.0

second best in the class. However, an incremental strategy may cause paradoxical results. As the data in Table 3 indicate, Design 001 has a higher capacity than the lower-bound design (888), but it also has higher V/C and ATT. Paradoxes can cause problems in design and must be avoided.

- Logic-based design strategy: Some students chose to try to develop more sophisticated design strategies. They wished to use mathematical principles to tackle the problem. To do this, they needed to develop a deeper understanding of various aspects of the design problem such as the design-to-performance and node-to-link relationships. For example, they knew that the low-cost designs would be preferred because the weight of "cost" is greatest (0.5). They determined that a link should have the highest possible capacity if it was congested in Design 999, in which all links had the maximum capacity setting. Similarly, they knew a link should be given the lowest capacity setting if it had no loading even in Design 888, in which all links had the maximum capacity setting. They devised efficient strategies for adding and deleting capacity. They knew that a network with an average V/C ratio of around 1.0 could produce the best combination of values for the criteria. Thus they simply added capacity in rough proportion to the V/C ratios. However, even with this knowledge, they could not always predict the results correctly. Student B used this type of design strategy. Table 5 gives his design results for the six

TABLE 3 DESIGN CRITERIA MEASURES OF DESIGNS OF STUDENT A

Design	Cost ($)	V/C	ATT (min)
888	356.62	3.94	4795.04
999	8,915.50	0.56	29.76
002	565.31	4.48	4881.95
002	804.51	3.06	1404.37
003	1,141.18	2.13	659.69
004	1,360.78	1.68	196.10
005	1,584.31	1.37	85.87
006	3,351.41	1.01	47.88

NOTE: 888 and 999 are bounding designs.

TABLE 4 MULTICRITERIA EVALUATION RANKS OF DESIGNS OF STUDENT A

	Design						
Run	888	999	002	003	004	005	006
1	2*	2*	1				
2	3*	3*	2	1			
3	4*	4*	3	2	1		
4	5*	5*	4	3	2	1	
5	6*	6*	5	3	2	1	4

NOTE: 1 is best. * = not above average.

TABLE 5 DESIGN EXPERIENCE OF STUDENT B

Link No	888 Cap	888 V/C	001 Cap	001 V/C	002 Cap	002 V/C	003 Cap	003 V/C	004 Cap	004 V/C	005 Cap	005 V/C	006 Cap	006 V/C
1	250	6.2	2000	1.5	2000	1.3	2000	1.2	2000	1.2	2000	1.6	2000	1.4
2	250	2.4	1300	1.4	1300	1.2	1300	1.1	1300	1.1	1300	1.7	2000	1.0
3	250	6.8	1300	1.4	2000	1.2	2000	1.4	2000	1.4	2000	1.8	2000	1.2
4	250	1.6	800	1.3	800	2.0	800	1.7	800	1.7	1300	1.8	1300	0.9
5	250	6.7	3000	1.5	3000	1.5	3000	1.4	3000	1.4	3000	2.1	3000	1.5
6	250	2.9	800	1.6	1300	1.2	1300	1.4	1300	1.3	1300	1.5	1300	1.2
7	250	5.0	1300	1.5	1300	1.3	800	1.8	800	1.7	1300	1.4	1300	1.3
8	250	7.7	800	1.0	250	2.1	250	1.7	250	1.6	800	1.5	800	1.3
9	250	3.7	1300	1.5	1300	1.5	1300	1.5	2000	1.0	1300	1.5	2000	1.0
10	250	3.7	1300	1.5	1300	1.5	1300	1.5	1300	1.5	1300	1.5	2000	1.0
11	250	5.6	2000	1.3	2000	1.4	2000	1.4	2000	1.5	2000	1.7	2000	1.5
12	250	2.2	250	0.2	250	1.4	250	1.4	250	1.5	250	1.1	250	1.1
13	250	4.6	2000	1.0	2000	1.1	2000	1.1	2000	1.1	1300	1.7	1300	1.0
14	250	0.0	250	0.0	250	0.0	250	0.8	250	0.9	250	0.0	250	0.0
15	250	4.0	1300	1.4	2000	1.1	2000	1.4	3000	1.0	2000	1.6	2000	1.3
16	250	1.1	250	1.4	250	1.5	250	1.4	250	1.4	250	0.6	800	0.6
17	250	3.7	800	1.1	250	1.8	800	1.1	800	1.2	800	1.4	800	1.1
18	250	2.0	250	0.3	250	0.7	250	1.4	250	1.5	250	0.4	250	0.2
19	250	1.4	250	0.9	250	1.7	250	1.3	250	1.3	250	0.4	250	0.0
20	250	3.1	800	1.5	800	1.0	250	1.5	250	1.6	1300	1.3	800	1.2
21	250	5.1	2000	1.3	2000	1.4	3000	1.0	2000	1.5	1300	1.7	2000	1.3
22	250	4.7	2000	1.4	2000	1.4	2000	1.3	2000	1.3	2000	1.6	2000	1.3
23	250	3.7	1300	1.5	2000	1.0	1300	1.5	800	2.5	1300	1.5	1300	1.5
24	250	6.9	3000	1.2	3000	1.2	3000	1.2	3000	1.2	2000	1.8	2000	1.8
Tot. Cap.	6000		30350		31850		31650		31850		30850		33700	
Avg. V/C		3.9		1.2		1.3		1.4		1.4		1.3		1.1

designs. He made dramatic changes on the first design and only marginal changes on the rest of the designs. Table 6 gives the performing measures of these six designs, and Table 7 gives their multicriteria evaluation rankings.

In general, the logic-based strategies produced good first designs but did not always produce high-performance designs quickly. As the data in Table 6 indicate, Student B finally reached his best design in the sixth design session and it was the best in the class. He had a good initial design (001), which was not very different from the best design. However, he did have problems making consistent progress (Table 7). The reason for this is that when a design is close to the optimum it is more difficult to predict the flow pattern that will be produced by link capacity changes. Actually, a network designer can never precisely predict the flow pattern that will be computed by the equilibrium assignment algorithm, but it is often possible to do so in general terms.

Ideally, a good design strategy should contain elements from both of these approaches. An incremental exploration strategy may be better for large problems because it is too difficult to do a sophisticated analysis before the first design action is taken. However, as much logic as possible should be used to reduce the size of the search space and to avoid paradoxes that usually occur when an incremental exploration strategy is used.

TABLE 6 DESIGN CRITERIA MEASURES OF DESIGNS OF STUDENT B

Design	Cost ($)	V/C	ATT (min)
888	356.62	3.94	4795.04
999	8,915.50	0.56	29.76
001	2,112.38	1.20	56.57
002	2,225.08	1.31	53.15
003	2,345.43	1.36	55.43
004	2,324.63	1.38	66.66
005	1,873.64	1.25	61.73
006	2,090.21	1.07	49.79

NOTE: 888 and 999 are bounding designs.

TABLE 7 MULTICRITERIA EVALUATION RANKS OF DESIGNS OF STUDENT B

	Design							
Run	888	999	001	002	003	004	005	006
1	2*	2*	1					
2	2*	2*	1	2				
3	3*	3*	1	2	3*			
4	4*	4*	1	2	3*	5*		
5	5*	5*	2	3	4	5*	1	
6	6*	6*	3	4	5*	7*	2	1

NOTE: 1 = best. * = not above average.

KNOWLEDGE REPRESENTATION

The design strategies identified in the design exercise are valuable for building an expert system. First, the incremental exploration strategy indicates that an incremental simulation approach is an effective way to deal with large design problems. A series of small improvements may be the easiest way to approach a good design. Second, the logic-based design strategy indicates that use of some deep knowledge can produce rapid progress toward the best design. This deep knowledge can be represented as facts and rules and used to construct the knowledge base for an expert system.

Expert System Shell

An expert system shell is a convenient tool for developing application-oriented expert systems. The PC-based expert system shell M.1 *(7)* was used in this study to provide a user-friendly interface and the capability to link with external functions. The extensive number crunching of EXPERT-UFOS was handled by using C-based external functions. By using these external functions, it is possible to maintain the transparency of the knowledge base while having the computational efficiency of a C-program. The relationship between the knowledge base and external functions is shown in Figure 3.

Facts and Rules in EXPERT-UFOS

The knowledge base of EXPERT-UFOS consists of various facts and rules. Facts are link specific and are represented by object-attribute-value (O-A-V) triplets. Objects are the specific links of a network. Attributes describe aspects of the network-related deep knowledge and form the basis for making effective design designs. The term "value" specifies the particular nature of an attribute for a given object. The attributes are further divided into static and dynamic categories. Attributes that are fixed during the entire design process are static attributes (e.g., type, criticality, and length). All other attributes are dynamic attributes (e.g., V/C, add-priority, or delete-priority). The dynamic attributes have to be recalculated during each consultation with EXPERT-UFOS. The full structure of the O-A-V framework is given in Table 8.

Rules are used to implement the heuristics for finding a series of successively better designs. There are four basic types of rules in EXPERT-UFOS:

1. Control rules: These rules control the main flow of a consultation. For instance, the following rule is used to determine if a best nondominated solution has been found:

```
if    no_more_improvement(1) is true
then  best_nondominated_solution is true.

if    evaluation_rank(CYCLE_N) = X
and   X > 1
then  no_more_improvement(CYCLE_N) is true.

if    CYCLE_M + 1 = CYCLE_N
then  nextcycle to CYCLE_M = CYCLE_N.

if    nextcycle to CYCLE_M = CYCLE_N
and   no_more_improvement(CYCLE_N) is true
then  no_more_improvement(CYCLE_M) is true.
```

The following rule controls the process of equilibrium assignment:

```
+-----------------+          +---------------+          +-----------------+
|   Knowledge     |          |   External    |          |                 |
|     Base        |  <---->  |   Functions   |  <---->  | External Files  |
| (Rules + Facts) |          |   (C-Codes)   |          |                 |
+-----------------+          +---------------+          +-----------------+
```

FIGURE 3 Relation between knowledge base and external functions.

TABLE 8 LINK O-A-V TRIPLET REPRESENTATION

Object		Attribute	Possible Values
Link #	Static	Type	(production, attraction, production-attraction, single-production, single-attraction, buffer)
	Static	Criticality	volume of free-flow loadings
	Static	Length	miles
	Static	Capacity	(250, 800, 1300, 2000 or 3000 vehicles per hour)
	Dynamic	V/C	real number as derived
	Dynamic	Add-priority	integer rank number
	Dynamic	Delete-priority	integer rank number

```
if    initial_equilibrium_assignment is true
and   do_shortest_path(ITER_N) is true
and   do_loading_assign(ITER_N) is true
and   do_fibonacci_search(ITER_N) is true
and   do_convergence_check(ITER_N) is true
then  equilibrium_ assignment(ITER_N) is true.
```

2. External rules: These rules activate the external functions for doing extensive calculations. For example, the following rules check the convergence of the assignment process:

```
if    no_of_links = NARC
and   ITER_N = ITER
and   external(eq_converg_check, [NARC,
      ITER]) = FLOW_STD
then  eq_convergence_check(ITER_N) = FLOW_STD.

if    eq_convergence_check(ITER_N) = FLOW_STD
and   ITER_N > 1
and   1.0 - FLOW_STD = X
and   convergence_criterion = Y
and   X > Y
then  do_convergence_check(ITER_N) is true.
```

3. Macrorules: These rules are used to determine the design goal and budget limit. For example, the following rules are used to determine the design goal:

```
if    weight(cost) = X
and   X > 0.5
then  design_goal = decrease_capacity.

if    weight(cost) = X
and   X <= 0.5
then  design_goal = increase_capacity.
```

The budget limit is determined by the following two macrorules and an external rule:

```
if    design_goal = decrease_capacity
then  search_type = downward.

if    design_goal = increase_capacity
then  search type = upward.

if    search_type = SEARCH
and   cost(CYCLE_N) = COST
and   vc(CYCLE_N) = VC
and   att(CYCLE_N) = ATT
and   cost_weight = CW
and   vc_weight = VCW
and   att_weight = ATTW
and   external(budget, [SEARCH,COST,VC,ATT,
      CW,VCW,ATTW]) = BUDGET
then  budget_check(CYCLE_N) = BUDGET.
```

4. Microrules: These rules are used to determine which links should have more or less capacity and how much. Such decision making is based on several facts, such as priority, higher-capacity, cost-factor, link-length, and criticality. For example, the following rules determine whether capacity of a link should be increased to a higher level:

```
if    design_goal = increase_capacity
and   capacity_check(LINK_N) is true
and   budget_check(LINK_N) is true
and   critical_check(LINK_N) is true
then  add_action(LINK_N) is true.
```

To verify that capacity_check(LINK_N) is true, the following two rules are used:

```
if    not(no_more_capacity(LINK_N))
then  capacity_check(LINK_N) is true.

if    design_goal = increase_capacity
and   capacity(LINK_N) = CA
and   CA>= 3000
then  no_more_capacity(LINK_N) is true.
```

The following rule is used to verify that the budget_check(LINK_N) is true to ensure that the capacity increase will not cause the budget limit to be exceeded:

```
if    link_cost(LINK_N) = X
and   budget(LINK_N) = Y
and   X<= Y
then  budget_check(LINK_N) is true.
```

The link cost is calculated by the following rule:

```
if    design_goal = increase_capacity
and   capacity_check(LINK_N)
and   capacity(LINK_N) = CA
and   higher_capacity(CA) = HC
and   cost_factor(CA,HC) = CF
and   length(LINK_N) = LN
and   CF * LN = Z
then  link_cost(LINK_N) = Z.
```

The higher-capacity and cost-factor data are provided by reference to the following facts:

```
higher_capacity(250) = 800.
higher_capacity(800) = 1300.
higher_capacity(1300) = 2000.
higher_capacity(2000) = 3000.

cost_factor(250, 800) = 2.2.
cost_factor(800, 1300) = 2.0.
cost_factor(1300, 2000) = 14.8.
cost_factor(2000, 3000) = 25.0
```

TESTING AND CONCLUSIONS

EXPERT-UFOS was tested by giving it the same design problem as was given to the students. The result was that EXPERT-UFOS needed only four cycles to conclude the best nondominated design (003) given in Table 9. Design 003 is substantially better than the three best student designs. The better rank (Table 10) indicates that Design 003 is the best known nondominated design for the given weighting scheme. This

TABLE 9 CRITERIA MEASURES OF EXPERT-UFOS DESIGN AND BEST STUDENT DESIGNS

Design	Cost ($)	V/C	ATT (min)
888	365.62	3.94	4795.04
999	8,915.50	0.56	29.76
003	1,525.76	1.33	126.07
S01	2,090.21	1.07	49.79
S02	1,584.31	1.37	85.87
S03	1,544.42	1.43	149.09

NOTE: S01 to S03 are the best student designs.

TABLE 10 MULTICRITERIA EVALUATION RANKS OF THE DESIGN CONTEST

	Concordance		Discordance			
Design	Dominance Value	Rank	Dominance Value	Rank	Average Rank	Final Rank
888	0	2	0.97	4	3.0	3*
999	0	2	0.97	4	3.0	3*
003	1.5	1	−0.51	1	1.0	1
S01	0	2	−0.47	3	2.5	2
S02	−0.5	3	−0.49	2	2.5	2*
S03	−1.0	4	−0.47	3	3.5	4*

NOTE: * = not above average.

result indicates that EXPERT-UFOS did find a solution that is better than all of the 79 designs generated by the students. EXPERT-UFOS performed well in its first test. It needed only four cycles to find its best design. Because an equilibrium assignment problem must be solved in each cycle, the fewer cycles needed, the greater the efficiency of the method. There can be no absolute measure of efficiency because different machines have different computational speeds. Because the equilibrium assignment algorithm is a standard procedure for finding an optimal flow pattern, the less execution time needed to solve the assignment, the more efficient the method. EXPERT-UFOS quickly reduced the search space to a minimum. Part of the success of EXPERT-UFOS is the result of its successful prevention of the design paradox. As the results show, increasing cost does reduce values of V/C and ATT. As long as EXPERT-UFOS can avoid paradoxes, the system should be able to find a high-performance design quickly using a cyclic approach.

Because good results were obtained in a few cycles, EXPERT-UFOS is cost-effective. However, this does not mean that EXPERT-UFOS will always be superior to interactive methods. The design exercise discussed was conducted by novice designers under loosely defined conditions. Trade-off functions were unknown, and all of the students were doing this design task for the first time. Given the trade-off functions, an experienced designer might produce a design the performance of which was the same as or even better than that of Design 003. However, this might be true only for a small network. It is unlikely that a human could deal effectively with a large network using only an intuitively guided approach. On the other hand, because EXPERT-UFOS can efficiently reduce the search space and effectively avoid paradoxes, it offers a reasonable approach for dealing with large problems. Tests of this type are currently under way.

REFERENCES

1. L. LeBlanc et al. An Efficient Approach to Solving the Road Network Equilibrium Traffic Assignment Problem. *Transportation Research*, Vol. 9, 1975, pp. 309–318.
2. R. Wong. Transportation Network Research: Algorithmic and Computational Questions. *Transportation Research*, Vol. 19A, 1985. pp. 436–438.
3. T. Friesz. Transportation Network Equilibrium, Design and Aggregation: Key Developments and Research Opportunities. *Transportation Research*, Vol. 19A, 1985, pp. 413–427.
4. T. Friesz and P. Harker. Multicriteria Spatial Price Equilibrium Network Design: Theory and Computational Results. *Transportation Research*, Vol. 17B, 1983, pp. 411–426.
5. F. Hayes-Roth et al., eds. *Building Expert Systems*. Addison-Wesley, Inc., Reading, Mass., 1983.
6. C. Hendrickson. Comment on Research in Transportation System Characteristics and Operations. *Transportation Research*, Vol. 19A, 1985, pp. 367–369.
7. *M.1 Reference Manual: Updated Edition for Software Version 1.3*. Teknowledge, Inc., Palo Alto, Calif., 1985.