On the Computational Complexity of Consumer Decision Rules

By A Norman, A. Ahmed, J. Chou, A. Dalal, K. Fortson, M. Jindal, C. Kurz
H. Lee, K. Payne, R. Rando, K. Sheppard, E. Sublett, and I. White

February 11, 2002

Abstract

A consumer entering a new bookstore can face more than 250,000 alternatives. The efficiency of compensatory and noncompensatory decision rules for finding a preferred item depends on the efficiency of their associated information operators. At best, item-by-item information operators lead to linear computational complexity; set information operators, on the other hand, can lead to constant complexity. We perform an experiment demonstrating that subjects are approximately rational in selecting between sublinear and linear rules. Many markets are organized by attributes that enable consumers to employ a set-selection-by-aspect rule using set information operations. In cyberspace decision rules are encoded as decision aids.

Classification codes
JEL: D1 Household Behavior, D4 Market Structure and Pricing

Send correspondence to:
Alfred Lorn Norman
Department of Economics
The University of Texas at Austin
Austin, TX USA 78712-1173
or send E-mail to:
norman@eco.utexas.edu

1 Introduction

This is the third in a series of papers defining a procedural model of a consumer. In this paper we consider the procedures consumers use to find a single item in their preferred bundle of goods.

Let us start by considering the standard calculus consumer utility model:

$$\max U(x_1, x_2, \ldots, x_n) \quad \text{subject to} \quad p_1 x_1 + p_2 x_2 + \cdots + p_n x_n = I \qquad (1)$$

where $U(x_1, x_2, \ldots, x_n)$ is the utility function, x_i is the amount of the ith good, p_i is the price of the ith good, and I is the income. If a consumer were to solve this problem with a numerical optimization technique, he or she might substitute the linear constraint into the objective function, calculate the gradient given a starting bundle of $n-1$ goods, and proceed iteratively using a conjugate gradient or variable metric algorithm. In general (1) is not computable because the algorithms will converge asymptotically. (See Velupillai (2000) for a comprehensive discussion of computability in economics.) An extremely superficial interpretation of Simon's satisficing would be that the consumer stops iterating after a finite number of steps with an ϵ approximation rather than converging to the optimum.

Nevertheless, what we observe in the marketplace is that consumers shop for their marketbaskets item-by-item and almost never consider bundles. Also, in the marketplace almost all purchases are made in discrete units and even those like gasoline are discretized to the smallest unit of coinage. Now, let us motivate why consumers almost always search for the items in their preferred bundle of goods item-by-item. Consider a grocery store organized so that consumers proceed down a line of shopping carts already filled with alternative bundles of goods and select the cart with the preferred bundle. To obtain some perspective why this alternative economic organization does not exist, let us assume that the number of goods categories equals 30 and that the number of alternatives in each category equals 10. For example, a typical goods category might by breakfast cereal. These numbers are

very conservative for a modern grocery store. In this case the number of shopping carts equals 10^{30}.

In experiments with pens, we found that a subject could make a binary comparison between two pens in 3.2 seconds, Norman et al (forthcoming a). If it only took 10 seconds to make a binary comparison between two bundles of 30 items, then it would take 1.59 $\times 10^{23}$ years to find the preferred bundle. If each bundle were placed in a shopping cart 3 feet long then the consumer would have to travel 5.68 $\times 10^{27}$ miles just to view all the bundles. The consumer shops item-by-item rather than as a bundle so that the number of alternatives increase linearly rather than multiplicatively with each category. Sellers organize their merchandise item-by-item and not by bundles to vastly reduce the required display space.

Even when searching for a single preferred item, consumers can face a very large number of choices. For example, the Tower Record store near the University of Texas campus has over 90,000 music CDs. The Barnes and Noble bookstore at the Arboretum has over 275,000 titles displayed for sale. If you examine the 1997 brochure for the Chevrolet Cavalier, there are over 20,000 possible combinations of options, colors and accessories for the Cavalier convertible alone.

In Section 2, we construct procedural models of these rules in the form of algorithms. In this paper we do not consider budgeting, which was considered in Norman et al (forthcoming b). For the case of a very large number of alternatives, these algorithms can be studied using the computational complexity concepts discussed in a brief survey in Appendix A. We show that the psychological rules found in the literature are linear rules. For computers linear processes are considered tractable; however, large linear processes are not tractable for humans because we process an operation in seconds not nanoseconds.

We show that Tversky's (1972) linear elimination-by-aspects (EBA) rule can be modified to create a sublinear set-selection-by-aspects (SSBA) rule. In Section 3 we perform an experiment to test whether subjects are approximately rational in

switching from an SSBA rule to a EBA rule when searching for an apartment with certain characteristics. In Section 4, we consider the organization of goods in stores both at physical locations and in cyberspace to show that stores organize goods to make the application of an SSBA rule feasible.

Finaly, in Section 5 we conclude.

2 Theory

In this paper we use computational complexity analysis in order to provide tools to study the case of a very large number of alternatives. Huber (1980) and Johnson (1979) proposed studying decision strategies by their elementary information processes. We provide a brief introduction to computational complexity or combinatorial complexity in Appendix A.

We shall model a consumer's selection of a preferred item from a set of goods and services. Then we will determine the complexity of consumer search using each of several alternative decision rules. At this point in the analysis our sole concern is identifing the equivalence classes of various consumer decision rules.

2.1 Compensatory Decision Rules

We will first show that finding a preferred item in a set using a binary comparison operator is a linear search.

Consider a simple model of a consumer searching for a preferred item from a finite set of close substitutes $X = \{x_1, x_2, \ldots, x_n\}$. The number of alternatives is n and each item has m attributes. For example, if X were a set of household vehicles, one attribute would be body type – 2-door, 4-door, SUV, SUV small, pickup, pickup small, sports car. Other attributes would be 2 or 4 wheel drive, the body color, and so on. The consumer determines the values of the attributes of x_i using the information operator, $D(x_i) = \{a_{i1}, a_{i2}, \ldots, a_{im}\}$ that has the property of completeness. Thus A is an $n \times m$ matrix with row i representing the values of

the attributes of x_i.

In order to determine preference between two items, x_i and x_j, consumer executes a binary ranking operator, $R(a_i, a_j) \rightarrow x_i \succeq x_j$ or $x_i \prec x_j$, that has the properties of completeness, reflexivity and transitivity. The execution of R compares the respective attribute values of rows a_i and a_j. In Appendix A we show that the compensatory decision rules proposed by psychologists are linearized variations of a utility function that can be formulated as variations of the binary comparison operator. As increasing experimental evidence since Tversky (1969) shows that humans are not always transitive, we point out that it does not change the computational complexity properties of the search, only the possibility that the outcome might be inconsistent.

The goal is to find a preferred item defined as follows: $x_i^* \in X$ such that $x_i^* \succeq x_j$ for all $x_j \in X$. Finding a preferred item with a ranking operator is a variation of finding the largest number in a set of numbers. Compare the first item with the second, then compare the larger with the third, and so on. Let $B(a_i, a_j) \in \{R, R_v, R_\triangle, R_I\}$ where $R_v, R_\triangle, R_I$ are defined in Appendix A.1.

Preferred Item Search Algorithm: $Pref$

Step 1: $max = 1$ $i = 1$ Perform $D(x_{max})$

Step 2: As long as $i < n$, repeat step 3.

Step 3: Let i increase by one. Perform $D(x_i)$. If $B(a_{max}, a_i) \neq x_{max} \succeq x_i$ then $max = i$. ◯

The algorithm terminates with max equal to the index of a preferred item.

Now let us consider the cost computational complexity of the problem of finding a preferred item with respect to the growth parameter n, the number of alternatives, and a fixed m, the number of attributes for each alternative. Thus, for this algorithm performed on s an element of the set of problems S consisting of the $n!$ combinations of the n items in the set, the cost function is:

$$C(Pref[s]) = C(D) \times L(D) + C(B) \times L(B) \tag{1}$$

where $C(\cdot)$ is the cost of executing the argument and $L(\cdot)$ counts the number of times the argument has been executed.

Now let us consider the cost of executing D and B. For simplicity, they are modeled as constant over the entire set of items. An inexperienced decision maker might gradually lower his or her information costs to some fixed value as the consumer develops an efficient procedure. In the case of such learning the costs are bounded between the initial costs and the efficient costs. The case of learning has no effect on the subsequent results provided there is a fixed lower bound. In the marketplace there would be considerable variation in executing these operators. The fixed cost can be considered the efficient average cost.

We will model D and B as elementary operations with costs c_D and c_B respectively. Since m is taken as a fixed constant, the cost of executing D is constant. B's execution cost is also fixed for each of the four possibilities, $R, R_v, R_\triangle$, and R_I. The first requires one nonlinear function evaluation over $2m$ arguments. The other three require $2m$ function evaluations over 1 argument and up to $2m$ arithmetic operations. We assume that all the functions are tractable by humans.

For this algorithm the worst case and expected computational complexity are the same because the algorithm must process every item and the cost does not depend on which combination is being processed.

Theorem 1: The worst case and expected computational complexity of finding a preferred item using D and B is n:

Proof: Any algorithm based on D and B must perform at least $n-1$ R operations to test all the items. Therefore the problem is bounded from below by $\Omega(n)$.

The preferred item search takes $n-1$ operations and n D operations; therefore, this algorithm is $O(n)$. By definitions D1-D3 in Appendix A the computational complexity of this problem is n. ◯

While the rules discussed in this section are linear, the absolute cost of exercising one of these rules can be very high when n is large. If a human is aided by a

computer, then linearly processing 2.5 million alternatives such as the list of books at Amazon.com is a tractable problem because even a personal computer can execute a billion operations per second. In contrast, a human may take seconds to execute one operation, and even at one second per item, a human would take almost 700 hours to linearly process the Amazon.com book list. Within the range of human capability, linear algorithms are not always tractable.

Up to now we have not assumed any organization of the set of items from which the consumer will choose the preferred item. Without a specified organization, the consumer has no criteria to create a reasonable stopping rule for satisficing performance. We will now show that consumers have more powerful rules than linear ones and that they face a market of goods organized to enable them to employ these more powerful rules.

2.2 Elimination by Aspects Rule

In this section we shall examine the elimination-by-aspects rule in order to determine in what sense it is more efficient than the compensatory rules considered in the previous section. Tversky [1972] defines the choice function underlying the elimination-by-aspects procedure formally. However, he defines the actual elimination-by-aspects procedure intuitively. The decision maker chooses an aspect and then eliminates all items which do not possess the aspect. He or she repeats the process until only one item remains.

To compare the EBA rule with the previously considered compensatory rules, we shall define the EBA as an algorithm beginning by defining an aspect. An *aspect* describes whether the item in question manifests some value or values for one or more of its attributes. For example, if the aspect is SUV small, then each of the items in X, a set of household vehicles, either has the aspect or does not have the aspect. We could also consider an aspect SUV small plus 4 wheel drive.

EBA Algorithm

Step 1: i = 1. Nomenclature: $X^0 = X$

Step 2: Define aspect γ_i over the m attributes.

Step 3: Sequentially examine each item in X^{i-1} and eliminate all those that do not possess aspect γ_i to obtain X^i.

Step 4: Repeat steps 2 and 3 until X^i contains one item. ◯

In order to compare the EBA rule with the previously considered compensatory rules we make two assumptions: (1) a consumer must make one information operation per item to determine whether the item possesses the aspect, and (2) m aspects $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_m\}$ are required to reduce the set X to X^m that contains only one item.

To consider the cost of executing EBA we must define the costs of the individual operations. Let D_{γ_i} and c_{γ_i} represent the operation and the cost of determining aspect γ_i respectively, and let D_{obs} and c_{obs} represent the operation and the cost of determining whether an item possesses γ_i. The cost function for the EBA algorithm is:

$$C(\omega[s]) = c_{\gamma_i} \times L(D_\gamma) + c_{obs} \times L(D_{obs}) \tag{2}$$

where as before $L(\cdot)$ counts the number of times the argument has been executed. To analyze this algorithm we define the worst case as each cycle of Steps 2 and 3 that removes only one item until the m^{th} cycle when $n-m$ items have been removed leaving X^m with one item. The expected case for each cycle removes a fraction $\frac{1}{k}$ truncated to an integer where k = 2,3,...

Theorem 2: The worst case and expected computational complexity of finding a preferred item using the EBA algorithm based on D_{γ_i} and D_{obs} is n.

The algorithm requires m D_γ operations. The number of D_{obs} operations is less than n for each of the m cycles in the worst case with the total D_{obs} operations less than mn For the expected case the number of D_{obs} operations for the first cycle is n , the expected number for the second is $\frac{1}{k}n$,..., the expected number for the m^{th} is $\frac{1}{k^m}$. The expected number of D_{obs} operations is $\leq \frac{k}{k-1}n$, which is the sum of the

associated geometric series. Thus the EBA algorithm is $O(n)$ in both the worst and expected cases.

n D_{obs} operations are required for the first cycle. Thus $\Omega(n)$ operations are needed. As these results do not depend on the order of the items in the search, the worst case and expected computational complexity are both n by definitions D1-D3 in Appendix A. ◯

While the EBA rule is a linear rule and therefore in the same equivalence class as the other rules considered, it is absolutely more efficient. This is because the EBA rule only requires the consumer to evaluate one aspect per item.

All of the rules considered so far have been linear rules. In appendix A.2 we show that other noncompensatory decision rules, such as the lexicographic, conjunctive, and disjunctive rules, also lead to linear decision rules. We shall consider a more efficient rule based on set information operators.

2.3 A Sublinear Decision Rule

One approach to creating a sublinear search rule involves making the information operator more efficient so that only one observation is required to determine the subset of items in X that possesses the aspect. Whether this is possible or not depends on how X is organized. For example, if all the items are heaped into a big pile, then the searcher will have to examine each item individually to observe the aspect under consideration. As we shall discuss in greater detail in a subsequent section, goods for sale are frequently organized by attributes. For example, a new car dealer organizes the models on his or her lot by make and model. Thus, a consumer in the phone directory can select the Honda dealers in one information operation and upon arriving at the lot can select the display of Civics in one information operation.

Let us now consider the efficiency of the EBA rule with a more powerful information operator, $Q(\gamma_i, X^{i-1}) = X^i$. We shall call this the set-selection-by-aspects

(SSBA) rule, where the information operator Q means that given X^{i-1} and γ_i the consumer can determine X^i in a single information operation with cost c_Q. For example, in a mens' clothing store, a consumer examining a circular rack of mens' pants selects in one operation all pants that have a particular waist band by observing the labels on the rack.

For the SSBA rule the order in which the γ_i are executed is crucial. For example, if a consumer in a metropolitan SMSA started a search for a new car by selecting that the car must have a CD player, the consumer would have to go to every car dealer, examine every car and create a giant list. Market organization dictates that a consumer can only apply the set information operator effectively for a small number of the possible sequences of aspects.

The SSBA is a refinement of Earl's (1986) clarification, characteristic filtering, of the EBA rule in which he pointed out that the sequence of EBA steps is not arbitrary, but ordered. In his examples, it is clear that the economic agents must sometimes be using set information operators and sometimes item information operators. Thus, the SSBA rule can be thought of as a clarification to Earl's characteristic filtering rule.

Also, market organization usually requires the consumer to switch to a linear rule before selecting the preferred item. In this subsection, however, we shall assume that the search items are organized so that the consumer can use the SSBA rule throughout the entire search in order to characterize its efficiency. We also assume that for a given sequence of Γ, X could be organized so that the SSBA rule could be applied.

SSBA Algorithm

Step 1: i = 1. Nomenclature: $X^0 = X$

Step 2: Define aspect γ_i over the m attributes.

Step 3: Execute $Q(\gamma_i, X^{i-1})$to obtain X^i.

Step 4: Repeat steps 2 and 3 until X^i contains one item. ◯

The cost function for the SSBA algorithm performed on problem set element s is:

$$C(SSBA[s]) = c_{\gamma_i} \times L(D_\gamma) + c_Q \times L(Q). \tag{3}$$

where $L(\cdot)$ counts the number of times the argument has been executed.

Theorem 3: The expected and worst case computational complexity of the SSBA algorithm is 1 (constant).

The SSBA algorithm requires no more than m D_γ or m Q operations; hence it is $O(1)$. Given the data structure, the SSBA algorithm requires at least m D_γ operations and at least m Q operations; hence the SSBA algorithm is $\Omega(1)$. The number of operations does not depend on the order of the items; hence by definitions D1-D3 the expected and worst case computational complexity of the SSBA algorithm is constant. ◯

To clearly understand the relationship between the EBA and SSBA rule assume that there are r values for each attributes and that $n = r^m$. Then, the SSBA rule constitutes a logarithmic reduction in cost from the EBA rule.

The SSBA rule is a powerful rule that offers searchers the possibility of better performance in finding a preferred item. We need to consider two questions related to this rule.

1. Are humans approximately rational in its use?
2. Are markets organized to facilitate its use?

3 Design of SSBA experiment

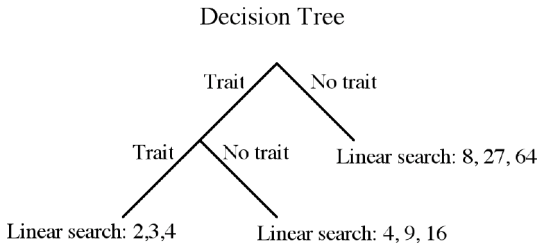
The purpose of the experiment is to test whether subjects are rational in the use of decision rule involving a set information operator. The experiment investigates how many times subjects use a SSBA rule before switching to a linear rule when searching for a specified apartment. By varying the time delay on the sublinear and linear rules we can control when it is rational to shift from the sublinear rule to the linear rule and see if subjects vary their shift points optimally with respect to the

changing time delays. The shift point is determined by the relative time delays and the number of alternatives.

Subjects were given incentives to find an apartment with specified traits from the following table as fast as possible :

No. Bedrooms	Location	Price
1	Riverside	\$200-300
2	West Campus	\$300-400
3	Hyde Park	\$400-500
4	Far West	\$500-600

To find the specified apartment, the subject may click on either trait buttons or object buttons. The trait buttons select those apartments with the specified trait and eliminate the rest. The object buttons represent randomly ordered apartments; clicking an object button tests that item individually to see if it has the desired attributes. The decision process that is three levels deep is represented as:



Initially there are 8 possible apartments in experiments 1 and 4, 27 possible apartments in experiments 2 and 5, and 64 apartments in experiments 3 and 6. If the subject clicks on the correct trait button, the number of possible apartments is reduced to 4, 9 or 16 respectively. If the subject clicks again on the correct trait button, the number is further reduced to 2, 3 or 4.

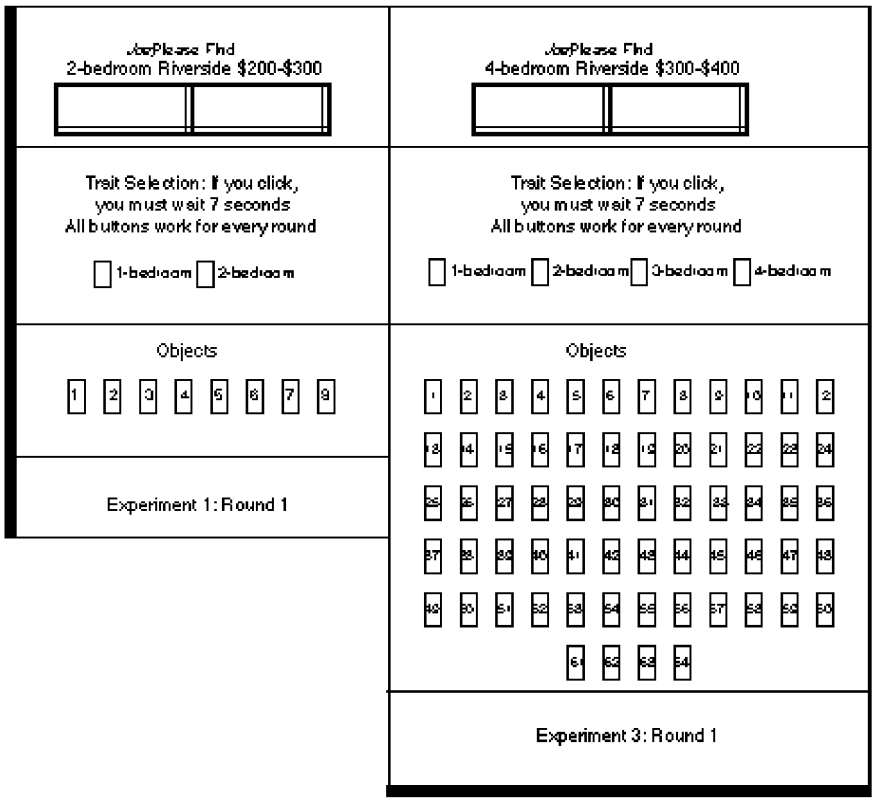
The subject might click on the trait buttons twice to reduce the options to 2, 3 or 4 individual buttons. They also could click on the trait button only once and then switch to a linear search over the 4, 9, or 16 remaining object buttons. Alternatively, they can simply start with a linear search by clicking on up to 8, 27, or 64 object buttons without using trait selection at all.

Whether it is rational to click on a trait button, reducing the number of object buttons over which a linear search must be conducted, depends on two factors. One is the longer delay incorporated into the trait buttons versus the 1 second delay incorporated into the object buttons. For experiments 1-3 the trait delay is 7 seconds and for experiments 4-6 the trait delay is 3 seconds. The other factor is how many alternatives of each trait are included in the experiment. The data are shown in the table below:

Exp	Trait Delay	No of Traits	No of Buttons	Apartment to Find
1	7 sec	$2 \times 2 \times 2$	8,4,2	2 bed-Riverside-$200-300
2	7 sec	$3 \times 3 \times 3$	27, 9, 3	1 bed-Hyde Park-$300-400
3	7 sec	$4 \times 4 \times 4$	64, 16, 4	4 bed-Riverside-$300-400
4	3 sec	$2 \times 2 \times 2$	8,4,2	1 bed-West Campus-$200-300
5	3 sec	$3 \times 3 \times 3$	27, 9, 3	3 bed-Riverside-$300-400
6	3 sec	$4 \times 4 \times 4$	64, 16, 4	3 bed-Far West-$300-400

To illustrate the decision process facing each subject, we display side by side the initial interfaces for experiments 1 and 3:

Initial interfaces for Experiments 1 and 3



First the subjects see the trait buttons for the number of bedrooms on top and the individual buttons for all the apartments on the bottom. Suppose in the first experiment the subject clicks on button for 2 bedrooms in experiment 3. The program eliminates all apartments that do not have 2 bedrooms and redraws the frame. In the second screen, the subject has the choice of 4 location trait buttons on top and a row of 16 individual apartment buttons on the bottom that all have 2 bedrooms. Now suppose the subject clicks on the Riverside trait button. The program redraws the frame with no trait buttons and a row of individual buttons on the bottom that all have the traits of 2 bedrooms and the Riverside location.

The subject clicks on the individual apartment buttons in succession until he or she finds the specified apartment, in this case the one priced $300-400.

Alternatively, the subject could simply click on the individual object buttons on the bottom and never click on any of the trait buttons. Instead of narrowing his search, the subject would look at all the possible apartments until he finds the correct one. Subjects must decide whether clicking on the trait buttons or clicking only on the object buttons is faster.

The text areas at the top of the interfaces provide the subjects with useful information. The text area on the right displays "Go" to inform the subject that the buttons will respond to a click. The text area on the left displays messages such as "wrong button" or ""FANTASTIC, YOU FOUND THE RIGHT ONE. Start the next round when the frame is redrawn."

In the experiment, each of the 25 undergraduate subjects was paid a flat $7 for participating and could win $10, 5, 3, or 2 if they had the lowest, second lowest, third lowest, or fourth lowest times. These incentives encouraged subjects to click on the buttons as fast as possible. There were three identical rounds in each experiment. At the end of three rounds the total time for the experiment is shown in the left frame and the winners are paid. Prior to starting the paid experiments the subjects performed 6 practice rounds. They differed from the paid rounds in two respects. The numbers of alternatives in each round were 2×2, 3×3, 4×4 and in the first round the subject had to use the trait buttons, in the second round the subject had to use only object buttons, and in the third the subject could use either. The purpose of the practice rounds was to ensure that the subjects had an intuitive grasp of the speeds at which the alternative decisions executed.

3.1 Experimental Results

Since the object buttons are in random order, a subject would on average have to click on 1/2 of them to find the specified apartment. The expected delay time in a

linear rule is compared with the delay time for a SSBA rule in the following table:

Expected time delays for SSBA versus Linear Rule

Exp ↓	1st Trait		2nd Trait		3rd Trait
Rule →	SSBA	Linear	SSBA	Linear	Linear
1	7	4	7	2	1
2	7	13.5	7	4.5	1.5
3	7	32	7	8	2
4	3	4	3	2	1
5	3	13.5	3	4.5	1.5
6	3	32	3	8	2

The mean number of times the trait and object buttons were clicked are displayed below:

	Trait Buttons			Object Buttons	
Exp	Forecast	Mean	SD	Mean	SD
1	0	0.35	0.55	3.70	2.44
2	1	0.77	0.60	7.49	6.65
3	2	1.39	0.53	7.21	6.85
4	1	0.72	0.70	2.94	1.90
5	1 or 2	1.32	0.54	4.71	4.87
6	2	1.79	0.41	3.96	3.98
SIG		.0001		.0001	

As expected, the subjects increased their use of the trait buttons when the total number of alternatives increased and when the trait execution delay decreased.

But, are these differences significant? The experimental design is a 3 × 2 balanced factor design with the dependent variable, *level*, 0, 1, or 2 levels of trait buttons clicked. The first factor is *size*, the number of alternatives: 2 × 2 × 2, 3 × 3 × 3, or 4 × 4 × 4; and the second factor is *delay*, 7 or 3 second delay on the execution of the trait buttons. The results are significant as shown in the table below:

Analysis of Variance Procedure
Dependent variable: *level* Independent variables: *size* and *delay*

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	80.12	16.02	50.88	0.0001
Error	336	105.82	0.31		
Corrected Total	341	185.94			

The Duncan test indicates that the two delays have significantly different affects and three sizes have significantly different affects. Another important consideration is whether the 6 practice experiments were sufficient for the subjects to develop their trait/object button strategy. The experimental design is a three factor design with *trait* being the dependent variable and *round*, that is round 1, 2, or 3, as the independent variable. If the subjects were changing their strategy in the experiments with prizes the means will be significantly different. The fact that there is no change in performance is shown in the following table:

Analysis of Variance Procedure
Dependent variable: *level* Independent variables: *round*

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	1.27	0.63	1.16	0.31
Error	339	184.68	0.54		
Corrected Total	341	185.94			

With the exception of experiment 1, subjects on average used slightly fewer trait buttons than forecasted. A much larger sample would be required to demonstrate that this shift is significant. If we consider the winners' use of the trait buttons, they generally used the trait buttons more often as the number of alternatives increased and the delay time on the trait button decreased, but they too used the trait button slightly less often than one would expect from the table of expected delays. For example, the winner of experiment 5 used the trait button once each round instead of twice, although the difference in time is slight. But, in experiment 4, he did not use the trait button and in experiment 6 he used it twice each round.

The experiment indicates that the subjects in general and the winners in particular were approximately rational in their use of the trait and object buttons.

4 Market Organization

Books on retail design, such as Reiwoldt (2000), Barr and Broudy (1986), and Barr and Field (1997), emphasize the visual display of goods and the creation of an appropriate ambiance for prospective customers. They do not, however, emphasize

that competition among sellers provides incentives for those sellers to organize their merchandise in a fashion that reduces the search costs of the consumer. Consider two stores that offer over 10,000 items for sale with the same selection and prices. The first store simply places the merchandise in a big heap without any sort of organization, forcing the consumer to search through the heap in order to find a preferred item. The second store organizes the merchandise into display cases by attributes. Most consumers would prefer the organized store because the average search costs will be lower if they can perform several SSBA steps before switching to linear rules. Such reasoning is implicit in store design and becomes explicit when there is a problem. For example, Redjacket redesigned its Virgin megastores to help customers locate departments and travel between them, Staff (2001).

What enables consumers to apply SSBA steps in their search is not that stores specifically organize goods in sets defined by the consumer's aspects. Rather, when stores organize goods in a nested structure by attributes, they make SSBA steps feasible. For example, clothing stores organize women's and men's clothes in different areas of the store. Within the men's clothing area, goods are further organized into sports clothes, suits, underclothes and so on. Also, automobile dealers organize their new car lots by make and model. This organization enables buyers to use many selection-by-aspect steps on aspects defined over the attributes because sellers provide customers with labels to recognize sets, organize goods in patterned displays which customers learn to recognize, organize goods in catalogues hierarchically through indices, and on web sites, provide search algorithms that return the set with the specified characteristics.

To perform several SSBA steps, the consumer must define a sequence of aspects consistent with the goods' organization. Then he or she can execute several SSBA steps in the search for the preferred item. For example, if a consumer started the search for a new car by insisting that the car had a CD player, the consumer would have to search every new car lot item by item to determine the set of new cars with

CD players. To use an SSBA rule in the initial steps of a search, the consumer must start with aspects that can be determined in a single operation such as the make and model of a car.

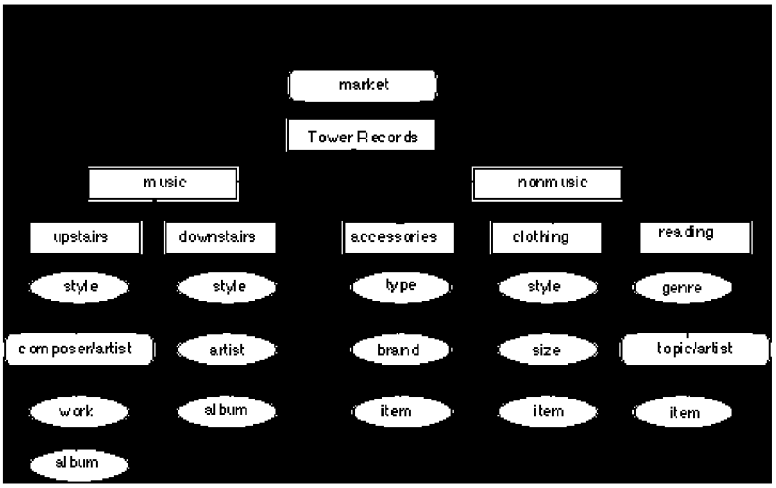
Organizing goods in a nested structure by attributes simplifies the administration of a store and makes searches by potential customers more efficient. The profit motive of the store overlaps with the desire for an efficient search for a consumer, but the two are not identical. Organization of goods in stores also has a marketing aspect which attempts to attract customers to goods that are the most profitable.

We will now consider in more detail the organization of markets in physical space and cyberspace.

4.1 Physical Space

We shall consider the organization of a department store, apartment listings, and automobile dealers' displays.

Tower Records in Austin is a CD department store that carries roughly 93,000 CDs. To increase the efficiency of consumers' searches Tower Records is organized by attributes such as musical styles and further organized into musical groups/composers that are organized alphabetically, and then by album as is shown in the figure below.



As can be seen there are two basic attributes: *music* and another to *nonmusic*. *Music* then splits into two branches, upstairs and downstairs. Both of these have almost identical structures, however upstairs also contains the choice of *composer*, along with *artist*, thus *upstairs* and *downstairs* cannot be combined. If a consumer wanted a recording of a Phillip Glass work, the consumer would execute SSBA steps to move upstairs, select a style, the artist Phillip Glass, and then can look for specific works. If all the CDs were in a heap, a consumer would need about 26 hours to select a particular recording using a linear rule if he or she could observe one CD per second on average. The organization of Tower Records is typical of a department store where the customer can perform several SSBA steps before switching to linear rules.

Now let us consider a student at the University of Texas at Austin might searching for apartment in physical space using: (1) the classified section of the *Austin-American Statesman*, (2) an apartment guide, or (3) an apartment locator service. Because there are about 1000 possible rentals typically listed in the Austin newspaper, a linear rule to find an apartment might take as many as 110 hours assuming the student made an appointment and drove to personally evaluate each rental. Each of these alternative processes is organized to allow students to use SSBA steps in their search.

The classified advertisements for rentals are first organized by type as (1) unfurnished apartments, (2) condos/townhomes, (3) unfurnished duplexes, or (4) unfurnished houses. Then each type is further organized by location in the Austin SMSA. The *Greater Austin Apartment Guide* provides data on over 250 apartment complexes divided into 5 sets by location. If a consumer knew the desired Austin location and rental type, he or she could use a SSBA step over this combined aspect before switching to a linear procedure. An apartment locator asks the student for his or her preference over price, location and number of bedrooms. In a single SSBA operation (from the perspective of the student) the locator shows most students no

more than four apartments to be evaluated by linear rules.

Now let us consider the organization of automobile displays at car dealerships. In the market for automobiles tens of thousands of items exist, a number that makes it impossible to operate a search using an item-by-item rule. At any given time there are approximately 300 brand new cars in any given lot. If a buyer were to test-drive each car at just one dealership for 20 minutes it would take him or her over four days of non-stop driving.

The organization of automobile data and auto displays at dealerships enables the consumer to execute several SSBA steps in a search for a preferred vehicle. Austin's Capital Chevrolet Dealership divides its lot into cars, trucks, sport utility vehicles, and vans. The main areas are then further classified by style. Then a potential customer, for example, could execute SSBA rules to examine alternative colors and options on an SUV. The Buick dealership has a slightly different approach. Buick has five different models. In the showroom they display two of each model, one custom version and one limited model. The customer walks around the showroom and decides which model they would like to purchase, rather than walking around in a parking lot. The salespeople at Buick cater to the more focused buyers and ask the consumers questions to determine the desired aspects. This requires an SSBA rule because the customer is deciding exactly what he or she wants in a car. On the outside, the Buick dealership is divided into five sections, similar to the Chevrolet divisions, and the salespeople can then select the set of cars that fits the given description.

4.2 Cyberspace

In cyberspace the sequence of decision rules is frequently presented to the consumer as a decision aid. Let us start with a student searching for an apartment in cyberspace. One such site is ApartmentGuide.com. On the first page the student would select the state and city. On the second page the student would select the

areas of town in which the student desires to live; the site provides maps for those who want them. The third page asks the student to indicate a price range, the preferred number of bedrooms and any desired amenities. The program presents the student with a list of all apartments matching the specified qualities. The various apartment complexes provide the prospective renter with the details concerning the apartment and apartment complex.

The new cards decision guide at AutoTrader.com is one example of a decision aid for selecting an automobile. The consumer starts by selecting the body type, such as sedan or SUV. Next, the consumer selects the maximum price. Third, the consumer ranks a number of options, such as air-conditioning, from no opinion to must-have. Then the consumer ranks a number of safety options. Next, the consumer specifies the gas mileage range, the number of cylinders, and the type of brakes. Then the consumer indicates a preference over the number of passengers the vehicle can carry, the number of doors and the amount of room per passenger. The consumer is asked to accept or reject various manufacturers and finally ranks the criteria to order the list of final selections.

Another site that has decision aids for a variety of products is ActiveBuyersGuide.com. These decision rules are compensatory and the consumer is asked questions to determine the weights among the various attributes of the product. Decision guides provide the consumer with quick, low cost tools to run scenarios that help assess the importance of various criteria. In addition many sites enable a consumer to create a table of alternative products with the attributes listed side by side. This is much more efficient than obtaining the same information from brochures for the various products. For most consumers, a necessary step in selecting an apartment is viewing the apartment, just as a key step in selecting an automobile is test-driving the vehicle. Internet sites can provide consumers with an efficient method of narrowing the search to final candidates that would then require direct sensory evaluation.

5 Concluding Remarks

A next topics in this series are (1) the information structure facing the consumer (2) consumer errors and (3) how decision aids affect performance.

References

1. Aho, A. V., Hopcroft, J. E., and Ullman, J.D., (1974), *The Design and Analysis of Computer Algorithms* (Reading: Addison-Wesley)

2. Barr, V and C. Broudy, (1986), *Designing to Sell*, (New York: McGraw-Hill)

3. Barr, V and K. Field, (1997), *Stores: Retail Display and Design*, (Glen Cove : PBC International)

4. Earl, P., (1986) *Lifestyle Economics*, (New York: St. Maritn's Press)

5. Hogarth, R. (1987) *Judgement and Choice* (New York: John Wiley and Sons)

6. Huber, O. (1980) The influence of some task Variables on Cognitive Operations in an Information-Processing Decision Model. *Acta Psychologica*, 45, 187-196.

7. Johnson, E. (1979) Deciding how to decide: The effort of making a decision Unpublished manuscript The University of Chicago

8. Knuth, D. E., (1973) *The Art of Computer Programming* (Addison-Wesley: Reading)

9. Newell, A. and H. Simon (1972) *Human Problem Solving* (New Jersey: Prentice Hall)

10. Norman, A., A. Ahmed, J. Chou, K. Fortson, C. Kurz, H. Lee, L. Linden, K. Meythaler, R. Rando, K. Sheppard, N. Tantzen, I. White and M. Ziegler, (forthcoming a), An Ordering Experiment, *Journal of Economic Behavior and Organization*

Available at http://www.eco.utexas.edu/Homepages/Faculty/Norman/innovation_central.htm

11. Norman, A., J. Chou, M. Chowdhury, A. Dalad, K. Fortson, M. Jindal, K. Payne, and M. Rajan, (forthcoiming b) Two Stage Budgeting, A difficult problem,

Computational Economics Available at URL listed above.

12. Pagan, F. G., (1981) *formal specification of programming languages* (Prentice-Hall: Englewood Cliffs)

13. Payne, J., Bettman, J., and Johnson, (1993) *The Adaptive Decision Maker* (Cambridge University Press-New York)

14. Riewoldt, O. (2000), *Retail Design* (Hong Kong: Lawrence King)

15. Simon, H. A., (1976), From substantive to procedural rationality, S. Latsis (ed), *Method and Appraisal in Economics*, (Cambridge University Press, Cambridge)

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

17. Staff, (2001) Virgin Megastores launching new design with new store in London *Design Week* 6, 16 August

18. Traub, J., Wasilkowski G. and Woźniakowski H. (1988), *Information Based Complexity*, (Boston: Academic Press, Inc.).

19. Tversky, A., (1969), "Intransitivity of Preferences", *Psychological Review*, Vol.76, No 1., 31-48

20. Tversky, A. (1972), Elimination by Aspects: A Theory of Choice, *Psychological Review*, Vol. 79, No. 4, 281-299.

21. Velupillai, K. (2000), *Computable Economics*, (Oxford: Oxford University Press)

A Computational Complexity

To study decision rules, the decision process is modeled as a computer algorithm, which could be defined in a variety of ways. Traub, Wasilkowski and Woźniakowski (1988) define an algorithm very abstractly as a function from the problem element to the solution range. A common practice is to define algorithms with respect to a particular language, such as a quasi-Algol, Aho, Hopcroft, and Ullman (1974),

or a particular computer model such as a Turing machine. Algorithms can also be described formally in computer languages using the Backus-Naur Form specification, Pagan (1981). In this paper we shall follow the example of Knuth (1973) and describe in English the algorithms that will be sequences of information operators and decision operators. This is an acceptable compromise between rigor and the need to communicate to a general audience.

Computer scientists compare computer algorithms based on the time required for execution, the cost involved, or the required memory. To make such comparisons the time, cost or memory are usually parameterized with respect to an important attribute of the problem, such as n, the number of items to be ordered. We can make these comparisons with a finite n, or make the comparisons as n increases. Computer scientists find the later method advantageous because asymptotically, only the dominant factor in an algorithm matters.

We now present the framework for the computational complexity analysis, which is standard combinatorial complexity with the addition of information operators. To define the cost function for an algorithm φ which solves a problem element $s \in S$, the problem set, we use the operators $\Upsilon = \{\upsilon_1, \upsilon_2, \ldots \upsilon_m\}$. These operators are the information and computational operators used in the algorithm φ. The cost function, C is: $C(\varphi[s]) = \sum c_{\upsilon_i} \cdot (\text{numberof } \upsilon_i \text{ operations})$ where c_l is the unit cost of executing υ_l. Similarly, we can also define a time function, T: $T(\varphi[s]) = \sum t_{\upsilon_i} \cdot (\text{numberof } \upsilon_i \text{ operations})$ where t_l is the unit time of executing υ_l. For the rest of the discussion we shall just consider the cost function. The development of the time function would be the same.

For this discussion, we shall only consider the worst case analysis by defining the cost function for an algorithm φ which will solve all $s \in S$ as: $C(\varphi[S])) = \sup_{s \in S} C(\varphi[s])$. The development of the expected case is similar. To solve S the consumer uses an efficient search, that is an algorithm φ^*; $\varphi^*(S) = \inf_{\varphi \in \Phi} C(S)$ where Φ is the set of all algorithms which solve S. Establishing that the cost of

an efficient consumer ordering algorithm has a particular cost (or time) function with respect to n is an example of computational complexity analysis. To define the computational complexity of S let $Y = Y(n)$ be a nonnegative function that we wish to compare with the cost function, $C = C_\varphi(n)$. Frequently Y is n, n^2 etc. Consider the following definitions:

D1. C is $O(Y)$ if there exist $i, j > 0$ such that $C(n) \leq jY(n)$ for all $n > i$.

D2. C is $\Omega(Y)$ if there exist $i, j > 0$ such that $C(n) \geq jY(n)$ for all $n > i$.

D3. S has computational complexity Y if there exists an algorithm $\varphi_i \in \Phi$ such that C_{φ_i} is $O(Y)$ and for all algorithms $\varphi_j \in \Phi$, C_{φ_j} is $\Omega(Y)$.

The concept of computational complexity divides problems into equivalence classes to facilitate comparison of the "difficulty" in problem solving. With these definitions, problems can be identified as easy, (for example, members of the n equivalence class) or hard, (for example, members of the exponential equivalence class).

It is important that the reader have a intuitive grasp of the difference between a sublinear decision rule and a linear decision rule. Consider a male customer searching for a pair of pants in a department store. The racks of pants are organized by waist band size, 28,30, 32 inches, and so on. The consumer can select all pants with a particular waist size, say 36, in a single operation. But to determine the pant length of pants with a waist band of 36 inches, he must examine each pair of pants in the set of pants with a waist band of 36 inches. Selecting the set of pants with waist band 36 is a sublinear decision; selecting a particular pant length is a linear decision. In the former the consumer can use one information operation to select a set and in the latter the consumer must perform one information operation per item.

B Other Rules

In order to keep the body of the paper focused we present other decision rules in this appendix.

B.1 Linear Compensatory Rules

Let us consider some linear compensatory models proposed by psychologists:

$$V(x_i) = \sum_j w_j \times U(A_{i,j}) \tag{4}$$

This equation gives the value of x_i as the sum of the products of the weight w_j for each aspect times the utility or scale value of that aspect $U(A_{i,j})$.

Two other variations of the linear model are the additive difference model and the ideal point model. The additive difference model can be specified as:

$$V_\triangle(x_i, x_j) = \sum_j w_j \times U_j(A_{i,j} - A_{k,j}) \tag{5}$$

The terms in the summation could be either positive or negative depending on whether the utility of the difference in the j^{th} option between x_i and x_k is positive or negative. The ideal point model is also a variation of the linear compensatory model:

$$V_I(x_i) = \sum_j w_j \times U_j(I_j - (A_{i,j})) \tag{6}$$

In this value function the utility or scale function measures the utility of the difference between the ideal I_j and the actual.

The following binary comparison operator can be constructed from these linear value functions:

$$\left.\begin{array}{l} R_v(x_i, x_j) \\ R_\triangle(x_i, x_j) \\ R_I(x_i, x_j) \end{array}\right\} \rightarrow x_i \succeq x_j \text{ if } \left\{\begin{array}{l} V(x_i) \geq V(x_j) \\ V_\triangle(x_i, x_j) \geq 0 \\ V_I(x_i) \geq V_I(x_j) \end{array}\right\} \text{ else } x_i \prec x_j \tag{7}$$

B.2 Other Noncompensatory Decision Rules

We now consider the lexicographic, conjunctive and disjunctive decision rules.

Lexicographic preferences are frequently used in economic analysis, for example Encarnacion (1987). The options for lexicographic rules are quantified with more being better than less, option by option as specified below:

$$x_i \succ x_j \text{ if } \begin{cases} a_{j1} > a_{j1} \\ a_{ik} = a_{jk} \text{ for } k = 1, 2, \ldots, v < m \quad a_{i,v+1} > a_{j,v+1} \end{cases} \tag{8}$$

To illustrate the lexicographic procedure let us represent each option as $Z_{ij} = a_{ij}/a_{max,j}$ where $a_{max,j}$ is the maximum of the j^{th} column of A. Consider the following Z matrix:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & .9 \\ 1 & 1 & 1 & 1 & 1 & .3 & .8 \\ 1 & 1 & 1 & 1 & .6 & .3 & .7 \\ 1 & 1 & 1 & .3 & .5 & .6 & .6 \\ 1 & .1 & .4 & .6 & .4 & .7 & .5 \\ 1 & .6 & .1 & .6 & .2 & .1 & .4 \\ .6 & .5 & .2 & .5 & .1 & 0 & .3 \end{pmatrix}$$

The lexicographic rule can be viewed as a variation of the elimination-by-aspects procedure where the order in which the columns will be processed is given, but processing each column requires an extra step, the determination of $a_{max,j}$. If we assume that determining the maximum must be performed by binary comparisons, then the determination of the maximum of a sequence of numbers is a minor variation of a preferred item search algorithm where $a_{ij} \succeq a_{ik}$ if $a_{ij} \geq a_{ik}$. Searching an entire column has complexity n. It is easy to show that since m is taken as fixed, the worst case and expected complexity of the lexicographic algorithm is n.

Now let us consider conjunctive and disjunctive decision rules. Hogarth (1987) describes a conjunctive model as "one in which the decision maker sets certain cutoffs on the dimensions such that any alternative which falls below a cutoff is eliminated." The number of variables upon which a cutoff has been established is $r = m$ and let the cutoffs be $\{k_i : i = 1, 2, \ldots, m\}$. The conjunctive procedure is to find x^* such that $a_{*,i} \geq k_i$ for $i = 1, 2, \ldots, m$.

In the disjunctive model a decision maker might permit a low score on a dimension provided there is a very high score on one of the other dimensions. There are several ways in which this psychological decision rule could be defined. Let us define it as $x^* = a_{*,1} \geq k_1$ or $a_{*,2} \geq k_2$ or ... or $a_{*,m} \geq k_m$. Defined this way the conjunctive rule is an 'AND' rule in terms of Boolean logic and the disjunctive rule is an 'OR' rule. For comparison purposes with the elimination-by-aspects rule we shall assume that only one item meets all the criteria for both the conjunctive and disjunctive rules and that the decision maker is not aware of this fact. The execution of either rule would require a single pass through the n items and it is easy to show that the worst case and expected complexity for both is n

Affiliation(all): The University of Texas at Austin