Simple Man / Lynyrd Skynyrd (https://youtu.be/8eNoms9wsGc?si=cXU_vhk6J3zLdNKN)

Be A Simple Kind of Man

Micah Beck

Associate Professor at University of Tennessee, Knoxville

June 30, 2024

> “Freedom and happiness are found in the flexibility and ease with which we move through change.” - Buddha

The history of reasoning about and design of computer systems is rife with exhortations of simplicity and restraint. Various words have been used to informally characterize different notions of design restraint, including “minimal”, “general”, and even “elegant”. A variety of arguments have also been made for the value of such restraint. But there is a common theme: less is more.

I will make a case that maximizing logical weakness in assumptions correlates with each of these types of design restraint, and that it may play an important role in the benefits that are thought to flow from each. This does not mean that this one formal property accounts for all of these intuitive notions. Similarly, there are trade-offs with other factors (such as the

orthogonality of a design or efficiency of implementation) that can outweigh the value of logical weakness. Understanding logical weakness can help explain the power of our intuitive notions of simplicity, but I do not claim that it is the only or even always the dominant consideration.

Simplicity is one of the oldest and most commonly invoked forms of discipline. Exhortations to simplicity in reasoning date back at least to Aristotle ("We may assume the superiority, other things being equal, of the demonstration which derives from fewer postulates or hypotheses.") and Ptolomy ("We consider it a good principle to explain the phenomena by the simplest hypothesis possible"). The most widely cited version is Occam's Razor, attributed to the 14th century Franciscan friar WIlliam of Okcam ("More things should not be used than are necessary.")

Simplicity and minimality in achieving a specific purpose seem intuitively valuable, and a variety of interpretations and explanations have been given, including some efforts at formalism. However to date there has not been a broadly applicable and strongly reasoned argument for the value of these characteristics. This discussion offers one.

The notion of logical weakness derives from the idea that the statements of any argument can be expressed formally (i.e. in some form of logic). For the purposes of this discussion, it is not important to say exactly what the logic is. It is enough to assume that there is one. (Formal reasoning about the structure of logical systems which is independent of any one particular logic is focus of Model Theory). It is worth noting that the description of a complex system using formal logic can be highly complex. In fact, logicians have developed specialized logical frameworks (e.g. modal logics) to make reasoning about complex systems more natural. With mixed success.

Intuitively, specifications such as manual pages of an operating system standard like POSIX or protocol specifications such as those published by the Internet Engineering Task Force can be thought of as approximations to formal logical descriptions. In principle, one could write them out formally in full detail, but the result would be unreadable by most people.

Specifications As Logical Theories

A set of logical statements, such as the specification of a system, is called a theory. Any formal reasoning about a system has to start with a theory of that system. Starting with these, facts about the system can then be proved. Intuitively, any closely reasoned argument about a system is an approximation to a formal proof based on its specification. Hence, the common response when faced with a question that can be easily derived from the spec: RTFM!

The specification of a system's behavior can be viewed as a theory. If the system is a programming language, it's theory describes its syntax and semantics. If the system is a network, it's theory describes the form & meaning of its protocols. If the system is an OS, its theory describes the type signatures of its system calls and the actions that they invoke. Each of these types of systems has a reference manual which specifies their form and function. Sometimes a partial formalism is used, such as grammars for syntax or type signatures for data structures and procedure calls. But in many cases behaviors are described less formally.

The specification of any system could in principle be completely formalized using some logic. For a real system the theory would be very complex and would probably be described using a modal logic. Regardless of how complex the logic or the theory was, this structural property is the same: a weaker theory has more models.

If we are considering two logical theories S and T, we say that T is logically weaker than S if every statement that is a logical consequence of S is also a logical consequence of T.

Intuitively, a logically weaker theory is one which makes fewer guarantees. This is not the same as specifying a lack of functionality, poor performance or unpredictability. It is simply the minimization of specific requirements. A canonical example of a service with a logically weak specification is the Internet's best effort packet delivery.

My claim is that an argument that has a weaker set of starting assumptions is often preferable to one that has a stronger set of starting assumptions. The reason for this is quite intuitive: the weaker set of assumptions holds true whenever the stronger set does. So if it turns out that the stronger

theory does not match the real world, the weaker set may still be valid. The inverse relationship between strength of a theory and the collection of possible worlds (called “models” that satisfy that theory is illustrated in Figure 1.

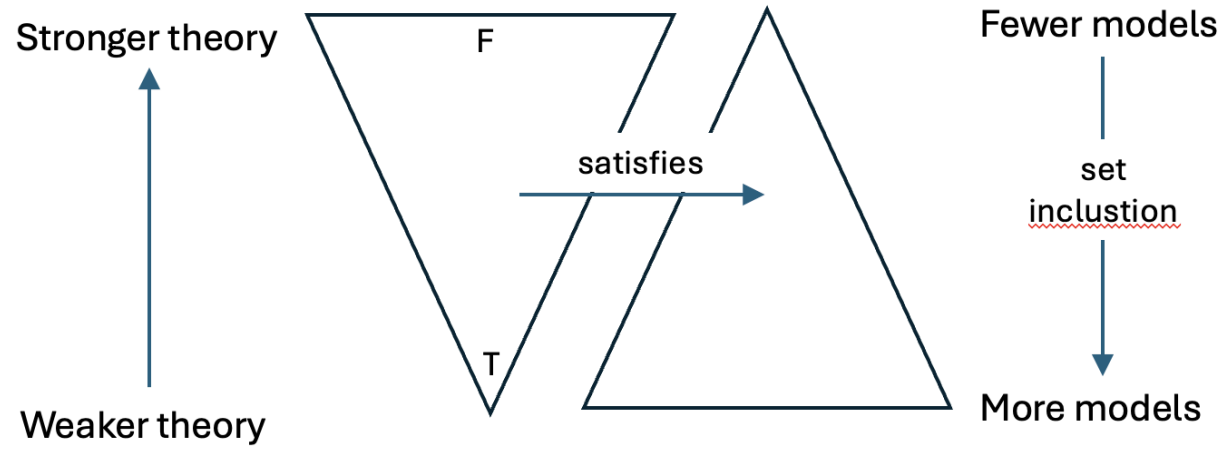

Figure 1: A stronger theory is satisfied by fewer models (possibe worlds).

Unfortunately, logical weakness is not a total or complete ordering: some theories are equivalent and many theories are not directly comparable on the basis of logical strength. This is a reason that the general application of this logical weakness as a design tool requires experience.

Logical Weakness Is Correlated With Minimality and Simplicity

If a theory T is a subset of another theory S, then T is logically weaker than S.

Thus, removing an assumption can only weaken a theory. For this reason, an argument which begins with few assumptions may also tend to have a weaker set of assumptions. However a single stong assumption can be stronger than a much larger set of very weak ones. Weakness is correlated to set inclusion of theories, not by the absolute number of statements.

Thus a theory may be weakened by removing unnecessary statements, making it simpler. The correlation between logical weakness and simplicity is harder to pin down because there is no widely accepted way to compare the “simplicity” of two theories if neither is a subset of the other.

The Wide Applicability of Logical Weakness

In my own work, I have applied logical weakness to the design of a “spanning layer”, which is the “thin waist” in the design of a layered system. I proved the Hourglass Theorem, which showed the way to achieve a common service layer that has the largest number of possible implementations. It is by choosing the logically weakest option which can still be used to achieve a given set of necessary goals. This result gives a specific meaning to the imprecise notion of “thinness” in the waist of layered systems that adopt the hourglass design methodology.

One can use logical weakness to understand the robustness or fragility of many common designs. Important systems have sometimes been criticized for design restraint that correlates with logical weakness:

- End-to-End Arguments advocate for a common communication service that is simple and generic. The Internet has been called a “dumb” network by advocates of circuit switching.
- The Unix kernel API was designed to be minimal for its intended purpose. To disparage it some said that “Unix is as much of Multics as [its author] could understand”.
- The C language type system is much weaker than strongly typed languages. This was one reason for C having been characterized as “assembly language with syntactic sugar” by advocates of higher level languages.
- The original Java Virtual Machine was a minimalist (logically weak) execution mechanism, leading to limited relevance in technical applications. The myriad native methods that were added eventually resulted in a system that was much stronger logically, at the cost of stability, security and adoption (although Java found its niche as a compiled language).

These are just a few examples of logically weaker designs that have proved highly durable and logically stronger ones facing challenges in deployment scalability. Logical weakness is a tool for understanding the impact of any design and of helping to predict the success of new ones. The challenge is to embrace the principle of logical weakness without requiring full formalism or measurable comparisons of designs. I believe that some of the greatest contributors to the current ICT environment, many of whom are ACM Turing

winners, had an intuition for logical weakness in system designs that have deployment scalability (viral adoption and success).