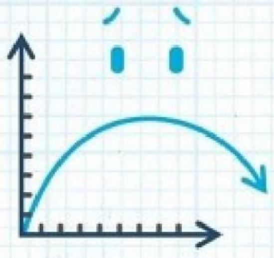
BAD CHOICES



HOW ALGORITHMS CAN HELP YOU THINK SMARTER AND LIVE HAPPIER

ALI ALMOSSAWI

AUTHOR OF BAD ARGUMENTS

VIKING
An imprint of Penguin Random House LLC
375 Hudson Street
New York, New York 10014
penguin.com

Creative and art direction by Ali Almossawi

Illustrations by Alejandro Giraldo

Library of Congress Cataloging-in-Publication Data
Names: Almossawi, Ali.
Title: Bad choices : how algorithms can help you think smarter and live happier / Ali Almossawi.
Description: New York : Viking, 2017. | Includes index.
Identifiers: LCCN 2016056684 (print) | LCCN 2016058049 (ebook) | ISBN 9780735222120 (hardcover) | ISBN 9780735222236 (ebook)
Subjects: LCSH: Decision making–Mathematical models. | Sequential analysis. Probabilities. | Algorithms.
Classification: LCC QA279.4 .A6275 2017 (print) | LCC QA279.4 (ebook) | DDC 518/.1–dc23
LC record available at https://lccn.loc.gov/2016056684

Version_1

YOU CAN SWIM ALL DAY IN THE SEA OF KNOWLEDGE AND NOT GET WET.

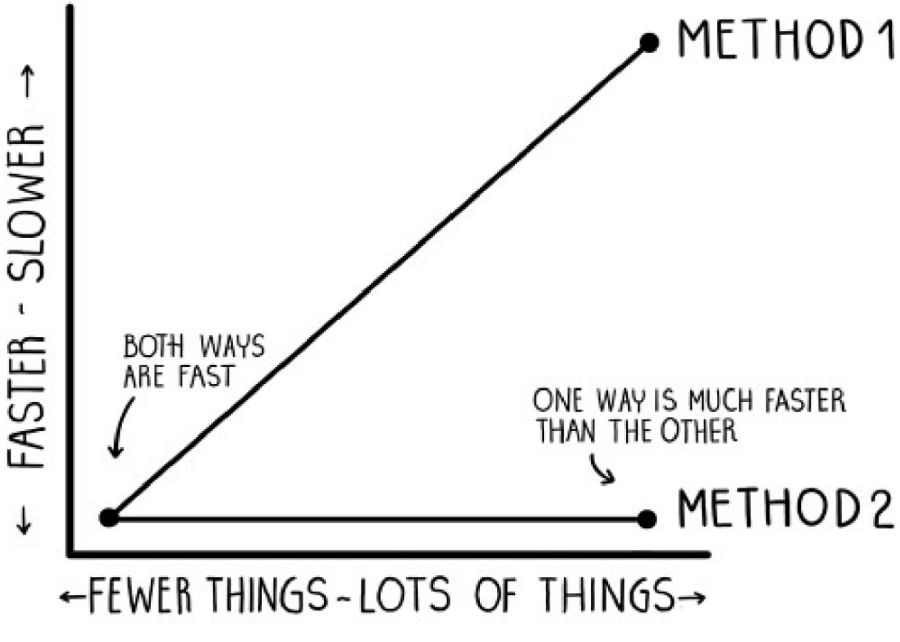
—Norton Juster, *The Phantom Tollbooth*

PREFACE

Did you know that Richard Feynman started developing the equations that won him the Nobel Prize after seeing someone throw a plate in the air? Or that John von Neumann modeled parts of his electronic computer on a friend's idea about how memories are stored in the human brain? Or that the sight of a kicking and screaming orangutan at the zoo led Charles Darwin to his big idea? What Feynman, von Neumann, Darwin, and others have in common is that they see physics and mathematics and science everywhere, way beyond the confines of their laboratories.

Even if you're not gunning for a Nobel Prize, you probably do things in your everyday life that can be modeled as algorithms. In fact, you apply them on a daily basis to solve various problems: finding pairs of socks in a pile of clothes, deciding when to go to the grocery store, determining how to prioritize your tasks for the day, and so on. An algorithm is a series of unambiguous steps that achieves some meaningful objective in finite time. The series might begin with some input and is expected to produce an output. Those are an algorithm's characteristics. What's fascinating is that Babylonian tablets from around 1800 to 1600 BCE reveal that ancient Babylonians wrote down their procedures for determining things like, say, compound interest or the width and length of a cistern given its height and volume using algorithms. That is to say, their procedures were made up of an unambiguous series of steps; they had some input, some output, they eventually terminated and they were useful. Algorithms can thus be found in the works of various contributors to mathematics over the centuries. After the advent of computers, these characteristics have proved crucial because they allow computers to carry out tasks in a way that is predictable.

Despite the importance of algorithms in our lives, texts on the subject tend to focus largely on intricate details—the “how”—while perhaps ignoring the more practical lessons of those algorithms that make them appealing. The seemingly simple everyday tasks we just mentioned can be undertaken in a number of different ways. The more aware we are of those ways, the better we can hone our ability to achieve a task in the most efficient way. Think of it like enhancing a general-purpose intuition that we all possess. That’s where *Bad Choices* comes in. This book aims to acquaint you with algorithmic thinking by highlighting different ways of approaching everyday tasks and pointing out how these approaches fare *relative* to each other. For instance, two methods of looking for a shirt in your size on a rack of shirts might be described like the graph below.*



Those shapes of lines have names like *linear* and *logarithmic*, which we will flesh out and discuss throughout the book. And while both approaches are comparable in terms of performance when we have a few things, notice how that changes as the number of things increases. This book includes twelve familiar scenes, such as a living room, a tailor shop, and a department store. In each scene, there are a number of potential tasks to be done. After each illustration, a paragraph describes the scene, and a few pages of commentary and discussion relate the scene to concrete concepts from computer science and highlight at least two possible ways of undertaking the fundamental task at hand. One that's slower and one that's faster. That difference is what the book's title aims to emphasize, albeit somewhat provocatively. The title is partly inspired by computer scientist Donald Knuth's talk of "good" algorithms, which is to say fast or effective ones.[*]

INTRODUCTION

WHY FOCUS ON RELATIVE MAGNITUDES?

Comparisons are amazingly powerful. One of the first things children learn are abstractions like *big* and *small*, which is why when a child asks, “How tall is that titanosaur that they now have at the Natural History Museum?” one finds that it is less meaningful for the child to hear the response, “Seventeen feet tall, little one,” and more meaningful to hear, “If Ms. Susan, Ms. Margaret, and Mr. Jascha were to stand on each other’s shoulders, Mr. Jascha would probably be able to tickle the titanosaur’s chin.”

Thinking in terms of relative magnitudes may in fact be an ability that we are all born with. Recent experiments seem to suggest that babies show as much brain activity in response to a change in an image they are seeing as they do in response to noticeable changes in the *number* of images they are seeing. Other experiments in remoter parts of the world suggest that people who have not been subjected to what we might call formal education reason about numbers in terms of orders of magnitude. It’s an intuition that we appear to have innately.

One subspecies of humans whose appreciation for this intuition is manifest is computer scientists. It’s what gives them the ability to recognize, fairly quickly, which of several competing approaches to solving a problem might be better. That fact is a reminder that seeing things in terms of relative magnitudes is an ability that remains useful even after you develop mastery of a field. Think of it like the mathematical notation you learned in primary school, which you continued to use throughout school and college and beyond.

This idea is a large part of the motivation behind writing this book. I had long used comparisons, estimates, and approximations to understand various concepts during my school and college years, but I dared not admit that to anyone because it felt like a less sophisticated way of learning. It wasn't until I read books like *The Strangest Man** and *The Society of Mind* that I realized I wasn't the only one who found that way of thinking useful. Much later, I read *The Art of Insight in Science and Engineering* and similar books, which talk of the same idea and its implications for insight.

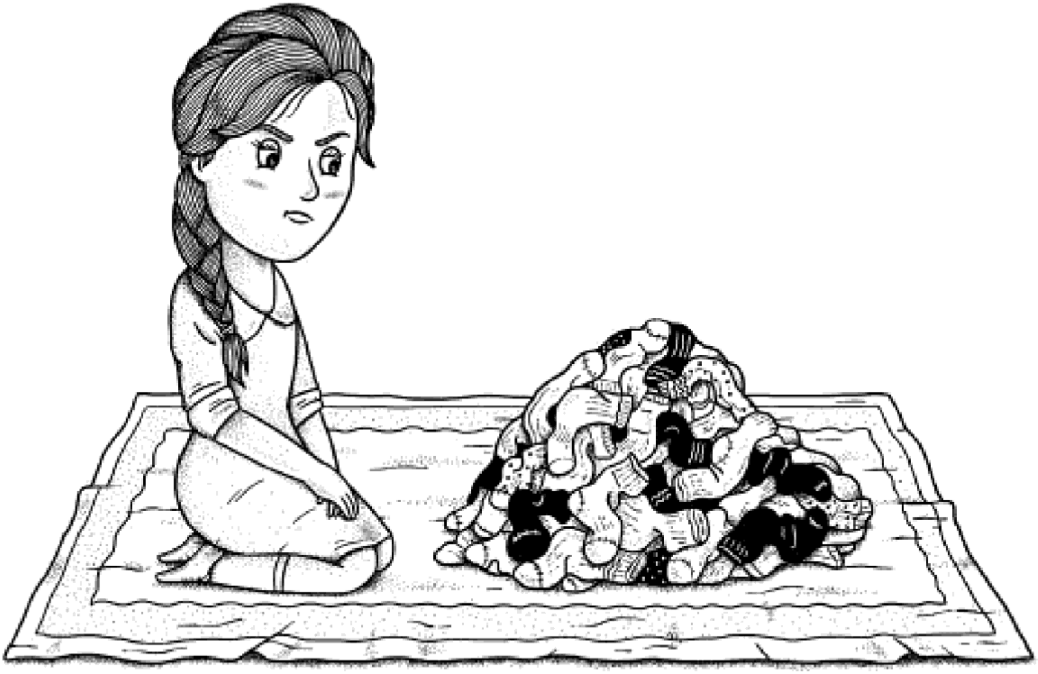
It is my hope that this book impresses on you the ability to better think about decisions throughout your life and better understand what trade-offs they come with. The book doesn't aim to teach you how to better match socks, an intuition that most people will likely already have, but rather to persuade you to turn the mirror on yourself and ask, "I didn't realize I could think about *my* socks in that way." Much like critical thinking, algorithmic thinking is a highly capable tool that has the potential to impact behavior for the better.

WHY FOCUS ON EVERYDAY TASKS?

Algorithms can be complex, but they're also critical and often already a part of our lives. We just don't know it or think about it much. By highlighting those parts of our lives that serve as good models for various algorithms, we end up with an approach that has several benefits.

It is relatable: Many of the explanations in this book leverage illustrations. Explaining by way of illustrations is not only useful because of the appeal that illustrations bring to an otherwise pedestrian work, but also because illustrations can place one in a world that is relatable, and relatable worlds are engaging and encourage you to be more sophisticated in your reasoning as you connect newly acquired knowledge with what you already know. That is precisely why analogies are so effective.

It is interactive: If you look at human history, you're likely to find that many of the names you recognize belong to people who were



1.

MATCH THOSE SOCKS

Baroness Margie Wana is a member of a formerly influential Viennese family who was recently indicted for smuggling Kinder Surprise eggs into the United States. She now works as an au pair in Bern and is folding clothes for the first time. Margie is shocked to discover that each member of her host family sweats through a pair of socks every half hour, making finding and matching pairs more time-consuming than she ever imagined. On the plus side, they have different shoe sizes and like different colors.

Hint: While there might be several tasks here, perhaps start with the fundamental one.

• • •

Have you ever thought about how essential a biological feature *memory* is in humans? The image of someone leaning back into her chair, putting one hand to her forehead, and pressing her eyes shut as she calls to mind a verse or an equation or a telephone number—that's the quintessential human. Imagine the struggle of having to go through life without that feature, as do sufferers of dementia. For starters, you would end up having to repeat a lot of the same work. Like in *Memento,* where every morning the protagonist has to fill up his mind all over again with all the bits of information that he needs to carry out his primary task.

I mention this at the outset because of the fact that faster methods of solving problems are often faster because they happen to leverage memory.[*] Consider AlphaGo, which last year beat a champion at the game of Go thanks to its ability to learn not only from expert humans but also from itself, thus amassing a greater memory from which to work.[*] Put differently, many of the faster ways of solving problems that we will encounter in this book, simple though they are, are fast because of their ability to avoid doing the same action on the same thing multiple times.

Let's not get ahead of ourselves though. Back to the socks, and to poor old Margie Wana, the irony of whose name was lost on the federal agents who recently seized her stash of chocolate goodness. Margie is facing the daunting task of matching pairs of socks from a humongous pile of clothes. Let us focus on one of several tasks that exist here and consider two possible methods for taking on that task:

OBJECTIVE: MATCH THE PAIRS OF SOCKS IN THIS PILE OF CLOTHES.

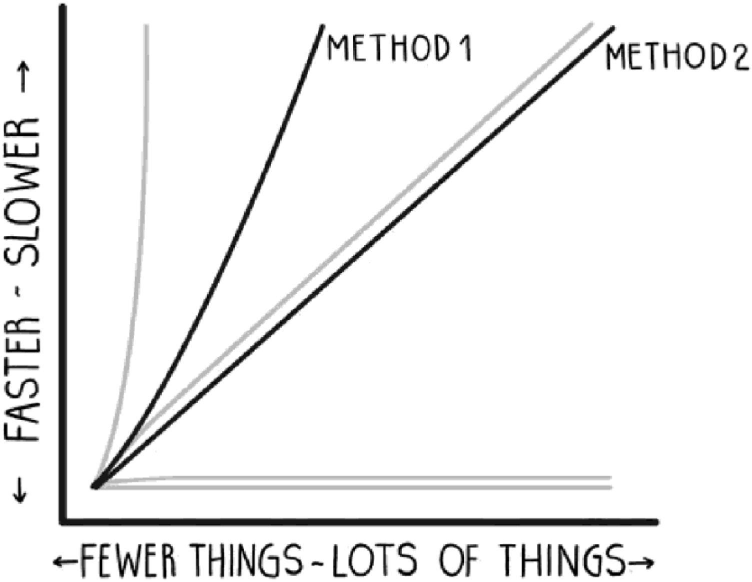
METHOD 1: PICK A SOCK, LOOK FOR ITS MATCH IN THE PILE, PUT IT TO ONE SIDE. THEN PICK ANOTHER SOCK, LOOK FOR ITS MATCH IN THE PILE, AND PUT IT TO ONE SIDE. AND SO ON.

METHOD 2: PICK A SOCK, PUT IT TO ONE SIDE. PICK ANOTHER SOCK. IF IT MATCHES ANY OF THE ONES SHE HAS PUT TO ONE SIDE, MATCH IT. OTHERWISE, ADD IT TO THE LINE OF UNMATCHED SOCKS, LUMPING IT WITH SOCKS THAT HAVE THE SAME COLOR OR SIZE.[*]

Before reading any further, I would suggest working through these scenes using pen and paper, props, or whatever else you feel comfortable with. Think about what achieving the objective entails in terms of individual steps and assumptions. Try to do that for all the scenes that follow.

With a pile of, say, four pairs of socks, it doesn't really matter which method Margie uses—she will be done fairly quickly. Now imagine Margie with hundreds of socks in front of her. If she opts for the first method, the chance that she will come across the same old sock over and over again is quite high, since she never takes it out of the pile. When she first comes across it, she simply doesn't glean any information from it. With the second method, however, she keeps a line of unmatched socks to one side, thus ensuring that she only ever comes across a sock in the pile once. The second method, therefore, ends up being faster because of its reliance on memory. More precisely, because of what's sometimes called a *lookup table* or *cache*. Though it need not be, it is useful to think of a lookup table as a collection of unique identifiers (keys) each pointing to some associated item of data (values) where you, quite literally, look up the values of keys. We call this type of representation a *key-value pair*. In this case, our keys would likely be "color." When Margie comes across, say, a red sock, she looks up "red" in her line of unmatched socks. If she finds a "red" area, she might then look for additional identifiers like, say, style or hue and take things from there. Otherwise, she would create a new "red" area with the solitary red sock.

Here are how the two methods compare.* Notice that Method 1 becomes noticeably slower than Method 2 as the number of socks in the pile grows. There are many more ways of tackling the tasks in this book than the two methods that are highlighted. The discussion is meant to emphasize two methods that are notably different in their asymptotic rates of growth, leaving out other methods whose performance may fall somewhere in between. With this scene, for instance, Margie might have chosen, alternatively, to find matches by way of the pigeonhole principle, which would have involved pulling six socks from the pile at a time and matching pairs that way.



Now, when we pick out a sock from the pile, we could probably tell fairly quickly if we've come across its matching pair. Most people's short-term memory is quite good at remembering half a dozen or so groups of things, which is what we have here. And so, coming across a sock in the pile that we've already placed to one side ought to elicit a near-immediate "Ah, I've seen that one before!" If you've ever played a card-matching game like *Memory,* the powers and limitations of that faculty ought to be familiar to you.

If we did have a bigger disparity in sock types and colors, however, our line of unmatched socks might grow quite a bit, forcing us to scan through the whole line every time we pick something new out of the pile. Scanning through a line of things, an *array,* can be time-consuming when the number of things is quite large, reason being that the thing you're looking for might be at the very end of that line. We would thus have to pass through the entirety of the array first.

ISN'T IT INTERESTING, HOW THE MUNDANE CAN TURN INTO SOMETHING SO ENGAGING WITH A SLIGHT SHIFT OF THE HEAD.

2.

FIND YOUR SIZE

It's the day after Christmas, and Eppy Toam, a nurse from Inverness, has started camping outside of a department store in preparation for this year's Boxing Day sale. Her shirt size is quite common and she wants to make sure she is the first one into the store so that she can grab all the shirts that are in her size. She is going to have to be fast. Things get pretty ugly. Last year, fifteen people were injured and the army had to intervene. How might Eppy increase her odds of grabbing the shirts she's after before everybody else?

Hint: Consider pushing the example to an absurd limit. What if the racks were as wide as the store?

• • •

If we are searching for an item in a collection of items, then surely, it must be the case that we face the possibility of having to look through all of those items before finding what we're looking for? In other words, mustn't it be the case that if we have one hundred items, we face the possibility of having to look through one hundred items, which is to say, taking *linear time*? Generally speaking, a linear function means that if it takes us a minute to find something among one hundred things, then we can expect to spend two minutes to find something in a pile of two hundred things. Ordinarily, yes. But there is a particular quality that a collection can have—namely, the quality of being sorted—that allows us to find an item in *logarithmic* time. In other words, in seven or so steps rather than 100. Recall that a logarithm is simply the inverse of an exponent. When writing computer programs, we assume the base of a logarithm to be two, and so the logarithm of 100 is $log_2 100$, which comes to around seven. That massive improvement you see when going from linear time to logarithmic time is why the logarithm is such an important concept, particularly when we talk about rates of growth. It is a concept that we will revisit often throughout the coming chapters.

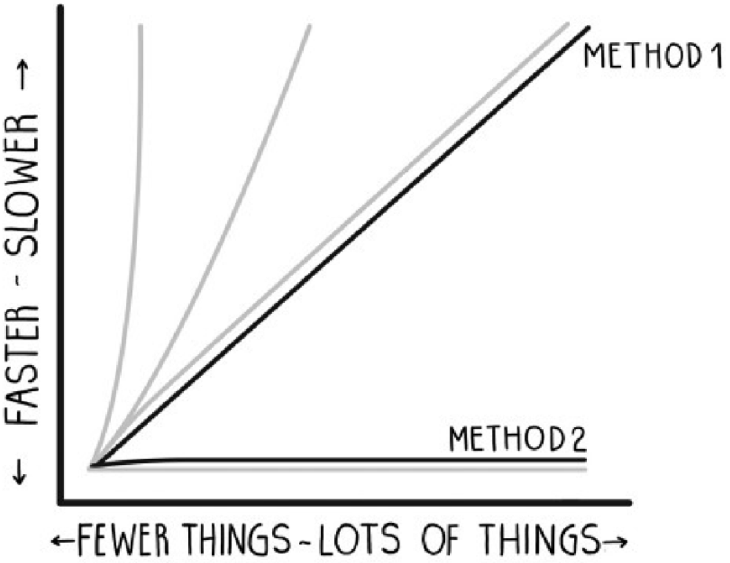
Let us first describe how Eppy might go about descending on the store, her face likely painted in the colors of glory, her tartan shawl billowing behind, her battle cries shooting past her rattling teeth and clinging on to the store's walls, for posterity to marvel at. She has been psyching herself up all morning.

OBJECTIVE: FOR A GIVEN RACK, FIND THE SHIRT IN HER SIZE.

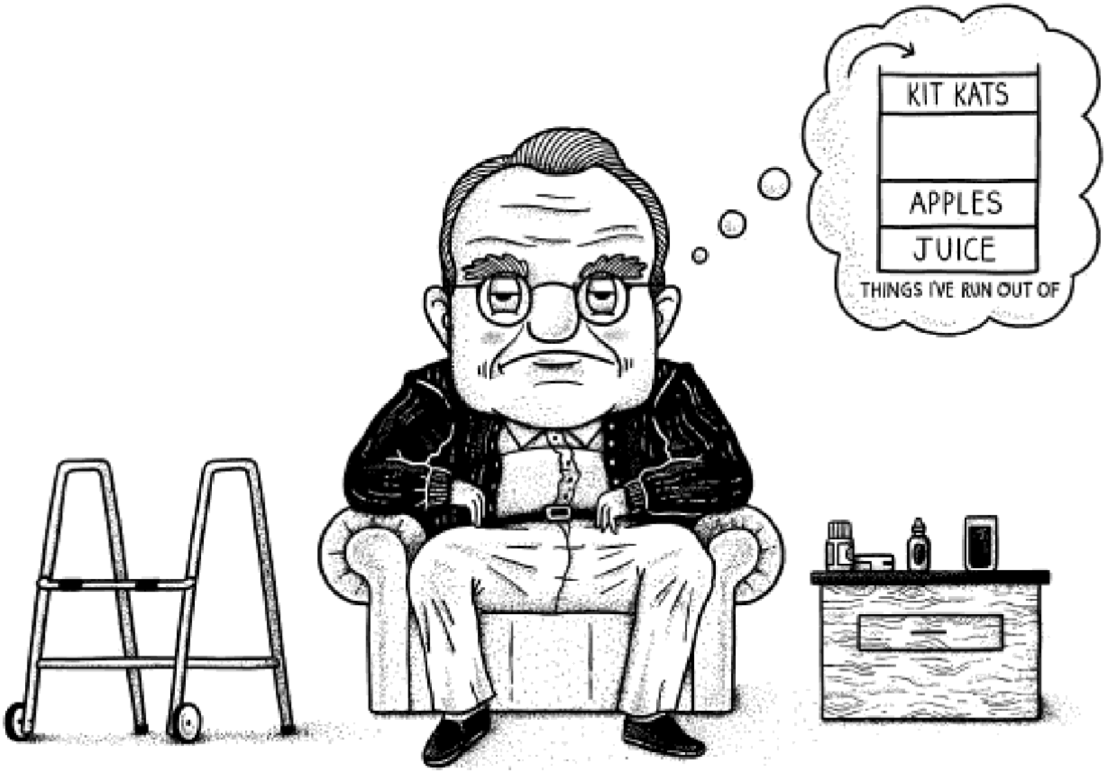
METHOD 1: FOR A GIVEN RACK, LOOK THROUGH THE SHIRTS FROM ONE END TO THE OTHER.

METHOD 2: FOR A GIVEN RACK, START LOOKING FOR THE CORRECT SHIRT SIZE SOMEWHERE NEAR THE MIDDLE. IF THE SHIRT IN THE MIDDLE IS LARGER THAN HER SIZE, MOVE TO THE LEFT. IF IT'S SMALLER THAN HER SIZE, MOVE TO THE RIGHT. AND SO ON.

Here are how the two methods compare. Notice that Method 1 becomes noticeably slower than Method 2 as the number of shirts on a rack grows.



As you might have guessed, Method 2 leverages two pieces of knowledge. One, that the shirts are most likely sorted on the rack by size. And two, that because Eppy’s size is a common size, which is to say, an average size, then it would likely be near the middle of the rack. Using that intuition not only to start from the middle, but to subsequently move left or right in jumps, thus halving the collection each time, is a signature of a *logarithmic-time* algorithm.* It is the same intuition we might use to look for a word in a dictionary or a name in a telephone directory or a topic in a book’s index. The same intuition we would use if we fall asleep reading a tedious novel and want to pick up where we left off the next day. More generally, we might characterize this approach as being one of discarding information.


KIT KATS
APPLES
JUICE
THINGS I'VE RUN OUT OF

3.

POP TO THE SHOPS

Iain Patoys is a retired linguistics teacher who lives in East London. He has a bad back from a fall a few years ago. He hates leaving the house because the neighbor's dog scares him, but alas, if he doesn't want to die of hunger, he has to pop to the shops every so often to buy his own groceries. This being London, it rains a lot, and Iain being Iain, doesn't like getting wet. How might Iain minimize the number of times he goes to the store within a given week without dying of hunger?

• • •

There is a timeless sketch that the British double act The Two Ronnies did during their many years on air.* In it, a customer walks into a hardware store and reads off the items on his list, one by one. Rather than waiting for his customer to read the entire list first, the store owner instead retrieves an item each time the customer reads it out, which ends up causing the store owner great distress.

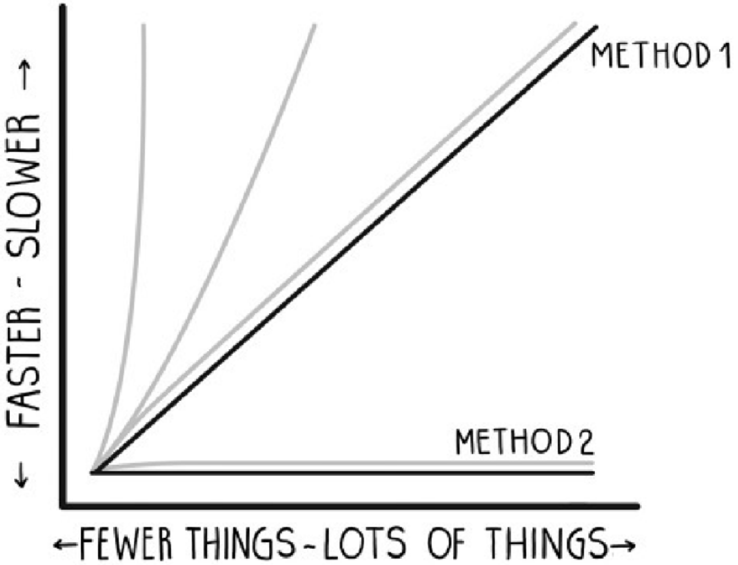
Keep this vignette in mind. We'll come back to it in a bit. But first, let's see how Iain might go about deciding how often to go to the store.

OBJECTIVE: MAKE AS FEW TRIPS TO THE GROCERY STORE IN A GIVEN WEEK AS POSSIBLE.

METHOD 1: REALIZE HE HAS RUN OUT OF SOMETHING. HEAD TO THE STORE TO BUY IT.

METHOD 2: MAINTAIN A LIST OF THINGS THAT HE HAS RUN OUT OF. GO TO THE STORE ONCE THE LIST REACHES A CERTAIN SIZE OR ONCE HE RUNS OUT OF AN ESSENTIAL LIFE-AFFIRMING FOOD ITEM, LIKE KIT KAT BARS.*

Here is a graph we are already familiar with that gives us a visual sense of how those two methods compare.



One interpretation of this scene would be to say that it’s essentially about avoiding repetitive work, much like why a secretary tasked with