An Optimization Methodology to Investigate Operator Impact on Quality of Service

Ling Rothrock, Jose Ventura, Sung Soon Park
The Pennsylvania State University
University Park, Pennsylvania
lrothroc@psu.edu, jav1@email.psu.edu, sup115@psu.edu

ABSTRACT

The 21st century battle space consists of vastly complex and dynamic environments to which human decision makers must adapt in order to achieve their objective. Moreover, the recent trend of manpower reductions necessitates the implementation of highly automated systems. Therefore, a decision maker must not only function under conditions of target ambiguity, time pressure, and information overload, he/she must also be able to effectively operate somewhat autonomous pieces of machinery. This paper presents the work in progress of a human decision making model for dynamic, time stressed tasks. We build upon a theoretical model of behavioral decision making generally used in judgment analyses in static contexts. For this paper, we focus on the mathematical formulation of the problem and some solution techniques.

Our model of the human decision maker seeks to account for noncompensatory (i.e., rule-based) behavior. In general, we present a model that uses inductive inference principles to generate rules in disjunctive normal form that are consistent with human decisions. More specifically, we formulate the problem as a multi-objective linear programming (MOLP) problem. We provide some sensitivity analyses of exhaustive solutions to simple decision problems and propose the use of heuristic algorithms for more complex problems.

ABOUT THE AUTHORS

Ling Rothrock is an assistant professor at the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering. He received his Ph.D. in Industrial Engineering from Georgia Institute of Technology in 1995 and served as an officer in the U.S. Army until 2000. His research areas include human-machine performance modeling and human-in-the-loop simulations.

Jose Ventura is a professor at the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering. Throughout his career, Dr. Ventura has developed a number of optimization algorithms to solve large-scale network equilibrium models, which have applications in telecommunication systems design, congestion analysis in traffic networks, and planning of production-distribution systems. He is the co-P.I. of a Pentagon-sponsored logistics project which objective is to develop a Windows-based application to provide Army planners with an "about right" solution to various mobility problems within the span of 15 minutes. Dr. Ventura has published more than fifty refereed journal papers. He is an associate editor of *IIE Transactions* and *Journal of Manufacturing Systems*, and past editor of *IIE Computer and Information Systems Newsletter*. Dr. Ventura is member of IIE and INFORMS.

Sungsoon Park is a graduate Ph.D. student at the Harold and Inge Marcus Department of Industrial and Manufacturing Engineering. His current focus is on the analysis and modeling of human decision making under dynamic changed environment. At Arizona State University, he was interested in analysis and modeling of discrete event scheduling and inventory control simulation area of operations research. He has a master degree of Industrial Engineering at Arizona State University.

An Optimization Methodology to Investigate Operator Impact on Quality of Service

Ling Rothrock, Jose Ventura, Sung Soon Park
The Pennsylvania State University
University Park, Pennsylvania
lrothroc@psu.edu, jav1@email.psu.edu, sup115@psu.edu

INTRODUCTION TO DECISION MAKING AND RELATED RESEARCH

In 1944, von Neumann and Morgenstern published the seminal book *Theory of Games and Economic Behavior* that set the standard for decision making research. The work introduced the von Neumann-Morgenstern axioms that specified a normative model of human choice behavior which favors the maximum expected utility. Other researchers sought to broaden the notion of utility maximization to subjective preferences (Savage, 1972) and multiple attributes (Edwards & Newman, 1982).

The standard soon came under attack, however, as Simon (Simon, 1955) (Simon, 1959) argued that humans have cognitive limitations and are incapable of implementing utility theory in any form. In fact, he proposed the notion of *satisficing* – of seeking satisfactory solutions and not optimal ones. Moreover, he also proposed the notion of *bounded rationality* to stipulate that humans intelligently use limited resources to make decisions.

A line of research, based on the work of Brunswik, developed near the time Simon presented his concepts of satisficing and bounded rationality. Brunswik labeled his approach *probabilistic functionalism* (Brunswik, 1955a) (Brunswik, 1955b) to emphasize the relationship between a human decision maker and his environment – a relationship which is mediated through the lens of probabilistic variables in the environment. A corollary to probabilistic functionalism is the concept of *representative design*. Brunswik argued that only through the study of behavior under naturally occurring (i.e., representative) conditions can the mediation between the criterion and the decision maker be observed. Perhaps more importantly, Brunswik's interest is shared by other investigators who have researched contingent decision making behavior (Payne, 1976), (Payne, Bettman, & Johnson, 1993), (Wright, 1974), (Maule, 1997). In a summary of previous findings, Ford (Ford, Schmitt, Schechtman, Hults, & Doherty, 1989) agreed that time pressure induces a switch of decision strategy from compensatory (strategies where the organizing principle is weighted averaging) to noncompensatory (strategies characterized by rules of logic such as conjunctive and disjunctive) decision rules. Moreover, Maule (Maule, 1997), (Maule & Hockey, 1993) argues that time pressure must be appraised in context of an individual's ability to cope with time demands.

MOTIVATION FOR THE RESEARCH

When formulating a framework for investigating decision making in military systems, we are guided by six principles inferred from the review of the literature:

1. Humans are indispensable elements of any military command-and-control system;
2. Formal utility theory-based accounts of human decision making are inadequate to characterize actual decision making behavior;
3. The study of command-and-control behavior must take place in a representative task;
4. Two distinct modes of cognitive processing exist in decision making – compensatory and noncompensatory;
5. Time pressure and workload induce noncompensatory decision strategies; and
6. Effective models of human decision making in a representative task will provide insight to the design and planning of command-and-control systems.

Recent interest in Simon's bounded rationality (Simon, 1955) has generated empirical data that support noncompensatory models. Gigerenzer and his colleagues (Gigerenzer & Goldstein, 1996), (Gigerenzer, Hoffrage, & Kleinbolting, 1991) argue that, consistent with Simon's (Simon, 1959) concept of satisficing and bounded rationality, humans utilize a family of heuristics that are simple yet powerful. Their research, however, has been focused on the comparison of the efficiency and parsimony of their heuristics against linear models. While the heuristics were generated based on observation of decision making in realistic settings, no technique has been reported to automatically generate the rules (Gigerenzer & Selten, 2001).

A COMPUTATIONAL DECISION MAKING FRAMEWORK

The proposed framework seeks to construct a noncompensatory representation of decision making using a mathematical model of a human controller. The formulation builds upon on existing work begun by one of the authors (Rothrock, 1999), (Rothrock & Kirlik).

As observed by the 18th century philosopher David Hume, any knowledge derived from induction cannot, in principle, be taken as certain. While investigators have advanced the field of causality (Pearl, 2000), the preconditions of causal calculus make an application toward dynamic environments, where truth regarding the world changes often, an unrealistic undertaking. Therefore, the purpose of induction in the proposed research is to generate plausible hypotheses relevant to a person's goals and consistent with his/her actions – admittedly a weaker interpretation than Pearl's Causal Modeling Framework (Pearl, 2000, p. 43). This interpretation is drawn from machine learning literature (Michalski, 1983), (Quinlan, 1986), (Holland, Holyoak, Nisbett, & Thagard, 1986), and serves as the basis for the noncompensatory form of the decision model.

A review of machine learning methods (Michalski, 1983), (Quinlan, 1986), (DeJong, Spears, & Gordon, 1993), (Vafaie & DeJong, 1994) showed that genetic algorithms (GAs) tend to be more robust in concept learning applications partly because search is done on the encoding of the genetic strings, not the strings themselves (Goldberg, 1989), (Chen, Shankaranarayanan, She, & Iyer, 1998), (Greene & Smith, 1993). In the formulation of the inductive learning problem, the investigators utilize genetic strings as the means to represent noncompensatory strategies because the evaluation of the encodings themselves gives interpretation to human behavioral patterns.

The noncompensatory decision model proposed in this research is an extension of work on the Genetics-based Lens Model (GBLM) (Rothrock & Kirlik). Each potential decision making strategy is represented in the model as a set of rules. To illustrate the form of a rule set, consider a conjunctive rule as a condition-action rule with N statements where:

IF (statement 1) $\cap$ (statement 2) $\cap$ ... $\cap$ (statement N) THEN (consequence statement)

To be consistent with the findings from Gigerenzer and Goldstein (Gigerenzer & Goldstein, 1996), the representation of each rule set in the model is made to conform to disjunctive normal form (DNF). Therefore, by preserving the order of antecedent and consequent statements, we remove the conditionals so that the conjunctive rule becomes:

(statement 1) $\cap$ (statement 2) $\cap$ ... $\cap$ (statement N) $\cap$ (consequence statement)

We can now represent each rule set in the population as a disjunction of conjunctive rules in the following form:

(statement I) $\cup$ (statement II) $\cup$ ... $\cup$ (statement M) $\cup$ (consequence statement)

where each disjunctive statement (I, II, etc.) consists of a conjunctive rule. Hence, the model results potentially can reflect not only fast and frugal heuristics, but also any logical strategy consisting of AND, OR, or NOT operators (Mendelson, 1997).

An example of a simple judgment domain will be used to illustrate the concepts underlying the induction approach and to clarify implementation details. Consider the case of a military pilot who is flying near a military reservation. The pilot sees two aircraft during the course of his flight, and makes judgments of their identity on the basis of two cues, speed and altitude. The aircraft characteristics and corresponding pilot judgments are shown in Table 1.

TABLE 1. Aircraft Characteristics and Pilot Judgments for Sample Domain

SPEED	ALTITUDE	JUDGMENT
Fast	Low	Fighter Aircraft
Fast	High	Fighter Aircraft

This judgment data will be used to demonstrate components of the inductive inference model and also the binary representation of candidate rule sets. Using the coding scheme where 1=fast, 0=slow, 1=high, 0=low, 1=fighter aircraft, and 0=transport aircraft, each of the judgments in Table 1 can be converted into the exemplars (i.e., instances of human judgment) shown in Table 2.

TABLE 2. Exemplar Representation for Sample

Exemplar No.	Characteristics Represented	Exemplar Representation
1	Fast, Low, Fighter	101
2	Fast, High, Fighter	111

For example, Exemplar No. 1 represents the following operator judgment:

(Speed = Fast) $\cap$ (Altitude = Low) $\cap$ (Judgment = Fighter Aircraft)

Each rule within the rule set is a similarity template – or schema (Holland, 1975), (Goldberg, 1989) – and is covered by the ternary alphabet {0,1,#} where "#" is a match-all character. Note that for simplicity's sake, the consequence statement is represented as part of the conjunctive statement. In addition, for the sake of illustration, consider the rule sets (i.e., potential strategies) shown in Table 3 as the population for the sample domain.

TABLE 3. Sample Rule Sets

Rule Set No.	Rule Set Representation
1	1#10#0
2	1#1

For example, Rule Set No. 1 matches both of the exemplar vectors in Table 2, and represents the following disjunctive rule set:

(Speed = Fast) ∩ (Altitude = anything) ∩ (Judgment = Fighter Aircraft)

OR

(Speed = Slow) ∩ (Altitude = anything) ∩ (Judgment = Transport Aircraft)

To reflect bounded rationality, the model uses the "Pittsburgh" string representation (DeJong et al., 1993), (Smith, 1983). In the Pittsburgh approach, each rule set is a candidate judgment strategy and each strategy has a variable number of rules – hence strategies with a few simpler and effective rules reflect a satisficing mode of interaction.

One advantage of genetic algorithms is that search is done on the encoding of the strings, and not the strings themselves. While the proposed research does necessarily use GAs as a search method, it does take advantage of the robustness of the string encodings. To start, the fitness of a rule set is considered to be the ability to classify a set of exemplars in a manner consistent with human decision making behavior (Simon, 1959). Therefore, a rule set should not only match a set of exemplars, but it should also resemble the types of noncompensatory judgment strategies a service provider typically uses in these tasks. This is done in the model through the use of a multi-objective function that evaluates fitness along three dimensions: completeness, specificity, and parsimony. The completeness dimension is based on work by DeJong and his colleagues (DeJong et al., 1993), and is a measure of how well a rule set matches the entire set of exemplars (i.e., human judgments in a data set). The specificity dimension was first suggested by Holland et al. (Holland et al., 1986), and is a measure of how specific a rule set is with respect to the number of wild cards it contains. Therefore, rule sets with less wild characters are classified as more specific. The parsimony dimension is a measure of the goodness of a rule set in terms of the necessity of each rule. Hence, in a parsimonious rule set, there are no unnecessary rules. The ideal rule set, therefore, will match all judgments, will be maximally specific, and maximally parsimonious. The mathematical formulation of each dimension is discussed in the following section.

Definition of Multiple Objectives

*Definition 1.1***:** An exemplar matrix, ***E***, consisting of a set of binary variable vectors, called exemplars, whose range is the set {0,1}. Each exemplar within ***E*** is represented as $\boldsymbol{e}_{i\bullet}$, for $i = 1, \ldots, r$, where r is the total number of exemplars. Each binary variable within $\boldsymbol{e}_{i\bullet}$ is represented as $e_{i,j}$ for $j = 1, \ldots, n$, where n is exemplar length. Thus, ***E*** is a r by n matrix in the form:

$$E = \begin{pmatrix} e_{1,1} & e_{1,2} & \cdots & e_{1,n} \\ e_{2,1} & e_{2,2} & \cdots & e_{2,n} \\ \vdots & \vdots & & \vdots \\ e_{r,1} & e_{r,2} & \cdots & e_{r,n} \end{pmatrix} \qquad \textbf{(2)}$$

*Definition 1.2***:** A rule set matrix is a matrix, ***S***, consisting of a set of ternary variable vectors, called a rule set, whose range is the set {0,1,#}. Each rule within ***S*** is represented as $\boldsymbol{s}_{k\bullet}$ for $k = 1, \ldots, q$, where q is the number of rules in the rule set. Each ternary variable within $\boldsymbol{s}_{k\bullet}$ is represented as $s_{k,j}$, for $j = 1, \ldots, n$ where n is the rule length. Therefore, ***S*** is a q by n matrix in the form:

$$S = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,n} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,n} \\ \vdots & \vdots & & \vdots \\ s_{q,1} & s_{q,2} & \cdots & s_{q,n} \end{pmatrix} \qquad \textbf{(3)}$$

Matching an exemplar with a rule set requires an indicator function. Therefore, given that $x \in \boldsymbol{E}$ and $y \in \boldsymbol{S}$, an indicator function is defined as $\boldsymbol{I}_A$ where $A = \{x, \#\}$, so that,

$$I_A(y) = \begin{cases} 1 \; if \; y \in A, \\ 0 \; if \; y \notin A \end{cases} \qquad \textbf{(4)}$$

The results of applying $\boldsymbol{I_A}$ to compare a rule set, ***S***, with the *i*th exemplar $\boldsymbol{e_{i\bullet}}$, is shown as a "matching matrix", $\boldsymbol{M_i}$, for exemplar *i* where

$$M_i = \begin{bmatrix} I_{\{e_{i,1},\#\}}(s_{1,1}) & I_{\{e_{i,2},\#\}}(s_{1,2}) & \cdots & I_{\{e_{i,n},\#\}}(s_{1,n}) \\ I_{\{e_{i,1},\#\}}(s_{2,1}) & I_{\{e_{i,2},\#\}}(s_{2,2}) & \cdots & I_{\{e_{i,n},\#\}}(s_{2,n}) \\ \vdots & \vdots & & \vdots \\ I_{\{e_{i,1},\#\}}(s_{q,1}) & I_{\{e_{i,2},\#\}}(s_{q,2}) & \cdots & I_{\{e_{i,n},\#\}}(s_{q,n}) \end{bmatrix} \quad \textbf{(5)}$$

Each row of the matching matrix represents how well an exemplar $\boldsymbol{e_{i\bullet}}$ matches a particular rule, $\boldsymbol{s_{k\bullet}}$, within the rule set. For simplicity's sake, rewrite $\boldsymbol{M_i}$ so that each element is represented as a binary variable, $m_{i,k,j}$, such that,

$$m_{i,k,j} = I_{\{e_{i,j},\#\}}(s_{k,j}) \quad \textbf{(6)}$$

Rothrock (Rothrock, 1995) proved that the matching matrix is a lattice under disjunction and conjunction so that it can be manipulated algebraically. Thus, by applying the disjunct operator on elements of the matching matrix, one can define that an exemplar is matched to a rule if and only if:

$$\prod_{j=1}^{n} m_{i,k,j} = 1 \quad \textbf{(7)}$$

That is, for any exemplar *i*, and any rule *k*, both having length *n*, a vector-wise match exists if and only if each exemplar value matches the corresponding rule value. Thus, a matching function, ***f***, between a rule set, ***S***, and an exemplar $\boldsymbol{e_{i\bullet}}$, can be formulated as,

$$f(M_i) = \bigvee_{k=1}^{q} \left(\prod_{j=1}^{n} m_{i,k,j} \right) \quad \textbf{(8)}$$

for *q* rules in the rule set, each with length *n*. A match, therefore, between an exemplar $\boldsymbol{e_{i\bullet}}$ and a rule set exists if and only if $\boldsymbol{f(M_i)} = 1$.

Definition 1.3: A rule set is said to be complete if it is able to match all the exemplars in the exemplar set. A scaled function to indicate rule set completeness, ***c***, follows:

$$c(M_1, M_2, \ldots, M_r) = \frac{\sum_{i=1}^{r} f(M_i)}{r} \quad \textbf{(9)}$$

for r exemplars. Thus, $0 \le \boldsymbol{c} \le 1$. Thus, although the completeness function is able to measure the degree to which a rule set matches an exemplar set, a rule set with optimal completeness does not necessarily represent a cognitively plausible judgment strategy (Klein, 1999), (Kirlik, Walker, Fisk, & Nagel, 1996), (Rouse, 1983), (Rasmussen, 1983). Two other fitness dimensions will now be introduced in an attempt to improve the capability of the fitness function to better achieve psychological plausibility.

Definition 1.4: A rule set is fully specified if there are no wild card characters in the rule set. A scaled function to show rule specificity, ***t***, follows:

$$t(S) = \frac{\sum_{k=1}^{q} \sum_{j=1}^{n} I_{\{0,1\}}(s_{k,j})}{(qn)} \quad \textbf{(10)}$$

for *q* rules of length *n* each. Thus, $0 \le \boldsymbol{t} \le 1$. While the specificity dimension promotes content, useless rules can exist within a rule set without penalty. Therefore, the final fitness dimension, parsimony, will now be introduced to eliminate useless rules from the rule set.

*Definition 1.5***:** A rule set is said to be parsimonious if each rule within the rule set matches at least one exemplar in the exemplar set. A scaled function to indicate rule set parsimony, ***p***, follows:

$$p(M_1, M_2, \ldots, M_r) = \frac{\sum_{k=1}^{q} \left(\bigvee_{i=1}^{r} \left[\prod_{j=1}^{n} m_{i,k,j} \right] \right)}{q} \quad \textbf{(11)}$$

For *r* exemplars and *q* rules of length *n* each. Thus, $0 \le \boldsymbol{p} \le 1$. Parsimony provides a means of discriminating between rule sets that are wholly useful (i.e., each rule matches at least one exemplar) and those that are not. Rothrock and Kirlik (Rothrock & Kirlik) provide examples to show that the combined completeness/specificity/parsimony value can represent viable decision strategies.

0-1 Multi-objective Linear Formulation

When the length of an exemplar or rule, *n*, is relatively small, ***S*** can be defined as a matrix which includes all possible rules. Since each element s_{kj} can take three possible values ($s_{kj} \in \{0,1,\#\}$), there exist 3^n different rules that define the rows of ***S***. In this case the problem can be models as a simple 0-1 multi-objective linear program. Let

$$x_k = \begin{cases} 1 & \text{if rule in row } k \text{ is selected,} \\ 0 & \text{otherwise.} \end{cases}$$

Let $w_{i,k} = \bigwedge_{j=1}^{n} m_{i,k,j}$. Thus, function

$f(M_i) = \bigvee_{k=1}^{q} \left(\bigwedge_{j=1}^{n} m_{i,k,j} \right)$ can be written as

$y_i = f(M_i) = \max_{k=1,\ldots,q} \{ w_{i,k} \cdot x_k \}$.

This new binary variable y_i can be used to define the completeness function:

$$c(y) = \frac{\sum_{i=1}^{r} y_i}{r}.$$

Since one of the objectives is to maximize *c(y)*, the y_i variables can be equivalently defined by the following set of constraints:

$$y_i \leq \sum_{k=1}^{q} w_{i,k} x_k, \qquad i=1,\ldots,r.$$

Let $v_k = \sum_{j=1}^{n} I_{\{0,1\}}(s_{k,j})$. Then, the specificity function can be defined as

$$t(x) = \frac{\sum_{k=1}^{q} v_k x_k}{qn}.$$

Finally, let $u_k = \max_{i=1,\ldots,r} \{ w_{i,k} \}$. Then, the parsimony function can be stated as

$$p(x) = \frac{\sum_{k=1}^{q} u_k x_k}{q}.$$

Now, it is possible to formulate the 0-1 multi-objective linear programming (MOLP) problem as follows:

(MOLP)

$$\text{maximize } z(x,y) = \{ c(y),\ t(x),\ p(x) \},$$

$$\text{subject to } \quad y_i \leq \sum_{k=1}^{q} w_{i,k} x_k, \qquad i=1,\ldots,r,$$

$$\sum_{k=1}^{q} x_k \leq b,$$

$$x_k \in \{0, 1\}, \quad k=1,\ldots,q,$$

$$y_i \in \{0, 1\}, \quad i=1,\ldots,r,$$

where *b* is the maximum number of rules that can be selected. Note that *b* is a psychologically relevant variable as it defines the number of rules a human can store and use as part of his decision strategy. Let ***F*** be the feasible region, i.e., $(x, y) \in \boldsymbol{F}$ if an only if (*x*, *y*) satisfy the above set of constraints. The objective of MOLP is to determine the set ***Ef*** of "efficient" solutions. Set ***Ef*** is a subset of ***F***, which main characteristic is that, for each solution outside the set but still within ***F***, there is an efficient solution in ***Ef*** for which all objective functions are unchanged or improved and at least one is strictly improved. The concept of efficient solution also appears in the literature under the names of "Pareto optimum" or "non-dominated" solution. An essential consequence of the fact that MOLP deals with discrete (non-continuous) variables when trying to find the set of all efficient solutions is that it is not sufficient to aggregate the objectives through weighted sums (Goicoechea, Hansen, & Duckstein, 1982). There usually exist efficient solutions, which are not optimal for any weighted sum of the objectives, even in cases where the constraint matrix is totally unimodular. These solutions are called non-supported efficient (NE) solutions, whereas the remaining are called supported efficient (SE) solutions.

Besides efficiency, there are other definitions of the max term in the formulation of MOLP. For example, one could consider lexicographic maximization, when the objectives are compared lexicographically. This could be done with one or all permutations of the objective functions. Another possibility is to maximize the worst (smallest) objective function, i.e,

$$\text{maximize } z(x,y) = \min \{ c(y),\ t(x),\ p(x) \},$$

$$\text{subject to } \quad (x,y) \in F.$$

We call this the min-ordering problem. A combination of the latter two is the lexicographic min-ordering problem, where the vector of objective values is first resorted in a non-decreasing order of its components, and the resulting vectors are compared lexicographically (Ehrgott, 1995). For lexicographic optimization, it is known that a lexicographically optimal solution is always efficient, and even supported efficient solution (Hammacher & Ruhe, 1994).

Computational Complexity of the Problem

To solve the MOLP problem through simple iterative search, the number of times z(*x y)* must be computed is:

$$C_q^{3^n} \quad \frac{(3^n)!}{q!(3^n ? q)!}$$

where *n* is the length of each rule (or length of the exemplar) and *q* is the maximum number of rules per rule set. TABLE 4 illustrates the exponential growth of the problem size based on rule lengths and maximum number of rules. The problem solving time also increases exponentially depending on rule length and maximum number of rules.

TABLE 4. Numbers of the Combinations based on Rule Length and Max Rule Number.

Rule length (n)

Max rule(b)

	1	2	3	4	5	6	7	8	9	10
1	3	9	27	81	243	729	2187	6561	19683	59049
2	3	36	351	3240	29403	265356	2390391	21520080	1.94E+08	1.74E+09
3	1	84	2925	85320	2362041	64304604	1.74E+09	4.71E+10	1.27E+12	3.43E+13
4		126	17550	1663740	1.42E+08	1.17E+10	9.51E+11	7.71E+13	6.25E+15	5.07E+17
5		126	80730	25621596	6.77E+09	1.69E+12	4.15E+14	1.01E+17	2.46E+19	5.98E+21
6		84	296010	3.25E+08	2.69E+11	2.04E+14	1.51E+17	1.11E+20	8.07E+22	5.89E+25
7		36	888030	3.48E+09	9.1E+12	2.11E+16	4.7E+19	1.04E+23	2.27E+26	4.96E+29
8		9	2220075	3.22E+10	2.68E+14	1.9E+18	1.28E+22	8.48E+25	5.58E+29	3.66E+33
9		1	4686825	2.61E+11	7.01E+15	1.52E+20	3.1E+24	6.17E+28	1.22E+33	2.4E+37
10			8436285	1.88E+12	1.64E+17	1.1E+22	6.76E+26	4.05E+31	2.4E+36	1.42E+41
11			13037895	1.21E+13	3.47E+18	7.18E+23	1.34E+29	2.41E+34	4.29E+39	7.62E+44
12			17383860	7.07E+13	6.72E+19	4.29E+25	2.43E+31	1.32E+37	7.04E+42	3.75E+48
13			20058300	3.75E+14	1.19E+21	2.37E+27	4.06E+33	6.62E+39	1.06E+46	1.7E+52
14			20058300	1.82E+15	1.96E+22	1.21E+29	6.3E+35	3.1E+42	1.5E+49	7.18E+55
15			17383860	8.14E+15	2.99E+23	5.77E+30	9.13E+37	1.35E+45	1.96E+52	2.82E+59
16			13037895	3.36E+16	4.26E+24	2.58E+32	1.24E+40	5.53E+47	2.41E+55	1.04E+63
17			8436285	1.28E+17	5.69E+25	1.08E+34	1.58E+42	2.13E+50	2.79E+58	3.62E+66
18			4686825	4.57E+17	7.15E+26	4.27E+35	1.91E+44	7.74E+52	3.05E+61	1.19E+70
19			2220075	1.51E+18	8.47E+27	1.6E+37	2.18E+46	2.67E+55	3.15E+64	3.69E+73
20			888030	4.69E+18	9.48E+28	5.68E+38	2.36E+48	8.72E+57	3.1E+67	1.09E+77
21			296010	1.36E+19	1.01E+30	1.92E+40	2.44E+50	2.72E+60	2.9E+70	3.06E+80
22			80730	3.72E+19	1.02E+31	6.17E+41	2.4E+52	8.08E+62	2.6E+73	8.21E+83
23			17550	9.54E+19	9.76E+31	1.9E+43	2.26E+54	2.3E+65	2.22E+76	2.11E+87
24			2925	2.31E+20	8.95E+32	5.58E+44	2.04E+56	6.26E+67	1.82E+79	5.18E+90
25			351	5.26E+20	7.84E+33	1.57E+46	1.76E+58	1.64E+70	1.43E+82	1.22E+94
26			27	1.13E+21	6.57E+34	4.26E+47	1.46E+60	4.11E+72	1.08E+85	2.78E+97
27			1	2.31E+21	5.28E+35	1.11E+49	1.17E+62	9.95E+74	7.87E+87	6.1E+100

Total elapsed processing time: 07:08:33

Total elapsed processing time: 15:46:47

The step-shape bold line, in the middle of TABLE 5, is a boundary of the combined single objective solution time, which we set to 24 hours, using an exhaustive enumeration of all combinations on a Pentium-4 class machine running MATLAB version 5.3. It takes less than 24 hours to solve the upper cases of the step-shape bold line and more than 24 hours to solve the bottom cases of the step-shape bold line. For example, in the case of four maximum rules (to match four exemplars with a rule length of four, the objective must be computed 1,663,740 times. This consumes over seven hours of processing time. In the case of three maximum rules with a rule length of five, the objective must be computed 2,362,041 times costing well over 15 hours of processing time. These example problems are relatively small in size compared to the encoding for a complex decision making task. Previous research on a small air traffic control task required 23 maximum rules with a rule length of 10 (Rothrock & Kirlik). This would require the objective to be computed $2.107 * 10^{87}$ times. The exhaustive search technique, therefore, is prohibitively costly and not currently feasible. To reduce the computational cost, we sought first to apply a weighted-sum scalarization approach to find the set of efficient solutions.

0-1 Single-objective Weighted Sum Scalarization

To begin our investigation of the noncompensatory decision making problem, we first worked on a weighted sum scalarization approach that combined all three objectives into a single objective. Varying the weights, it is known that all supported efficient solutions can be found (Isermann, 1979).

To apply the weighted sum scalarization approach, we need simple modification of the previous 0-1 MOLP into the 0-1 single-objective linear programming (SOLP). There are three weights for objective function such as w_c, w_t and w_p. And the constraints and the variables are same as 0-1 MOLP which we proposed previous section.

(SOLP)

$$\text{maximize } z(x,y) = w_c c(y) + w_t t(x) + w_p p(x),$$

subject to $y_i \le \sum_{k=1}^{q} w_{i,k} x_k$, $i=1,\ldots,r,$

$$\sum_{k=1}^{q} x_k \le b,$$

$x_k \in \{0, 1\},$ $k=1,\ldots,q,$

$y_i \in \{0, 1\},$ $i=1,\ldots,r,$

where $w_c = \{0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9\}$

$w_t = \{0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9\}$

$w_p = \{0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9\}$

We solve the 0-1 SOLP through the 1000 combinations of w_c,w_t and w_p. Among the various numbers of combinations shown in TABLE 4, we choose a sample problem to apply the weighted sum scalarization approach. First, we decide the length of examplar as four (n=4). The all possible combinations of the examplars are sixteen ($2^4=16$) because the examplar matrix ***E*** is a set of binary variable vectors in Definition 1.1. Among the sixteen possible examplars, we randomly chose nine for our exemplar matrix (***E***). Therefore, ***E*** is 9 by 4 matrix. And the rule set matrix ***S*** has ternary variable vectors in Definition 1.2. Therefore, ***S*** is 81 by 4 matrix. Thus, our problem is represented as $n = 4$, $r = 9$, $q = 81$ and $b = 4$.

$$E = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix} \qquad S = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \# \\ & \cdot & & \\ & \cdot & & \\ & \cdot & & \\ \# & \# & \# & 0 \\ \# & \# & \# & 1 \\ \# & \# & \# & \# \end{pmatrix}$$

While the sample problem will not prove that the 0-1 SOLP approach will be effective in general, it can expose the inadequacies of such an approach in approximating the solution for the multi-objective problem. Using these two matrix as input data for the 0-1 SOLP, we solved for the values of $w_{i,k}$, ,v_k and u_k by LINGO software (version 6). The values of $w_{i,k}$ are presented in TABLE 5 of the appendix. The values of $w_{i,k} = \bigvee_{j=1}^{n} m_{i,k,j}$ can be defined as binary value, $w_{ik} \in \{0,1\}$, by applying the disjunctive operator on elements of the matching matrix. These values are needed for the completeness function. Values of the v_k are shown in TABLE 6 of the appendix. The values of the v_k can be defined, $v_k \in \{0,1,2,3,4\}$, by equation of $v_k = \sum_{j=1}^{n} I_{\{0,1\}}\left(s_{k,j}\right)$. These values are needed for the specificity function. The values of the u_k are shown as TABLE 7 in the appendix. The values of the u_k can be defined as binary value, $u_k \in \{0,1\}$, where $u_k = \max_{i=1,\ldots,r}\left\{ w_{i,k} \right\}$. These values are needed for the parsimony function. We now have assembled all the necessary components to solve the weighted sum scalarization 0-1 SOLP problems.

The TABLE 8 shows the results of the 0-1 SOLP. The results are collapsed into twelve cases of different variable solutions. The variable x_k and y_i have binary value. For example, case 2 in TABLE 8 represents the case in which the variables x_1, x_2, x_{61}, and x_{64} have value of one and the other subscripts k of the x_k have value of zero. Moreover, the weights of w_c,w_t are fixed as zero and the changing value of w_p from 0.1 to 0.9.

The collapsed twelve case results of the 0-1 SOLP problems, from one thousand weighted sum combination 0-1 SOLP problems, can be decomposed to eight case results of the 0-1 MOLP in TABLE 9.

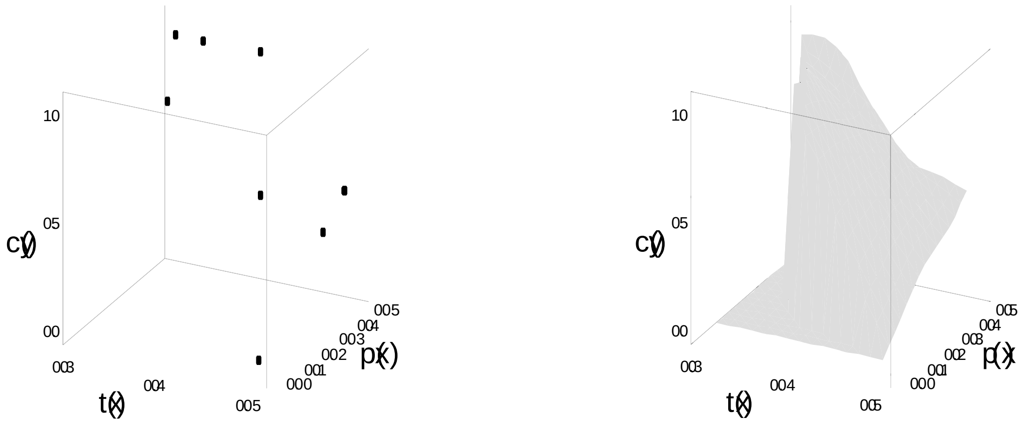
TABLE 8. The 0-1 SOLP LINGO result summary of the one thousand weighted combinations.

				X(k)={0,1}				Y(i)={0,1}								
	W(c)	W(t)	W(p)	k				1	2	3	4	5	6	7	8	9
Case1	0	0	0	4	14	28	29	0	0	0	0	0	0	0	0	0
case2	0	0	0.1~0.9	1	2	61	64	1	1	0	1	0	0	0	0	0
case3	0	0.1~0.9	0	4	10	40	41	0	0	0	1	0	0	0	1	1
case4	0	0.1	0.1~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.2	0.2~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.3	0.2~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.4	0.3~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.5	0.3~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.6	0.4~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.7	0.4~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.8	0.5~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
	0	0.9	0.5~0.9	1	13	40	41	1	0	0	0	0	1	0	1	1
case5	0	0.2	0.1	1	2	10	40	1	1	0	1	0	0	0	1	0
	0	0.4	0.2	1	2	10	40	1	1	0	1	0	0	0	1	0
	0	0.6	0.3	1	2	10	40	1	1	0	1	0	0	0	1	0
	0	0.8	0.4	1	2	10	40	1	1	0	1	0	0	0	1	0
case6	0	0.3	0.1	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.5	0.2	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.7	0.2~0.3	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.6	0.2	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.8	0.3	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.9	0.3	1	10	11	41	1	0	0	1	1	0	0	0	1
	0	0.9	0.4	1	10	11	41	1	0	0	1	1	0	0	0	1
case7	0	0.4	0.1	10	11	13	41	0	0	0	1	1	1	0	0	1
	0	0.8	0.2	10	11	13	41	0	0	0	1	1	1	0	0	1
case8	0	0.5~0.9	0.1	2	5	10	32	0	1	1	1	0	0	1	0	0
	0	0.9	0.2	2	5	10	32	0	1	1	1	0	0	1	0	0
case9	0.1	0	0.9	2	5	6	81	1	1	1	1	1	1	1	1	1
case10	0.1~0.9	0	0	14	28	29	81	1	1	1	1	1	1	1	1	1
case11	0.1~0.9	0	0.1~0.8	13	15	17	81	1	1	1	1	1	1	1	1	1
case12	0.1~0.9	0.1~0.9	0~0.9	27	32	40	41	1	1	1	1	1	1	1	1	1

TABLE 9. Decomposition Result from 0-1 SOLP to 0-1 MOLP

	X(k)={0,1}				Y(i)={0,1}									c(y)	t(x)	p(x)
	k				1	2	3	4	5	6	7	8	9			
case1	4	14	28	29	0	0	0	0	0	0	0	0	0	0	0.049383	0
case2	1	2	61	64	1	1	0	1	0	0	0	0	0	0.333333	0.040123	0.049383
case3	4	10	40	41	0	0	0	1	0	0	0	1	1	0.333333	0.049383	0.037037
case4	1	13	40	41	1	0	0	0	0	1	0	1	1	0.444444	0.049383	0.049383
case5	1	2	10	40	1	1	0	1	0	0	0	1	0			
case6	1	10	11	41	1	0	0	1	1	0	0	0	1			
case7	10	11	13	41	0	0	0	1	1	1	0	0	1			
case8	2	5	10	32	0	1	1	1	0	0	1	0	0			
case9	2	5	6	81	1	1	1	1	1	1	1	1	1	1	0.033951	0.049383
case10	14	28	29	81	1	1	1	1	1	1	1	1	1	1	0.037037	0.012346
case11	13	15	17	81	1	1	1	1	1	1	1	1	1	1	0.030864	0.049383
case12	27	32	40	41	1	1	1	1	1	1	1	1	1	1	0.040123	0.049383

The eight case results of the 0-1 MOLP are plotted on three-objective space in Figure 1. The Figure 1.a) represents the eight case efficient solution results and the Figure 1 b) represents the surface of the efficient solutions. The *c(y)* axis represents the completeness value. The *t(x)* axis represents specificity value. And *p(x)* axis represents the parsimony value.



a) 3D plot of efficient solution

b) 3D Surface plot of efficient solution

Figure 1. SOLP Solutions Shown as Points and as a Surface.

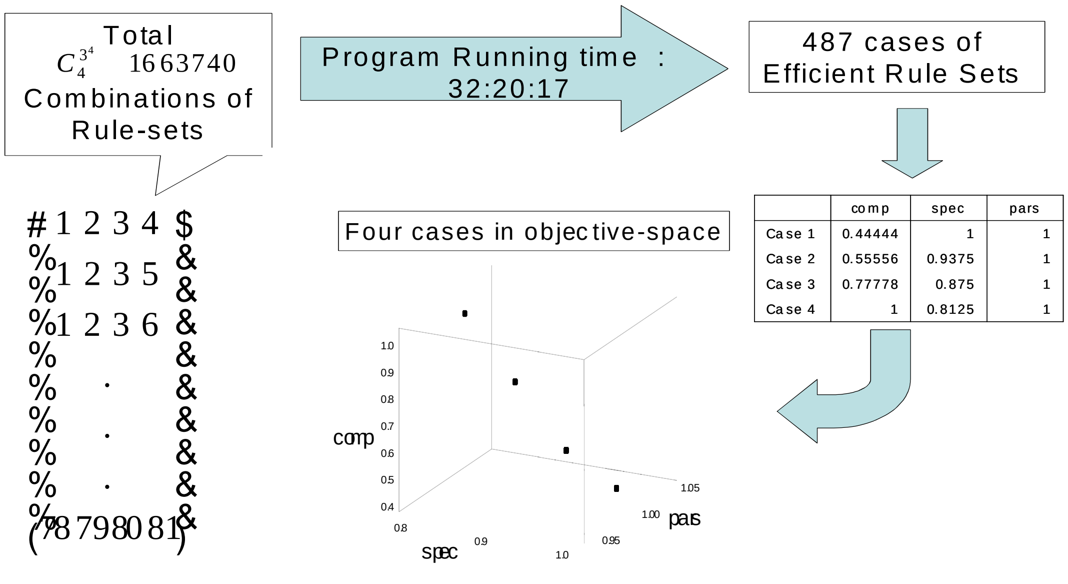
Finding the set of non-dominated solutions for the 0-1 MOLP

It turns out that finding or counting efficient solutions for problems similar to MOLP is in general NP-complete (Ehrgott & Gandibleux, 2000). This is true even for problems, which have efficient algorithms in the single objective case (Emelichev & Perepelitsa, 1991). Another aspect related to the difficulty of MOLP is the number of efficient solutions. It may be exponential in the problem size, thus prohibiting any effective method to determine all efficient solutions. Numerical results available for the knapsack problem (Visee, Teghem, Pirlot, & Ulungu, 1998) show that the number of supported solutions grows linearly with the problem size, but the number of non-supported solutions grows following an exponential function.

We apply two types of enumeration methods to find the non-dominated solutions for the 0-1 MOLP. One is multi-objective enumeration method and the other is a combined single-objective enumeration method. We use the same problem for the enumeration methods as we used for the weighted sum scalarization approach (n=4,r=9,q=81,b=4). The total combinations of the defined problem are $C_{4}^{3^4} \quad 1,663,740$. Each rule is covered by the ternary alphabet {0,1,#} and the represented rule can be expressed by rule index number like as below matrix:

$$\begin{bmatrix} 0000\ 0001\ 000\#\ 0010 \\ 0000\ 0001\ 000\#\ 0011 \\ 0000\ 0001\ 000\#\ 001\# \\ \cdot \\ \cdot \\ \cdot \\ \#\#1\#\ \#\#\#0\ \#\#\#1\ \#\#\#\# \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 5 \\ 1 & 2 & 3 & 6 \\ & \cdot & & \\ & \cdot & & \\ & \cdot & & \\ 78 & 79 & 80 & 81 \end{bmatrix}$$

The result of the multi objective enumeration method by MATLAB software can be summarized as shown in Figure 2. It took over thirty two hours to exhaustively enumerate through the search space to find efficient rule sets. A total of 487 efficient rule sets were found and could be collapsed to four non-dominated cases as seen in the plot shown in Figure 2.



	comp	spec	pars
Case 1	0.44444	1	1
Case 2	0.55556	0.9375	1
Case 3	0.77778	0.875	1
Case 4	1	0.8125	1

Figure 2. Process for Multi-Objective Enumeration Method.

The solution time of the single objective enumeration method by MATLAB software is shown in TABLE 4. The summary of the three methods, single objective enumeration, multi objective enumeration and sum scalarization methods is shown in TABLE 10. We can compare the three methods with several terms such as number of combination, number of efficient sets, plot number on objective space and solution time. The solution time of sum scalarization method can not measure exactly because the weight, w_c,w_t and w_p, are changed manually by person but it takes one second to solve the each case of the one thousand combinations.

TABLE 10. The Comparison of Three Types Methods

Method	**Single Objective Enumeration**	**Sum Scalarization**	**Multi-Objective Enumeration**
Combination	$C_4^{3^4}$ 1,663,740	1000	$C_4^{3^4}$ 1,663,740
Efficient solution	1 case	12 cases	487 cases
On objective space	1 point	8 points	4 points
Solution time	07:08:33 hrs	< 1 hr	32:20:17 hrs
Solution Software	MATLAB 5.3	LINGO6	MATLAB 5.3

Discussion of Results and Suggestions for Future Study

The outcomes of our analyses suggest that while the sum scalarization provides a fast way of computing near-optimal solutions (12), it misses the majority of non-dominated solutions found by a multi-objective enumeration approach (487).

We plan to use other search techniques to solve our noncompensatory decision making problems. First, we intend to use single-objective search enumeration techniques (Balas, 1963), (Balas, 1965) to either explicitly or implicitly enumerate all possible integer vectors to identify the entire set of efficient solutions. The performance of these techniques depends on the effectiveness the branching strategy and the chosen implicit enumeration tests (such as the ceiling test, cancellation test, etc.).

We also intend to approach the problem as a multi-objective problem for large values of *n*. When the length of an exemplar or rule *n* is large, it is not possible to generate the rule set matrix ***S*** by including all possible rules since the number of rules increases exponentially with n. In this case, the problem needs to be reformulated by defining ***S*** as the unknown matrix of chosen rules. The components $s_{i,k,j}$ of ***S*** will be the ternary variables of the problem.

Approximation methods, such as metaheuristic algorithms, will be used to find efficient solutions. A metaheuristic refers to an iterative master strategy that guides and modifies the operations of subordinate heuristics by combining intelligently different concepts for exploring and exploiting the search space (Glover & Laguna, 1997), (Osman & Laporte, 1996), (Deb, 2001). A metaheuristic may manipulate a complete (or incomplete) solution or a collection of solutions at each iteration. The family of metaheuristics includes constant logic programming, genetic algorithms, evolutionary methods, neural networks, simulated annealing, tabu search, non-monotonic search strategies, greedy randomized adaptive search, colony systems, etc. A comprehensive list of 1380 references on the theory and application of metaheuristics is presented in Osman and Laporte (Osman & Laporte, 1996).

The development of a new framework based on mathematical programming techniques to optimize the fit between models of alternative human strategies and actual decisions in complex command-and-control systems is an attempt to integrate Cognitive Science and Operations Research. The potential of technology transfer of the proposed work is promising. The investigators focused on the aviation domain because of the relevance of the present research toward next generation aviation systems – namely the National Aeronautics and Space Administration's vision of free flight by the year 2025. One company, MicroAnalysis and Design, Inc., is currently engaged in research on free flight (Gore & Corker, 2000) and has expressed interest in extending the findings from the proposed research into an operational domain.

REFERENCES

Balas, E. (1963, December 5-7). *Linear Programming with 0-1 Variables (in Rumanian).* Paper presented at the Third Scientific Session on Statistics, Bucharest.

Balas, E. (1965). An Additive Algorithm for Solving Linear Programming with Zero-One Variables. *Operations Research, 13*(4), 517-548.

Brunswik, E. (1955a). In Defense of Probabilistic Functionalism: A Reply. *Psychological Review, 62*(3), 236-242.

Brunswik, E. (1955b). Representative Design and Probabilistic Theory in a Functional Psychology. *Psychological Review, 62*(3), 193-217.

Chen, H., Shankaranarayanan, G., She, L., & Iyer, A. (1998). A Machine Learning Approach to Inductive Query by Examples: An Experiment Using Relevant Feedback, ID3, Genetic Algorithms, and Simulated Annealing. *Journal of the American Society for Information Science, 49*(8), 693-705.

Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. New York: John Wiley & Sons, Ltd.

DeJong, K. A., Spears, W. M., & Gordon, D. F. (1993). Using Genetic Algorithms for Concept Learning. *Machine Learning, 13*(2-3), 161-188.

Edwards, W., & Newman, J. R. (1982). *Multiattribute Evaluation*. Beverly Hills, CA: Sage Publications Inc.

Ehrgott, M. (1995). *Lexicographically Max-Ordering - A Solution Concept for Multicriteria Combinatorial Optimization.* Paper presented at the 5th Workshop of the DGOR - Working Group Multicriteria Optimization and Decision Theory, Universitat Kaiserslautern, D-67663 Kaiserslautern, Germany.

Ehrgott, M., & Gandibleux, X. (2000). *An Annotated Bibliography of Multiobjective Combinatorial Optimization* (Report in Wirtschaftsmathematik Nr 62/2000). D-67663 Kaiserslautern, Germany: Universitat Kaiserslautern.

Emelichev, V. A., & Perepelitsa, V. A. (1991). Complexity of Vector Optimization Problem on Graphs. *Optimization, 22*, 903-918.

Ford, J. K., Schmitt, N., Schechtman, S. L., Hults, B. M., & Doherty, M. L. (1989). Process Tracing Methods: Contributions, Problems, and Neglected Research Questions. *Organizational Behavior and Human Decision Processes, 43*, 75-117.

Gigerenzer, G., & Goldstein, D. G. (1996). Reasoning the Fast and Frugal Way: Models of Bounded Rationality. *Psychological Review, 103*, 650-669.

Gigerenzer, G., Hoffrage, U., & Kleinbolting, H. (1991). Probabilistic Mental Models: A

Brunswikian Theory of Confidence. *Psychological Review, 98*, 506-528.

Gigerenzer, G., & Selten, R. (Eds.). (2001). *Bounded Rationality: The Adaptive Toolbox*. Cambridge, MA: The MIT Press.

Glover, F., & Laguna, M. (1997). *Tabu Search*. Dordrecht: Kluwer Academic Publishers.

Goicoechea, A., Hansen, D. R., & Duckstein, L. (1982). *Multiobjective Decision Analysis with Engineering and Business Application*. New York: Wiley.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. New York: Addison-Wesley Publishing.

Gore, B. F., & Corker, K. M. (2000). *A systems engineering approach to behavioral predictions of an advanced air traffic management concept.* Paper presented at the The 19th Digital Avionics Systems Conference (DASC), Philadelphia, PA.

Greene, D. P., & Smith, S. F. (1993). Competition-based Induction of Decision Models from Examples. *Machine Learning, 13*, 229-257.

Hammacher, H. W., & Ruhe, G. (1994). On Spanning Tree Problems with Multiple Objectives. *Annals of Operations Research, 52*, 209-230.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI: The University of Michigan Press.

Holland, J. H., Holyoak, K. F., Nisbett, R. E., & Thagard, P. R. (1986). *Induction*. Cambridge, MA: The MIT Press.

Isermann, H. (1979). Proper Efficiency and the Linear Vector Maximum Problem. *Operations Research, 22*, 189-191.

Kirlik, A., Walker, N., Fisk, A. D., & Nagel, K. (1996). Supporting Perception in the Service of Dynamic Decision Making. *Human Factors, 38*(2), 288-299.

Klein, G. A. (1999). Applied Decision Making. In P. A. Hancock (Ed.), *Handbook of Perception and Cognition: Human Performance and Ergonomics* (pp. 87-108). New York: Academic Press.

Maule, A. J. (1997). Strategies for Adapting to Time Pressure. In R. Flin & E. Salas & M. Strub & L. Martin (Eds.), *Decision Making under Stress: Emerging Themes and Applications* (pp. 271-279). Brookfield, VT: Ashgate.

Maule, A. J., & Hockey, G. R. J. (1993). State, Stress and Time Pressure. In O. Svenson & A. J. Maule (Eds.), *Time Pressure and Stress in Human Judgment and Decision Making* (pp. 83-101). New York: Plenum.

Mendelson, E. (1997). *Introduction to Mathematical Logic* (4th ed.). London: Chapman & Hall.

Michalski, R. S. (1983). A Theory and Methodology of Inductive Learning. In R. S. Michalski & J. G. Carbonell & T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach*. Palo Alto, CA: Tioga Publishing Co.

Osman, I., & Laporte, G. (1996). Metaheuristics: A Bibliography. *Annals of Operations Research, 63*, 513-623.

Payne, J. W. (1976). Task Complexity and Contingent Processing in Decision Making: An Information Search and Protocol Analysis. *Organizational Behavior and Human Performance, 16*, 366-387.

Payne, J. W., Bettman, J. R., & Johnson, E. J. (1993). *The Adaptive Decision Maker*: Cambridge University Press.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. New York: Cambridge University Press.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning, 1*, 81-106.

Rasmussen, J. R. (1983). Skills, Rules and Knowledge: Signals, Signs, Symbols, and Other Distinctions in Human Performance Models. *IEEE Transactions on Systems, Man, and Cybernetics, 13*(3), 257-266.

Rothrock, L. (1995). *Performance Measures and Outcome Analyses of Dynamic Decision Making in Real-time Supervisory Control.* Unpublished Ph.D. dissertation, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta.

Rothrock, L. (1999). *Genetics-Based Inductive Inference Model to Represent Human Decision Strategies in a Supervisory Control Task.* Paper presented at the Artificial Neural Networks in Engineering Conference, St. Louis, MO.

Rothrock, L., & Kirlik, A. (In Press). Inferring Rule-based Strategies in Dynamic Judgment Tasks. *IEEE Transactions on Systems, Man, and Cybernetics.*

Rouse, W. B. (1983). Models of Human Problem Solving: Detection, Diagnosis, and Compensation for System Failures. *Automatica, 19*, 613-625.

Savage, L. (1972). *The Foundation of Statistics*. New York: Dover Publications, Inc.

Simon, H. A. (1955). A Behavioral Model of Rational Choice. *Quarterly Journal of Economics*, *69*, 99-118.

Simon, H. A. (1959). Theories of Decision-Making in Economics and Behavioral Science. *American Economic Review*, *XLIX*(3), 253-281.

Smith, S. F. (1983). *Flexible Learning of Problem Solving Heuristics through Adaptive Search.* Paper presented at the International Joint Conference on Artificial Intelligence, Karlsruhe, GE.

Vafaie, H., & DeJong, K. (1994). Improving a Rule Induction System Using Genetic Algorithms. In R. Michalski & G. Tecuci (Eds.), *Machine Learning: A Multistrategy Approach* (Vol. Vol. IV). San Mateo, CA: Morgan Kaufman Publishers.

Visee, M., Teghem, M., Pirlot, M., & Ulungu, E. L. (1998). Two-Phase and Branch-and-Bound Procedures to Solve the Bi-Objective Knapsack Problem. *Global Optimization*, *12*, 139-155.

Wright, P. (1974). The Harassed Decision Maker: Time Pressures, Distractions, and the Use of Evidence. *Journal of Applied Psychology*, *59*(5), 555-561.

APPENDIX

TABLE 5. The Values of the $w_{i,k}$.

		i								
		1	2	3	4	5	6	7	8	9
k	1	1	0	0	0	0	0	0	0	0
	2	0	1	0	0	0	0	0	0	0
	3	1	1	0	0	0	0	0	0	0
	4	0	0	0	0	0	0	0	0	0
	5	0	0	1	0	0	0	0	0	0
	6	0	0	1	0	0	0	0	0	0
	7	1	0	0	0	0	0	0	0	0
	8	0	1	1	0	0	0	0	0	0
	9	1	1	1	0	0	0	0	0	0
	10	0	0	0	1	0	0	0	0	0
	11	0	0	0	0	1	0	0	0	0
	12	0	0	0	1	1	0	0	0	0
	13	0	0	0	0	0	1	0	0	0
	14	0	0	0	0	0	0	0	0	0
	15	0	0	0	0	0	1	0	0	0
	16	0	0	0	1	0	1	0	0	0
	17	0	0	0	0	1	0	0	0	0
	18	0	0	0	1	1	1	0	0	0
	19	1	0	0	1	0	0	0	0	0
	20	0	1	0	0	1	0	0	0	0
	21	1	1	0	1	1	0	0	0	0
	22	0	0	0	0	0	1	0	0	0
	23	0	0	1	0	0	0	0	0	0
	24	0	0	1	0	0	1	0	0	0
	25	1	0	0	1	0	1	0	0	0
	26	0	1	1	0	1	0	0	0	0
	27	1	1	1	1	1	1	0	0	0
	28	0	0	0	0	0	0	0	0	0
	29	0	0	0	0	0	0	0	0	0
	30	0	0	0	0	0	0	0	0	0
	31	0	0	0	0	0	0	0	0	0
	32	0	0	0	0	0	0	1	0	0
	33	0	0	0	0	0	0	1	0	0
	34	0	0	0	0	0	0	0	0	0
	35	0	0	0	0	0	0	1	0	0
	36	0	0	0	0	0	0	1	0	0
	37	0	0	0	0	0	0	0	0	0
	38	0	0	0	0	0	0	0	0	0
	39	0	0	0	0	0	0	0	0	0
	40	0	0	0	0	0	0	0	1	0
k	41	0	0	0	0	0	0	0	0	1
	42	0	0	0	0	0	0	0	1	1
	43	0	0	0	0	0	0	0	1	0
	44	0	0	0	0	0	0	0	0	1
	45	0	0	0	0	0	0	0	1	1
	46	0	0	0	0	0	0	0	0	0
	47	0	0	0	0	0	0	0	0	0
	48	0	0	0	0	0	0	0	0	0
	49	0	0	0	0	0	0	0	1	0
	50	0	0	0	0	0	0	1	0	1
	51	0	0	0	0	0	0	1	1	1
	52	0	0	0	0	0	0	0	1	0
	53	0	0	0	0	0	0	1	0	1
	54	0	0	0	0	0	0	1	1	1
	55	1	0	0	0	0	0	0	0	0
	56	0	1	0	0	0	0	0	0	0
	57	1	1	0	0	0	0	0	0	0
	58	0	0	0	0	0	0	0	0	0
	59	0	0	1	0	0	0	1	0	0
	60	0	0	1	0	0	0	1	0	0
	61	1	0	0	0	0	0	0	0	0
	62	0	1	1	0	0	0	1	0	0
	63	1	1	1	0	0	0	1	0	0
	64	0	0	0	1	0	0	0	0	0
	65	0	0	0	0	1	0	0	0	0
	66	0	0	0	1	1	0	0	0	0
	67	0	0	0	0	0	1	0	1	0
	68	0	0	0	0	0	0	0	0	1
	69	0	0	0	0	0	1	0	1	1
	70	0	0	0	1	0	1	0	1	0
	71	0	0	0	0	1	0	0	0	1
	72	0	0	0	1	1	1	0	1	1
	73	1	0	0	1	0	0	0	0	0
	74	0	1	0	0	1	0	0	0	0
	75	1	1	0	1	1	0	0	0	0
	76	0	0	0	0	0	1	0	1	0
	77	0	0	1	0	0	0	1	0	1
	78	0	0	1	0	0	1	1	1	1
	79	1	0	0	1	0	1	0	1	0
	80	0	1	1	0	1	0	1	0	1
	81	1	1	1	1	1	1	1	1	1

TABLE 6. Values of the v_k.

V(1)	4	V(28)	4	V(55)	3
V(2)	4	V(29)	4	V(56)	3
V(3)	3	V(30)	3	V(57)	2
V(4)	4	V(31)	4	V(58)	3
V(5)	4	V(32)	4	V(59)	3
V(6)	3	V(33)	3	V(60)	2
V(7)	3	V(34)	3	V(61)	2
V(8)	3	V(35)	3	V(62)	2
V(9)	2	V(36)	2	V(63)	1
V(10)	4	V(37)	4	V(64)	3
V(11)	4	V(38)	4	V(65)	3
V(12)	3	V(39)	3	V(66)	2
V(13)	4	V(40)	4	V(67)	3
V(14)	4	V(41)	4	V(68)	3
V(15)	3	V(42)	3	V(69)	2
V(16)	3	V(43)	3	V(70)	2
V(17)	3	V(44)	3	V(71)	2
V(18)	2	V(45)	2	V(72)	1
V(19)	3	V(46)	3	V(73)	2
V(20)	3	V(47)	3	V(74)	2
V(21)	2	V(48)	2	V(75)	1
V(22)	3	V(49)	3	V(76)	2
V(23)	3	V(50)	3	V(77)	2
V(24)	2	V(51)	2	V(78)	1
V(25)	2	V(52)	2	V(79)	1
V(26)	2	V(53)	2	V(80)	1
V(27)	1	V(54)	1	V(81)	0

TABLE 7. Values of the u_k.

U(1)	1	U(28)	0	U(55)	1
U(2)	1	U(29)	0	U(56)	1
U(3)	1	U(30)	0	U(57)	1
U(4)	0	U(31)	0	U(58)	0
U(5)	1	U(32)	1	U(59)	1
U(6)	1	U(33)	1	U(60)	1
U(7)	1	U(34)	0	U(61)	1
U(8)	1	U(35)	1	U(62)	1
U(9)	1	U(36)	1	U(63)	1
U(10)	1	U(37)	0	U(64)	1
U(11)	1	U(38)	0	U(65)	1
U(12)	1	U(39)	0	U(66)	1
U(13)	1	U(40)	1	U(67)	1
U(14)	0	U(41)	1	U(68)	1
U(15)	1	U(42)	1	U(69)	1
U(16)	1	U(43)	1	U(70)	1
U(17)	1	U(44)	1	U(71)	1
U(18)	1	U(45)	1	U(72)	1
U(19)	1	U(46)	0	U(73)	1
U(20)	1	U(47)	0	U(74)	1
U(21)	1	U(48)	0	U(75)	1
U(22)	1	U(49)	1	U(76)	1
U(23)	1	U(50)	1	U(77)	1
U(24)	1	U(51)	1	U(78)	1
U(25)	1	U(52)	1	U(79)	1
U(26)	1	U(53)	1	U(80)	1
U(27)	1	U(54)	1	U(81)	1