Every Good Key Must Be A Model Of The Lock It Opens

(The Conant & Ashby Theorem Revisited)

By

Daniel L. Scholten

Table of Contents

Introduction

The title of the present paper is really a metaphor for what might have been a more literal title:

Every Good Solution Must be a Model of the Problem it Solves.

It is also a metaphor for the so-called "Conant and Ashby Theorem" referred to in the sub-title. That theorem – published in 1970 by Roger C. Conant and W. Ross Ashby – establishes that,

Every Good Regulator of a System Must be a Model of that System.[1]

What all of this means, more or less, is that the *pursuit* of a goal by some dynamic agent (Regulator) in the face of a source of obstacles (System) places at least one particular and unavoidable *demand* on that agent, which is that the agent's behaviors *must* be executed in such a reliable and predictable way that they can serve as a *representation* (Model) of that source of obstacles. As a refinement to this paraphrase, we could specify that the particular *style* of pursuit be both *optimal* and *maximally simple,* where "optimal" means that the actualized goal is as close as possible to its ideal form given the circumstances, and "maximally simple" means that the agent is achieving this "best-attainable" goal without unnecessary expense or effort.

A careful reading of Conant and Ashby's paper[2] reveals a distinction that can be made between such an idealized "good-regulator model", which is really a *dynamic* entity, and its "technical specification", which we might call its *control-model*. Another distinction to be recognized is that whereas the good-regulator model is dynamic, the control-model may be either static or dynamic. As an example of a *static* control-model, consider an inexperienced cook attempting to make a roast duck with the help of a recipe. In this case, the system to be regulated consists of the various ingredients and kitchen tools to be used to create the meal, the dynamic good-regulator model is the human being doing the cooking, and the recipe is what we are calling the *static* "control-model". The recipe is a control-model because the human being uses it, like a technical specification, to guide (control) his behavior and thus to "turn himself into" (i.e. to act *as-if* he were) a good-regulator model. As an example of a *dynamic* control-model, consider the case in which a child learns to use an idiomatic expression such as "two wrongs don't make a right" by overhearing an adult use that expression in a conversation. In this case the system to be regulated is a

[1]Roger C. Conant and W. Ross Ashby, "Every Good Regulator of a System Must be a Model of that System," *International Journal of Systems Science*, 1970, vol 1., No. 2, 89-97.
[2]"A Primer For The Conant And Ashby Theorem", Available on the internet at www.goodregulatorproject.org.

particular portion of some conversation in which the child is participating, the dynamic good-regulator model is the child, and the *dynamic* control-model is the adult role-model. The idea here is that the adult's behavior serves as a type of dynamic technical specification that the child then uses to control his or her own behavior in the context of the given conversation.

It is important to make these distinctions between a dynamic good-regulator model and its static or dynamic technical specification because otherwise the theorem appears to prove that the *technical specification* (control-model) is necessary, which is really a misunderstanding of the theorem. The theorem only proves that the *good-regulator* model is necessary, although it does appear to be an empirical fact that such technical specifications are also necessary, or at least extremely useful.

One objective of the present paper is to offer the key-lock and solution-problem formulations as useful metaphorical equivalents to this important result by Conant & Ashby (henceforth C&A). But the present paper's *primary* objective is to explore a line of inquiry that this theorem establishes, and which, as far as I can tell, has yet to be explored.

In what follows, I will show how C&A's "good-regulator theorem", despite being just one of potentially various necessary conditions to successful regulation, has yet a great deal more to say about the nature of these idealized good-regulator models[3]. In particular, I will show that the theorem points the way to a search-algorithm (of the type used in the field of Operations Research) that can be implemented by a computer program[4] and used to find excellent approximations to these idealized entities given certain high-level measurements taken on the system in question and the resources available to regulate that system[5]. Furthermore, the analysis leading to this search algorithm will also lead us to an alternate proof of the C&A theorem which is both more general and allows us to assert that "Even A Decent Regulator Of A System Must Be A Model Of That System". This more general theorem has two primary benefits. First of all, however valuable or interesting a true good-regulator model might be, any realistic scenario has such an astronomically large search-space that finding such unicorns is a truly rare occurrence. Much more likely is the discovery of a decent approximation to the beast, and this more general version of Conant and Ashby's theorem allows us to include these much more common "decent-regulator models" in our discussions of actual good-regulator models. The second benefit is that the proof of this more general version of the good-regulator theorem is substantially easier for a lay-person to understand, mainly because it doesn't rely on the Shannon Entropy function. Such a simpler proof should go a long way toward making this important theorem accessible to a much wider audience.

In addition to these two primary results, I will also examine a number of related results. One of these is a corollary which we might call a "Law of Requisite Back-Up Plans" the gist of which tells us that a good-regulator must have its priorities straight.

[3] http://en.wikipedia.org/wiki/Good_Regulator
[4] This program is currently in development.
[5] If the System to be regulated and the available regulatory resources are not too complicated, then the search algorithm will actually find the associated ideal good-regulator models. The problem is that most real-world applications will probably involve Systems that do not fulfil this simplicity criterion.

This corollary establishes that a good-regulator must know, given a set of possible outcomes, which of these is its favorite, which is its second favorite, and so on, all the way through to its least favorite outcome. As it pertains to us humans, it tells us, contrary to what we might like to think, that our most lofty aspirations are not more important than what we will do if we fail to attain them. And I will also discuss a principle that I have come to think of as *Ashby's First Law*, or perhaps, the *Any-Port-In-A-Storm Theorem*. This result appears to explain a great deal of human behavior, especially our tendency to sanctify and ferociously defend even the most delusional ideas. The gist of Ashby's First Law is that *any model is better than no model at all*, which basically means that even if we can't figure out the truth of a situation, we are *almost always* better off just making up any old thing rather than muddling around in the muck of our own ignorance. If you have ever wondered how a person could believe so seriously in, say, faeries or werewolves, *to the point where they are willing to hurt themselves or someone else*, Ashby's First Law may provide a meaningful clue. Finally, I will also discuss two useful metaphors for the C&A theorem, namely the key-lock and solution-problem metaphors referred to above and in this paper's title. This latter metaphor, in particular, casts the C&A Theorem as a fundamental theorem of problem solving and as such shows us how to make meaningful progress toward the solution of *any problem whatsoever*.

After discussing these topics, I will turn to a question that is grounded in the observation that the C&A Theorem is itself a control-model (technical specification) for a good-regulator model. The system to be regulated in this case is nothing other than the system of good- and decent-regulator models that we humans are using more and more to regulate our interactions with each other and the world. Recognition of this fact raises the question as to why the C&A theorem is not more famous than it is. Given the preponderance of control-models that are used by humans (the evidence for this preponderance will be surveyed in the latter part of the paper), and especially given the obvious need to regulate that system, one might guess that the C&A theorem would be at least as famous as, say, the Pythagorean Theorem ($a^2 + b^2 = c^2$), the Einstein mass-energy equivalence ($E = mc^2$, which can be seen on T-shirts and bumper stickers), or the DNA double helix (which actually shows up in TV crime dramas and movies about super heroes). And yet, it would appear that relatively few lay-persons have ever even heard of C&A's important prerequisite to successful regulation. This state-of-affairs strikes the present author as a problem in need of a solution, a lock that needs a key. It is my hope that the present paper can contribute to finding that key.

Note: throughout the following discussion I will assume that the reader has studied Conant &Ashby's original paper, possesses the level of technical competence required to understand their proof, and is familiar with the components of the basic model that they used to prove their theorem, e.g. the payoff matrix, the outcome mapping $\psi : R \times S \to Z$, and the definition of regulation in terms of the Shannon entropy function, etc. If the current reader does not already fit this profile but would like to, he or she is invited to study the references in the footnote at the end of this sentence.[6]

[6] The reference to the original paper can be found in a previous footnote. The paper is also readily available on the internet at http://pespmc1.vub.ac.be/Books/Conant_Ashby.pdf. Everything you need to

Finding Good Regulators: A Search Algorithm

In order to motivate the present discussion, we will use a concrete example which is organized around the following "payoff" matrix:

$$\begin{array}{c|cccccc} \psi & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ \hline r_1 & a & h & d & g & b & h \\ r_2 & c & f & i & e & c & d \\ r_3 & f & d & e & b & a & i \\ r_4 & a & d & c & e & a & f \end{array}$$

The above matrix represents every possible outcome that can occur when a behavior from some system with behavioral repertoire $S = \{s_1, s_2, s_3, s_4, s_5, s_6\}$ combines with a behavior from some other system with behavioral repertoire $R = \{r_1, r_2, r_3, r_4\}$ to produce outcomes from a set $Z = \{a, b, c, d, e, f, g, h, i\}$, as determined by the mapping $\psi : R \times S \to Z$.

In addition, we assume that S is executing its behaviors according to some probability distribution $p(S) = \{p(s_1), p(s_2), p(s_3), p(s_4), p(s_5), p(s_6)\}$ that R is responding to these behaviors according to some conditional distribution $p(R \mid S) = \{p(r \mid s) : r \in R, s \in S\}$ and that together these two distributions determine, via the rule $p(r_i, s_j) = p(r_i \mid s_j)\, p(s_j)$, the following distribution for Z:

$$p(Z) = \left\{ \begin{array}{l} p(a) = p(r_1,s_1) + p(r_3,s_5) + p(r_4,s_1) + p(r_4,s_5), \\ p(b) = p(r_1,s_5) + p(r_3,s_4), \\ p(c) = p(r_2,s_1) + p(r_2,s_5) + p(r_4,s_3), \\ p(d) = p(r_1,s_3) + p(r_2,s_6) + p(r_3,s_2) + p(r_4,s_2), \\ p(e) = p(r_2,s_4) + p(r_3,s_3) + p(r_4,s_4), \\ p(f) = p(r_2,s_2) + p(r_3,s_1) + p(r_4,s_6), \\ p(g) = p(r_1,s_4), \\ p(h) = p(r_1,s_2) + p(r_1,s_6), \\ p(i) = p(r_2,s_3) + p(r_3,s_6) \end{array} \right\}$$

learn about the Shannon entropy function in order to understand C & A's proof can be acquired by reading the first 26 pages (especially problem 1.6) of *Information Theory*, by Robert B. Ash, 1990 (1965), Dover Publications, New York. For a self-contained exposition, the reader can also consult the present author's *A Primer For The Conant & Ashby Theorem* available at www.goodregulatorproject.org.

Also, we can use the above distribution to calculate the Shannon Entropy associated with the outcomes of Z, which is given by $H(Z) \equiv -\sum_{z \in Z} p(z) \log p(z)$.

Furthermore, we will assume, given any example such as the one we are considering, that the distribution $p(S)$ has been given to us as a set of fixed probabilities over which we have no control, and that, on the contrary, we have complete freedom to set the conditional probabilities of $p(R \mid S)$ to any values we choose. And finally, we will assume that the goal of regulator design is to set these conditional probabilities of $p(R \mid S)$ so that $H(Z)$ is made as small as possible, given the constraints imposed by the mapping $\psi : R \times S \to Z$, in which case, the regulator is said to be *optimal*.

Now, given only these assumptions, we can observe that in the most general case, the task of finding a regulator (i.e. a distribution $p(R \mid S)$) that will minimize $H(Z)$ is daunting indeed. The problem is that the solution space, even for a relatively simple example such as the one we are considering, is nothing short of huge. If all we have is the option of grinding through every possible distribution $p(R \mid S) = \{ p(r \mid s) : r \in R, s \in S \}$ then the search space is infinite and the task is impossible. What we need is some way to reduce the search space.

In their 1970 paper, C & A showed the world how to cut that search space down to a much more manageable size. In particular, they showed us a lemma (henceforth, the C&A lemma) which tells us that whenever R is behaving so that $H(Z)$ is as small as possible, then it must be the case that R is picking out *exactly one outcome per column of the table*. In other words, their lemma establishes that optimal regulation can only occur via some sort of mapping from the set S to the set Z. Note that C&A chose to define *model* in terms of just such a mapping and we will continue with that convention here, although this particular "model" is *not* the one referred to in the title of their paper and, in fact, the whole purpose of regulation is make this "model" an especially poor one. The idea here is that the regulator acts as a buffer between the system and the outcomes produced, so that the outcomes become a lousy representation of the system. Let's call the mapping that creates this poor "underdog" model of the system $u : S \to Z$, where "*u*" stands for "underdog".

C&A then went on to show that by adding the economically reasonable assumption that R should achieve *as simply as possible* its one outcome per column – meaning that in response to a given column $s_j \in S$ the *same* $r_j \in R$ is *always* executed in order to produce that column's unique outcome – then yet a *different* mapping would be established from S to R, which we will refer to here as $g : S \to R$ ("*g*" for "good regulator"). It is this latter mapping that creates the model referred to by the C&A theorem – the model that is the "good regulator" of the system being regulated. Contrary to the relatively poor "underdog" model referred to in the previous paragraph, this good-regulator model will tend to be a fairly detailed representation of the system.

Now, without plugging in some actual numbers for $p(S)$, we really can't say much else about the example under consideration, but let's just imagine that there is some example we could use for $p(S)$ such that the following mapping $u : S \to Z$ will minimize $H(Z)$:

$$u \downarrow \begin{array}{cccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ \hline a & d & d & b & a & d \end{array}$$

The implication of this mapping is that the probability distribution $p(Z)$ is a much simpler function of just the column probabilities of $p(S)$. Let's illustrate this by working through an example. Referring to the table for $p(Z)$ above, we have that:

$$p(a) = p(r_1,s_1) + p(r_3,s_5) + p(r_4,s_1) + p(r_4,s_5)$$

or

$$p(a) = p(r_1 \mid s_1)\, p(s_1) + p(r_3 \mid s_5)\, p(s_5) + p(r_4 \mid s_1)\, p(s_1) + p(r_4 \mid s_5)\, p(s_5)$$

or

$$p(a) = \left[\, p(r_1 \mid s_1) + p(r_4 \mid s_1) \right] p(s_1) + \left[\, p(r_3 \mid s_5) + p(r_4 \mid s_5) \right] p(s_5)$$

So far, $p(a)$ appears to be a function of both $p(S)$ and $p(R \mid S)$. But the mapping $u : S \to Z$ implies that we, as regulator designers, have respected the following constraint on $p(R \mid S)$:

$p(r_1 \mid s_1) + p(r_4 \mid s_1) = 1$ and $p(r_3 \mid s_5) + p(r_4 \mid s_5) = 1$, which, in turn, means that,

$p(a) = 1 \cdot p(s_1) + 1 \cdot p(s_5) = p(s_1) + p(s_5)$.

The reader can verify that performing this sort of analysis for each element in Z for our chosen example yields the following:

$$p(Z) = \left\{ \begin{array}{l} p(a) = p(s_1) + p(s_5), \\ p(b) = p(s_4), \\ p(c) = 0, \\ p(d) = p(s_2) + p(s_3) + p(s_6) \\ p(e) = 0, \\ p(f) = 0, \\ p(g) = 0, \\ p(h) = 0, \\ p(i) = 0 \end{array} \right\}$$

Now, we can make two observations about this state-of-affairs. I will make them here with respect to our current example, but similar observations can be made in general about *every* such example. These are as follows:

1. Because the probabilities in the distribution $p(Z)$ are real numbers, they have a natural ordering. That is, there is a way to rename the elements in the set Z using indexes such that $Z = \{z_1, z_2, ..., z_9\}$ and $p(z_1) \geq p(z_2) \geq ... \geq p(z_9)$. To put it most generally, we can say that there is a mapping from the set Z to the set of natural numbers (excluding zero), let's call it $f : Z \rightarrow \mathbb{N}_1$, such that $Z = \{z_1, z_2, ..., z_{|Z|}\}$ and $p(z_1) \geq p(z_2) \geq ... \geq p(z_{|Z|})$.
2. The probability $p(b)$ does not use up all of the "column probability" that it might use. That is, since the outcome *b* is available in both columns s_4 as well as s_5, it might have been the case that $p(b) = p(s_4) + p(s_5)$, but this did not happen. For some reason, our optimal regulator *preferred* to give the column probability $p(s_5)$ to outcome *a* instead of to outcome *b.*

This natural ordering of the outcome probabilities coupled with the observation that an optimal regulator behaves as if it preferred some outcomes over others, suggests that at we can jump to the following conclusion:

> When all is said and done and we have managed to find an optimal regulator (i.e., a $p(R\,|\,S)$ that minimizes $H(Z)$), then the behavior of this optimal regulator will accord with at least one "preference ranking" on the outcomes in Z. Although there may be many such preference rankings, one of these is the mapping $f : Z \to \mathbb{N}_1$ that is determined by the natural ordering of the probabilities in $P(Z)$. Finally, this preference ranking is logically equivalent to the optimal regulator, which means that they both produce exactly the same outcomes and thus exactly the same minimized value of $H(Z)$.

Now, I say that we can jump to this conclusion, but it really needs to be proven. The proof relies on the following lemma which uses the same "useful property" of the Shannon Entropy function that C&A used to prove their lemma:

> Given some $\psi : R \times S \to Z$ and a $p(R\,|\,S)$ that minimizes $H(Z)$, if z_k is the outcome selected for column s_j, and if z_h is any other outcome in the column s_j, then $p(z_k) > p(z_h)$.

> Proof: Suppose to the contrary that $p(z_k) \le p(z_h)$. Then we can change $p(R\,|\,S)$ so that z_h is selected for column s_j instead. Doing so will increase the difference $p(z_h) - p(z_k)$ and thus reduce $H(Z)$ thus contradicting our assumption that $H(Z)$ is a minimum. The contradiction proves the result.

In order to prove the main proposition under consideration, we have to show that the mapping $f : Z \to \mathbb{N}_1$ produces exactly the same outcomes as the distribution $p(R\,|\,S)$ that minimized the entropy function. This has to be proven because even though a minimized $H(Z)$ implies $f : Z \to \mathbb{N}_1$, it is conceivable that $f : Z \to \mathbb{N}_1$ doesn't "return the favor". That is, perhaps we are dealing with a simple one-way material conditional and not a material bi-conditional.

To accomplish this proof, consider an arbitrary $s_j \in S$ and suppose that z_k is the outcome produced by $p(R \mid S)$. If z_h is any other available outcome in s_j, we know (by the lemma proven above) that $p(z_k) > p(z_h)$ which means that our preference ranking favors z_k over z_h. Since $s_j \in S$ was assumed to be arbitrary, such a result will hold for every column of the table which means that our preference ranking will produce exactly the same outcomes as $p(R \mid S)$. This establishes the result.

Now, how does all of this translate into a search algorithm? To see how, we observe that the existence of this preference ranking implies that $p(z_1)$ (the largest of all the probabilities in $p(Z)$) is the sum of *all* of the column probabilities associated with the outcome z_1. That is, whenever z_1 is available in some column s_j, then $p(s_j)$ is included as an addend in the sum for $p(z_1)$. Yet another way to put it is that $p(z_1)$ "gobbles up" all of the column probabilities that are available to produce the outcome z_1. Likewise, $p(z_2)$ "gobbles up" whatever is leftover for z_2, and so on until nothing is left for the remaining outcomes and everybody else gets nothing. This means that we can begin our search for an optimal regulator by creating a table of these "total probabilities" for each possible outcome in Z and then trying to identify which of these total probabilities is really our $p(z_1)$. That's the first step in the search. For the next step we adjust our table, removing from consideration the column probabilities that were used to calculate $p(z_1)$, and then try to identify which of the remaining total probabilities is $p(z_2)$ and so forth until there aren't any column probabilities left to use.

Now, in each step I say we proceed by "trying to identify" which total probability is the next in the sequence. This raises the question of the best way to accomplish this. There are two intuitively appealing approaches to this problem, and although they are both useful heuristics, it turns out that neither works in all cases. These are as follows:

1. At each step in the process, choose the outcome with the greatest total probability available.
2. At each step in the process, choose the outcome that occupies the greatest number of columns.

I have written a computer program that compares both of these approaches to a brute-force exhaustive search of the solution space for relatively small tables (e.g. $|R| = 5$, $|S| = 7$ and $|Z| = 12$) and I have also tried various ways to combine them both and the upshot is that it appears that the only way to always find the real minimum for

$H(Z)$ is just to check every path[7]. This can still be quite a large search space, but it is substantially less than the brute-force approach. Furthermore, if we organize that search by always beginning with the outcome with the greatest total probability, and then proceeding at each level with the next largest, and then the next largest, and so forth, we greatly increase the chances of encountering the solution sooner rather than later. This approach has the added benefit of being easily convertible to a very useful search heuristic and I have found experimentally that limiting the search to the first two of these (largest and next largest probability) captures very close to every case, at least for the relatively small tables that I have tested[8].

The Search Algorithm, Formally Stated

The purpose of the preceding discussion is to motivate and to illustrate the gist and justification of a formal search algorithm for the identification of the underdog mapping(s) $u : S \to Z$ that minimize(s) $H(Z)$. This algorithm, which can be implemented in any high-level programming language (I have used Java), consists of three steps: an initialization step, an iterato-recursive step, and a completion step. I will now present the complete and formal details of this algorithm. Unfortunately, the completeness and formality of these details might appear at first glance as something of a pedagogical quagmire. Therefore, the reader is encouraged to skim over these details for the time being and then read back over them more carefully as we work through an illustrative example. I assure you that the following algorithm is no where near as complicated as it may look:

1. **Initialization**: this step takes as input the information contained in the payoff matrix $\psi : R \times S \to Z$ as well as the distribution $p(S)$ and uses it to construct a data structure that we will call $Total\text{-}Probability\text{-}Set(\ )$ (the meaning of the empty parentheses will be explained in step 2.b.ii), which will be used as the initial input to the iterato-recursive process of step (2). The construction of $Total\text{-}Probability\text{-}Set(\ )$ is accomplished as follows:

    a. For each outcome element $z \in Z$, calculate $p_{tot}(z)$, i.e. the "total probability available to the outcome z", as follows: for each $s \in S$, examine the contents of the corresponding column in the payoff matrix ψ to determine if the outcome z is available in that column.

[7] Strictly speaking there are ways to eliminate a few additional paths, but I'm trying to summarize the gist of the process at this point; these refinements will be covered shortly. In any case, it may be that these refinements won't enhance the performance of a computer program that implements them to a degree that would justify the additional complexity that they add to such a program.

[8] I have to restrict my tests to relatively small tables because my "control population" is generated with a brute-force, every-possible-combination algorithm and if the tables get too big, the time required to run through all of these possibilities becomes too long.

If it is available, then include $p(s)$ as an addend in the sum for $p_{tot}(z)$.

b. Place each of these *total-probability* elements in an ordered set, sorted top to bottom from largest to smallest. These can be indexed as $p_{tot}(z)_{j,0}$, for $j \in \{1,2,3,...\}$, where $p_{tot}(z)_{1,0}$ is the largest, $p_{tot}(z)_{2,0}$ the second largest, etc. This descending-ordered set of total-probability elements is what we are calling $Total\text{ - }Probability\text{ - }Set(\ )$.

2. **Iterato-Recursive process**: the goal of this process is to produce a list of values for $H(Z)$ which are candidates for the smallest possible such value, which will be determined in (3), the completion step, below. Toward this end, various distributions for $p(Z)$ are calculated, which are in turn used to calculate the values for $H(Z)$. I'm calling this an *iterato-recursive* process because it is comprised of an iterative process that occurs within a larger recursive process. By definition, a *recursive process* is any process that includes an execution of *itself*, which creates a sequence of *recursion-levels*. We will refer to these as the 1st recursion-level, 2nd recursion-level, etc., and in the general case, the *k*th recursion-level. Furthermore, within each of these recursion-levels occurs an *iterative process* that performs an identical set of operations over a sequence of distinct elements (to be explained momentarily). We will index these as element$_{1,k}$, element$_{2,k}$, etc., and in the general case, element$_{j,k}$, where the index *k* indicates the recursion-level inside of which these iterations are performed.

Now, the input to the *k*th recursion-level, which we will call $Total\text{ - }Probability\text{ - }Set(j_1, j_2, ..., j_{k-1})$ (the sequence $j_1, j_2, ..., j_{k-1}$ will be explained in step 2.b.ii) is either the output from step (1) (i.e. $Total\text{ - }Probability\text{ - }Set(\ )$) or else the output from the *(k-1)*th recursion level. On the other hand, the output from the *k*th recursion-level is either a value for $H(Z)$, or else a value for $p(z_k)$ along with $Total\text{ - }Probability\text{ - }Set(j_1, j_2, ..., j_k)$, where the method used to produce this latter *Total-Probability-Set* is explained below in sub-step 2.b.ii. To begin this iterato-recursive process, the input $Total\text{ - }Probability\text{ - }Set(j_1, j_2, ..., j_{k-1})$ is first examined to see if it contains any non-zero total-probability elements (where the fact that it *might* will be explained in sub-step 2.b.ii). The answer to this question determines which of the following is executed:

a. If the input $Total\text{ - }Probability\text{ - }Set(j_1, j_2, ..., j_{k-1})$ contains no non-zero total-probability elements the process has reached the end of a search-path. In this case the distribution $p(Z)$ is complete and is used to calculate a value for $H(Z)$ which is stored in a list, which will later become the input to the completion step (3, below). After calculating and storing the value for $H(Z)$, the process returns to the *(k-1)*th recursion level where it resumes its iterations on sub-step (2.b).

b. If the input $Total\text{-}Probability\text{-}Set(j_1, j_2, ..., j_{k-1})$ is found to contain at least one non-zero total-probability element, then two tasks (i and ii, explained below) are executed *for each* non-zero total-probability element in $Total\text{-}Probability\text{-}Set(j_1, j_2, ..., j_{k-1})$, beginning first with the largest such element, i.e. $p_{tot}(z)_{1,k-1}$, and proceeding to the second largest, i.e. $p_{tot}(z)_{2,k-1}$ and so forth. These are the "elements", referred to above, that are processed iteratively and which are indexed in general as $p_{tot}(z)_{j,k-1}$. Note that the elements with value of zero are excluded. When both of tasks i and ii have been performed for each such non-zero total-probability element, the *k*th recursion level is deemed completed and the process returns to the *(k-1)*th recursion level where it resumes its iterations on sub-step (2.b) at that level.

Now, the two tasks that are performed in each iteration are as follows:

i. On the *(j,k)*th iteration, the outcome z that is associated with $p_{tot}(z)_{j,k-1}$ is assigned the index *k*, meaning that $p(z_k)$ is assigned the value $p_{tot}(z)_{j,k-1}$. The significance of this is that $p_{tot}(z)_{j,k-1}$ is *assumed* to be the *k*th largest probability in a distribution $p(Z)$ that will be used to calculate the smallest value for $H(Z)$. Whether this assumption will turn out to be correct will be tested below in the completion step (3). This value for $p(z_k)$ is stored in a list that represents $p(Z)$ which, when complete, will be used as input into sub-step (2.a).

ii. Next, the process produces $Total\text{-}Probability\text{-}Set(j_1, j_2, ..., j_k)$ and sends it as input into the *(k+1)*th recursion-level. Note that the invocation of the *(k+1)*th recursion-level temporarily interrupts the iterative process occurring on the *k*th recursion-level, but this *k*th recursion-level iterative process will be resumed upon completion of the *(k+1)*th recursion-level.

Now, regarding the production of $\mathit{Total\text{-}Probability\text{-}Set}\left(j_1, j_2, \ldots, j_k\right)$, this is accomplished by first making a clone (copy) of $\mathit{Total\text{-}Probability\text{-}Set}\left(j_1, j_2, \ldots, j_{k-1}\right)$ and then deleting the total-probability element $p_{tot}\left(z\right)_{j,k-1}$ that was used in sub-step (2.b.i) as the candidate for z_k. Next, the remaining total-probability elements are adjusted by removing from each of their respective summations, any of the column probabilities that were used to calculate $p_{tot}\left(z\right)_{j,k-1}$. Finally, the remaining total-probability elements are sorted, top to bottom, from largest to smallest, re-indexed to reflect both the new ordering and the *k*th recursion-level, and this freshly processed, reorganized "clone" is then renamed $\mathit{Total\text{-}Probability\text{-}Set}\left(j_1, j_2, \ldots, j_k\right)$.

Now it is time to explain the meaning of the sequence $j_1, j_2, \ldots, j_k$. You may have noticed that it gets longer with each recursion-level, which is to say that the index *k* tracks these recursion-levels. On the other hand, the variable j_i tracks the iteration that was interrupted in order to begin the (*i*+1)th recursion-level. Thus, for example, $\mathit{Total\text{-}Probability\text{-}Set}\left(3,4,2\right)$ is the input to the 4th recursion-level, and was produced after interrupting, respectively, the 3rd iteration of the 1st recursion-level, the 4th iteration of the 2nd recursion-level, and the 2nd iteration of the 3rd recursion-level. This will become clearer as we work through a couple of specific examples, below.

3. **Completion**: In this step the list of $H\left(Z\right)$ values that were produced by sub-step (2.b) is searched for the smallest (minimized) value. The identification of this smallest value for $H\left(Z\right)$ completes the algorithm.

Working Through An Example

The completeness and formality of the above presentation of this algorithm may have left the reader feeling a little baffled as to just how it works. As I said, it's really not as complicated as it might appear. The following example should go a long way toward illustrating its fundamental simplicity and efficacy.

We will continue with the illustrative example used earlier, e.g. the payoff matrix $\psi : R \times S \to Z$ and the distribution $p(S)$, reproduced here:

$$\begin{array}{c|cccccc} \psi & s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ \hline r_1 & a & h & d & g & b & h \\ r_2 & c & f & i & e & c & d \\ r_3 & f & d & e & b & a & i \\ r_4 & a & d & c & e & a & f \end{array}$$

$$P(S) = \{ p(s_1), p(s_2), p(s_3), p(s_4), p(s_5), p(s_6) \}$$

And in order to make the following discussion more concrete, we will arbitrarily assign the following numbers to the probabilities in $P(S)$ as follows:

$$p(S) = \left\{ \begin{array}{l} p(s_1) = 0.05, \\ p(s_2) = 0.20, \\ p(s_3) = 0.15, \\ p(s_4) = 0.10, \\ p(s_5) = 0.30, \\ p(s_6) = 0.20 \end{array} \right\}$$

The three steps of the algorithm are applied as follows:

Initialization Step:

1. For each element $z \in Z$, calculate $p_{tot}(z)$, i.e. the sum of all of the column probabilities that are available to that element and list these with appropriate *(j,0)* indexes in a descending-ordered table to be called $Total\text{-}Probability\text{-}Set(\ )$. For our chosen example, the table we would create would be the following:

$Total\text{ - }Probability\text{ - }Set(\ \ )$

$$\boxed{\begin{aligned}
&p(d)_{1,0} = p(s_2) + p(s_3) + p(s_6) = 0.20 + 0.15 + 0.20 = 0.55,\\
&p(c)_{2.0} = p(s_1) + p(s_3) + p(s_5) = 0.05 + 0.15 + 0.30 = 0.50,\\
&p(f)_{3,0} = p(s_1) + p(s_2) + p(s_6) = 0.05 + 0.20 + 0.20 = 0.45,\\
&p(a)_{4,0} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35,\\
&p(b)_{5,0} = p(s_4) + p(s_5) = 0.10 + 0.30 = 0.40,\\
&p(h)_{6,0} = p(s_2) + p(s_6) = 0.20 + 0.20 = 0.40,\\
&p(i)_{7,0} = p(s_2) + p(s_6) = 0.20 + 0.20 = 0.40,\\
&p(e)_{8,0} = p(s_3) + p(s_4) = 0.15 + 0.10 = 0.25,\\
&p(g)_{9,0} = p(s_4) = 0.10,
\end{aligned}}$$

The above table is our initial input into the subsequent iterato-recursive process.

Iterato-Recursive Process:

2. First, as instructed by the algorithm, we examine $Total\text{ - }Probability\text{ - }Set(\ \ )$ to see if it contains any non-zero total-probability elements, which it clearly does (the alternative is trivial and wouldn't have made a very good illustrative example). Thus we execute sub-step (2.b) of the algorithm by performing tasks 2.b.i and 2.b.ii over each of the non-zero total-probability elements in $Total\text{ - }Probability\text{ - }Set(\ \ )$, beginning with the largest at the top of the list and working our way down. We will not walk through each of these iterations here because they will combine multiplicatively with the iterations in subsequent recursion-levels and produce, needless to say, far too many examples than we actually need to illustrate the algorithm. But we will begin with at least the first which is $p(d)_{1,0}$. Task 2.b.i requires that we set $z_1 = d$ and $p(z_1) = p(d)_{1,0} = 0.55$, which is to say that we hypothesize that $H(Z)$ will be minimized with a distribution $p(Z)$ in which $p(z_1) = p(d)_{1,0} = 0.55$. At this point this is only a hypothesis, although in the tests I have done with relatively small tables it turns out to be true in the large majority of cases. The whole purpose of the search algorithm, however, is to find all of the cases and this can only be done by checking all of the relevant candidates, hence the iterative part of this step.

Next, we store this hypothesized value for $p(z_1)$ in a list, e.g. $p(Z) = \{ p(z_1) = 0.55, ... \}$, and proceed with task 2.b.ii. In this task we make a clone (copy) of $Total\text{ - }Probability\text{ - }Set(\ )$, delete $p(d)_{1,0}$ and adjust the remaining total-probability elements by removing from their respective summations any column probability that was used in the sum for $p(d)_{1,0}$, i.e. $p(s_2) = 0.20$, $p(s_3) = 0.15$ and $p(s_6) = 0.20$. Then we re-index and re-sort the total-probability elements, re-name the clone to $Total\text{ - }Probability\text{ - }Set(1)$, and finally we interrupt the 1st recursion-level's iteration and invoke the 2nd recursion-level, using $Total\text{ - }Probability\text{ - }Set(1)$ as the input.

Let's walk through this one step at a time. First we make the clone:

$Total\text{ - }Probability\text{ - }Set(\ )$ (clone)

$$\boxed{\begin{aligned}
&p(d)_{1,0} = p(s_2) + p(s_3) + p(s_6) = 0.20 + 0.15 + 0.20 = 0.55,\\
&p(c)_{2.0} = p(s_1) + p(s_3) + p(s_5) = 0.05 + 0.15 + 0.30 = 0.50,\\
&p(f)_{3,0} = p(s_1) + p(s_2) + p(s_6) = 0.05 + 0.20 + 0.20 = 0.45,\\
&p(a)_{4,0} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35,\\
&p(b)_{5,0} = p(s_4) + p(s_5) = 0.10 + 0.30 = 0.40,\\
&p(h)_{6,0} = p(s_2) + p(s_6) = 0.20 + 0.20 = 0.40,\\
&p(i)_{7,0} = p(s_2) + p(s_6) = 0.20 + 0.20 = 0.40,\\
&p(e)_{8,0} = p(s_3) + p(s_4) = 0.15 + 0.10 = 0.25,\\
&p(g)_{9,0} = p(s_4) = 0.10,
\end{aligned}}$$

Now we delete $p(d)_{1,0}$ from the clone, and remove the column probabilities $p(s_2) = 0.20$, $p(s_3) = 0.15$ and $p(s_6) = 0.20$ from where ever else they appear in the clone, which gives us:

$Total\text{ - }Probability\text{ - }Set(\ \ )$ (clone, partially adjusted)

$$\boxed{\begin{array}{l}
p(c)_{2.0} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(f)_{3,0} = p(s_1) = 0.05, \\
p(a)_{4,0} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(b)_{5,0} = p(s_4) + p(s_5) = 0.10 + 0.30 = 0.40, \\
p(h)_{6,0} = 0, \\
p(i)_{7,0} = 0, \\
p(e)_{8,0} = p(s_4) = 0.10, \\
p(g)_{9,0} = p(s_4) = 0.10,
\end{array}}$$

Now we re-sort and re-index the total-probability elements that remain in the clone and re-name the clone to $Total\text{ - }Probability\text{ - }Set(1)$, thus:

$Total\text{ - }Probability\text{ - }Set(1)$

$$\boxed{\begin{array}{l}
p(b)_{1,1} = p(s_4) + p(s_5) = 0.10 + 0.30 = 0.40, \\
p(c)_{2.1} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(a)_{3,1} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(e)_{4,1} = p(s_4) = 0.10, \\
p(g)_{5,1} = p(s_4) = 0.10, \\
p(f)_{6,1} = p(s_1) = 0.05, \\
p(h)_{7,1} = 0, \\
p(i)_{8,1} = 0,
\end{array}}$$

Note that the (1) suffixed to the name of this *Total-Probability-Set* indicates that it was created as input to the 2nd recursion-level by interrupting the 1st iteration of the 1st recursion-level.

3. Now, we interrupt the iterative process of the 1st recursion-level and begin the 2nd recursion-level using $Total\text{ - }Probability\text{ - }Set(1)$ as the input. (Note that if we were to walk through all of the iterations at each recursion-level then we would resume the 1st recursion-level iterative process only when the 2nd recursion-level has been completed). Because it contains non-zero total-probability elements this means we execute step (2.b), iterating, as we are in

the process of doing for the 1st recursion-level, from the top to the bottom of the elements in $Total\text{ - }Probability\text{ - }Set(1)$. Thus, we begin the 2nd recursion-level's iterative process with $p(b)_{1,1}$ and hypothesize that $z_2 = b$ and thus that $p(z_2) = p(b)_{1,1} = 0.40$. Thus, again, we are *assuming* that $H(Z)$ will be minimized with a distribution $p(Z)$ in which $p(z_2) = p(b)_{1,1} = 0.40$. As we did in the 1st recursion-level, we store this hypothesized value for $p(z_2)$ in the list $p(Z) = \{p(z_1) = 0.55, p(z_2) = 0.40, \ldots\}$.

In task (2.b.ii) we make a clone (copy) of $Total\text{ - }Probability\text{ - }Set(1)$, delete $p(b)_{1,1}$ and adjust the remaining total-probability elements by removing from their respective summations any column probability that was used in the sum for $p(b)_{1,1}$, i.e. $p(s_4) = 0.10$ and $p(s_5) = 0.30$. Then we re-index and re-sort the total-probability elements, re-name the clone to $Total\text{ - }Probability\text{ - }Set(1,1)$, and finally we interrupt the 2nd recursion-level's iterative process and invoke the 3rd recursion-level, using as input $Total\text{ - }Probability\text{ - }Set(1,1)$ which (the reader can check) appears as follows:

$$Total\text{ - }Probability\text{ - }Set(1,1)$$

$$\boxed{\begin{array}{l} p(a)_{1,2} = p(s_1) = 0.05, \\ p(c)_{2.2} = p(s_1) = 0.05, \\ p(f)_{3,2} = p(s_1) = 0.05, \\ p(e)_{4,2} = 0, \\ p(g)_{5,2} = 0, \\ p(h)_{6,2} = 0, \\ p(i)_{7,2} = 0, \end{array}}$$

4. Because $Total\text{ - }Probability\text{ - }Set(1,1)$ has non-zero total-probability elements we initiate the iterative process beginning with $p(a)_{1,2} = p(s_1) = 0.05$. This means that we set $z_3 = a$ and $p(z_3) = p(a)_{1,2} = 0.05$. Also, we store this information in the list (which is now a complete probability distribution since its elements sum to unity):

$$p(Z) = \{ p(z_1) = 0.55, p(z_2) = 0.40, p(z_3) = 0.05 \}$$

Next we clone $Total\text{ - }Probability\text{ - }Set(1,1)$ and use it to make $Total\text{ - }Probability\text{ - }Set(1,1,1)$, which (the reader should check) appears as follows:

$Total\text{ - }Probability\text{ - }Set(1,1,1)$

$$\boxed{\begin{array}{l} p(c)_{1.3} = 0, \\ p(f)_{2,3} = 0, \\ p(e)_{3,3} = 0, \\ p(g)_{4,3} = 0, \\ p(h)_{5,3} = 0, \\ p(i)_{6,3} = 0, \end{array}}$$

5. Now we interrupt the 3rd recursion-level's iterative process and use $Total\text{ - }Probability\text{ - }Set(1,1,1)$ as input to the 4th recursion-level. This time we see that there are no non-zero total-probability elements, which means we execute step (2.a). In other words, we have come to the end of our current search path, as also indicated by the previously noted fact that $p(Z)$ is a complete probability distribution, and we can now use $p(Z)$ to calculate a value for $H(Z)$. This value is calculated thus:

$$H(Z) = (0.55)\log_2(0.55) + (0.40)\log_2(0.40) + (0.05)\log_2(0.05) \approx 1.219241$$

As I mentioned earlier, as far as I have been able to ascertain there is really no way to know if this is the absolute minimum value for the entropy that could be attained for the given example. In this example we have completed just one of many possible search paths, and in order to know if the value obtained above is really the minimum, all of these paths have to be explored and the resulting candidate values for $H(Z)$ must be compared in order to determine the true minimum. The search algorithm detailed above will accomplish exactly that, and it will do so much more quickly than a brute-force approach. On the other hand, in the simulations I have run with relatively small tables, this first value has most often turned out to be the minimum, and when it isn't, it's very close to the actual minimum. Thus, the search-algorithm presented here is easily converted to a search heuristic, simply by limiting the number of iterations performed at each recursion-level. In the example we just completed we performed just one iteration per recursion-level. However, for the sake of illustration, let's quickly walk through another.

6. Step (2.a) now requires that we store the value that we calculated for $H(Z)$ (we'll just leave it where it is on the page above) and then return to the 3rd recursion-level in order to resume that level's iterative process. We won't actually do this because a glance at the table shows us that the remaining iterations will produce the exact same value for $H(Z)$. Instead, we'll jump ahead to the end of those 3rd recursion-level iterations, at which point we return to the 2nd recursion-level to continue with the iterative process at that level. Note that in order to make this shift up two levels we have to restore $p(Z)$ to its form at that level, which means we have to remove the values we calculated for $p(z_2)$ and $p(z_3)$. Thus, we have that $p(Z) = \{ p(z_1) = 0.55, ... \}$.

   At the 2nd recursion-level we are processing $\mathit{Total\text{-}Probability\text{-}Set}(1)$ which we can reproduce here:

   $$\mathit{Total\text{-}\ Probability\text{-}\ Set}(1)$$

   $$\boxed{\begin{array}{l}
p(b)_{1,1} = p(s_4) + p(s_5) = 0.10 + 0.30 = 0.40, \\
p(c)_{2.1} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(a)_{3,1} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35, \\
p(e)_{4,1} = p(s_4) = 0.10, \\
p(g)_{5,1} = p(s_4) = 0.10, \\
p(f)_{6,1} = p(s_1) = 0.05, \\
p(h)_{7,1} = 0, \\
p(i)_{8,1} = 0,
\end{array}}$$

   Now, the first iteration at this level hypothesized that $z_2 = b$, and now we will move down and take the next largest total-probability element in the list which is $p(c)_{2.1} = p(s_1) + p(s_5) = 0.05 + 0.30 = 0.35$. This time we set $z_2 = c$ and thus $p(z_2) = p(c)_{2.1} = 0.35$, which means that $p(Z) = \{ p(z_1) = 0.55, p(z_2) = 0.35 ... \}$. Next, we perform task (2.b.ii) on $\mathit{Total\text{-}Probability\text{-}Set}(1)$ to construct $\mathit{Total\text{-}Probability\text{-}Set}(1,2)$, which is as follows:

$$Total\text{ - }Probability\text{ - }Set(1,2)$$

$$\boxed{\begin{array}{l} p(b)_{1,2} = p(s_4) = 0.10, \\ p(e)_{2,2} = p(s_4) = 0.10, \\ p(g)_{3,2} = p(s_4) = 0.10, \\ p(a)_{4,2} = 0, \\ p(f)_{5,2} = 0, \\ p(h)_{6,2} = 0, \\ p(i)_{7,2} = 0, \end{array}}$$

7. Once again, we interrupt the 2nd iteration of the 2nd recursion-level and begin the 3rd recursion-level iterations, using as input $Total\text{ - }Probability\text{ - }Set(1,2)$. Since we are beginning a totally new recursion-level we begin with the first iteration in that level, i.e. $p(b)_{1.2} = 0.10$. Thus, we set $z_3 = b$ and $p(z_3) = p(b)_{1.2} = 0.10$, which we store in

$$p(Z) = \{ p(z_1) = 0.55, p(z_2) = 0.35, p(z_3) = 0.10\}$$

Next we perform task 2.b.ii on $Total\text{ - }Probability\text{ - }Set(1,2)$ in order to produce $Total\text{ - }Probability\text{ - }Set(1,2,1)$, which appears as follows:

$$Total\text{ - }Probability\text{ - }Set(1,2,1)$$

$$\boxed{\begin{array}{l} p(e)_{2,2} = 0, \\ p(g)_{3,2} = 0, \\ p(a)_{4,2} = 0, \\ p(f)_{5,2} = 0, \\ p(h)_{6,2} = 0, \\ p(i)_{7,2} = 0, \end{array}}$$

8. Taking $Total\text{ - }Probability\text{ - }Set(1,2,1)$ as input to the 4th recursion-level, we see that once again (both because there are no non-zero total-probability elements and because $p(Z)$ is a complete probability distribution) we have come to the end of a search path and can calculate another value for $H(Z)$:

$$H(Z) = (0.55)\log_2(0.55) + (0.35)\log_2(0.35) + (0.10)\log_2(0.10) \approx 1.33667$$

The algorithm directs us to store this value for $H(Z)$ along with the one we obtained earlier and then to return to the 3rd recursion-level to resume that level's iterative process. A quick glance at $\textit{Total - Probability - Set}(1,2)$ reveals that doing so will only produce the exact same value for $H(Z)$. In any case, having worked our way through two complete search paths will hopefully have accomplished the goal of clarifying in the reader's mind the way this algorithm works. The last step of the algorithm, the completion step, is straight forward. The list of values for $H(Z)$ that were accumulated in the iterato-recursive step is searched for the absolute smallest. Clearly there will be at least one of these, but often there will be more than one.

Finding Excellent Approximations to Ideal Good-Regulator Models

Although the above algorithm is much more efficient than any brute force search approach, even a moderately complex system will probably have a search space that will simply be too large to completely cover in any practical length of time. Even in such cases, however, the algorithm is useful because the very first step in the algorithm will always yield an excellent approximation to the ideal good-regulator model. This first step can be achieved very rapidly even with very complex systems, and depending on how much search time is available to continue past the first step, the algorithm has a good chance of turning up improvements on that excellent approximation.

Real-World Applications: Goals for Future Research

This completes our current discussion of the search algorithm. From here I can see three logical next steps to take along this line of inquiry. The first step, clearly, is to write a computer program to implement this search algorithm. Although I have already completed various prototypes of such a program and have used these to answer various questions regarding the performance of the basic algorithm in an informal way, I still have a good deal of programming and testing to do in this direction.

The second and third steps are to show how such a computer program can be used to solve, on the one hand, problems for which we already have solutions, and on the other, new problems which are currently unsolved. The taking of these two steps obviously requires the completion of the programming step. Regarding the plausibility of success on these second and third steps, I can only point out that the input to the search algorithm can be obtained at a very high level. Only the distribution $p(S)$ and the mapping $\psi : R \times S \to Z$ need to be specified. The algorithm will calculate the "underdog" mapping $u : S \to Z$ from which it is a simple matter to find the "good regulator" mapping $g : S \to R$. These sorts of measurements are well within the reach of current measurement technologies and resemble the sorts of measurements made, for example, in the field of Operations Research. In short, I see no reason to hesitate in the taking of these next three logical steps along the line of inquiry established by the Conant and Ashby theorem. Although, as for any investigation, there can be no guarantee that taking these steps will produce anything useful, in my opinion the available evidence warrants a fairly high degree of confidence that it will.

Even Decent Regulators Must Be Models

We will now turn our attention to the more general version of the Good-Regulator Theorem. As discussed in the introduction, the value of this theorem is two-fold. First of all, the type of idealized, maximally optimal and simple good-regulator model described by the C&A theorem is a difficult thing to find, even with the search-algorithm described in the previous pages. The fact is that the overwhelming majority of system-regulators that exist in the world are really just decent to excellent *approximations* to their ideal versions. It would be nice to understand the relationship between these sorts of effective-but-not-quite-perfect regulators and their ideal counterparts referred to by the C&A theorem, and it is this more general version of the good-regulator theorem that makes the connection. Secondly, the proof of this more general version is extremely simple and thus has great pedagogical value. Whereas C&A's theorem is probably beyond the lay-person's ability to understand without a good deal of training, the version of this theorem we are about to examine requires little more than the ability to read and count. Such a simpler and more general version should go a long way toward making the Good-Regulator Theorem accessible to a wider audience.

The idea for this version of C&A's theorem comes partly from acknowledging the *role* played in C&A's original proof by the lemma that they used to establish what I am calling the "underdog" mapping $u : S \to Z$. As far as I can see, the role of their lemma is nothing more or less than to establish that mapping, and as regulator designers we are free to constrain the regulator to such a mapping for *any reason at all*, not simply because we want to minimize the entropy. I think that in the most general case, we could just *assume* that there is some sort of justification for constraining the regulator to behave according to an underdog mapping, i.e., that a change in outcome can only arise from a change in column, and that we could make this assumption regardless of any given outcome's impact on the entropy function. Another way to put it is that we can set aside the entropy function and open the door to other ways to define the idea of "successful regulation". The only criterion we need to fix, at this point, is that whatever else it does, it must obey some sort of an "underdog" mapping $u : S \to Z$. Then the only other assumption we would need is the economical one that the regulator should be as simple as possible. Such an approach would have maximum generality and be so simple that a child could understand it.

But there is also *slightly* less general way that has widespread applicability and that still doesn't require any sophisticated mathematics (such as the Shannon Entropy function). This version of the theorem posits that a regulator is optimal to the extent that it is selecting outcomes from Z *according to a specified preference ranking* on Z (i.e., some mapping $f : Z \to \mathbb{N}_1$). Note that this definition of successful regulation does not require the entropy function and thus it does not require that we know anything about $p(S)$ or $p(Z)$. All we need to know is that some outcome $z_1 \in Z$ is said to be the "best" outcome, z_2 is "second best", and so forth. We will say that R is behaving optimally as long it responds to S with respect to this preference ranking. Now, if we also require that it achieves this result as simply as possible, then it turns out that R will also be conducting itself according to some mapping $g : S \to R$ (i.e., R will be a model of S).

The proof is nearly trivial and requires only the recognition that no matter what S does, there will always be a "best available outcome", as defined by the preference ranking, and that therefore there could never be a rational justification for R to choose, say, that "best available" outcome on some occasions and some "less than best available" on others. Thus, R will always choose just one outcome per column, which establishes the underdog mapping $u : S \to Z$. From there, the economical simplifying assumption then establishes the good-regulator mapping $g : S \to R$.

One point we can recognize here is that C&A's Good-Regulator Theorem is actually a special case of this "preference ranking" version. Although C&A do not specifically refer to such a preference-ranking, it is implicit in their discussion. That is a fairly strong claim, but it was proven several pages back when I showed that any $p(R \mid S)$ that minimizes $H(Z)$ is logically equivalent to the preference ranking that is established by the natural ordering of the probabilities in $p(Z)$ that were used to calculate that minimized value for $H(Z)$. The thing to notice here is that whenever a regulator manages to minimize the entropy function, it is only able to do so because it is selecting its outcomes according to a particular preference ranking that allows it to do so.

Another point to recognize is that although the proof of this more general version of the Good-Regulator Theorem is nearly trivial, the theorem itself is profoundly important because the type of regulatory process it describes – possibly sub-optimal and driven by a preference-ranking – is *ubiquitous* throughout the biological world in general and throughout the world of human behavior in particular[9], vastly more so than the type of idealized maximally optimal regulator described by C&A's version. The main difference between the two is that this more general version ignores the importance of stability. C&A's version places the important constraint on the regulator that its preference-ranking maximally *stabilize* the outcomes it produces. Such stability is almost always important to the very integrity of the regulator – any regulator that ignores the stability requirement runs the risk of self-destruction. On the other hand, the requirement that the regulator achieve the *maximum* amount of stability that is possible under the given circumstances has the unfortunate effect of obscuring the theorem's relevance to any discussion of real-world regulators. By ignoring this criterion on the preference-ranking, the more general version allows us to use the Good-Regulator Theorem in reference to all of the almost-certainly sub-optimal regulator processes we see in the world. Of course, this increased generality also means that the theorem applies to regulators that might destroy themselves with their so-called "successful" regulation, but because they do tend to destroy themselves, we don't encounter very many of those.

[9]The process of Natural Selection is just one really important example that explains how species come to represent their habitats. From the domain of human behavior, our efforts to "get our priorities straight" reflect an implicit understanding of the "Law of Requisite Back-up Plans", discussed in the next section.

A Law of Requisite Back-Up Plans

Another result that drops out of this analysis is something I have come to think of as "The Law of Requisite Back-Up Plans". Whether it is actually a law or perhaps just an axiom of regulation depends on whether we want to define regulation in terms of minimizing the entropy function or in terms of a preference ranking on the outcomes. If we define it as a preference ranking on the outcomes, then we are simply assuming it as an axiom, but if we define regulation in terms of the entropy function, then it is deducible as an actual law, which can be stated as follows:

The Law of Requisite Back-Up Plans

> Successful regulation requires the prior establishment of a complete set of contingency plans over the set of all possible outcomes Z.

In this context, a "contingency plan" is simply a rule that specifies, for any given outcome $z \in Z$, the *next best* outcome to accept in case the outcome z is not available. Thus, "a complete set" of such contingency plans, defined "over the set of all possible outcomes Z," would provide such a "back-up" plan for each outcome in Z. That this law should hold (provided we have defined successful regulation as minimized entropy) can be seen by recognizing the logical equivalence of such a set of contingency plans and a preference ranking over the outcomes in Z. That is, to assert that outcome z_1 is preferable to outcome z_2, which, in turn, is preferable to outcome z_3, and so forth, is to say nothing more or less than that outcome z_2 is the "contingency plan" for outcome z_1, that outcome z_3 is the contingency plan for outcome z_2, etc. Of course, if we define successful regulation in terms of such a preference ranking regardless of its impact on the entropy function, then the above law is reduced to the status of an axiom.

Part of the utility of this law is that it clarifies a major responsibility of anyone who would try to design a successful regulator. What it tells us is that it is not enough just to concern ourselves with what the possible outcomes are, or even with what the most desirable outcomes are. In order to design a successful regulator, not only do we have to know what *all* of the outcomes are, but we also have to rank them *all* with respect to each other.

This runs counter to the popular advice that places great emphasis on so-called "positive thinking" and focusing only on our highest aspirations. "Don't dwell on the negative", "Keep your eyes on the prize" and so forth. The Law of Requisite Back-Up Plans tells us that if we really want to be successful, we need to step back and take in a larger view. It isn't enough to focus only on our highest aspirations. We must know in advance what we will do if our loftiest goals just aren't possible.

Ashby’s First Law: The Any-Port-In-A-Storm Theorem

The C&A theorem specifies a necessary condition for the successful regulation of any system. As such, it can be cast in the logical form of a material conditional (i.e. “if P then Q”):

If a system is being optimally and most simply regulated,
then some model of that system is driving the regulation.

This is a simple but important point. The C&A theorem is *not* a sufficient condition for successful regulation, nor is it any sort of promise that the theorem’s converse (the “if Q then P” version) might hold for any given case. The converse of the C&A theorem would be:

If a model of a system is driving its regulation, then that
system is being optimally and most simply regulated.

It is important to keep the distinction clear between these two propositions, the first of which is true, while the second is maybe true but maybe false, depending on the particular model that is driving the regulation. This point really needs to be emphasized because despite the profound utility of the C&A theorem, its status as necessary and not sufficient condition (let’s be frank about it) *is a bit of a disappointment*. I say that somewhat facetiously, of course, because my own assessment of this theorem (in case it isn’t obvious) borders on the kind of dogmatic reverence that is usually reserved for prophetic scripture. But let’s face it, as useful as it is to specify any given prerequisite or necessary condition to achieve some valued objective, what we would really like more than anything is the complete set of all the prerequisites, which is to say, the *sufficient* condition and the C&A theorem falls far short of such a thing. This state-of-affairs is perfect for provoking the kind of wishful thinking that might otherwise blind us to the way the world really works, thus tricking us into misunderstanding the C&A theorem.

On the other hand, there is a proposition which, though not really the converse of the C&A theorem, bears a useful resemblance to it, and it is this proposition that I am calling *Ashby’s First Law*, or the *Any-Port-In-A-Storm theorem*. Perhaps the simplest way to put it is as follows:

Any model is better than no model at all

This result follows from a basic property of the entropy function that in turn follows from the facts that $1 \cdot \log 1 = 1 \cdot 0 = 0$ and also that for the purposes of the entropy function, the quantity $0 \cdot \log 0$ is defined to be zero. Thus, for any discrete probability distribution $P = \{p_1, p_2, ..., p_n\}$, such that $p_i = 0$ or 1, for all $p_i \in P$, we have that

$H(p_1, p_2, ..., p_n) = -\sum_{i=1}^{n} p_i \log p_i = 0$. Now, let's consider a specific example as represented by the following table:

ψ	s_1	s_2	s_3	s_4	s_5	s_6
r_1	b	e	d	g	b	h
r_2	c	f	i	e	g	d
r_3	f	d	e	b	a	i
r_4	a	d	c	e	i	f

One thing to notice about this example is that there are no duplicate outcomes in any of the rows, although there are in some of the columns. This is a restriction that we did not make in our previous discussion of the search algorithm. The result of this assumption is that the following relationship, the utility of which will be made clear in a moment, holds:

$$H(S \mid R) = H(Z \mid R)$$

To motivate and illustrate what follows, let's begin by recalling that when a regulator is behaving in accord with a mapping $h : S \to R$, the associated conditional distribution that defines the regulator, $p(R \mid S) = \{p(r \mid s) : r \in R, s \in S\}$, is such that $p(r_i \mid s_j) = 0 \text{ or } 1$, for each $p(r_i \mid s_j) \in p(R \mid S)$. But as regulator designers, we are not required to obey this like dogma. It just so happens that – by the C&A theorem – we must do it if we wish to minimize $H(Z)$, the entropy associated with the resulting outcomes. But if that is not our wish, then we only have to obey the rule that $0 \le p(r_i \mid s_j) \le 1$. Because of this, a regulator could be designed to *maximize* $H(Z)$, or perhaps to attain some value in between the maximum and minimum.

As a general rule-of-thumb (meaning that important exceptions exist) the greater the number of outcomes ultimately selected, the greater the resulting entropy that is associated with those outcomes. Conversely – and again, as a general rule-of-thumb – the fewer the outcomes, the lower the entropy. It is important to realize that this is not strictly true, and plenty of counter examples are easily found. For example, a regulator that produces three outcomes $\{a, b, c\}$, such that $p(a) = .80$ and $p(b) = p(c) = .10$ has an entropy of $-.8\log.8 - .1\log.1 - .1\log.1 \cong 0.922$ whereas a different regulator that produces only two outcomes $\{a, b\}$ such that $p(a) = p(b) = .50$ has an entropy of $-.5\log.5 - .5\log.5 = 1.00$. The notion of stability that is captured by the concept of minimized entropy is more complicated than can be described merely with the number of outcomes produced – what Ashby called *Variety*. What is also important is the frequency with which these outcomes appear. It is ironic that Ashby's notion of *Variety* turns out to lack the variety requisite to the task it was meant to accomplish – the measurement of stability. In any case, even though it is not strictly true, it's true often enough to make the following discussion meaningful. We will deal with the counter examples later.

In the case we are considering there are a total of 9 possible outcomes $Z = \{a,b,c,d,e,f,g,h,i\}$, and any regulator that maximized $H(Z)$ would assign some sort of positive probability to each one of those outcomes. On the other hand, any regulator that behaves according to some mapping $h : S \to R$ – any mapping at all – must necessarily produce fewer than 9 outcomes, and thus (again, as a rule-of-thumb with exceptions very possible) a lower entropy. To see why this is the case, consider the following three arbitrary mappings, $h_i : S \to R$, for $i \in \{1,2,3\}$, along with the outcomes they produce :

1. $h_1 \downarrow \dfrac{s_1 \quad s_2 \quad s_3 \quad s_4 \quad s_5 \quad s_6}{r_2 \quad r_3 \quad r_1 \quad r_2 \quad r_4 \quad r_3}$, produces the outcomes $\{c,d,e,i\} \subset Z$

2. $h_2 \downarrow \dfrac{s_1 \quad s_2 \quad s_3 \quad s_4 \quad s_5 \quad s_6}{r_3 \quad r_3 \quad r_3 \quad r_3 \quad r_3 \quad r_3}$, produces the outcomes $\{f,d,e,b,a,i\} \subset Z$

3. $h_3 \downarrow \dfrac{s_1 \quad s_2 \quad s_3 \quad s_4 \quad s_5 \quad s_6}{r_4 \quad r_1 \quad r_1 \quad r_3 \quad r_4 \quad r_2}$, produces the outcomes $\{a,b,d,e,i\} \subset Z$

Notice that in each case, the number of outcomes actually produced is less than $|Z|$, the total number than could be produced if we didn't use a mapping, and, as already mentioned, when this sort of thing happens, the resulting entropy tends to be lower (with important exceptions).

These examples are meant to illustrate the gist of Ashby's First Law: any model is better than no model. Unfortunately, these examples rely on Ashby's concept of variety – i.e. the number of distinct elements in a set – which as, already pointed out, lacks the variety requisite to the task before us. In order to handle the exceptions, we need the entropy function; we proceed as follows:

Another type of entropy that we can calculate is that associated with the regulator's responses for any given column, say the column associated with system behavior $s_4 \in S$. This is defined to as follows:

$$H(R \mid S = s_4) = -\sum_{i=1}^{4} p(r_i \mid s_4) \log p(r_i \mid s_4)$$

In the most general case we have that $0 \le p(r_i \mid s_j) \le 1$, but for any mapping $h : S \to R$ it must be the case that $p(r_i \mid s_j) = 0 \text{ or } 1$, and so, by the property described above, $H(R \mid S = s_4) = -\sum_{i=1}^{4} p(r_i \mid s_4) \log p(r_i \mid s_4) = 0$. Since this holds for each $s_j \in S$, we have that,

$$H(R \mid S) = E\left[H(R \mid S = s_j)\right] = \sum_{j=1}^{|S|} p(s_j) \left(\sum_{i=1}^{|R|} p(r_i \mid s_j) \log p(r_i \mid s_j) \right) = \sum_{j=1}^{|S|} p(s_j)(0) = 0$$

A point I want to highlight here is that this is true for *any* mapping $h : S \to R$, regardless of its impact on the entropy function.

Now, in his discussion of his *Law of Requisite Variety*[10], Ashby performs the following derivation:

$$H(R,S) = H(S \mid R) + H(R) = H(R \mid S) + H(S)$$

$$H(S \mid R) = H(R \mid S) + H(S) - H(R)$$

$$H(Z) \geq H(Z \mid R) = H(R \mid S) + H(S) - H(R)$$

Where the last line in this derivation makes use of our assumption that there are no duplicate outcomes in any given row of the table, that is $H(S \mid R) = H(Z \mid R)$. The purpose of this argument is three-fold. First of all, it shows that there is an absolute minimum below which further reduction in $H(Z)$ is impossible. Secondly, it shows us that as regulator designers, we can have quite an impact on that lower limit. And thirdly, it shows us two ways to lower that limit. The first way is to use a mapping $h : S \to R$, which will set $H(R \mid S) = 0$, and the second is to increase $H(R)$.

Ashby's Law of Requisite Variety states that provided we have used a mapping $h : S \to R$ that fixes $H(R \mid S) = 0$, an assumption that extends the above derivation to the following line:

$$H(Z) \geq H(S) - H(R),$$

then the only way to further lower the absolute limit on $H(Z)$ is to increase the value of $H(R)$, the entropy associated with the regulator's responses. This is what I think of as Ashby's *second* law – the Law of Requisite Variety. Actually, and for reasons already mentioned, the term *variety* is insufficient to the task it was meant to perform, and so it would be more accurate to call Ashby's second law the *Law of Requisite (Average) Surprise*, because the quantity $-\log p(r)$ is considered a measure

[10]http://pespmc1.vub.ac.be/Books/AshbyReqVar.pdf, although my notation here is a little different from that used by Ashby.

of the surprise we would experience at the observance of the execution of behavior $r \in R$ and so $H(R)$ is the expected (average) value of that surprise[11].

But what I am calling Ashby's *First* Law is this fact that a mapping – any mapping (i.e. model) at all – will also improve the situation by lowering that limit. I think of it as his first law because he uses it to derive the Law of Requisite Variety, that is, his second law. I also think of it as the "Any-Port-In-A-Storm Theorem", because this floor lowering effect can be accomplished with any model at all.

One additional caveat is in order regarding this notion of a floor or lower limit on $H(Z)$. The fact that $H(Z) \geq H(R|S) + H(S) - H(R)$ is an inequality means that we are ignorant about the actual location of $H(Z)$ with respect to the floor. Perhaps it is close, perhaps it is far away, we can't really know. What this means is that raising or lowering the floor might have absolutely no impact whatsoever on the actual value of $H(Z)$ that is obtained for a given regulator. Such is the nature of prerequisites and necessary conditions. They make things possible, but they don't make things certain. What this means is that we might very well regulate with a model ($H(R|S) = 0$) and with maximum possible average surprise in the regulator ($H(R)$ as large as possible), and we might still end up with a high value for $H(Z)$. On the other hand, Ashby's First and Second Laws tell us that if we don't handle these prerequisites, we will certainly be stuck with a value that is *at least* $H(Z) \geq H(R|S) + H(S) - H(R)$.

This caveat means that we have to clarify what we mean by *any model is better than no model at all*. What will always be improved with a model (given the assumption that $H(S|R) = H(Z|R)$) – *regardless of the model actually used* – is that the floor on $H(Z)$ will be lowered. To the extent that this is a good thing, the situation has been improved. However it does not mean, in general, that $H(Z)$ will actually be lowered. Whether that actually happens will depend on the particularities of the system and regulator in question. Still, $H(Z)$ cannot be actually reduced until the capacity for such reduction has been created, which means that we have to fulfill Ashby's First and Second Laws.

The above caveat notwithstanding, it is easy to find examples, especially in human behavior, that appear to be explained by Ashby's First Law. In particular, it appears to explain why human beings can come to venerate and tenaciously defend certain belief systems which are otherwise completely incoherent with the workings of the world as determined through controlled experiment. Why does so-and-so believe there is a unicorn living in his attic? Because holding such a belief causes an increase in stability (i.e. lowers the entropy) in some aspect of his life over what he would have in the absence of such a belief. *Any port in the storm*. What does he fight so vehemently against the evidence that contradicts his belief? Because to give up that belief would

[11] http://en.wikipedia.org/wiki/Self-information

leave him in an uncomfortable state of confusion regarding the mysterious sounds he hears coming from his attic late at night. *Any model is better than no model at all.*

Seen in this way, Ashby's First Law has important implications for the art of persuasion. If you have ever been frustrated by your failure to persuade someone to your point of view despite your best efforts to lay out what appears to you to be overwhelmingly convincing evidence that you are right, then it could be because you are not addressing the deeper issue that the beliefs you are arguing against have a powerful, stabilizing effect on the life and well-being of the person who holds those beliefs and that your attempts to change those beliefs are threatening to destabilize the well-being of that person. The obvious conclusion to draw here is that if you really want to change the belief, then you have to address this issue of stability.

Another part of human behavior that this theorem appears to explain is the existence of habits. Try the following thought experiment. Try to imagine what your life would look like if you had no habits whatsoever. Every waking moment would be an unpredictable mess. Some mornings you would wake up and eat breakfast, and some you'd wake up and glue all of your shoes to the ceiling. Sometimes you might answer the phone when it rang and other times you'd try to see how fast you could count backwards from 53. If you take a disciplined look at the things you do every day, or every week, or every time you eat lunch, you will see that many of them are pretty arbitrary and could easily be swapped for different ones without much difficulty, except for the fact that you never really do that. Habits are habits. Arbitrary or not, if you went around changing too many of them your life could get pretty messy.

So we have these arbitrary habits simply because they are models and although they may be somewhat arbitrary and far from the best models we can find, by virtue of the fact that they help us keep things under control, we adopt them and resist giving them up.

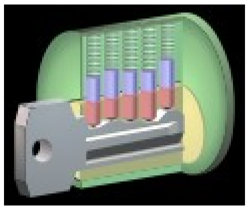
Ashby's First Law and the C&A theorem work together to form a team. On the one hand, we know that if we want to regulate a system (solve a problem, open a lock), we need to have a model of that system (problem, lock). On the other hand, it isn't always easy to find the best such model for the job at hand. But Ashby's First Law tells us that in some respects, it might not matter so long as we at least have some sort of model. Clearly this is true in only a limited sense, and only if the alternative is no model whatsoever. Once we have some sort of model, we can reasonably begin to consider whether it is the best we could have. Of course, the widespread existence of conflicting habits, beliefs, customs and cultural practices would suggest that many people just sort of stop after they get their hands on some kind of model. The whole point and practice of Science, of course, is to continually try to improve on our models.

Every Good Solution Must Be A Model Of The Problem It Solves

It has become something of a hobby for me to find metaphorical equivalents to the C&A theorem. One objective of this paper is to offer what I believe are two of the most useful[12]:

1. Every Good Key Must Be A Model Of The Lock It Opens,
2. Every Good Solution Must Be A Model Of The Problem It Solves.

The first metaphor has a luminous tangibility that makes the truth of the theorem utterly obvious. A glance at any standard pin-tumbler key shows that the ridges of the key form a precise model of the contours that the lock's inner pins must assume in order to liberate the tumbler and allow it to turn the bolt, as can be seen in the following image[13]:



The second metaphor casts the C&A theorem as a fundamental theorem of problem solving. Understood as such, the theorem equates the process of problem solving with the process of problem modeling and especially the process of problem *re-modeling*. As Herbert Simon observed, "Solving a problem simply means representing it so as to make the solution transparent."[14] In other words, the solution to the problem *is the very model* that renders the solution transparent.

Examples of this principle are abundant and easily found. One ready example can be seen in these very pages, in the various ways that I have represented the search algorithm. One such representation was the "formal" version, which was almost certainly impossible to understand at first reading. Another representation can be seen in the worked examples that followed the formal presentation. These worked examples can be seen as a control-model for the more formal presentation, and were almost certainly much easier for you to follow, and (hopefully) allowed you to understand the formal presentation. To the extent that understanding the formal representation was a problem, the worked examples (i.e. the control-model) were the solution to that problem.

[12] A few others: "Every successful species must be a model of its ecological niche", "Every good manager must be a model of the company she manages," "Every good replicator must be a model of itself." These versions cast the C&A theorem as a fundamental theorem of Biology, Management Science and Mimetics, respectively.

[13] This image is one of a series that can be found, along with a more detailed explanation of the mechanism at http://en.wikipedia.org/wiki/Pin_tumbler_lock

[14] H.A.Simon, 1981, *The Sciences of the artificial*, 2nd edition, MIT Press, Cambridge, MA, as cited in Donald A. Norman, *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine*, pg. 53, 1993, Basic Books, New York, NY.

A plentiful source of especially clear examples of this principle can be found in the study of Linear Algebra, in particular the matrix reduction techniques used to solve systems of linear equations. By representing such a system as an augmented matrix, finding the solution set is equivalent to the mechanical process of applying the 3 basic row operations, each of which effectively "re-models" the system, until the solution set is obvious. This final representation of the original system of linear equations, i.e., the final *model of that system*, is the solution[15]. Let's quickly walk through an example to illustrate this process. Consider the following system of linear equations:

$$\begin{aligned} 8x_1 - 3x_2 - 11x_3 &= -25 \\ 5x_1 - x_2 - 6x_3 &= -13 \\ -2x_1 + 5x_2 + 28x_3 &= 75 \end{aligned}$$

This system can be represented (modeled) with the following augmented matrix of coefficients:

$$\left[\begin{array}{ccc|c} 8 & -3 & -11 & -25 \\ 5 & -1 & -6 & -13 \\ -2 & 5 & 28 & 75 \end{array}\right]$$

Recall that we are free to execute any of the following three basic row-operations on the above matrix and the resulting matrix will correspond to a system of linear equations that has the exact same solution set as the original system:

1. Replace any row with a non-zero constant multiple of itself (we will represent this process in general as $k \cdot R_i \to R_j$),
2. Swap any two rows ($R_i \leftrightarrow R_j$), and
3. Replace any row with a row constructed by adding it to a constant multiple of any other row ($k \cdot R_i + R_j \to R_j$).

Performing the first row operation as $\frac{1}{8}R_1 \to R_1$ on the above augmented matrix yields the following, what we can of think of as a *re-modeled* version of the original:

$$\left[\begin{array}{ccc|c} 1 & -3/8 & -11/8 & -25/8 \\ 5 & -1 & -6 & -13 \\ -2 & 5 & 28 & 75 \end{array}\right]$$

Recall that although this matrix appears to be a bit different from the original augmented matrix, it represents the following system of linear equations which has the exact same solutions set as the original system:

[15] A representative text is Howard Anton and Chris Rorres, *Elementary Linear Algebra*, *Applications Version*, 9th edition, 2005, John Wiley & Sons, Hoboken, NJ.

$$\begin{aligned} x_1 - \tfrac{3}{8}x_2 - \tfrac{11}{8}x_3 &= -\tfrac{25}{8} \\ 5x_1 - x_2 - 6x_3 &= -13 \\ -2x_1 + 5x_2 + 28x_3 &= 75 \end{aligned}$$

Because it has the same solution set, we can view it as just a different way to represent (model) the original system.

Continuing on with the new matrix:

$$\left[\begin{array}{ccc|c} 1 & -\tfrac{3}{8} & -\tfrac{11}{8} & -\tfrac{25}{8} \\ 5 & -1 & -6 & -13 \\ -2 & 5 & 28 & 75 \end{array}\right],$$

we can execute the first row operation two more times in sequence as $\frac{1}{5}R_2 \to R_2$ and $\frac{-1}{2}R_3 \to R_3$, performing the operation each time on the output matrix from the previous row operation. Doing so yields the following:

$$\left[\begin{array}{ccc|c} 1 & -\tfrac{3}{8} & -\tfrac{11}{8} & -\tfrac{25}{8} \\ 1 & -\tfrac{1}{5} & -\tfrac{6}{5} & -\tfrac{13}{5} \\ 1 & -\tfrac{5}{2} & -14 & -\tfrac{75}{2} \end{array}\right]$$

Next, if we perform the third row operation two separate times as $-R_1 + R_2 \to R_2$ and $-R_1 + R_3 \to R_3$ we get:

$$\left[\begin{array}{ccc|c} 1 & -\tfrac{3}{8} & -\tfrac{11}{8} & -\tfrac{25}{8} \\ 0 & \tfrac{7}{40} & \tfrac{7}{40} & \tfrac{21}{40} \\ 0 & -\tfrac{17}{8} & -\tfrac{101}{8} & -\tfrac{275}{8} \end{array}\right]$$

Two executions of the first row operation as $\frac{40}{7}R_2 \to R_2$ and $-8R_3 \to R_3$ yields:

$$\left[\begin{array}{ccc|c} 1 & -\tfrac{3}{8} & -\tfrac{11}{8} & -\tfrac{25}{8} \\ 0 & 1 & 1 & 3 \\ 0 & 17 & 101 & 275 \end{array}\right]$$

Next, we execute the third row operation twice as $-17R_2 + R_3 \to R_3$ and $\frac{3}{8}R_2 + R_1 \to R_1$ to produce:

$$\left[\begin{array}{ccc|c} 1 & 0 & -1 & -2 \\ 0 & 1 & 1 & 3 \\ 0 & 0 & 84 & 224 \end{array}\right]$$

Then we execute the first row operation as $\frac{1}{84}R_2 \to R_2$ to produce:

$$\left[\begin{array}{ccc|c} 1 & 0 & -1 & -2 \\ 0 & 1 & 1 & 3 \\ 0 & 0 & 1 & 8/3 \end{array}\right]$$

And finally, two executions of the third row operation as $-R_3 + R_2 \to R_2$ and $R_3 + R_1 \to R_1$ yields:

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 2/3 \\ 0 & 1 & 0 & 1/3 \\ 0 & 0 & 1 & 8/3 \end{array}\right]$$

Remember that at each step in this process, the resulting matrix represents a system of linear equations that has the exact same solution set as the original system. Because of this, we can see the production of these intermediate matrices as sequence of models of the original system and the whole process as a re-modeling process that produces the final reduced row echelon form (RREF) of the original matrix. This final RREF matrix represents the following system of linear equations:

$$\begin{aligned} 1\cdot x_1 + 0\cdot x_2 + 0\cdot x_3 &= x_1 = 2/3 \\ 0\cdot x_1 + 1\cdot x_2 + 0\cdot x_3 &= x_2 = 1/3 \\ 0\cdot x_1 + 0\cdot x_2 + 1\cdot x_3 &= x_3 = 8/3 \end{aligned}$$

Again, this system has the same solution set as the original system, and thus can be thought of as a model of that system. Furthermore, as was described by Simon, this final model represents the original problem in such a way as to make the solution completely obvious. It other words, *it is this model that is the solution to the original problem*.

Of course, the preceding sort of analysis does not constitute a proof that "every good solution must be a model of the problem it solves." It is a plausibility argument only. One way to construct such a proof would be to piggy-back on Conant & Ashby's original argument by defining a problem in terms of, say, a "scenario repertoire" $S = \left\{ s_1, s_2, \ldots, s_{|S|} \right\}$, where the elements in S represent mutually exclusive scenarios that a given problem might present. The proof, then, would otherwise be exactly the same as for the original theorem. As solution designers, we would assume that we are equipped with a repertoire of possible responses $R = \left\{ r_1, r_2, \ldots, r_{|R|} \right\}$ that combine with the problematic scenarios in S to produce outcomes in Z according to some mapping $\psi : R \times S \to Z$. From here we could define a "good solution" either as one that produces outcomes from Z so as to minimize $H(Z)$, or more generally according to some specified preference ranking on those outcomes. Of course, if we wish to take the minimized entropy approach, we will also have to specify a probability distribution $p(S)$. The upshot here is that we don't really need a proof of this fundamental theorem of problem solving, beyond recognizing that any problem can be viewed as a system in need of regulation. Conant and Ashby have already proven the theorem.

But they did so in the technical language of Systems and Regulators, which tends to bring to mind clunky images of Thermostats and Watt-Governors and if it is not your job to build or maintain such devices, you may tend to think that the C&A theorem is something you don't really need to know about. But *everybody* has to solve problems, and what could be more useful than a fundamental theorem of problem solving that tells us how to proceed to solve them?

What must we *always* do?

Make a model of the problem.

How do we know we have to do that?

Because we know that *every good solution must be a model of the problem it solves*. Whatever else we do, we must do at least that.

Of course, most of the time this approach will fail, at least on the first attempt, but only because there are many, many ways to model any given problem, and only a relatively few will make the solution transparent. But if after modeling the problem the solution is not transparent, then we also know that we have to come up with a different model. How do we know this? Again, *because every good solution must be a model of the problem it solves*. If the model we currently have doesn't solve the problem, then we must find some other way to model the problem.[16] As illustrated above with the matrix reduction example, the C&A theorem shows us that the process of problem solving is equivalent to the process of problem modeling, and especially, the process of problem *re-modeling*.

[16] For a compelling popular examination of the importance of problem representation to problem solving, see Donald A. Norman's *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine*. 1993. Basic Books. New York, NY.

Conclusion

Have you ever stopped to notice all the models and representations we humans seem to use? A little disciplined reflection reveals that they are really just about *everywhere*. And I'm not just talking about the obvious ones – e.g. the scale model buildings used by architects; the model airplanes, trains, boats and cars played with by children and hobbyists; the fashion models that show us how to dress, adorn ourselves, stand, walk and wear our hair; and of course the computer software we use to model everything from virtual desktops to entire virtual worlds. I do mean *at least* those, of course, but I also mean the ones that may not be so obvious. For example, consider a device as simple and as common as a grocery list, which is a model of the items you need from the store. Every list, in fact, is such a model – a to-do list, a guest list, the index and table of contents for a book, a travel itinerary, a list of ingredients, etc. Or consider a typical wall-mounted light switch which models the level of light in a room – i.e. when it's in the "up" position the light is on (usually) and when it's in the "down" position the light is off. And I'll bet that every morning as you prepare for your day – as you dress, comb your hair, shave, apply make-up, etc. – you make use of the model of your appearance that is reflected back to your eyes from the surface of a mirror. I'm referring to *all* of the models we use. They really seem to be *everywhere*. If you step back and take an objective look at our modern civilized way of life what you will see is a biological species that literally walks, drives, talks, eats, works and plays with models. Here are a few more examples to illustrate the point:

- A city street map is a model of the actual city streets (thus, we walk and drive with models).
- Any spoken or written sentence is a model of the real-world events or objects that form the topic of that sentence (thus, we talk with models).
- A restaurant menu is a model of the food the restaurant prepares and sells (thus, we eat with models).
- An accounting register is a model of a company's financial activity (thus, we work with models).
- A set of instructions for a game, such as Chess, are a model of that game (thus, we play with models).

And in order to really emphasize the point, here are several more examples (in no particular order) of common models that we either use deliberately or else depend on in some important way:

- A photograph is a model of the actual subject(s) depicted in the photograph.
- Your annual tax return is a model of your annual financial activity.
- An audio or video recording is a model of the actual sounds or images used to make the recording.

- A job description is a model of an employee's role and responsibilities in a company.
- A memory in your brain is a mental-model of some experience you lived.
- A piece of sheet music is a model of a given piece of music.
- A cooking recipe is a model of a process used to prepare a given dish.
- A library's catalog is a model of the library's inventory of documents.
- The DNA in your cells is a model of the process used to build your physical body.
- A template, such as a rubber stamp or a stencil, is a model of some pattern, form, block of text, etc.
- A business plan is a model of a business.
- Any given particular performance of a habit is a model for all of the other performances of that habit.
- A census is a model of a given population.
- A project manager's work-plan is a model of the tasks to be accomplished throughout a project.
- A representative sample is a model of the substance or population that provided the sample.
- An abstract symbol (e.g. a red cross, the word *pencil*) is a model of the actual thing, idea or institution represented by that symbol (e.g. the Red Cross organization, an actual pencil).
- When light reflects off of any physical object it is structured into a model of that object.
- An ethical rule, such as "be kind to strangers" or "always tell the truth" is a (mental) model of some ideal behavior.
- Many children's toys are models – cars, boats, airplanes, dolls, puppets, stuffed animals, houses, kitchen appliances, assembled puzzles, game pieces (e.g. Monopoly, Battle Ship, etc.). Also, many children's toys are used for making models – blocks, Legos, Lincoln Logs, erector sets, scale model kits, etc.
- A university chemistry textbook is a model of the basic chemistry knowledge to be learned by a chemistry student.
- A song written, for example, in the key of C major, is a model of the same song transposed, for example, to the key of F major.
- A written constitution is a model of an organization, such as a state, a club or an educational institution.
- A sculpture is a model of the artist's idea for that sculpture.

- A dead influenza virus, such as is used in a flu vaccine, is a model of a living influenza virus.
- An immune system antibody possesses a surface that is a model of an antigen's surface.
- A fossilized organism is a model of the original organism.
- A quantitative measure of some attribute of a thing, (e.g. its length, weight, density, etc.) is a model of that attribute.
- Your reputation – i.e. the ideas, evaluations, memories, etc. that others have of you in their heads – is a (mental) model of you. Your reputation can take on more durable forms as well, for example: your career résumé, your credit report, your academic transcript, or your profile in an online social network (Facebook.com, Classmates.com, etc.)
- A "friendly hacker" of the kind hired by organizations in order to test their computer network security systems is a model of a real hacker of the kind who tries to break into such systems in order to steal data.
- A history book is a model of historical events.
- A key is a model of a lock's keyhole.
- Honey bees use a kind of dance to model the location of a source of nectar[17].
- A legal contract is a model of the behavior of those bound by the contract.
- An understanding or an explanation of some thing or phenomenon (e.g. a mechanic's understanding of the way a combustion engine works or a physicist's explanation of lightening) is a (mental) model of the actual thing or phenomenon.
- A system of classification (e.g. the periodic table of the elements, or the spectrum of colors) is a model of the classified items.
- In a ball and socket joint, such as in the human shoulder, the ball is a model of the socket.
- A (school, company, home) fire-drill is a model of the events that ought to occur during an actual fire in order to ensure the safety of the participants in the drill during an actual fire.
- A U.S. one dollar bill is a model of one dollar's worth of purchasing power.
- A belief system is a (mental) model of the way the world works.
- A teacher's course syllabus is a model of the course he or she will teach.
- A scientific theory or mathematical theorem, such as Einstein's famous $E=mc^2$ or Darwin's theory of evolution by natural selection is a model of some aspect of the way the real world works.

[17] See James L. Gould's paper "The Dance Language Controversy", *The Quarterly Review of Biology*, 1976, Vol. 51, No. 2, pp. 211 -244.

I'll stop there, but I'd like to point out that this list could go on and on. The above list (a type of model) is only a sampling (another type of model) of the all of the models we humans use or on which we depend on a daily basis and it is only intended to give you a rough idea (e.g. a mental model) of the astonishing *ubiquity* of models throughout the human domain, which you can start to glimpse in this list, and which you can observe even more profoundly, now that you know what I'm talking about, by keeping your eyes open in your daily life. We are literally *surrounded* by models. They seem to be just about everywhere and we seem to use them deliberately (maps, menus, lists, etc.) or benefit from them passively (DNA, immune system antibodies, honey bee dances[18]) in nearly *everything* we do. They form a solid cornerstone of human civilization and a fundamental element of the human habitat, much as our air, water and food supplies. We are constantly making and using them. We start off playing with them as children (dolls, toy trucks, etc.) and then we grow up and use them in just about everything we do from grocery shopping to constructing office buildings. In analogy to the terms *biosphere* and *biophilia*, (if you can excuse my mixing of the Latin and Greek roots) we might say that we humans live within a *modelosphere*, and that we are *modelophilic*, i.e. in love with models[19].

Such an important and widespread phenomenon cries out for an explanation. What is the purpose of all this model-based behavior? The answer can easily be seen when you go back over the above examples and try to imagine what life would be like *without* these models. Imagine trying to shave or put on make-up in the morning without a mirror to help you. Imagine trying to navigate through a foreign city without a map of that city. Imagine trying to order a meal in a restaurant without a menu. And when you conduct these kinds of thought experiments, be careful not to cheat. It's no fair asking the waiter for his memorized menu! Without any sort of menu you'd have to just guess until you stumbled onto something the restaurant makes. Of course, if there were no architectural models, business plans, or recipes then you wouldn't even get that far because there would be no restaurants! The key word here is *complexity* and these models help us to manage it. As soon as behavior becomes much more complex than the basics open to *any* animal – sleeping, eating, wandering around, etc. – then we pretty much need some sort of model or representation in order to make it happen.

In the introduction I referred to a distinction that can be made between a good-regulator model and its "technical specification", or *control-model*. I gave the example of a recipe for roast-duck which is the control-model that a human being uses to become the dynamic good-regulator model for the system of ingredients and kitchen tools that produces the roast-duck. The same reasoning applies to most of the models

[18] Our food supply is heavily dependent on honey bees and other insects for the pollination of the plants we eat.

[19] It is not clear to me that there is much of a difference between what I am calling a "model" and what semioticians call a "sign". To the extent that these are the same, then what I am calling the Modelosphere is what semioticians call the *Semiosphere*, or "culture as a system of signs". See page 39 of Danesi, Marcel, *Messages, Signs, And Meanings: A Basic Textbook in Semiotics and Communication Theory*, 2004, Canadian Scholars' Press Inc, Toronto.

and representations we have just examined[20], which is to say that these can be seen as technical specifications that we humans use to transform ourselves into good-regulator models of the systems that we hope to regulate with their help. Furthermore, although the C&A theorem does not actually prove that these control-models are necessary to the regulation of these systems, the above sorts of thought experiments constitute a kind of inductive argument in favor of this position. Furthermore, given the astonishing complexity of the Human Modelosphere (briefly surveyed above) we can now reasonably wonder about the sort of high-level technical specification (control-model) that we will surely need in order to become (or otherwise build) good-regulators of *that* very complicated system. And although it is hard to imagine that those technical specifications could ever be squeezed into a single essay such as the one written by Conant and Ashby in 1970, it does seem compellingly obvious that these hypothetical technical specifications will simply have to contain, at the very least, the information their paper encapsulates. The upshot here is that with respect to our attempts to regulate this Modelosphere, the C&A theorem would appear to be very, very important.

Here we can return to the question raised in the introduction to this paper as to why the C&A theorem is not anywhere near as famous, say, as the Pythagorean Theorem. For that matter, why isn't this theorem being taught as part of the standard science curriculum? Why doesn't it show up on tee-shirts and bumper stickers and in movies about super heroes? After all, this system of control-models is already enormous and still growing and the C&A theorem explains much about what is required to regulate that system. One would think that these facts would have struck themselves together like flint and steel and ignited a conflagration of fascination for models and everything that has to do with them.

That last statement needs to be explained. Of course, in a sense, there is already such a widespread fascination with models. That is exactly what I mean when I say that we humans are modelophilic. We do love models. They are everywhere precisely because on some level we are fascinated by them. But what is missing is widespread high-level recognition and understanding of this fact. In this sense we are like birds who fly, but who have no clue about Newton's Laws or the Bernouilli Principle. It is the lack of this *higher level* fascination that strikes me as odd, if not a little precarious. The Conant & Ashby Theorem is the conceptual key that could unlock the door to such a higher level fascination, and yet, few people have ever even heard of it.

I won't pretend to have any sort of final solution to this problem, but I can offer what I believe are a couple of useful insights. For one thing, I believe that the theorem has suffered from poor public relations. This is partly because the version of the theorem that Conant and Ashby proved relies on some mathematics that is probably a little intimidating to the lay-person, to say the least. My hope is that the simpler version of the theorem, presented earlier, will go a long way to solving that problem. Furthermore, the core idea of the theorem is couched in the technical jargon of "systems" and "regulators". I think the theorem would be a lot more famous if it could be couched in more concrete terms that more people can relate to, hence the

[20]Excepting the likes of DNA, honeybee dances, etc. which we rely on but don't really use in the same way we use a grocery list.

"Every good key..." and "Every good solution..." metaphors that I have offered in this paper.

But public relations can only go so far. I suspect that the Pythagorean Theorem never needed a public relations campaign in order to achieve its fame and it is always expressed in the language of algebra and geometry that probably frightens most adults despite its simplicity. Of course the reason it has achieved its fame is because it has always been *useful*. It is my hope that the search algorithm I have presented here will ultimately demonstrate the usefulness of this theorem to the point where it earns the status of a kind of "Pythagorean Theorem" for, at least, the field of Operations Research and possibly other fields as well (Game Theory, Evolutionary Theory, a General Theory of Problem Solving, Semiotics, Cognitive Science, Genetics, Memetics, etc.) Once it has established its usefulness in this way, the next step will be for it to make its way into the standard science education curriculum, and after that, perhaps onto tee-shirts and bumper stickers.

Whether it can actually fulfill such lofty ambitions remains to be established, but I hope I have left you intrigued by the possibility of such an outcome.