A General Theory of Information Cost Incurred by Successful Search

William A. Dembski[1]*, Winston Ewert[2] and Robert J. Marks II[2]

[1]*Discovery Institute, 208 Columbia Street, Seattle, WA 98104.* [2]*Electrical & Computer Engineering, One Bear Place #97356, Baylor University, Waco, TX 76798–7356.* (**Corresponding author: dempski@discovery.org*)

Abstract

This paper provides a general framework for understanding targeted search. It begins by defining the search matrix, which makes explicit the sources of information that can affect search progress. The search matrix enables a search to be represented as a probability measure on the original search space. This representation facilitates tracking the information cost incurred by successful search (success being defined as finding the target). To categorize such costs, various information and efficiency measures are defined, notably, *active information*. Conservation of information characterizes these costs and is precisely formulated via two theorems, one restricted (proved in previous work of ours), the other general (proved for the first time here). The restricted version assumes a uniform probability search baseline, the general, an arbitrary probability search baseline. When a search with probability q of success displaces a baseline search with probability p of success where $q > p$, conservation of information states that raising the probability of successful search by a factor of $q/p(>1)$ incurs an information cost of at least $\log(q/p)$. Conservation of information shows that information, like money, obeys strict accounting principles.

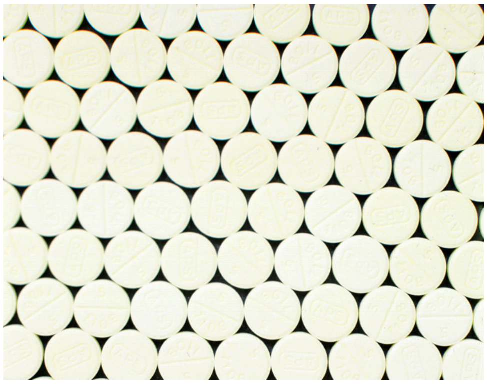
Key words: Search matrix, targeted search, active information, probabilistic hierarchy, uniform probability, conservation of information

1. The Search Matrix

All but the most trivial searches are needle-in-the-haystack problems. Yet many searches successfully locate needles in haystacks. How is this possible? A successful search locates a target in a manageable number of steps. According to conservation of information, nontrivial searches can be successful only by drawing on existing external information, outputting no more information than was inputted [1]. In previous work, we made assumptions that limited the generality of conservation of information, such as assuming that the baseline against which search performance is evaluated must be a uniform probability distribution or that any query of the search space yields full knowledge of whether the candidate queried is inside or outside the target. In this paper, we remove such constraints and show that conservation of information holds quite generally. We continue to assume that targets are fixed. Search for fuzzy and moveable targets will be the topic of future research by the Evolutionary Informatics Lab.

In generalizing conservation of information, we first generalize what we mean by targeted search. The first three sections of this paper therefore develop a general approach to targeted search. The upshot of this approach is that any search may be represented as a probability distribution on the space being searched. Readers who are prepared to accept that searches may be represented in this way can skip to section 4 and regard the first three sections as stage-setting. Nonetheless, we suggest that readers study these first three sections, if only to appreciate the full generality of the approach to search we are proposing and also to understand why attempts to circumvent conservation of information via certain types of searches fail. Indeed, as we shall see, such attempts to bypass conservation of information look to searches that fall under the general approach outlined here; moreover, conservation of information, as formalized here, applies to all these cases.

In first generalizing targeted search before generalizing conservation of information, we introduce the search matrix. The elements that constitute the search matrix may be illustrated as follows. Imagine a gigantic table that is miles in both length and width. Covering the table are upside-down dixie cups that are tightly packed, such as the following hexagonal packing:



Under each dixie cup resides a single pea. The cups are opaque, so the peas are not visible unless the cup is lifted. The peas come in two varieties, high-yield and low-yield (the difference being that high-yield peas, if planted, produce lots of peas whereas low-yield peas produce only few). The low-yield peas far outnumber the high-yield peas. Our task is to locate a high-yield pea. The high-yield peas therefore form the target. Because the table is so large and the cups are tightly packed, for a human to try to walk around the table and turn over cups is infeasible.

We therefore imagine a remote-controlled toy helicopter flying over the table, hovering over individual cups, and lifting a given cup to examine the pea under it. Each act of lifting a cup to examine the pea under it constitutes a single query.

Because the table is so large and the high-yield peas are so few, this search constitutes a needle-in-a-haystack problem. As with all such problems, the number of queries (i.e., attempts to locate the needle, or, in this case, to locate a high-yield pea) is strictly limited. Moreover, because the needle is so small in relation to the haystack, unless the queries are chosen judiciously, the needle will in all likelihood elude us. Our search therefore is limited to at most *m* queries, which we call the *sample size*. This, in the case at hand, is the maximal number of dixie cups the search can turn over. We therefore imagine that the remote controlled toy helicopter flies over the gigantic table of dixie cups, hovers over a given cup, turns it over to examine the pea under it, replaces the cup, and then moves on, repeating this action at most *m* times.

Within the constraints of this scenario, how do we find the target? The helicopter has *m* queries in which to locate the target (i.e., to find a high-yield pea). At each query, the helicopter does three things:

(1) It identifies a given pea by removing and replacing the dixie cup over it.
(2) It extracts information that bears on the pea's probability of belonging to the target.
(3) It receives information for deciding where to fly next to examine the next pea.

The helicopter's search for the target may therefore be characterized as the following $3 \times m$ matrix, which we call the *search matrix*:

$$\begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_m \\ \alpha_1 & \alpha_2 & \alpha_3 & \dots & \alpha_m \\ \beta_1 & \beta_2 & \beta_3 & \dots & \beta_m \end{bmatrix}$$

Here the first row lists the actual peas sampled, the second row lists the information extracted about the peas that bears on their probability of belonging to the target, and the third row lists the information for deciding where the helicopter is to fly next. Moreover, each column represents a single query, with the columns listed in the order of search. A successful search is then one that on the basis of the search matrix explicitly identifies some x_i in the first row that belongs to the target.

Note that in this general search scenario, given columns may query the same element more than once. Thus, for separate columns that record x_i, x_j, x_k, etc., these first-row elements of the search matrix might all be identical. Such repetitions could occur if each time the helicopter lifts a dixie cup and queries a pea, it can extract only partial information about it, and so the search may need to query the pea again to obtain still more information about it. We could have distinguished between queries that lift a dixie cup to access a pea and queries that subsequently extract further information about a pea once the dixie cup is lifted. But since one may not want to query a pea immediately after having already queried it but rather wait until other peas have been queried (information about other peas might help to elucidate information about the given pea), in the interest of generality it is best to allow for only one type of query, namely, a combined query that lifts a dixie cup and then extracts information about the underlying pea.

The search matrix is not identical with the search. Rather, the search matrix records key information about the progress of the search. Specifically, it records the elements sampled from the search space, including any repetitions (this information appears in the first row); it records information bearing on the probability that an element sampled belongs to the target (this information appears in the second row); and it records information for deciding where to sample next (this information appears in the third row). All this information contained in the search matrix comes to light through the activity of a search algorithm. Success of the search therefore depends on how effectively the algorithm uses as well as fills in the information contained in the search matrix. Does the search algorithm, in sampling x_i, have complete memory of the prior information sampled? Or is it a Markov process with access only to the latest information sampled? Does the algorithm contain additional information about the target so that regardless of the information in rows two and three, the algorithm will, with high probability, output an x_i that is in the target? Or is the target-information available to the algorithm restricted entirely to the search matrix in the sense that its probability of successfully locating the target depends entirely on the information contained in those two rows [2]? The options here are wide and varied.

We consider next several examples of how the search matrix might work in practice.

Example 1.1: Uniform random sampling with perfect knowledge

In this case, each x_i is selected according to a uniform distribution across the dixie cups, each α_i records whether x_i belongs to the target (1 for yes, 0 for no), and each β_i directs the helicopter to take a uniform random sample in locating the next point in the search space (that being x_{i+1}). The reference to "perfect knowledge" here signifies that for each query we know exactly whether the pea sampled (each x_i)

belongs to the target (in which case $\alpha_i = 1$) or not (in which case $\alpha_i = 0$). If any α_i equals 1, we can stop the search right there and produce x_i as an instance of a successful search. Alternatively, we can fill out the search matrix rather than leave it incomplete, and then produce the first x_i for which α_i equals 1 (producing simply x_1 if none of the α_is equals 1). Given that the proportion of high-yield peas (i.e., the target) has probability p, the probability that this search is successful is $1-(1-p)^m$.

Example 1.2: Uniform random sampling with zero knowledge

In this case, as before, each x_i is selected according to a uniform distribution across the dixie cups; moreover, each β_i directs the helicopter to take a uniform random sample in locating the next point in the search space (that being $x_i + 1$). This time, however, examining the peas reveals nothing about whether they belong to the target. This might happen, for instance, if high-yield and low-yield peas are visually indistinguishable and we have no way of otherwise discriminating them (as we might through genetic analysis or actually planting them). The reference to “zero knowledge” therefore signifies that for each query we know nothing about whether the pea sampled (x_i) belongs to the target. In this case, each of the α_is may be treated as equal to 0. Given that the proportion of high-yield peas (i.e., the target) has probability p, the probability that this search successfully identifies a particular x_i in the target is simply p. Accordingly, a sample size of m greater than 1 does nothing here to improve on the probability of locating the target if we have no means of obtaining knowledge about the peas we are sampling.

Note that the probability that some element in the first row of the search matrix belongs to the target is $1-(1-p)^m$. This is the probability of successful search as calculated in the previous example, which presupposed perfect knowledge. Nevertheless, for a search to be successful in the present example, it is not enough for the search matrix merely to have a target element appear in the first row. In addition, it must be possible to explicitly identify one element in the first row taken to be the best candidate for belonging to the target. Moreover, because this is a needle-in-the-haystack problem, successful search requires that the candidate selected must belong to the target with probability considerably larger than p. With zero knowledge about whether elements in the first row of the search matrix belong to the target, however, no candidate selected from that row stands a better chance of belonging to the target than any other. In that case, each such candidate has the very small probability p of belonging to the target.

Example 1.3: Uniform random sampling with partial knowledge

In this example, as in the previous two, each x_i, when first selected, follows a uniform distribution across the dixie cups. Yet, to determine whether a given x_i actually does belong to the target, two agricultural tests may need to be performed on it. The tests work as follows: if both yield a positive result (denoted by a 1), then the candidate x_i belongs to the target; if one or both yield a negative result (denoted by 0), then it does not belong to the target. Moreover, the performance of each of these tests requires a single query. Thus, to determine whether an x_i that is in the target actually does belong to it, the dixie cup over it will have to be removed and replaced twice, meaning that x_i itself will appear twice in the top row of the search matrix, implying that under those appearances the corresponding α_is will both be 1.

On the other hand, if on either of the tests, the first query performed yields a 0, then there's no point in performing the other test, and x_i need appear only once in the top row. Given a query that for the first appearance of x_i yields an α_i equal to 1, x_i will need to be queried again to determine whether it indeed belongs to the target. Once a given x_i's inclusion in or exclusion from the target is determined, the next query is uniformly random across the dixie cups. In this case, the probability p' of hitting the target over m queries will be strictly between the probabilities determined in the last two examples, i.e., $p < p' < 1 - (1 - p)^m$, where p is the zero-knowledge lower bound and $1 - (1 - p)^m$ is the perfect-knowledge upper bound. The exact value of p' will depend on Bayesian considerations relating to how negative results on the two agricultural tests are distributed (in terms of prior probabilities) among the non-target elements.

Example 1.4: Smooth gradient fitness with single peak

In this case, we begin by turning over a randomly chosen dixie cup, examining the pea under it (x_1), and recording its fitness (α_1). We assume that the fitness function over the peas has a smooth gradient (in other words, fitness does not zigzag up and down as we move in a given direction over the large table, which is our search space) and that it has a single peak (in other words, any local maximum is also the [unique] global maximum). In this case, a hill-climbing strategy is appropriate, so each β_i directs the helicopter to search in the neighborhood of the x_i that, so far in the search, has attained the highest value of fitness α_i, looking to increase the fitness still further. There's no reason in this case to repeatedly query a given pea since we assume that fitness can be precisely determined in a given query and that fitness does not vary from one query to another (in other words, the fitness function here is "time independent"). Once m queries in this search have been carried out, we consult the search matrix and choose, as our best candidate for landing in the target, the x_i that attains the highest value of fitness α_i. The probability that such a search is successful will depend on the sample size m, the initialization (i.e., the procedure for deciding where the search begins), the precise characteristics of the fitness function, and how efficiently the search samples points in the neighborhood of an already sampled point to improve fitness (i.e., to "climb the hill").

2. General Targeted Search

A precise mathematical characterization of the general search scenario described in the last section now looks as follows. Let $\Omega = \{\omega_1, \omega_2, ..., \omega_K, \omega_{K+1}, ..., \omega_N\}$ be the search space, which we assume to be finite (this assumption can be relaxed and we have done so in other work, but doing so entails no substantive gain in generality). Let $T = \{\omega_1, \omega_2, ..., \omega_K\}$ be the target and define the probability $p = K/N$. A search of Ω for T then consists of the following 6-tuple: $(\iota, \tau, O_\alpha, O_\beta, \mathcal{A}, \Delta)$. The items in this 6-tuple denote respectively the *initiator*, the *terminator*, the *inspector*, the *navigator*, the *nominator*, and the *discriminator*. Here is what these six items mean:

Initiator. The initiator ι, denoted by the Greek *iota*, starts the ball rolling. It is the procedure by which the search determines where to begin. The initiator ι is responsible for x_1, and possibly additional members of the search space x_2 through x_k, that appear as the first entries in the first row of the search matrix. In many searches the initiator does nothing more than choose a single search space element (i.e., x_1) at random according to some probability distribution.

Terminator. The terminator τ, denoted by the Greek *tau*, provides a stop criterion for ending the search. Because all searches are limited to a maximum number of queries m (i.e., the sample size), the terminator can always simply be identified with the policy to cut off the search after m queries. In practice, however, terminators often end a search before the maximal number of queries have been made because the search is deemed to have achieved success before this maximal number. In that case, the search matrix may be incomplete, with missing entries in the columns to the right of the last column for which a point in the search space was queried. Without loss of generality, we can then fill up the columns that are missing entries by repeating the last complete column. Alternatively, we can just leave the columns empty.

Inspector. The inspector O_α is an oracle that, in querying a search-space entry, extracts information bearing on its probability of belonging to the target T. We assume that O_α is a function mapping into some range of values capable of providing information about the degree to which members of the search space Ω give evidence of belonging to the target T. Quite often, the domain of O_α is merely Ω, and O_α maps into {0,1}, returning a 1 if an element of Ω is in the target, 0 otherwise. Alternatively, O_α may map into the singleton {0}, returning the same element regardless of the element of Ω in question, thus providing zero information about target elements. O_α may even assume misleading values, suggesting that search-space entries are in the target when they are not and vice versa. Besides taking on discrete values, O_α may also take on more continuous values, signaling the degree to which a search-space entry is likely to be included in a target, as with a fitness function. The possible forms that O_α can take are wide and varied. Without loss of generality, however, we assume that the range of values that the inspector can take is finite.

As the inspector, O_α's task is to fill the second row of the search matrix and thus provide evidence about the degree to which corresponding elements in the first row may belong to the target. Accordingly, all the α_is in the second row take values in O_α's range. Nevertheless, given that a single query may not provide all the information that the inspector is capable of providing about a given element from the search space, the inspector may perform multiple queries on a given search-space element and may even use information gained from different previously queried elements in answering the present query. Thus, given an element x_i in the search space that's just been selected, its value as assigned by O_α can depend on the entire partial matrix

$$\begin{bmatrix} x_1 & \cdots & x_{i-1} & x_i & ** \\ \alpha_1 & \cdots & \alpha_{i-1} & * & ** \\ \beta_1 & \cdots & \beta_{i-1} & * & ** \end{bmatrix}.$$

Here ellipses denote elements of the search matrix that have been filled in, single asterisks denote individual missing entries, and double asterisks denote possibly multiple missing entries. In the case at hand, O_α uses the partial search matrix given here to determine α_i. If it ignores all entries of the partial search matrix prior to column $i - 1$, then we say that O_α is *Markov*. If it determines α_i solely on the basis of x_i, we say that O_α operates *without memory* (otherwise, *with memory*).

Navigator. Like the inspector O_α, the navigator O_β is an oracle. Given that we are at

$$\begin{bmatrix} x_1 & \cdots & x_{i-1} & x_i & ** \\ \alpha_1 & \cdots & \alpha_{i-1} & \alpha_i & ** \\ \beta_1 & \cdots & \beta_{i-1} & * & ** \end{bmatrix}$$

in the search process, the navigator takes this partial search matrix and returns the value β_i, which directs (navigates) the search as it attempts to locate the next entry in the search space (i.e., x_{i+1}). O_β maps into a fixed range of values, which in the search matrix we denote by βs. As with the inspector, if O_β ignores all entries of the partial search matrix prior to column $i - 1$, then we say that O_β is *Markov*. If it determines β_i solely on the basis of x_i and α_i, we say that O_β operates *without memory* (otherwise, *with memory*).

The type of information that O_β delivers can be quite varied. It can provide distance of search-space elements from the target. It can provide information about the smoothness of fitness. It can provide information about how likely neighbors of a given search-space element are to be in the target. Moreover, it can combine these types of information. Whereas the inspector O_α confines itself to extracting information that bears on the probability of search-space elements residing in the target, the navigator O_β focuses on information that helps guide the search to the target. As with the inspector, we assume that the range of values the navigator may take is finite. For (mathematically) smooth fitness functions, this will entail discretizing the values that the fitness function may assume. Yet, because the degree to which searches can discriminate such information is always strictly limited (in practice, distinct measurements when sufficiently close become empirically indistinguishable), assuming a finite range of values for the navigator entails no loss of generality.

Nominator. The nominator $\mathcal{A}$ is the update rule that, given a search matrix filled through to the ith column and thus incorporating the most current information from the inspector and navigator, explicitly identifies (and thereby "nominates") the next element to be queried, namely x_{i+1}. Thus $\mathcal{A}$ takes us from the search matrix

$$\begin{bmatrix} x_1 & \cdots & x_i & * & ** \\ \alpha_1 & \cdots & \alpha_i & * & ** \\ \beta_1 & \cdots & \beta_i & * & ** \end{bmatrix}$$

to the updated search matrix

$$\begin{bmatrix} x_1 & \cdots & x_i & x_{i+1} & ** \\ \alpha_1 & \cdots & \alpha_i & * & ** \\ \beta_1 & \cdots & \beta_i & * & ** \end{bmatrix}$$

We denote the nominator by $\mathcal{A}$ (for "algorithm") because, in consulting the inspector and navigator to determine the next search-space element to be queried, it acts as the basic underlying algorithm of the search, running through all the target candidates that the search will consider. We say that the nominator is *Markov* if its selection of x_{i+1} depends solely on the ith column of the search matrix. We say that it operates without memory if its selection of x_{i+1} is independent of prior columns of the search matrix.

To say that the nominator nominates an element x_{i+1} based on the partial search matrix

$$\begin{bmatrix} x_1 & \cdots & x_i & * & ** \\ \alpha_1 & \cdots & \alpha_i & * & ** \\ \beta_1 & \cdots & \beta_i & * & ** \end{bmatrix}$$

may seem to entail a loss of generality since in many searches (e.g., genetic algorithms and particle swarms), multiple candidates from the search space tend to be generated in batches. Thus with genetic algorithms, for instance, all candidates of a given reproduction cycle appear at the same time. Accordingly, if, say, 100 offspring are generated at each reproduction cycle, the new partial search matrix is not

$$\begin{bmatrix} x_1 & \cdots & x_i & x_{i+1} & ** \\ \alpha_1 & \cdots & \alpha_i & * & ** \\ \beta_1 & \cdots & \beta_i & * & ** \end{bmatrix}$$

but rather

$$\begin{bmatrix} x_1 & \cdots & x_i & x_{i+1} & \cdots & x_{i+100} & ** \\ \alpha_1 & \cdots & \alpha_i & * & ** & * & ** \\ \beta_1 & \cdots & \beta_i & * & ** & * & ** \end{bmatrix}.$$

Given this last matrix, we can then let the inspector and navigator fill in the columns below x_{i+1} to x_{i+100} one column at a time proceeding left to right. Alternatively, we can simply require the nominator to proceed one column at a time (thus taking a given batch of candidates one by one in sequence), letting the inspector and navigator fill in that column before proceeding to the next. Both cases are mathematically equivalent. For some searches, it makes better intuitive sense for the nominator to nominate a whole batch of search-space elements at a time. But this can always be made equivalent to nominating one element of the batch at a time until the entire batch is exhausted. For simplicity, we tend to adopt this latter approach. Another possibility is to change the search space so that each element consists of multiple elements from the original search space (see example 3.5).

Discriminator. Once a search matrix

$$\begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_m \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \beta_3 & \cdots & \beta_m \end{bmatrix}$$

that's compatible with $\mathcal{A}$ has been formed, it's time to decide which x_i that appears in the first row is most likely to belong to the target T. With a complete search matrix in hand, it's not enough to suspect that some entry somewhere in the first row belongs to T. For the search to be successful, we need to know which of these entries in fact belongs to T or, if definite knowledge of inclusion in T is not possible, then which of these entries is more likely than the rest to belong to T. Choosing from the first row of the search matrix the most likely candidate that belongs to T is the job of the discriminator Δ. As such, the discriminator is a function into the search space Ω from possible search matrices (i.e., from $3 \times m$ matrices whose first row consists of elements from Ω, whose second row consists of elements from the range of the inspector, and whose third row consists of elements from the range of the navigator). For each such search matrix, the discriminator outputs the element x_i in the first row that it regards as most likely to belong to T.

Discriminators can vary in quality. *Self-defeating discriminators* that, whenever possible, select first-row entries belonging outside the target are an option. For a given search matrix, such discriminators minimize the probability of successfully locating the target. Also an option are *independent-knowledge discriminators* that can identify whether a first-row entry belongs to the target with greater certainty than is possible simply on the basis of the information delivered by the inspector and navigator (information found in the second and third rows of the search matrix). Thus, the discriminator might have access to a source of information about target inclusion that is less ambiguous than what is available to the inspector and navigator. Such discriminators would thereby introduce information external to the search matrix to locate those elements in the first row most likely to belong to the target. By contrast, *no-independent-knowledge discriminators* would select x_i from the first row based solely on information contained in the second and third rows of the search matrix. Such variations among discriminators are easily multiplied and formalized. We leave doing so as an exercise to the reader.

Although the discriminator Δ as here described is a function from complete search matrices to the search space Ω, in fact we allow Δ also to be a function from partial search matrices to Ω, in keeping with the terminator's ability to stop a search when success in fewer than m queries has likely been achieved. Recall that partial search matrices can always be filled up with redundant columns and thus turned into complete search matrices. Hence, allowing partial search matrices entails no gain in generality, nor does restricting ourselves to complete search matrices entail a loss of generality.

Each of the six components of a search $S = (\iota, \tau, O_\alpha, O_\beta, \mathcal{A}, \Delta)$ can be stochastic. Thus, the initiator might choose x_1 according to some probability distribution.

Likewise, the terminator may end the search depending on chance factors relevant to success being achieved. The inspector and navigator, at any given stage in forming the search matrix, may draw on a stochastic source to randomize its outputs. So too, the nominator and discriminator may choose their candidates in part randomly. It follows that a search S can be represented as a random search matrix consisting of three discrete stochastic processes X, Y, and Z:

$$S = \begin{pmatrix} X_1 & X_2 & \dots & X_m \\ Y_1 & Y_2 & \dots & Y_m \\ Z_1 & Z_2 & \dots & Z_m \end{pmatrix}.$$

Here X represents the search-space elements delivered by the nominator (or the initiator for X_1), Y the corresponding outputs of the inspector, and Z the corresponding outputs of the navigator. X therefore takes values in Ω, Y in the range of O_α, and Z in the range of O_β.

Alternatively, S can be conceived as a vector-valued stochastic process $\vec{W}$ where each

$$\vec{W}_i = \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix},$$

in which case

$$S = \left(\vec{W}_1 \ \vec{W}_2 \ \cdots \ \vec{W}_m \right).$$

Applying the discriminator Δ to this random search matrix thus yields an Ω-valued random variable $\Delta(S)$, which we denote by X_S. As an Ω-valued random variable, X_S therefore induces a probability distribution μ_s on Ω that entirely characterizes the probability of S successfully locating the target T. In this way, an arbitrary search S can be represented as a single probability distribution or measure μ_s on the original search space Ω. This representation will be essential throughout the sequel.

As noted at the start of this paper, this representation of searches as probability measures is central to our formalization of conservation of information. If it were obvious that searches could in general be represented this way, we might just as well have omitted these first three sections. But given that a general characterization of targeted search is itself a point at issue in determining the scope and validity of conservation of information, these preliminary sections were in fact necessary. Logically speaking, however, these sections come up only tangentially in the sequel by guaranteeing that searches can indeed be represented as probability measures.

3. Search Examples

In this section, we consider several further examples of targeted search, expanding on the examples given at the end of section 1.

Example 3.1: Uniform random sampling with perfect knowledge and without replacement

In the last section, we considered a search of m queries in which, at each query, the entire search space was sampled uniformly. This led to independent and identically distributed uniform random variates in the first row of the search matrix, 0s and 1s in the second row depending on whether the corresponding entry in the first row was respectively outside or inside the target, and in the third row a directive simply to continue uniform random sampling. The discriminator in this case simply looked for a first-row entry with a 1 directly below it in the second row. Accordingly, with uniform probability $p = K/N$ of the target T in the search space Ω, we calculated the probability of successful search at $1 - (1 - p)^m$. This probability, however, assumes that the first row of the search matrix was sampled *with replacement* and thus may repeat elements of the search space.

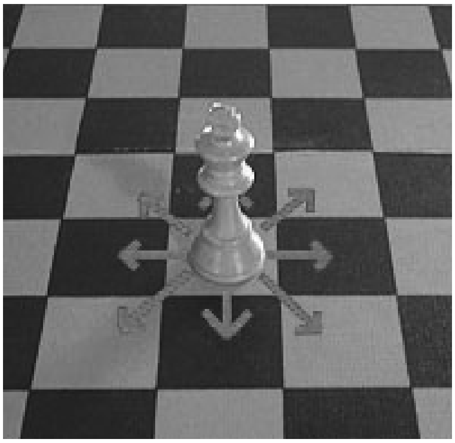
We can, on the other hand, have the navigator direct the search to avoid elements in the search space previously queried (*this implies* a memory of previously queried elements). If all other aspects of the search are kept the same, the search space is then sampled *without replacement* so that each query is uniform with respect to elements of the search space yet to be queried. This sampling procedure yields a hypergeometric distribution. Thus for $\Omega = \{\omega_1, \omega_2, ..., \omega_K, \omega_{K+1}, ..., \omega_N\}$, $T = \{\omega_1, \omega_2, ..., \omega_K\}$, $p = K/N$, and m not exceeding $N - K$, the probability that this search locates the target is then

$$1 - \frac{\binom{N-K}{m}}{\binom{N}{m}}.$$

Moreover, if N is much larger than m, this probability approximately equals the "with replacement probability" $1-(1-p)^m$, underscoring the well-known fact that sampling without replacement only negligibly improves the efficiency of search compared to sampling with replacement unless the sample size m is large [3].

Example 3.2: Easter egg hunt

Imagine a very large two-dimensional grid with Easter eggs hidden under various squares of the grid. You are able to move around the grid by going from one square to an adjacent square. Thus you can move one square vertically, horizontally, or diagonally, like a king in chess:



You start out on a randomly chosen square, which is determined by the initiator. The terminator gives you at most m squares to examine. When you are on a given square, the inspector tells you whether you are over an Easter egg (by saying "yes") or not (by saying "no"). If "yes," uncover the square on which you are standing, locate the egg underneath, and end the search.

Given that you have moved from one square to another with neither being over an Easter egg, the navigator tells you whether the square you are currently on is closer to, the same distance from, or further from the nearest Easter egg (by saying "warmer," "same," or "colder"; distance between squares A and B is calculated as minimum number of steps needed to reach B from A). Notice that the navigator cannot provide such information until the initiator has designated the first square and the nominator has designated the second. Thus, for the very first square chosen by the initiator, the navigator simply puts down "same."

If for your current square the navigator says “warmer,” the nominator says to choose that square from your immediate neighborhood that takes you in the same direction as your last move. If for your current square the navigator says “same,” the nominator says to choose at random a square that you have not yet visited in the immediate neighborhood of the current square. If for your current square the navigator says “colder,” the nominator says to return to the previous square and randomly choose a square in its immediate neighborhood that you have not yet visited. Proviso: the nominator ignores any column with “colder,” in subsequent search treating it as though it were not part of the search matrix except for not revisiting its square when sampling nearest neighbors. This proviso prevents the search from getting stuck. Finally, the discriminator returns the first square under which an Easter egg was found if an egg is indeed found; otherwise, it returns the square chosen by the initiator.

The Easter egg hunt so described falls within our general framework for targeted search.

Example 3.3: Competitive search

In competitive search, elements of the search space Ω are conceived as “players” whose skill can be evaluated and ranked according to certain “performance criteria.” Evolutionary computing typically employs a single performance criterion given by a fitness function. Fitness thus provides a *single-objective* measure of optimality — one and only one thing needs to be optimized, and when it is optimized we have the undisputed best player. In many circumstances, however, optimality is *multi-objective*, that is, there are several competing things we are trying to optimize simultaneously, where a rise in one leads to a drop in another. Optimization with multiple performance criteria thus requires a balancing or compromise among rival objectives. How these criteria are balanced determines what we regard as the “best players,” that is to say, the target.

Just what constitutes the right balance of performance criteria is not written in stone but constitutes a judgment call [4]. Consider a search space consisting of all college men’s basketball players in a given year. Professional NBA teams are seeking the best basketball players in this search space — the very best presumably being the player picked first in the first round of the NBA draft. But what determines the best players? Many performance criteria come to mind: speed, height, field-goal percentage, three-point percentage, average number of rebounds per game, average number of blocked shots per game, average number of assists per game, etc. etc. All these performance criteria need to be suitably combined to determine who are the best players and thus what constitutes the target of the search. Some years, this balancing of performance criteria is straightforward, so that one player stands out head and shoulders above the rest. At other times, different teams may have different needs, leading them to emphasize certain performance criteria over others, so that no player is completely dominant and no target is universally agreed upon.

How we combine and balance performance criteria depends on our needs and interests. Suppose, to change examples, you are a college admissions officer. Your search space is all graduating high school students and you are trying to find those who will thrive at your institution. This is your search, that is, to find the "right" students. Prospective students need to take the Scholastic Aptitude Test (SAT). The test provides two main scores, a verbal and a math score (each varying between 200, which is worst, and 800, which is best) [5]. Each of these scores corresponds to a performance criterion and requires a search query. With these two queries performed on each high school student, how do you now select the best students for your school (leaving aside other performance criteria such as GPA and recommendations)? Do you add the scores together, as is commonly done? Or do you weight one more than the other and, if so, how?

If your school focuses mainly on liberal arts, you will want to weight the verbal portion more strongly than the math portion. Thus, even though you may want to see a combined score of 1200 or better, you will favor students who get a 750 verbal/450 math over students who get a 450 verbal/750 math. If, on the other hand, yours is an engineering school, then you will prefer the latter over the former. Some schools don't discriminate the two scores but simply add them to give a combined performance measure for the test. Besides adding scores or weighting them, one can introduce arbitrary cut-offs. Thus, one might require that no student be admitted who performs less than 500 on either test, thereby ensuring that both verbal and math scores exceed a certain threshold. This suggests a maxi-min approach to combining performance measures: take the minimum of the two SAT scores and try to recruit those students whose minima are maximal. The precise formulation of such combined performance measures is straightforward. The trick is to find the right combination that suits one's purposes.

In such examples of competitive search, evaluating how a search space element fares with respect to the various performance criteria is the job of the inspector. To evaluate a search space element's competitiveness, the inspector may need to query it several times. Sometimes, however, a single query is enough.

In basketball, for instance, a player whose free throw percentage is less than 10 percent can be eliminated from consideration for the NBA draft without needing to consult any other performance criteria. Alternatively, a player who scores over 100 points a game on average (a performance achieved just once in the entire history of the NBA, as it is, by Wilt Chamberlain) will rise to the very top of the player pool even if we don't know any of his other stats. Knowing which queries to make conditional on which queries have already been made is essential to constructing an effective competitive search.

Example 3.4: Tournament play

Tournament play is a special case of competitive search in which the players display their competitive abilities by playing against each other. In tournament play, there are as many performance criteria as there are players, each player's competitiveness being gauged by how well one performs in relation to the others. Basketball is an example of tournament play, though in this case the unit of search is not the individual player (as it was in the last example) but the team. In chess, on the other hand, the unit of search tends to be the individual player (though team play is also known, as when local chess clubs play each other).

Tournament play is typically represented by a square anti-symmetric matrix with blanks down the diagonal (players don't play themselves) and opposite outcomes mirrored on either side of the diagonal. For instance, in the St. Petersburg Chess Congress of 1909, the tournament matrix was as follows [6]:

St. Petersburg 1909

		1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9		
1	Rubinstein	*	1	1	1	½	½	½	1	1	1	½	1	0	1	½	1	1	1	1	14½	875 Rubles
2	Lasker	0	*	½	1	½	1	1	1	½	1	1	1	0	1	1	1	1	1	1	14½	875 Rubles
3	Spielmann	0	½	*	1	0	1	1	½	1	½	½	½	1	0	½	1	½	½	1	11	475 Rubles
4	Duras	0	0	0	*	0	1	½	0	½	1	0	1	1	1	1	1	1	1	1	11	475 Rubles
5	Bernstein	½	½	1	1	*	0	1	0	1	1	1	1	½	0	0	0	½	½	1	10½	190 Rubles
6	Teichmann	½	0	0	0	1	*	0	½	½	½	½	1	1	½	1	½	1	1	½	10	120 Rubles
7	Perlis	½	0	0	½	0	1	*	½	½	1	½	1	1	½	1	½	0	0	1	9½	80 Rubles
8	Cohn	0	0	½	1	1	½	½	*	0	0	1	½	½	0	½	½	½	1	1	9	40 Rubles
9	Schlechter	0	½	0	½	0	½	½	1	*	1	0	0	1	1	½	0	1	½	1	9	40 Rubles
10	Salwe	0	0	½	0	0	½	0	1	0	*	½	1	1	1	½	0	1	1	1	9	40 Rubles
11	Tartakower	½	0	½	1	0	½	½	0	1	½	*	0	0	0	½	1	1	1	½	8½	
12	Mieses	0	0	½	0	0	0	0	½	1	0	1	*	½	1	1	1	0	1	1	8½	
13	Dus Chotimirsky	1	1	0	0	½	0	0	½	0	0	1	½	*	½	½	½	1	0	1	8	
14	Forgács	0	0	1	0	1	½	½	1	0	0	1	0	½	*	½	½	½	0	½	7½	
15	Burn	½	0	½	0	1	0	0	½	½	½	½	0	½	½	*	1	½	½	0	7	
16	Vidmar	0	0	0	0	1	½	½	½	1	1	0	0	½	½	0	*	½	1	0	7	
17	Speyer	0	0	½	0	½	0	1	½	0	0	0	1	0	½	½	½	*	½	½	6	
18	Von Freyman	0	0	½	0	½	0	1	0	½	0	0	0	1	1	½	0	½	*	0	5½	
19	Znosko Borovsky	0	0	0	0	0	½	0	0	0	0	½	0	0	½	1	1	½	1	*	5	

Emanuel Lasker, who tied with Akiba Rubinstein for first place, was at the time the world champion. Even so, Rubinstein, though he never played Lasker in a title match (back then challengers had to raise sufficient funds before they could arrange such a match with the world champion), was in the five years preceding World War I regarded as the strongest player in the world (in 1912 he won five international tournaments in a row, a feat unparalleled). Yet both Rubinstein and Lasker were defeated by Dus Chotmirsky, a chess player who would be lost to history except for this feat.

In chess tournaments, winners are decided by summing performance across all games (assigning 1 to a win, ½ to a draw, and 0 to a loss) and then selecting the player(s) with the highest total. This is standard practice, but one can imagine variations of it. We might want simply to focus on victories and count them. In that case, Lasker would have won the tournament outright (with 13 victories), Rubinstein would have come in second (with 12 victories), and Duras would have come in third (with 10 victories). On the other hand, we might want to choose the victor based on fewest losses. In that case Rubinstein would have been the outright victor (with a single loss), Lasker would have taken second (with 2 losses), and Spielmann would have taken third (with 3 losses). Other options for balancing performance criteria in tournament play are possible as well. For instance, Chotmirsky, for having defeated the two top performing players as determined by conventional tournament standards, might have been rewarded extra points for doing so.

In tournament play, exhaustive search means each player playing all the other players and recording the outcomes. Most chess tournaments, however, have so many players that an exhaustive search is not possible. The St. Petersburg tournament was a select invitational meet. Most tournaments are open to the chess community. In such tournaments, players are initially matched in line with their official chess ratings (1600 for amateur, 2000 for expert, 2200 for master, 2500 for grandmaster), with weaker players initially playing stronger players so that the best players don't cancel each other out early. Then, as the rounds proceed (typically between six to eight rounds total), players with the same tournament record, or as close a record as available, are matched. Note that weaker players, as gauged by their rating coming into the tournament, tend at each round to be matched with stronger players. At the end of the tournament, one's wins and draws are summed (1 for a win, ½ for a draw). The tournament winner(s) are then decided in terms of this raw total as well as an algorithm that, in the case of ties, takes into account the strength, as determined by chess rating, of one's opponents.

In terms of the search matrix, especially when the pool of tournament players is quite large, each player can play only a few other players (each game played constituting a single query, the outcomes of these games being noted by the inspector). Given that one wants to discover the strongest players (for some specified method of balancing performance criteria, these players, taken jointly, constitute the target), the search needs to be judicious in the choice of players it uses to query individual players. Is it more effective, given at most *m* queries, to query as many players as possible by playing them against only one or a few other players? Or is it better to hone in on a few players and put them through their paces by having them play a wide cross-section of other players? It all depends. It depends on whether player strength tends to function transitively (if A is able to defeat B and B is able to defeat C, is A able to defeat C?). It depends on whether, in the case of a single player testing other players, this player is strong or weak.

The discriminating power of strong players is important in tournament play. A strong player, by definition, loses only to a few players (i.e., to other strong players), and thus will clearly discriminate strong from weak players. In contrast, a weak player, by losing to most players, will fail to discriminate all but the weakest players. To change the game from chess to baseball, if the test of a team is whether it performs well against the New York Yankees, any team that does reasonably well against them could rightly be considered excellent. But if the "test team" is drawn from the local little league, then it would not provide a useful way of determining teams of national excellence. But notice, using the New York Yankees as a test team may not be very useful either — a team that beats or keeps it close with the Yankees is surely top notch, but if all the teams tested fail miserably against the Yankees, we may learn nothing about their relative strength. Players that are overly strong or overly weak are poor discriminators of play excellence.

In sum, tournament play is a special case of competitive search that fits within our general search framework but in which performance is assessed by the players (i.e., the search space elements) playing each other directly and then by noting the winner and, if applicable, the margin of victory [7].

Example 3.5: Population search

For some searches, the concern is not simply with individuals exhibiting certain characteristics but with whole populations exhibiting those characteristics. Thus, in population genetics, the emergence of a trait in a lone individual is not enough. Traits can come and go within a population. The point of interest is when a trait gets fixed among enough members of a population to take hold and perpetuate.

To represent such a scenario, we might imagine all the members of a population as drawn from a set of individuals Ω'. Moreover, Ω' may contain a target T'

consisting of all members exhibiting the trait(s) in question. Yet the actual search is not for T' in Ω' but for sets, whether ordered or unordered, of members from Ω'; what's more, the target will consist of such sets that have a sufficient proportion of members from T'. Thus, in a standard evolutionary computing scenario, the actual search space Ω might consist of all 100-tuples of elements drawn from Ω' (each element of the first row of the search matrix would belong to this Ω). Moreover, the actual target T might, in this case, consist of 100-tuples drawn from Ω' for which 75 or more of their elements belong to T'. In this case, successful search would require 75 percent of the population to have acquired the given trait(s).

Many ways of transitioning from Ω' to Ω and T' to T are possible here depending on the population size (is it fixed or variable?), on whether the order of elements in the population is important, and on the threshold that determines whether individually determined characteristics are widespread enough for the population to have properly acquired them. Even though the natural search space may seem to be one we have called Ω', representing the search within the general framework outlined in this paper may require identifying another space, which we called Ω. The actual target we are trying to locate would thus belong not to Ω' but to Ω. Note that such an Ω will invariably have more structure than Ω', even supplying a metric of comparison in terms of how many members of Ω' the members of Ω share.

4. Information and Efficiency Measures

In a general theory of search that avoids arbitrary assumptions about underlying probability distributions, uniform probabilities nonetheless play a salient role. Consider our general set-up, a search space $\Omega = \{\omega_1, \omega_2, ..., \omega_K, \omega_{K+1}, ..., \omega_N\}$ with target $T = \{\omega_1, \omega_2, ..., \omega_K\}$, where the target has uniform probability $\mathbf{U}(T) = p = K/N = |T|/|\Omega|$, where $|*|$ is the cardinality of *. In any such scenario, we can always do at least as good as take a single uniform random sample and thereby attain a target element with probability p. We might conduct our search to improve this probability or we might conduct it to diminish this probability. The natural probability distribution attaching to Ω, given the idiosyncrasies of this search space, may be very different from uniform. But it is always, in principle, possible to enumerate the elements of a finite space Ω and choose one of them randomly so that no element is privileged over any other. Uniformity, even if destined to miss the target in any nontrivial search, is always an option.

To take a single uniform random variate from the search space Ω will be called the *null search*. This search becomes the baseline against which we compare all other searches. Any search different from the null search will be called an *alternative search* [8]. The null search induces the uniform probability distribution $\mathbf{U}$ on Ω (see section 2). This is the probability measure we get by setting our sample size at $m = 1$ and letting the discriminator act on the corresponding 3×1 search matrix whose first row element is simply a uniformly chosen element of the search space. In practice, p is so small that a null search over Ω for T is extremely unlikely to succeed. Success therefore demands that in place of the null search, an alternative search S be implemented that succeeds with a probability q that is considerably larger than p. The search S thus induces, in the notation of section 2, a probability distribution μ_s on Ω that entirely characterizes the probability of S successfully locating the target T. For simplicity, we denote μ_s simply by μ. In this way, an alternative search S reduces to a single probability distribution μ on the original search space Ω where the probability of the target is $\mu(T) = q$.

In comparing null and alternative searches, it is convenient to convert probabilities to information measures (note that all logarithms in the sequel are to the base 2). We therefore define the *endogenous information* I_Ω as $-\log(p)$, which measures the inherent difficulty of a blind or null search in exploring the underlying search space Ω to locate the target T. We then define the *exogenous information IS* as $-\log(q)$, which measures the difficulty of the alternative search S in locating the target T. And finally, we define the *active information* I_+ as the difference between the endogenous and exogenous information: $I_+ = I_\Omega - I_S = \log(q/p)$. Active information therefore measures the information that must be *added* (hence the plus sign in I_+) on top of a null search to raise an alternative search's probability of success by a factor of q/p.

In the null search, the sample size is fixed at 1 (a single uniform random variate is taken) whereas in the alternative search the sample size is m (m queries are made). If we make m explicit, then we can define q_m as the probability that the alternative search locates the target in m queries, and write $I_s^m = -\log(q_m)$ and $I_+^m = I_\Omega - I_s^m$. The behavior of I_+^m as a function of m now provides a measure of the efficiency of search. Suppose, for instance, that S conducts its search by taking independent and identically distributed random variates. In that case, assuming m to be much less than $1/p$, $q_m = 1 - (1 - p)^m$ is approximately equal to mp, and I_+^m is approximately $\log(m)$. If, instead, S conducts its search by, at each query, cutting in half the search space ("interval halving"), then the probability of finding the target increases by a factor of 2 for every query, and I_+^m is approximately m (i.e., the active information is linear rather than logarithmic in m). Interval halving is therefore a much more efficient search strategy (if it can be implemented) than uniform random sampling. I_+^m, as a function of m, therefore measures the efficiency of the search.

By comparing the performance of a search S against the endogenous information baseline I_Ω, I_+^m provides an absolute measure of efficiency of the search. Indeed, in specifying S, we define $I_+^m(S) = I_+^m$, conceived as a function of m, as the *absolute efficiency* of S. Given two searches, S and S', we define $(I_+^m(S' \mid S) = I_+^m(S) - I_+^m(S')$, again conceived as a function of m, as the *relative efficiency* of S' given S. Thus, if S represents uniform random sampling and S' represents interval halving, the relative efficiency of S' given S, $I_+^m(S' \mid S)$ is $m - \log(m)$. In general, for a given m, if S' induces a probability of r_m on T and if S induces a probability of q_m on T, then $I_+^m(S' \mid S) = \log(r_m/q_m)$. Absolute and relative efficiency can also be negative: for a given m, S does worse in locating the target than a single uniform random sample if and only if $I_+^m(S) < 0$; likewise, for a given m, S' does worse in locating the target than S if and only if $I_+^m(S' \mid S) < 0$. Note that if S represents a single uniform random sample, so that the search matrix has only a single column and is incomplete for all remaining $m - 1$ columns (the first entry in the first row is therefore a uniform random variate), then $I_+^m(S' \mid S) = I_+^m(S')$.

5. Liftings and Lowerings

Conservation of information tracks the information that goes into constructing a search, showing that the amount of information exhibited by the search in locating a target can never exceed the amount of information inputted in its construction. Accordingly, conservation of information addresses not just the search for a given target in the original search space, but also a search for the information that goes into rendering such a search successful. Conservation of information therefore is not about search per se but about the search for a search. In other words, it is about a higher-level search for the information required to render a lower-level search successful. We abbreviate “the search for a search” by S4S.

In section 2 we represented an arbitrary search (i.e., S) as a probability measure on a search space (i.e., μ_s). Given that the search for a search (S4S) is itself a search, it must likewise be representable as a probability measure. Such an S4S probability measure assigns probabilities to a higher order search space consisting of probability measures on the original search space. Formulating conservation of information requires the ability to project probability measures up and down a probabilistic hierarchy of search spaces. We show how this is done in this section. This section thus provides the formal background for the conservation of information theorems proved in the next section.

We consider again our general set-up, a search space $\Omega = \{\omega_1, \omega_2, ..., \omega_K, \omega_{K+1}, ..., \omega_N\}$ with target $T = \{\omega_1, \omega_2, ..., \omega_K\}$, where the target has uniform probability $\mathbf{U}(T) = p = K/N = |T|/|\Omega|$. We assume that $1 \leq K < N$. Define $\mathbf{M}(*)$ as the set of all Borel probability measures on * where * is any compact metric space. Ω, as a finite set, is compact in the discrete topology, which is given by any metric on it. Any probability measure m on Ω therefore has the form

$$\sum_{i=1}^{N} a_i \delta_{x_i},$$

where each a_i is a nonnegative real number, the a_is together sum to 1, and each δ is a point mass (assigning probability 1 to the corresponding x_i). It follows that **M**(Ω) is the set of all these convex linear combinations of point masses. Note that when each a_i equals 1/N, this sum of point masses is the uniform probability **U** in **M**(Ω).

We can think of the point masses δ_{x_i} (for $1 \le i \le N$) as N independent vectors in an N-dimensional vector space. Because these vectors are all added as convex linear combinations to form **M**(Ω), **M**(Ω) in fact sits in an (N – 1)-dimensional subspace, forming an N-simplex with Euclidean metric. Moreover, as a closed, bounded subset of Euclidean space, **M**(Ω) is compact. It follows that the uniform probability on **M**(Ω) is ordinary Lebesgue measure (suitably normalized). We denote this uniform probability over **M**(Ω) as $\bar{\mathbf{U}}$. $\bar{\mathbf{U}}$ resides in **M**(**M**(Ω)). For convenience, we therefore define $\mathbf{M}^0(\Omega) = def\, \Omega$, $\mathbf{M}^1(\Omega) = def\, \mathbf{M}(\Omega)$, $\mathbf{M}^2(\Omega) = def\, \mathbf{M}(\mathbf{M}(\Omega))$, and in general $\mathbf{M}^{j+1}(\Omega) = def\, \mathbf{M}(\mathbf{M}^j(\Omega))$.

Thus, to recap, the uniform probability **U** over Ω resides in **M**(Ω) and is defined as

$$\mathbf{U} = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i};$$

moreover, the uniform probability $\bar{\mathbf{U}}$ over **M**(Ω) resides in $\mathbf{M}^2(\Omega)$ and is isomorphic to normalized Lebesgue measure on the N-simplex $\{(a_1, \dots, a_N) \in \mathbf{R}^N \mid a_i \ge 0, \Sigma_{1\le i\le N} a_i = 1\}$. We call the various $\mathbf{M}^j(\Omega)$, taken together, the *probabilistic hierarchy* over the search space Ω. Note that we give each of these spaces in the probabilistic hierarchy the weak topology. It then follows by Prohorov's theorem that each of these spaces is compact (indeed, they form compact metric spaces in the Kantorovich-Wasserstein metric, which induces the weak topology on these spaces) [9][i].

The probabilistic hierarchy allows for considerable interaction among its measure spaces, so that structures associated with $\mathbf{M}^j(\Omega)$ have corresponding structures both up and down the hierarchy at $\mathbf{M}^{j+1}(\Omega)$ and $\mathbf{M}^{j-1}(\Omega)$. We speak of a structure at $\mathbf{M}^j(\Omega)$ projected up to $\mathbf{M}^{j+1}(\Omega)$ as a *lifting* and a structure at $\mathbf{M}^{j+1}(\Omega)$ projected down to $\mathbf{M}^j(\Omega)$ as a *lowering*. To see how this works, we take the higher-order space **M**(Ω) and the lower-order space Ω and examine how structures associated with these spaces can be projected to the other. Our discussion here will focus on the base of the probabilistic hierarchy (i.e., Ω and $\mathbf{M}(\Omega)$), but our observations readily generalize up the probabilistic hierarchy. Accordingly, structures associated with Ω may be *lifted* to structures associated with $\mathbf{M}(\Omega)$ and structures associated with $\mathbf{M}(\Omega)$ may correspondingly be *lowered* to structures associated with Ω.

To start, consider a real-valued function f on Ω (note that because Ω is finite and has a discrete topology, f is bounded, measurable, and even topologically continuous). The function f now lifts to a real-valued (continuous) function $\overline{f}$ on $\mathbf{M}(\Omega)$ that takes any probability measure θ in $\mathbf{M}(\Omega)$ and assigns its integral with f, i.e., $\overline{f}$ is the mapping from $\mathbf{M}(\Omega)$ to $\mathbf{R}$ such that

$$\theta \mapsto \int_\Omega f(x)d\theta(x).$$

Note that for $\theta = \delta_x$ (i.e., the point mass at x), $\overline{f}(\theta) = \overline{f}(\delta_x) = f(x)$. Call $\overline{f}$ the *lifting* of f from Ω to $\mathbf{M}(\Omega)$. Likewise, for F a real-valued function on $\mathbf{M}(\Omega)$, define $\tilde{F}$ on Ω as $x \mapsto F(\delta_x)$. Call $\tilde{F}$ the *lowering* of F from $\mathbf{M}(\Omega)$ to Ω. It then follows that $\tilde{\overline{f}} = f$, but it need not be the case that $\overline{\tilde{F}} = F$ (lowerings can lose information whereas liftings do not). In general, under the weak topology, liftings and lowerings of functions preserve measurability and continuity.

Next, consider a probability measure μ on Ω (μ is therefore in $\mathbf{M}(\Omega)$). Because Ω is finite, all probability measures in $\mathbf{M}(\Omega)$ are absolutely continuous with respect to the uniform probability $\mathbf{U}$. Absolute continuity of μ with respect to $\mathbf{U}$ means that every set of nonzero probability under μ also has nonzero probability under $\mathbf{U}$. By the Radon-Nikodym theorem, it follows that μ can be rewritten as the product of a density, denoted by $\frac{d\mu}{d\mathbf{U}}$, times the measure $\mathbf{U}$. This means that for $f = \frac{d\mu}{d\mathbf{U}}$, μ can be written as $f \cdot d\mathbf{U}$. In other words, for a set A contained in Ω,

$$\mu(A) = \int_A f(x)d\mathbf{U}(x)$$

In particular, if $\mu = \sum_{i=1}^{N} a_i \delta_{x_i}$, $f(x_i) = \frac{d\mu}{d\mathbf{U}}(x_i) = a_i \cdot N$.

It follows that by lifting f from a function on Ω to a function $\overline{f}$ on $\mathbf{M}(\Omega)$, we can now lift μ from a probability measure in $\mathbf{M}(\Omega)$ to a probability measure $\overline{\mu}$ in $\mathbf{M}^2(\Omega)$. Specifically, for B a measurable subset of $\mathbf{M}(\Omega)$, we define

$$\overline{\mu}(B) = \int_B \overline{f}(\theta)d\overline{\mathbf{U}}(\theta)$$

where

$$\bar{f}(\theta) = \int_{\Omega} f(x)\,d\theta(x).$$

To see that $\bar{\mu}$ is indeed a probability measure over $\mathbf{M}(\Omega)$, we need the following result.

Proposition 5.1 (Consistency of Uniformity)

$$\mathbf{U} = \int_{\mathbf{M}(\Omega)} \theta\, d\bar{\mathbf{U}}(\theta).$$

Remarks. The integral on the right side of this equation is vector-valued [10]. Such integrals exist provided that in applying continuous linear functionals to them (which, in this case, amounts to integrating with respect to all bounded continuous real-valued functions on Ω), one gets the same result as integrating over the continuous linear functions applied inside the integral. Linear functionals thereby reduce vector-valued integration to ordinary integration. Thus, the equality in the statement of this theorem means that for all continuous real-valued *h* on Ω,

$$\int_{\Omega} h(x)\,d\mathbf{U}(x) = \int_{\mathrm{M}(\Omega)} \left[\int_{\Omega} h(x)\,d\theta(x) \right] d\bar{\mathbf{U}}(\theta).$$

Because Ω is finite, all real-valued functions on Ω are continuous, so this equality needs to hold for all real-valued *h*. As we move up the probabilistic hierarchy, subsequent $\mathbf{M}^j(\Omega)$ are compact metric spaces, so continuity actually does place a restriction on the continuous linear functionals used in calculating vector-valued integrals. Because these are all compact metric spaces, existence and uniqueness of such vector-valued integrals is not a problem [11]. For equality to hold in $\mathbf{U} = \int_{\mathbf{M}(\Omega)} \theta\, d\bar{\mathbf{U}}(\theta)$ means that averaging all probability measures on $\mathbf{M}(\Omega)$ with respect to the uniform probability $\bar{\mathbf{U}}$ is just the uniform probability $\mathbf{U}$ on Ω. This establishes measure-theoretic consistency in lifting the uniform probability $\mathbf{U}$ on Ω to the uniform probability $\bar{\mathbf{U}}$ on $\mathbf{M}(\Omega)$.

Proof. This result follows from exchangeability — **U** is the only probability measure invariant under permutation of the elements of Ω. The vector-valued integral in question can immediately be seen to have this same property — its value does not depend on any point in Ω to which it is applied. A detailed proof is available elsewhere [12].

Suppose now that μ is a probability measure on Ω that is absolutely continuous with respect to $\mathbf{U}$ (in fact, because Ω is finite, this assumption holds for all probability measures on Ω). Let $\frac{d\mu}{d\mathbf{U}}$ denote the Radon-Nikodym derivative of μ with respect to $\mathbf{U}$ and let $\overline{\frac{d\mu}{d\mathbf{U}}}$ denote its lifting. If we now define the lifting of μ as $\bar{\mu} = \overline{\frac{d\mu}{d\mathbf{U}}} d\bar{\mathbf{U}}$, then $\bar{\mu}$ is a probability measure on $\mathbf{M}(\Omega)$. Moreover, since $\mathbf{U}$ is absolutely continuous with itself such that $\frac{d\mathbf{U}}{d\mathbf{U}}$ is identically equal to 1 on Ω, it follows that the lifting of $\frac{d\mathbf{U}}{d\mathbf{U}}$, i.e., $\overline{\frac{d\mathbf{U}}{d\mathbf{U}}}$, is identically equal to 1 on $\mathbf{M}(\Omega)$, and thus the lifting of $\mathbf{U}$, as so defined, is in fact the uniform probability on $\mathbf{M}(\Omega)$. Thus, whether we interpret $\bar{\mathbf{U}}$ as the uniform probability on $\mathbf{M}(\Omega)$ as ordinary Lebesgue measure (suitably normalized) on an N-simplex (which is isomorphic to $\mathbf{M}(\Omega)$), or as the lifting of the uniform probability $\mathbf{U}$ on Ω, both signify the same probability measure on $\mathbf{M}(\Omega)$.

To see that all the claims in the previous paragraph hold, it is enough to see that $\bar{\mu}$ is indeed a probability measure on $\mathbf{M}(\Omega)$, and for this it is enough to see that

$$\begin{aligned}\int_{\mathbf{M}(\Omega)} \overline{\frac{d\mu}{d\mathbf{U}}}(\theta) d\bar{\mathbf{U}}(\theta) &= \int_{\mathbf{M}(\Omega)} \left[\int_{\Omega} \frac{d\mu}{d\mathbf{U}}(x) d\theta(x) \right] d\bar{\mathbf{U}}(\theta) \\ &= \int_{\Omega} \frac{d\mu}{d\mathbf{U}}(x) d\left[\int_{\mathbf{M}(\Omega)} \theta \, d\bar{\mathbf{U}}(\theta) \right](x) \\ &= \int_{\Omega} \frac{d\mu}{d\mathbf{U}}(x) d\mathbf{U}(x) \text{ [by Cons. of Unif.]} \\ &= \int_{\Omega} d\mu \\ &= 1.\end{aligned}$$

Lastly, we need to be able to lift targets from Ω to $\mathbf{M}(\Omega)$. Thus, given the target T in Ω, we define a corresponding higher-order target $\bar{T}_q$ in $\mathbf{M}(\Omega)$, indexed by q in the unit interval ($0 \le q \le 1$), namely,

$$\bar{T}_q = \{\theta \in \mathbf{M}(\Omega) \mid \theta(T) \ge q\}.$$

$\bar{T}_q$ equals $\mathbf{M}(\Omega)$ when q is 0 and grows smaller as q increases. Elsewhere [13] we have shown that for the search space $\Omega = \{\omega_1, \omega_2, ..., \omega_K, \omega_{K+1}, ..., \omega_N\}$ with target $T = \{\omega_1, \omega_2, ..., \omega_K\}$, where the target has uniform probability $\mathbf{U}(T) = p = K/N = |T|/|\Omega|$, the (higher-order) uniform probability of $\bar{T}_q$ is given by

$$\bar{\mathbf{U}}(\bar{T}_q) = \frac{\Gamma(N)}{\Gamma(N(1-p))\Gamma(Np)} \int_0^{1-q} t^{N(1-p)-1}(1-t)^{Np-1} dt.$$

Note that this last expression describes a cumulative beta distribution with first parameter $r = N(1-p)$ and second parameter $s = Np$ [14].

6. Conservation of Information — The Uniform Case

We are now in a position to prove two conservation of information theorems: the special case for uniform probabilities, which we have proved elsewhere and recap here in this section; and the general case for arbitrary probabilities, which we prove for the first time in the next section [15]. We begin with the special case.

Theorem 6.1 (Conservation of Information — Uniform Case)

Let T be a target in Ω. Assume Ω is finite and T is nonempty. Let $\mathbf{U}$ denote the uniform probability distribution on Ω and let $p = |T|/|\Omega| = \mathbf{U}(T)$ (which we take to be extremely small). Next, let μ be a probability distribution on Ω such that $q = \mu(T)$ (which we take to be considerably larger than p). Suppose that μ characterizes the probabilistic behavior of an alternative search S, so that the endogenous information is $I_\Omega = -\log(p)$ and the exogenous information is $I_S = -\log(q)$. Then the (higher-order) uniform probability of $\overline{T}_q$ in $\mathbf{M}(\Omega)$, denoted by $\overline{\mathbf{U}}(\overline{T}_q)$, is less than or equal to p/q. Equivalently, the (higher-order) endogenous information associated with finding the (higher-order) target $\overline{T}_q$ in $\mathbf{M}(\Omega)$, i.e., $-\log(\overline{\mathbf{U}}(\overline{T}_q))$, is bounded below by the (lower-order) active information $I_+ = -\log(\mathbf{U}(T)) + \log(\mu(T)) = \log(q/p)$.

Proof. Let $\Omega = \{x_1, x_2, ..., x_K, x_{K+1}, ..., x_N\}$ and $T = \{x_1, x_2, ..., x_K\}$ so that $p = K/N$. As we saw in the last section, it then follows that

$$\overline{\mathbf{U}}(\overline{T}_q) = \frac{\Gamma(N)}{\Gamma(N(1-p))\Gamma(Np)} \int_0^{1-q} t^{N(1-p)-1}(1-t)^{Np-1}\,dt,$$

which is a cumulative beta distribution with first parameter $r = N(1-p)$ and second parameter $s = Np$.

Integration by substitution shows that this expression can be rewritten as

$$\frac{\Gamma(N)}{\Gamma(Np)\Gamma(N(1-p))} \int_q^1 t^{Np-1}(1-t)^{N(1-p)-1}\,dt,$$

which describes a cumulative beta distribution with first parameter $r = Np$ and second parameter $s = N(1-p)$. It is well known that the mean for this distribution is $r/(r + s)$ [16]. In consequence,

$$\begin{aligned}
\frac{\Gamma(N)}{\Gamma(Np)\Gamma(N(1-p))}\int_q^1 t^{Np-1}(1-t)^{N(1-p)-1}\,dt &= \frac{\Gamma(N)}{\Gamma(Np)\Gamma(N(1-p))}\int_q^1 \tfrac{q}{q}\cdot t^{Np-1}(1-t)^{N(1-p)-1}\,dt \\
&= \frac{1}{q}\cdot\frac{\Gamma(N)}{\Gamma(Np)\Gamma(N(1-p))}\int_q^1 q\cdot t^{Np-1}(1-t)^{N(1-p)-1}\,dt \\
&\le \frac{1}{q}\cdot\frac{\Gamma(N)}{\Gamma(Np)\Gamma(N(1-p))}\int_0^1 t\cdot t^{Np-1}(1-t)^{N(1-p)-1}\,dt \\
&= \frac{1}{q}\cdot\frac{Np}{Np+N(1-p)} \\
&= \frac{p}{q}.
\end{aligned}$$

It follows that –log $(\overline{\mathbf{U}}(\overline{T}_q)),$ is bounded below by the active information $I_+ = \log(q/p)$. This proves the theorem.

This theorem characterizes the probability costs incurred by a search for a search. Given a vast search space Ω and a tiny target T, the probability of finding the target via the null search is effectively nil ($p = |T|/|\Omega|$). To find the target, we thus need an alternative search S that is able to find it with a probability q that is much larger than p. But where did S come from? Because the complexities and idiosyncrasies associated with the construction of searches in general, the first three sections of this paper focused on simplifying our representation of searches, first by representing them as search matrices and then by representing them as probability measures μ on the original search space Ω such that $\mu(T) = q$.

So the question now is, Where did μ come from? In statistics, whenever confronted with a given outcome, the statistician attempts to situate it among a collection of possible outcomes that are *at least as extreme* as the one in question and then inquires into the improbability of that collection. For instance, given a thousand coin tosses and six-hundred heads, the statistician's first impulse will be to ask how likely it is that a fair coin (the null hypothesis) could have led to six-hundred *or more* heads. In this case, the statistician wants the probability of the tail of a binomial distribution. Leaving aside Bayesian considerations, which can always be incorporated later, if the probability of this tail is extremely small, the statistician will be inclined to question whether the coin responsible for six-hundred heads was fair, thereby implicating an alternative hypothesis. As it is, six-hundred or more heads in a thousand coin tosses represents a departure from expectation by more than six standard deviations. Such a result with a fair coin would be very improbable indeed.

Returning now to our search space Ω and target T, the outcome that confronts us is not a sequence of coin tosses but a search S represented by the probability measure μ. If we set aside that the search is the product of intelligent design, then μ presumably results from some statistical process. Moreover, the collection of outcomes as extreme as μ is then

$$\overline{T}_q = \left\{\theta \in \mathbf{M}(\Omega) \mid \theta(T) \geq q\right\}.$$

In our analogy with statistical practice, $\overline{T}_q$ may then be conceived as the "tail" associated with the "outcome" μ. It would follow that the improbability of this tail is crucial to deciding whether μ is the outcome of a (higher-level) null search.

The parallel here between coin tossing and the search for a search, though far from exact, is suggestive and illuminating. Each coin toss, under the null hypothesis, is a Bernoulli trial, with probability of ½ for heads and ½ for tails. These trials are probabilistically independent, and thus in one-thousand trials should conform to a null hypothesis characterized by a binomial distribution with parameters $N = 1{,}000$ and $p = ½$. The lower-order Bernoulli trials, as it were, "lift" to a higher-order binomial distribution. Similarly, the null search of Ω for T, characterized by the uniform probability **U** on Ω, lifts to a null search of **M**(Ω) for $\overline{T}_q$, characterized by the (higher-order) uniform probability $\overline{\mathbf{U}}$. Conservation of information then shows that the uniform probability of this higher-order target is bounded above by p/q.

Conservation of information is essentially an accounting rule for probabilities associated with search. Here is how it works: finding the original target T within Ω had the very low probability of p under the null search. Fortunately, an alternative search S was available to raise this probability to q. But the probability cost of locating this alternative search, represented by μ, was less than or equal to p/q. Thus, when the cost of locating the alternative search is factored in, nothing is gained over the original null search. The original search, as it were, purchased the target at the "high" probability cost p. The alternative search, correspondingly, purchased the target at the "cheaper" probability cost q, but then itself incurred a probability cost of at least p/q in a higher-order search space since the alternative search itself had to be accounted for. Thus, when the full probability costs incurred by the alternative search are factored in, the total cost is the same as or even worse than the probability cost associated with the original null search.

In fact, the cost tends to be much worse. Conservation of information in the uniform case states that $\overline{\mathbf{U}}(\overline{T}_q) \leq p/q$. Nevertheless, we have shown elsewhere [17] that for $\Omega = \{x_1, x_2, \ldots, x_K, x_{K+1}, \ldots, x_N\}$ and $T = \{x_1, x_2, \ldots, x_K\}$, provided that $p = K/N$ and $N \geq (2q - 1)/(q - p)$,

$$\overline{\mathbf{U}}(\overline{T}_q) < \sqrt{N} \cdot \frac{\sqrt{p}}{q} \cdot \left[1 - (q-p)^2\right]^N.$$

This (strict) inequality shows that the (higher-order) uniform probability of the lifted target $\overline{T}_q$ decreases exponentially with the absolute size N of the search space Ω. As an upper bound on $\overline{\mathbf{U}}(\overline{T}_q)$, p/q is therefore very conservative.

To see how the probability costs associated with null and alternative searches relate, it is instructive to consider the following two quasi-Bayesian ways of reckoning these costs:

$$\begin{aligned}
\mathbf{P}(\text{locating } T \text{ via null search}) &= \mathbf{P}(\text{null search locates } T \text{ \& null search is available}) \\
&= \mathbf{P}(\text{null search locates } T \,|\, \text{null search is avail.}) \\
&\quad \times \mathbf{P}(\text{null search is avail.}) \\
&= \mathbf{U}(T) \times 1 \text{ [because the availability of null search is} \\
&\quad \text{taken for granted]} \\
&= p.
\end{aligned}$$

$$\begin{aligned}
\mathbf{P}(\text{locating } T \text{ via alt. search}) &= \mathbf{P}(\text{alt. search locates } T \text{ \& alt. search is available}) \\
&= \mathbf{P}(\text{alt. search locates } T \,|\, \text{alt. search is avail.}) \\
&\quad \times \mathbf{P}(\text{alt. search is avail.}) \\
&= \mu(T) \times \overline{\mathbf{U}}(\overline{T}_q) \\
&\leq q \times p/q \\
&= p.
\end{aligned}$$

It follows that $\mathbf{U}(T) \geq \mu(T) \times \overline{\mathbf{U}}(\overline{\mathrm{T}}_q)$ and therefore, by taking negative logarithms, that $I_\Omega \leq I_S - \log(\overline{\mathbf{U}}(\overline{T}_q))$, or equivalently that $-\log(\overline{\mathbf{U}}(\overline{T}_q)) \geq I_+ = \log(q/p)$, inasmuch as $I_+ = I_\Omega - I_S$, $I_\Omega = -\log(\mathbf{U}(T)) = -\log(p)$, and $I_S = -\log(\mu(T)) = -\log(q)$. According to conservation of information, the higher-order endogenous information $-\log(\overline{\mathbf{U}}(\overline{T}_q))$, required to find a search qua probability measure that has probability q or better of locating T, is always at least that of the lower-order active information I_+. To say that information is conserved is thus really to say that in the search for a search, information leading to success of the original search is at best conserved when moving to a higher-order search space and may in fact grow considerably higher (in some circumstances, exponentially higher). This rise in the information/probability cost associated with higher-level search should not be surprising given that spaces comprising searches tend to be bigger and structurally richer than the spaces they are searching [18].

7. Conservation of Information — The General Case

We turn now to a generalization of the previous conservation of information theorem. The previous theorem was formulated in terms of a uniform probability baseline. We now lift this restriction. Processes that exhibit stochastic behavior arise from what may be called a *natural probability*. The natural probability characterizes the ordinary stochastic behavior of the process in question. Often the natural probability is the uniform probability. Thus, for a perfect cube with distinguishable sides composed of a rigid homogenous material (i.e., an ordinary die), the probability of any one of its six sides landing on a given toss is 1/6. Yet, for a loaded die, those probabilities will be skewed, with one side consuming the lion's share of probability. For the loaded die, the natural probability is not uniform. Now, if the natural probability for all search spaces Ω were the uniform probability $\mathbf{U}$, we'd be done — the conservation of information theorem proved in the last section would suffice. Yet despite Bernoulli's principle of insufficient reason, which we have argued elsewhere rightly makes the uniform probability the default in many searches [19], the natural probability associated with some searches need not be uniform.

Given structural and external factors influencing search, the natural probability need not be $\mathbf{U}$ but some probability measure μ that assigns probability q to the target T. It's thus convenient to extend the notion of a null search to include not just uniform or blind searches but any searches that accord with such a natural probability. Accordingly, we may then say that μ characterizes the null search of Ω for T. Moreover, the alternative search will then be characterized by a probability measure ν that assigns probability r to T. As the natural probability on Ω, μ is not confined simply to Ω but lifts to $\mathbf{M}(\Omega)$, so that its lifting, namely $\overline{\mu}$, becomes the natural probability on $\mathbf{M}(\Omega)$ (this parallels how the uniform probability $\mathbf{U}$, when it is the natural probability on Ω, lifts to the uniform probability $\overline{\mathbf{U}}$ on $\mathbf{M}(\Omega)$, which then becomes the natural probability for this higher-order search space). When μ is the natural probability associated with a search space, treating it as the null search and ν as the alternative search now leads to a general conservation of information theorem, one that point for point parallels the previous formulation for uniform probabilities.

Theorem 7.1 (Conservation of Information — General Case)

Let T be a target in Ω. Assume Ω is finite and T is nonempty. Let $\mathbf{U}$ denote the uniform probability distribution on Ω and let $p = |T|/|\Omega| = \mathbf{U}(T)$ (which we take to be extremely small). Next, let μ and ν be probability measures on Ω such that $q = \mu(T)$ and $r = \nu(T)$. We assume that $p \le q < \mathrm{r}$ (the rationale for assuming that q is no less than p is discussed at the end of this section). Suppose that μ characterizes the probabilistic behavior of a search S and that ν characterizes the probabilistic behavior of a search S'. We treat μ as the null search and ν as the alternative search, thus making μ the natural probability associated with Ω. Accordingly, $I_S = -\log(q)$ becomes the endogenous information and $I_S' = -\log(r)$ the exogenous information. It then follows that the (higher-order) natural probability of $\overline{T}_r$ in $\mathbf{M}(\Omega)$, i.e., $\overline{\mu}(\overline{T}_r)$, is less than or equal to q/r. Equivalently, the (higher-order) endogenous information associated with finding the (higher-order) target $\overline{T}_r$ in $\mathbf{M}(\Omega)$, i.e., $-\log(\overline{\mu}(\overline{T}_r))$, is bounded below by the (lower-order) active information $I_+ = -\log(\mu(T)) + \log(\nu(T)) = \log(r/q)$.

Remark 1. The probabilities r and q in this theorem correspond respectively to q and p in Theorem 6.1. We changed notation because it seemed best to let p continue to denote the uniform probability of the target. Outside the notation of this theorem, however, we shall typically refer to a null search as setting a baseline probability p and an alternative search as giving an improved probability of success q. Thus, outside the notation of this theorem, we shall generally refer to the active information cost of search in terms of $\log(q/p)$ rather than $\log(r/q)$.

Remark 2. Regressing up the probabilistic hierarchy (i.e., Ω, $\mathbf{M}(\Omega)$, $\mathbf{M}^2(\Omega)$, $\mathbf{M}^3(\Omega)$, etc.) does nothing to mitigate the information cost of successful search. In fact, it intensifies the cost. Searching for a target T in the original search space Ω against a baseline natural probability μ in $\mathbf{M}(\Omega)$, we find that the difficulty of the search is only exacerbated by searching for the higher-order target $\overline{T}_r$ with respect to the higher-order natural probability $\overline{\mu}$ in $\mathbf{M}^2(\Omega)$. The proof below can now be applied again up the probabilistic hierarchy, showing that the search for a still higher-order target aimed at resolving the original search requires the still higher-order natural probability $\overline{\overline{\mu}}$ in $\mathbf{M}^3(\Omega)$, and that this move again only intensifies the difficulty of the search. And so on, up the probabilistic hierarchy.

From the vantage of conservation of information, *searches are no less real than the objects being searched*. Just as the existence and structure of objects require explanation, so too the existence and structure of the searches that locate those objects require explanation. It follows that searches, by residing in a space of searches, are themselves objects to be searched. This implies a hierarchy of searches: the original search, the search for that search, the search for the search for that search, etc. Conservation of information entails that as we regress up this search hierarchy, the search problem never becomes easier and may in fact become more difficult.

Proof. Let $\Omega = \{x_1, x_2, ..., x_K, x_{K+1}, ..., x_N\}$ and $T = \{x_1, x_2, ..., x_K\}$ so that $p = K/N$. Since Ω is finite, the probability measures μ and ν are absolutely continuous with respect to **U**, and so there exist non-negative real-valued functions f and g such that $\mu = f \cdot d\mathbf{U}$ and $\nu = g \cdot d\mathbf{U}$. As we saw in section 5, the lifting of μ is now defined as $\overline{\mu} = \overline{f} \cdot d\overline{\mathbf{U}}$, where $\overline{\mathbf{U}}$ is the uniform probability on $\mathbf{M}(\Omega)$. Thus, for B a measurable subset of $\mathbf{M}(\Omega)$,

$$\overline{\mu}(B) = \int_B \overline{f}(\theta) d\overline{\mathbf{U}}(\theta),$$

where for θ in $\mathbf{M}(\Omega)$,

$$\overline{f}(\theta) = \int_\Omega f(x) d\theta(x).$$

In section 5 we showed that $\overline{\mu}$ is indeed a probability measure over $\mathbf{M}(\Omega)$. Because μ is the natural probability on Ω, $\overline{\mu}$ is the natural probability on $\mathbf{M}(\Omega)$.

Next, for the lifted target $\overline{T}_r = \{\theta \in \mathbf{M}(\Omega) \,|\, \theta(T) \geq r\}$ define the following measure on Ω resulting from vector-valued integration:

$$\mathbf{V}_r = def \int_{\overline{T}_r} \theta \, d\overline{\mathbf{U}}(\theta)$$

This definition holds for any r in the unit interval. Note that when $r = 0$ (an equality that does not in fact holds since we assume that r is no smaller than p), then $\overline{T}_r$ is all of $\mathbf{M}(\Omega)$; on the other hand, when $r > 0$, then $\overline{T}_r$ is a proper subset of $\mathbf{M}(\Omega)$. It follows that $\mathbf{V}_r$ is a probability measure on Ω only if $r = 0$ and is a sub-probability measure otherwise (i.e., it assigns measure less than 1 to Ω if $r > 0$).

What value less than 1 does $\mathbf{V}_r$ assign to all of Ω? The answer can be seen from the following equation:

$$\mathbf{V}_r(\Omega) = \left[\int_{\overline{T}_r} \theta \, d\overline{\mathbf{U}}(\theta) \right](\Omega) = \int_{\overline{T}_r} \theta(\Omega) d\overline{\mathbf{U}}(\theta) = \int_{\overline{T}_r} 1 \cdot d\overline{\mathbf{U}}(\theta) = \overline{\mathbf{U}}(\overline{T}_r),$$

which, by conservation of information in the uniform case (Theorem 6.1), we know to be bounded above by p/r.

By consistency of uniformity (Proposition 5.1), we know that $\mathbf{V}_0$ is just the uniform probability **U**. For $r > 0$, it is easily seen that $\mathbf{V}_r$ is exchangeable on T and on its complement T^c. In other words, $\mathbf{V}_r$ is invariant under permutations of T and of T^c. Since the only such exchangeable measures are those proportional to uniform probabilities, this means that there exist positive constants a_r and b_r such that

$$\mathbf{V}_r = a_r \mathbf{U}(\cdot \,|\, T) + b_r \mathbf{U}(\cdot \,|\, T^c)$$

Here $\mathbf{U}(\cdot\,|\,T)$ is the uniform probability on T and $\mathbf{U}(\cdot\,|\,T^c)$ is the uniform probability on T^c . Note that because $\mathbf{V}_0$ is the uniform probability on Ω, $a_0 = p$ and $b_0 = 1 - p$.

Now, because $\mu = f \cdot d\mathbf{U}$ with $\mu(T) = q$ and $\mu(T^c) = 1 - q$, integrating f with respect to $\mathbf{V}_r$, which is proportional to a uniform probability measure on T and on T^c, is the same as integrating the function $\frac{q}{p}1_T + \frac{1-q}{1-p}1_{T^c}$ with respect to $\mathbf{V}_r$, where 1_T and 1_{T^c} are the indicator functions for T and T^c respectively. This is because integrating a real-valued function with respect to a uniform probability measure is the same as integrating its average value with respect to a uniform probability measure (the average of f on T being $\frac{q}{p}$ and the average of f on T^c being $\frac{1-q}{1-p}$).

It follows that

$$
\begin{aligned}
\overline{\mu}(\overline{T}_r) &= \int_{\overline{T}_r} \overline{f}(\theta)\,d\overline{\mathbf{U}}(\theta) \text{ [unpacking definitions]}\\
&= \int_{\overline{T}_r}\left[\int_{\Omega} f(x)\,d\theta(x)\right] d\overline{\mathbf{U}}(\theta) \text{ [unpacking definitions]}\\
&= \int_{\Omega} f(x)\,d\left[\int_{\overline{T}_r} \theta\, d\overline{\mathbf{U}}(\theta)\right](x) \text{ [by vector-valued integration]}\\
&= \int_{\Omega} f(x)\,d\mathbf{V}_r(x) \text{ [by definition]}\\
&= \int_{\Omega}\left[\frac{q}{p}1_T + \frac{1-q}{1-p}1_{T^c}\right] d\mathbf{V}_r(x) \text{ [as noted above]}\\
&\le \frac{q}{p}\cdot \int_{\Omega} d\mathbf{V}_r(x) \text{ [because } q \ge p\text{]}\\
&= \frac{q}{p}\cdot \overline{\mathbf{U}}(\overline{T}_r) \text{ [because } \mathbf{V}_r(\Omega) = \overline{\mathbf{U}}(\overline{T}_r)\text{]}\\
&\le \frac{q}{p}\cdot\frac{p}{r} \text{ [by cons. of info., unif. case]}\\
&= \frac{q}{r}.
\end{aligned}
$$

This proves the theorem.

The theorem just proved assumes that the null search assigns a probability q to T that is at least as large as the uniform probability p. But what if the "natural" probability on the search space entails a null search that is worse at locating the target than uniform random sampling, so that q is strictly less than p? We put "natural" in scare quotes here because, we submit, natural probabilities need never do worse than uniformity. To see this, consider a deck of cards and imagine we are searching for the ace of hearts. Presented with a deck, face down, we are told to draw the first card on top. What is the probability that it will be the ace of hearts? If we knew that the deck had just been thoroughly shuffled, then we would be justified in assigning the uniform probability of 1/52 to the top card being the ace of hearts. But if we knew absolutely nothing about how the deck came to assume its order, the uniformity assumption becomes questionable, requiring for its justification Bernoulli's disputed principle of indifference [20].

Now imagine we learn that that the deck gets thoroughly shuffled, but that whenever the ace of hearts appears on top, a coin is flipped so that heads leaves it there but tails moves it to the bottom of the deck. Given this way of randomly arranging the deck, the probability of the top card being the ace of hearts is not $1/52$ but $1/52 \times 1/2 = 1/104$. In this case, the probability of drawing the ace of hearts is strictly less than its uniform probability. Considerations such as this suggest that the uniformity assumption, though appropriate in many circumstances, doesn't hold universally for search. But this example additionally suggests that we don't need to stay with a sub-uniform probability when conducting a search. Precisely because we are searching for the ace of hearts, we don't have to sample the first card at the top of the deck. Search implies we have freedom to move about the search space and thus, in the present example, to sample any card in the deck. Hence, by suitably randomizing the selection, we can ensure that the card picked had the uniform probability 1/52 of being the ace of hearts. In general, then, when conducting a search, we are in our rights to assume that we can always do at least as well as uniformity. Doing worse, at least for search, is *unnatural*. Thus, in the context of search, any natural probability that replaces the uniform probability may be taken to assign a higher probability of success in locating the target than the uniform probability.

8. Regulating the Information Industry

Conservation of information supplies the information industry with a balance sheet, ensuring that the information output on one side of the ledger does not exceed the information input on the other. Specifically, conservation of information guarantees that any search that proportionately raises the probability of finding a target by q/p requires, in its construction, an amount of information not less than the active information $I_+ = \log(q/p)$. Simply put, raise the probability of successful search by a factor of q/p, incur an information cost of $\log(q/p)$.

At the time of this writing, the United States government is much exercised about regulating the financial industry. Essential to any such regulation is accurate accounting of money — where it originates, how it flows, and where it ends up. Conservation of information shows that active information, like money, obeys strict accounting principles. Just as banks need money to power their financial instruments, so searches need active information to power their success in locating targets. Moreover, just as banks must balance their books, so searches, in successfully locating targets, must balance their books — they cannot output more information than was inputted.

Regulation of the financial industry is necessary because it is too easy to mask liabilities as assets and thereby attempt to escape one's obligations. Likewise, regulation of the information industry is necessary because it is too easy to focus on the success of a search and forget the information that paid for that success. The temptation is to inflate the creative power of search programs by conveniently forgetting the creative power of the programmers who impart the information that makes those programs successful. In short, the temptation is to ignore conservation of information in the hopes of a free lunch.

Conservation of information keeps the search practitioner honest.

Acknowledgment

The authors thank Peter Olofsson and Dietmar Eben for helpful feedback on previous work of the Evolutionary Informatics Lab, feedback that has found its way into this paper.

References and Notes

1. See, for instance, William A. Dembski and Robert J. Marks II, "Conservation of Information in Search: Measuring the Cost of Success," *IEEE Transactions on Systems, Man and Cybernetics A, Systems & Humans* 5(5) (September 2009): 1051–1061 and also William A. Dembski and Robert J. Marks II, "Life's Conservation Law: Why Darwinian Evolution Cannot Create Biological Information," in Bruce Gordon and William Dembski, eds., *The Nature of Nature* (Wilmington, Del.: ISI Books, 2010).
2. Compare Joseph Culberson's remarks about evolutionary and adaptive algorithms: "Evolutionary algorithms (EAs) are often touted as 'no prior knowledge' algorithms. This means that we expect EAs to perform without special information from the environment. Similar claims are often made for other adaptive algorithms." From

Joseph C. Culberson, “On the Futility of Blind Search: An Algorithmic View of ‘No Free Lunch’,” *Evolutionary Computation* 6(2) (1998): 109–127.

3. For the hypergeometric distribution and its limiting behavior, see G. R. Grimmett and D. R. Stirzaker, *Probability and Random Processes* (Oxford: Oxford University Press, 1982), 46–47.
4. “In contrast to single-objective optimization, a solution to a multi-objective problem is more of a concept than a definition. Typically, there is no single global solution, and it is often necessary to determine a set of points that all fit a predetermined definition for an optimum. The predominant concept in defining an optimal point is that of Pareto optimality.” Quoted from R. T. Marler and J. S. Arora, “Survey of Multi-Objective Optimization Methods for Engineering,” *Structural and Multidisciplinary Optimization* 26 (2004): 371.
5. In 2005, the SAT introduced a writing component and thus another score from 200 to 800. For simplicity, we ignore this change in the test.
6. Taken from http://www.endgame.nl/stpeter.htm (last accessed April 7, 2010).
7. Because our framework for search includes these last two examples, all the results in this paper apply to coevolutionary scenarios of the sort described in David Wolpert and William Macready, “Coevolutionary Free Lunches,” *IEEE Transactions on Evolutionary Computation* 9(6) (December 2005): 721–735.
8. In other work of the Evolutionary Informatics Lab (www.evoinfo.org), we have referred to the null search as an “unassisted search” or a “blind search” and the alternative search as an “assisted search” — see for example Dembski and Marks, “Conservation of Information in Search.” The language of “null” and “alternative” searches is in analogy to statistics.
9. See William A. Dembski and Robert J. Marks II, “The Search for a Search: Measuring the Information Cost of Higher Level Search,” *Journal of Advanced Computational Intelligence and Intelligent Informatics* 14(5) (2010): 475–486.
10. For vector-valued integration, see Nicolae Dinculeanu, *Vector Integration and Stochastic Integration in Banach Spaces* (New York: Wiley, 2000).
11. See, for instance, Donald L. Cohn, *Measure Theory* (Boston: Birkhäuser, 1997), 220, theorem 7.3.5.
12. See Dembski and Marks, “The Search for a Search.”
13. This result is somewhat buried in Dembski and Marks, “The Search for a Search.” It is proven explicitly in William A. Dembski, “Searching Large Spaces,” typescript, available at www.designinference.com (last accessed April 27, 2010). Note that the result, as stated in this last paper, is for $q > p$. In fact, nothing in the proof requires this restriction. Indeed, q is free to vary across the entire unit interval.
14. Robert J. Marks II, *Handbook of Fourier Analysis and Its Applications* (Oxford: Oxford University Press, 2009), 165.

15. We proved the special case in Dembski and Marks, “Life’s Conservation Law.”
16. See Marks, Handbook of Fourier Analysis, 165.
17. Dembski, “Searching Large Spaces.”
18. Search spaces whose elements are themselves searches may be thought of as consisting of probability measures or fitness landscapes or other functions on the original search space. Spaces of functions on an original space are always richer than the original space and, in case the original space is finite, grow exponentially or even super-exponentially in cardinality. Spaces of probability measures, for instance, are always infinite. Cf. William A. Dembski, *No Free Lunch: Why Specified Complexity Cannot be Purchased without Intelligence* (Lanham, Md.: Rowman and Littlefield, 2002), ch. 3.
19. William A. Dembski and Robert J. Marks II, “Bernoulli’s Principle of Insufficient Reason and Conservation of Information in Computer Search,” *Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, Texas (October 2009): 2647–2652.
20. Though compare the reference in the previous note, which argues that uniformity, even if not holding precisely, is, in the absence of definite knowledge about the underlying probabilities, the probability distribution most reasonably assumed.