Trust Systems

TRUST SYSTEMS

A Journey Through Trust, its Applications, Calculations and Limitations

STEPHEN MARSH

Open Education Lab, Ontario Tech University
Oshawa

CONTENTS

Contents

DEDICATION

mom & dad

patricia

dawn, owain, addison, anika, layne

without whom, darkness

ACKNOWLEDGEMENTS AND ATTRIBUTION

OER Equal Love

Contributors

Author: Dr. Stephen Marsh

Editors: Pranjal Saloni, Shreya Patel, Noopa Kuriakose, Abida Choudhury, Hasan Ahmad, Divya Sharma, Fazal Rahman

Reviewers: Joanna Bakler

Project Managers: Rebecca Maynard, Sarah Stokes

About This Work

This project is made possible with funding by the Government of Ontario and through eCampusOntario's support of the Virtual Learning Strategy. To learn more about the Virtual Learning Strategy visit: https://vls.ecampusontario.ca.

About the OE Lab

Ontario Tech University is proud to host the OE Lab – a student-run, staff-managed group that brings content and technological expertise to the timely **creation of high quality OER that will be used directly in an Ontario Tech course by Ontario Tech students**.

If you adopt this book, you will be using a work created by students as an experiential learning and employment opportunity. Please let us know if you are using this work by emailing oer@ontariotechu.ca.

How to Attribute This Work

Suggested Attribution for This Work: ***Trust Systems by Stephen Marsh (OE Lab Edition) 2021, licensed under a CC BY SA International License.***

ACCESSIBILITY STATEMENT

The Open Education (OE) Lab is committed to providing course resources that are free, open, and available to anyone who wishes to use them. As such, we strive to make all our resources accessible and easy to use.

Accessibility Features of This Resource

The web version of **Trust Systems** has been designed with accessibility in mind by incorporating the following features:

- It has been optimized for people who use screen-reader technology.
    - all content can be navigated using a keyboard
    - links, headings, and tables are formatted to work with screen readers
    - images have alt text and separate long descriptions if needed
- Information is not conveyed by colour alone.
- The resource is available to download in multiple formats.

Conformance Status

The Web Content Accessibility Guidelines (WCAG) defines requirements for designers and developers to improve accessibility for people with disabilities. It defines three levels of conformance: Level A, Level AA, and Level AAA. **Trust Systems** is fully conformant with WCAG 2.1 level AA. Fully conformant means that the content fully conforms to the accessibility standard without any exceptions.

Feedback

We welcome your feedback on the accessibility of this resource. If you encounter accessibility barriers, please feel free to contact us by email at oer@ontariotechu.ca

THIS BOOK IS A CONVERSATION

Hello.

(see what I did there?)

This book is a conversation starter. it was written in order to try a different way to get academic knowledge and thought into a wider sphere, and to help other people to be able to actually take part in the process of writing and creating knowledge.

How can this work? What do I mean by a conversation? More prosaically, why would I even want to do this? Let me answer this, as Anne Lamott's father would have it, *Bird by Bird*. Bear with me, there's a lot going on here and it is all quite important, possibly even more important than the book itself.

I call it a conversation starter because that is how I see it. Here's how it goes: when people get together there is usually some form of a conversation, about shared interests, books, politics, the weather, you name it. Often they engage in what we call 'small talk[1]. Generally, someone starts by saying something which other people (or the other person) get to respond to somehow. We use words, gestures, thoughts, our faces and our bodies and if we are really lucky we get to be involved in a conversation that actually means something important to us[2]. This could be something we disagree with, or a problem, someone we wish we could understand and honestly seek more information from, things like that. This book is one of those. You know, a conversation starter.

It's written as a collection of chapters, each of which might refer to others, but each of which can also be read on its own. It is the sum of thirty-plus years of thought and practice in the fields of trust and computers. It is also completely unlike pretty much any other science-type book. Oh, sure, you can read it like a science book–it has references and everything, and I sometimes know what I am talking about–but you can also read it as someone who has an interest in how some things work. Like people and security and risk and more. Given that I've been thinking about this stuff for a long time, there's some good stuff in here, even if I say so myself, but there's also a lot of myself in that it is also a personal reflection on what I think matters in the field. Two for the price of one, as it were.

But there is one thing that I want you to bear in mind before you start thinking all those clever thoughts. Have you ever been involved in a conversation and, whilst the other person is talking, found yourself thinking about what you are going to say next because it is going to change the world, or at least they way they think?

[1] Why we call it small talk has always been a source of mystery to me, given the massive importance it seems to have to the way people actually exist together.

[2] As it happens, this is one reason I got into academia... The number of fascinating, totally immersed people in this place is incredible.

Yes, you.

Don't be that person. Stop and actually, actively listen to what is being said, then take the time you need to figure out what it is you want to say back[3]. This takes active listening and it's worth it. If you've never done it before, practice. So, I consider this to be that start of a conversation with *you*. Imagine someone standing in front of you for thirty-odd years before actually getting up the nerve to say something to you[4]

Now, where was I? Oh yes! When you are reading this, whichever bits you read[5], take the time to *really* read it, however much of it you want, before you think about a response. You may agree or disagree with what I say. I may make you angry enough to want to throw the book out of the window[6]. It's all good. And then, you get to respond. The book has been written on a platform that enables social commentary using the social annotation engine that the clever folks at Hypothesis.is made.

Think of it sort of like a book you can answer back to.

With this system you can highlight, make comments and annotations, ask questions and so on, all either privately or publicly (your choice). If you send them to me privately I will endeavour to respond likewise, by the way. Public comments are there for others to see and respond to. Including me, which is fun. I will be delighted to respond, in fact.

Comments and responses will become part of the book. It's that simple. Tell me stuff and, assuming it makes sense, it'll stay there forever. Okay, that also is creepy, but it may become part of the fabric of the book. What does 'makes sense' mean? Mostly, relevant to what it is commenting on – the world doesn't need more stupid comments, but it could use all the sensible thought it can get, so the comment might be a personal anecdote, a thought, a correction, a reference, a question, you name it. Just don't be that person. You know who they are.
I promise a couple of things. The first, simply, is that I won't delete sensible comments or questions or their responses, and that I will respond as often as I am able.

The second is a bit more concrete. If you read a book, you are usually reading a version that has been edited and published and so on, and perhaps undergone revision, which basically means "had things chopped out" so that you don't get to see what has changed, how knowledge and truth has grown and changed, things like that. This book won't be doing that.

What is written is written and it won't change[7]. But my promise is this: I won't delete stuff post the first proper publication (I'm aiming for October 21 2021! It's my birthday, you see).

[3] If the person you are conversing with is genuine, they'll wait.

[4] Actually now that I have written that, it sounds creepy. Don't imagine this if it is creepy to you.

[5] I sincerely hope you want to read it all – there's a lot of me in here.

[6] Given that this was written to be read on a screen, do not do this thing. If you are reading it on paper, don't do it either: you can always make hats with it.

[7] I can't resist: "The Moving Finger writes; and, having writ, Moves on: nor all thy Piety nor Wit Shall lure it back to cancel half a Line, Nor all thy Tears wash out a Word of it." Omar Khayyám.

Not *ever*.

Things might change though, and instead of deleting I will just strike them out, because that way you get to see what went first, second, third and whatever and how my thoughts changed, either because so did I, or my knowledge, or because someone changed my thoughts, perhaps by commenting on it.

That's it. No deletions, just obvious change-making. I hope that this will help you see what I mean by a conversation. And that you can stand every chance of discussing with me and others what the book brings to mind.

Oh, there's one more thing, and I mention it in the next few sections too: my writing style is definitely idiosyncratic and not for everyone. If you don't like it, you are free not to read it. Feel free to write your own book. But don't comment on how I say things because I promise you, those comments won't survive.

If you are reading this book on paper, please read on and consider how you might want to join the conversation, I would love you to be part of it.

Meanwhile, I've said my bits, I've started this conversation and I'm genuinely excited about what you are thinking about responding with.

Joy Oladokun |
In Defence of My Own Happiness

THE CONTENTS AND THE AUTHOR

The Book

Trust Systems presents a social-annotation-enhanced personal view of trust and its workings, more specifically computational trust, attained through more than three decades of research and practice in the field. It is aimed primarily at senior undergraduate and graduate students looking for an accessible introduction to the concept that points toward more in-depth works. It is also written for general interest science in the hope that they find it interesting. It was written as an electronic book but can be happily printed and read on good old-fashioned paper.

Whilst much of the focus is on trust as a computational concept, it is applicable to readers in the social science who are interested in trust, since it also discusses the phenomenon with reference to sociological and social-psychological works.

The book explores trust and trust systems from first principles through to examining their role in many of the tools and technologies we encounter today. It also explores some of the future and what that means for us as humans, whilst explicitly acknowledging that humans are emphatically not the only beings that can, and do, think in terms of trust.

The Writer

My name is Steve (the good looking one in the picture is Sally, who is in training to be my next Service Dog).

Steve and Sally

I've been working in the field of Computational Trust for more than thirty years. I actually founded it, which is a bit of luck because I guess that means I can also write about it like this. I was born in Dudley, England, was educated in Gornal, Malvern (all Primary schools), Upton upon Severn (Junior High), Hanley Castle (High School), Stirling University (BSc in Computing Science and PhD in the same, but basically Computational Trust). I taught for a while in Stirling University before being offered a Postdoctoral Fellowship with the National Research Council (NRC) of Canada in 1996, and moving over to Canada as a result. This turned into being a Research Associate and eventually Research Officer at the NRC, where I worked on ACORN, an agent-based information system that delivered information to people based on interests and recommendations and the idea of Socially Adept Technology. I also helped start up the PST (Privacy Security and Trust) conference as well ask the IFIPTM Conference (which followed on from iTrust). After NRC I went to work at the Communications Research Centre in Ottawa (also Canadian government) where I worked on things like privacy and information security and (of course) trust. In 2012, I went to work at Ontario Tech University (then UOIT) where I am currently associate professor of trust systems.

Phew.

I teach introductory management courses, a senior undergrad level course on Trust Systems (which this book started out being for), graduate level courses in Trust Systems as well as Information Security Policies and Risk Management, and have taught a bunch of Information Systems courses. In my past life in Stirling I taught programming courses mostly, and helped create a brand new first year program for computing science.

I live in Eastern Ontario on a tiny farm where we have a few animals and try to grow things to eat. I'm good at tomatoes.

A PROLOGUE, SOME THOUGHTS, AND A WARNING

Thanks for getting this far!

We are almost at the bit that actually has content. First, a few more words about the book, specifically about how it is written. When you write an (academic) book, there is a certain expectation that it be a serious endeavour. This book was no exception. It started well, and then I got to looking at it, and thought, "*well this looks boring*". And then I thought, "*it really doesn't have to just because it's academic*."

At the same time, I was getting very excited about sketchnoting and sketching in general. I'm really not very good at either, but to be honest, that doesn't matter. I wasn't that good at it when I was a child and it didn't stop me. I was also really excited about using Explain Everything to do just about *everything*. If you've never heard of Explain Everything and you are in any way associated with education I highly recommend looking at it.

No, I don't get paid by them.

I don't get paid by Apple either but the iPad Pro, an iPad Mini and my trusty Apple Pencils have been invaluable tools along this path.

It may well be that putting together a bunch of newbie sketchnotes and 'attempts' at drawing arrows and stick people is a mistake. Nevertheless here we are. It's a bit different, so, let's begin with a brief set of thoughts about its objectives and why it is how it is.

The objective of any book is to engage. The objective of a scientific book is to impart knowledge. The assumption is that the writer is capable of doing so. Unfortunately very few scientific tracts are engaging. I grant you that this is a subjective opinion. That said, the fact that Bill Bryson can write a science book better than most scientists is a fair indication that scientists can try a little harder.

I'm nowhere near the writer Bryson is. And that's okay.

I am pretty good with trust though. That is, I'm pretty good with the topic as a field of study, having been in it for over thirty years as I write. Whether I'm good at it as a human being is more open to debate. This is so for all of us. We're pretty much all of us 'experts' on trust, but many of us get quite flustered if we're asked what it really means to us. Try it — asking people why they trust something or someone is like that game your 5 year old played with you. The one where they always say "Why?" To everything you say.

Well, I use the word 'game' rather loosely… (It drives me up the wall.)

This book was written *primarily* as an electronic book. As such, it follows different kinds of, shall we say, conventions from physical books. Margins can be different, images and videos can be included all over the place, even widgets can be plopped in occasionally to help engage. I wrote the first drafts of it in Ulysses, which means it was written with a markup language, and this also meant I could export it in many different forms.

So there's an HTML version around someplace (maybe you are reading it now) as well as an eBook in various forms. But it's still at heart a *book*. Sure you can print it. There's bound to be a way.

Why did I write it like this? I wanted to experiment with imparting knowledge differently. There are (Bryson aside) plenty of good examples of this — James Burke, a science historian, has written books like *Connections* and *The Day The Universe Changed* that left me in awe as a young child and almost certainly put me on the path I tread today (as it happens, his was also probably one of the first voices I heard on TV when he covered the Moon Landing in 1969 when I was just 1 year old); Ted Nelson's *Literary Machines* foresaw and demonstrated in book form how Hypertext is used, whilst Burke (again) took that idea for a spin in his book, *The Knowledge Web* (see also here)[1]. And some time ago I read Brian Fawcett's book, *Cambodia*. It has an interesting layout where there are in fact two different things going on. In the top bit of each page is a story, but in the bottom is a sub-story, probably the real point of the book, about what happened in Cambodia and why we should care. This isn't just because what happened was terrible, that the 20th century in Cambodia was basically all but wiped out by the Khmer Rouge. It's because of why we all missed it, and what our consumer culture does to us. It's a fascinating book and troubling. But as a written document it appealed to me because of the way the subtext worked: it was a separate but connected part of the message being conveyed in the short stories above it.

I have tried in this book to inject some of the same kinds of oddity that make for interesting reading: there are (hopefully) fun and definitely information-rich images as part of the book. The footnotes, whilst not as complex as Fawcett's, hopefully add to the message. There are links shown here and there when they make sense, both to different parts of the book as well as external material. Every so often there's a movie too. This means that on some eReaders you won't get the movies but you can always grab them from the website for the book, which also has different versions of the book as well as other bits of writing and a blog on it. There are also exercises wherever possible to help you figure out things that are important[2]. Finally, at the end of most sections or chapters, there's often the name of the music I was listening to as I wrote, more for posterity than anything else but also as a reminder that you don't have to suffer in silence when you write.

One note about the characters you may encounter in the book. Computer security uses different characters to give its stories focus. First introduced in Rivest et al. (1978), they have been extended somewhat and from our original Alice and Bob we now have Carol, Chuck, Charlie, Dan, Eve, and many more. We'll use some of these characters in the book as we move forward. Sometimes they just appear by accident. Sometimes there is a purpose.

[1] One thing: when I reference things I will put links to them if at all possible, or to a place you can get them from, or follow links to in order to read them or watch them. This may be Wikipedia links, or links to various online bookstores, or doi links. Regardless, the things I do reference will also be listed at the end of the book in a largely traditional reference list as well as a growing further reading 'chapter.'

[2] Yes, it's subjective, but I think they're important so that's at least one person.

The many images in this book use pretty basic stick people. One reason for this is because I am, as I have already noted, a rubbish artist. However, there is a more serious reason for using stick people. They have no gender, no race, no obvious creed, indeed they are as anonymous as we can make them. Sometimes you can infer gender (but there may well be cis gendered males called Alice, and lots of blokes wear kilts).

Sometimes you'll see my dogs too: Charlie, Jessie, Sally and Ash. They are important, and immortalizing them in 'print' seems to be a rather apt way to thank them for their love.

The point is that we can imagine these characters, these stick people, as all kinds of things: humans, computer-based 'agents', Artificial Intelligences, and so on. It's a great shorthand for the things for which trust has a meaning.

Plus, I'm a rubbish artist (did I already say that? I may, to paraphrase Dickens, also be a large absent-minded scientist…)

The title mentions a warning. Here it is: if you haven't realized it yet[3], this is a serious book about a serious subject but it doesn't present the serious subject as seriously as some would undoubtedly like it to. If you would rather not read it as a result, this is entirely your choice. My own take on it is this: serious things don't always have to be presented entirely seriously. There is much to be gained from looking at them in different ways, with different kinds of prose or presentation. Of course, this doesn't always work and there are some parts of the book that are more serious than others and that's fine too. But consider yourself warned, it is not your average textbook. At all.

I hope you enjoy reading this as much as I did writing[4] it. If you spot any errors, however small, or have any feedback, or any questions at all, please get in touch: stephen.marsh@ontariotechu.ca

Huge thanks are due to the OER Lab at Ontario Tech University, who put this into PressBook for me and worked on exercises and more – the end product is amazing because of you all. Finally, thanks to eCampus Ontario for the grant which enabled this to be done.

On that note: Are you sitting comfortably? Then I'll begin.

Steve Marsh
Dalkeith, September 2021

powder! go away : laika
still wants go home

[3] If I haven't been subtle enough…

[4] And drawing.

CONTENTS

Let’s get started…

INTRODUCTION

Let's get started! Introductions are important trust builders.

> *"The chief lesson I have learned in a long life is that the only way you can make a man trustworthy is to trust him, and the surest way to make him untrustworthy is to distrust him."*
>
> *Henry L. Stimson*

This is a book about trust. More specifically, it's about what I call **trust systems**. I'll discuss what these are in more detail through the book, but for an introduction, think of it this way: A system is things working together to do something. A trust system is no different.

A trust system is a collection of three necessary and sufficient things:

- people
- process
- place

We can look at these things in isolation and examine how they all work together to help agents in a trust relationship make decisions — where the decisions come from and how they got there. People in the trust system are the agents in an interaction. They think about, act on, and otherwise engage with trust as a phenomenon. Processes are how trust is thought about, measured, decided. Places are the environments or contexts (real or virtual or imagined) that the trust decisions are made in. Trust is highly contextual, so it's important to think about that. For instance, I would trust my brother to drive me to the airport but not to fly the plane. The plane and the car are *places* in the trust system between my brother and myself where, of course, the process is flying or driving. Sometimes, process and place can be quite difficult to peel apart, but that's okay. So much for a system. We'll come back to it in much more detail, particularly when we talk about how[1] trust works.

Another question may have occurred to you as you read. What, exactly, is trust? That's a more difficult

[1] I see

question to answer than you might think. I mean, it's pretty easy for anyone to tell you what trust means to *them* (some languages don't have a separate word for trust, though). But what is it, well for all of us, specifically (or generally)?

Put simply, it's complicated.

A basic way of looking at it, that I've used in a lot of my work, is wonderfully described by Piotr Cofta (Cofta, 2007): trust is a way of accepting the fact that we can't control everything that happens around us. As Niklas Luhmann (Luhmann, 1979) would put it, it's a way of simplifying the decisions we have to make every day, and not think about the many things that, if we did think about them, would make us not want to get out of bed in the morning. Things like crossing the road. Things like riding a bike in heavy traffic. Things that we take on trust: that people driving a big red bus won't deliberately try to drive over us, for instance.

Trust is the foundation of a functioning society because of this (Bok, 1982). It is the acceptance of risk in a society where we can't control everyone. It exists because risk exists and we can't just stand still and do nothing. It's a social construct that doesn't even exist outside our heads but that, if it is betrayed, can hurt us. And, whilst we can't control people, trusting them allows us to shape their behaviour in interesting ways. As a matter of fact, as I type you can see this kind of thing playing out in places where COVID vaccinations and mask mandates exist – a lack of trust in people to act in the best interests of those around them leads to societal impositions and fractures.

All very well, but why *study* trust?

There are a few reasons. It's a fascinating phenomenon which, as you already heard, makes societies work. It's a slippery phenomenon that defies singular definitions. And yet, what would we do without it? It is almost impossible to imagine an environment or society where trust isn't, and if we can, it's dystopian and horrific, truly a Hobbesian world where life is 'solitary, poor, nasty, brutish and short.' Indeed there have been examples of such societies, and it's probably possible to see them today (more's the pity) where the lives of people are severely constrained. But we won't get too far into that particular problem of humankind. At least, not yet.

If you'll forgive a personal anecdote in what is a personal exploration, the time that trust became much more interesting to me was when I was doing some research for my PhD back in the '90s. I was trying to understand how individual artificial autonomous agents[2] in a society of multiple other self-interested autonomous agents could make decisions about each other in a meaningful way. As you do when you're starting on a Ph.D., I was reading a lot. One PhD thesis I came upon was a great one by Jeff Rosenschein. You can grab the thesis from here. In it, things like working together for agents were discussed in great detail. And on page 32 of the thesis there was this line:

[2] What are these? A simple explanation is that they are bits of code that are independent from our control out there doing things 'for' us. It can get a bit more complex because, you see, robots can be seen as such agents. And then it gets even more complex because we can, hopefully, take the idea and apply it to biological organisms like animals and other people. It's all quite fun.

> “It is essential for our analysis that the agents are known to be trustworthy; the model would have to be developed further to deal with shades of trustworthy behaviour on the part of agents.”

I think I sat there for some time looking at that sentence and wondering to myself exactly what was meant by trustworthy and by extension (it seemed to me at the time) trust. And that’s how it all began.

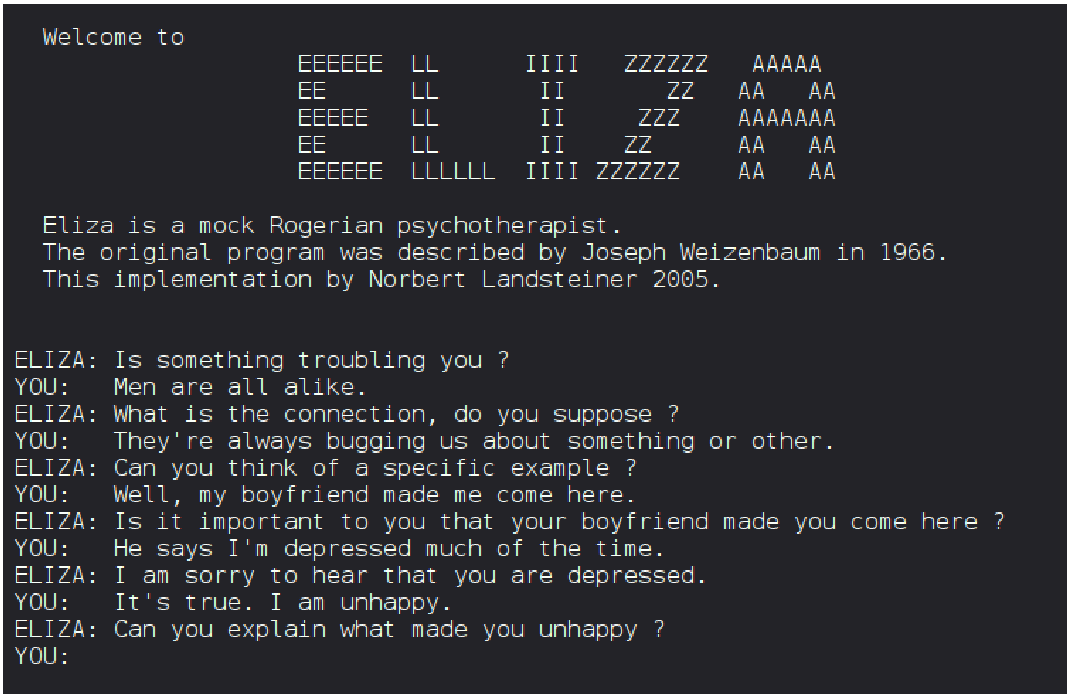
You see, it’s all very well to insist that trust has to exist in such situations or societies (because we made them and they are made of us). That kind of misses the point though, because when we let these ‘agents’ go to do their work we are no longer in a state of control. This includes self-driving cars, an AI that tweets, algorithms that trade in stocks, an AI that monitors students as they do their exams online, an AI that predicts recidivism, and much more. Whilst trust has been a human province for as long as there have been humans, we’re now in a situation where we really can’t afford to ignore the artificial autonomous entities around us. Like it or not, this has something to do with trust.

In this book, I build on a foundation of the cumulative understanding of trust from many centuries of work. I take it and examine it in the light of computational systems, those autonomous agents I mentioned before. Since 1990 and that line in Rosenchein’s Ph.D. thesis, I have worked on what has come to be called **computational trust**[3]. Simply put, it’s the exploration of how trust might be incorporated into the decisions artificial autonomous agents make. More, it’s an aspect of how we as humans might think about these agents.

Some would say that thinking about trusting an autonomous agent or an AI is a misunderstanding for the simple reason that we can’t trust AI. Indeed, we don’t need to because we can build in things like reliability and accountability and transparency (Bryson, 2018). Accountability is a big one here, and we’ll come back to it, but really this kind of argument is all about control and controlling the computational (autonomous) systems around us. Because when we control the thing, we don’t *actually* need to trust it at all. Because when the thing is completely transparent to us, we don’t need to trust it. We can *see* it and how it works (and why).

That’s possibly true if we want to see humans as rational creatures who do sensible things. They aren’t and they don’t. Human beings see technology as a social actor. As Reeves and Nass found out a long time ago – and as we will see a bit more when I start talking about Trustworthy AI – people see media and computers (and by extension artificial actors) as social actors (Reeves & Nass, 1996). Anyone who has heard of Eliza and Weizenbaum (1976) would tell you, people see artificial things differently than we would, perhaps, want them to.

[3] To toot my own horn, I created the field.



```
  Welcome to
                    EEEEEE  LL      IIII   ZZZZZZ   AAAAA
                    EE      LL       II       ZZ   AA   AA
                    EEEEE   LL       II     ZZZ    AAAAAAA
                    EE      LL       II    ZZ      AA   AA
                    EEEEEE  LLLLLL  IIII ZZZZZZ    AA   AA

  Eliza is a mock Rogerian psychotherapist.
  The original program was described by Joseph Weizenbaum in 1966.
  This implementation by Norbert Landsteiner 2005.


ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:   Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU:   He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:   It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:
```

Figure 1.1: A conversation with Eliza, a mock Rogerian psychotherapist program (from Wikipedia Commons)

This doesn’t mean that it’s correct to think this way. It doesn’t mean it’s incorrect either. it’s just a reality. And that means that we can either tell people not to do it and be ignored. Or we can help them do it better.

There’s another reality that is just as important. There is constant pressure these days to make AI more ‘trustworthy’ *so that people will trust it more*. This is interesting not because it is right or wrong, but because it totally misses the point: what are we trusting the AI *for*? What are we expecting it to do for us, and what is the context we are expecting it in. And it wouldn’t stop there in any case because as anyone who has interacted with dogs or cats or horses or many other animals before could tell you, you can both trust and be trusted by them. The real question is, “What for?”

Actually, the real question is more like this: why do you think someone would trust something or someone else just because it (or they) is trustworthy. That’s not how trust works.

To compare with humans, anyone paying attention to the world around them would hear about things like the lack of trust in politicians, or the need to think about trusting reporters more, or scientists more, and so on.. The question is the same. What for? As we will see shortly, Onora O’Neill has her own views about this.

PIONEERS

Introduction

In any field of research there are people who have helped us to change the way we think about a topic. Trust, despite being a field of study that is centuries old, is no exception. This chapter explores the thinking of some of these trust pioneers. Of necessity it's a somewhat subjective, personal choice of material. That's how research works, particularly in fields like this. There is an immense amount of literature revolving around trust, much of it recent. For instance, the Social Science Citation Index lists more than 46,000 articles on the subject of trust in the past 25 years or so. To put that into perspective, only around 4,500 were published in the previous 1,000 years. That's *just* the social sciences.

The result is that you tend to, if not pick and choose, at least get excited about certain writings. That they fit your own narrative is inevitable–they, after all, pretty much shape it. Note this: different researchers in a field such as this will have different thoughts about it. That is quite healthy: it would be quite a boring world if everyone agreed on everything. My own take on the research that is important is in this chapter specifically and in this book in general. There will be people who think I have missed certain authors or seminal papers or whatever. That's okay. For one thing, I can't possibly synthesize 46,000 papers, and nor would I want to. For another, the point of being a so-called expert is that you get to curate stuff in much the same way as a museum curator shows you the things that they think are important from the vastly larger collection the museum may have. It's a fun part of the job.

Bear in mind, important doesn't necessarily mean right. In this book I aim show a few different viewpoints from different researchers and thinkers. Sometimes I'll show why I think these are 'correct' or 'incorrect' but often I won't: the things I present are important to me regardless of their correctness. Your own practice will ultimately determine the rightness of them. That's part of the joy of doing the research!

This is a book about **trust systems** and **computational trust**. It's not, specifically, a book about trust in human beings, but it is informed by the phenomenon of trust in human beings. So it works like this: I show some of the important work that explores trust in humans which helps to inform the field of computational trust. I'll show how and why this is the case. I will also, in this chapter and throughout the book, show several different views of how trust in human *and* computational systems works. Guess what? One such viewpoint is my own (that's also part of the joy of research: you get to think about and create your own stuff and, if you're really lucky, change the way people think).

This chapter is titled *Pioneers*. As such, it will explore the people and research that excites, bothers, informs and moves the field forward. I'll do it person by person because to a large extent the difference in our thinking were shaped by these individuals as they have presented their work. This is often true of many fields. Take a look back at physics, for instance. Copernicus and Galileo for their solar system and telescopes; Newton and his apple and his three laws (amongst other things!); Einstein and his theory of relativity. Or perhaps Ampére, Ohm, Faraday and Watt. The point isn't just that they (in some cases) made individual strides forward for the fields they were in. The point is that they took what others had done and moved the world forward in its understanding. Could it have been different people? Sure. Of course in some sense they were the right minds at the right time. But they were the right minds, which is why when we talk about physics in all its guises, their names always come up. Science is like that. The field of the study of trust is not any different.

So who are the pioneers of trust? Read on and find out! For different reasons, the writers and researchers I talk about in this chapter have had an important impact on the field of trust as it relates to trust systems. In many cases the impact has been far-reaching across the study of trust as a whole. For the interested reader (and I hope you are), there is a comprehensive further reading section toward the end of this book. Since the book is electronic rather than paper (but do feel free to print it!) that section and the chapter as a whole will continue to evolve, so do jump in.

Trust Systems Pioneers

Morton Deutsch

Morton Deutsch was a pioneer of peace studies and conflict resolution. It goes without saying then that he would have studied trust. After all, given what we as humans perceive trust to be, it would hardly be possible to have peace of some form or other without trust. In fact, Deutsch was foremost in the study of trust as some form of a computational phenomenon. His definition of an individual focusing on trust, which is foundational, is:

> "(a) the individual is confronted with an ambiguous path, a path that can lead to an event perceived to be beneficial (Va+) or to an event perceived to be harmful (Va−);
>
> (b) he perceives that the occurrence of Va+ or Va− is contingent on the behaviour of another person; and
>
> (c) he perceives the strength of Va− to be greater than the strength of Va+.
>
> If he chooses to take an ambiguous path with such properties, I shall say he makes a trusting choice; if he chooses not to take the path, he makes a distrustful choice." (Deutsch, 1962, page 303)

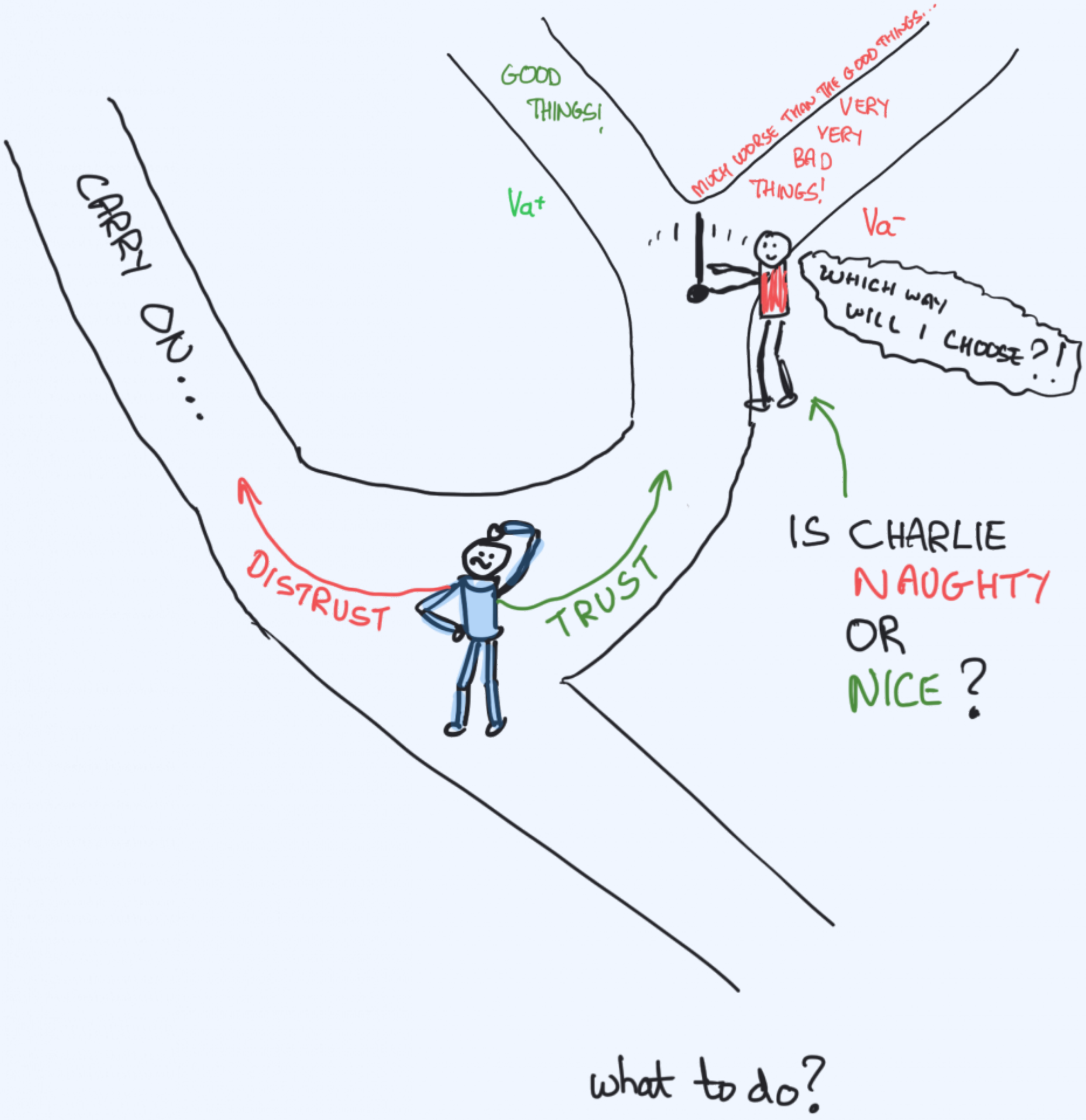

GOOD THINGS!
Va+
MUCH WORSE THAN THE GOOD THINGS...
VERY VERY BAD THINGS!
Va-
WHICH WAY WILL I CHOOSE?!
CARRY ON...
DISTRUST
TRUST
IS CHARLIE NAUGHTY OR NICE?
what to do?

Figure 2.1: Making a trust choice, Deutsch style...

“Well, sure,” you’re saying to yourself, “but what does that actually mean?” Perhaps it’s easier to see in a picture. Which is good, because I drew one.

To begin, it’s important to tell a story. It’s one that Deutsch himself used (Deutsch, 1973) to make things a little clearer, as well as to discuss things like confidence. We’ll come to confidence, but let’s hear the story first.

Stories are good. I've taken the liberty of bringing this one into the 21st Century because I can. And because it matters that everyone feels valued. So it's an adaptation of *The Lady or the Tiger*.

Okay, so there's this princess. Princesses are fine. This one, let's call her Alice (see, I knew those characters would come up!). Anyway, Alice falls in love with a beautiful but poor writer in the local town, whose name is Carol. Carol and Alice have an unfortunately brief partnership because Alice's father, King Bob, finds out about it and is less than pleased. Perhaps not for the reasons you think, after all this is the 21st Century and even kings move with the times. Bob is stuck somewhere around 1970 but for royalty fifty-odd years out of date is pretty hip. In fact, Bob just wants to be able to choose Alice's life partner himself – alliances are made in such a manner and the blood can't be diluted etc., etc. (Okay, maybe he's stuck in 1870).

Here's where the picture I promised comes in. You can see it on this page. King Bob has poor Carol arrested and brought before him. With a distraught Alice standing next to him he has Carol taken to face two identical doors.

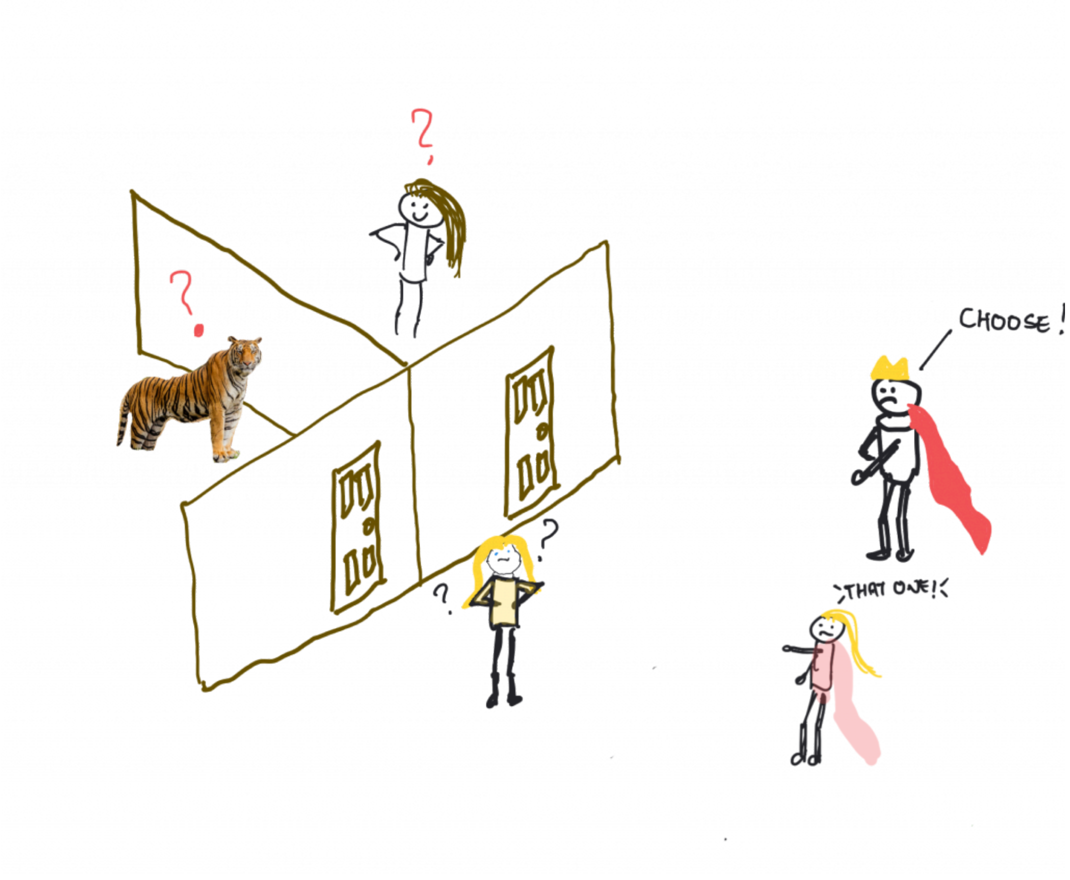


Figure 2.2: Bob forces Carol to choose!

Bob goes on to inform Carol, in suitably kingly tones, that behind one door there is a beautiful someone, who (it is assumed) will steal her heart. *Whatever*. Anyway, behind the other door there is a tiger. You know, the roaring kind, which almost certainly means almost certain death for Carol (perhaps he's stuck in 1570). What to do?

Well, Carol is about to make a choice when she notices Alice pointing subtly toward one of the doors. What would you do if you were Carol? (No, not be captured in the first place is definitely not an option.)

Carol chooses that door instantly and steps through to... well, that's where the story ends. Make your own choice. But that's precisely the point, you see: you get to make your own choice. If you think back to Deutsch's definition of trust you'll remember a few things.

First, there's the ambiguous path that can lead to a beneficial or a not so beneficial outcome. I'd say we have that situation here. The choice faced by Carol is one that leads to good (technically – I mean, the beautiful lady may not be Alice but at least Carol is alive) or bad (definitely – well, on the assumption that being eaten by a tiger is bad.)

Second, the "occurrence of" the good or the bad is dependent on the actions of another person, who is not under Carol's control. We've definitely got that. After all, Alice is pointing at a door, Carol has no way of making Alice choose one or the other, it just is what it is and it's Alice's choice.

Third, the bad has to be worse than the good. We'll come back to this because it's important where trust is concerned. In this story, well, as I said above, being eaten is probably seen as worse than not.

Here's where it gets interesting. The choice to go through a specific door is Carol's, not Alice's. So whilst Alice may choose a specific door, Carol is under no obligation to head through it. She chooses to. So, Deutsch's final condition is met because in the story, Carol chooses to go through the door that Alice points to.

In other words, from Deutsch's definition, Carol trusts Alice to have made the "right" decision for her. We'll come back to the reasoning in a bit, but before we do there's one more important thing to notice in the definition. It's that Deutsch uses the word perceives a lot. What does this say about trust? That's right! It's an individual's choice based on what the individual subjectively sees in the situation. Trust is subjective. It's something that is for us to give. This is a vital thing to understand where trust is concerned. It is entirely possible for someone to ask you to "trust me" or "trust the police" or whatever may be. It is entirely wrong for them to demand that you trust them, the police, politicians, and so on. Because trust is a personal choice. You are the one that chooses to trust, based on, well, we'll see.

Why does this matter? Quite apart from the obvious that I already mentioned (police, politicians, etc.) it matters greatly in situations where people talk about making 'users' trust AI (**Artificial Intelligence**). Nobody can make you trust anything, let alone AI. The choice is yours. No-one can design an AI you will automatically trust because you must. The only thing that can be done is to design systems that are worthy of your trust in some way. Trustworthiness is something separate from but related to trust in very interesting ways. We will get to those in this book and undoubtedly take the time to reinforce this important fact: trust is something you give.

Let's get back to our tiger, who is either now very hungry or very full of Carol. You'll remember that I asked the question, why does Carol choose the door Alice points to at all? We've kind of established that Carol makes a trusting choice – that Alice will do the 'right' thing for Carol, Alice or both. The thought process might be difficult but the choice is clear: choose the door Alice points at. In his 1973 book, Deutsch explores several different reasons why Carol would make the choice, all of them related to a specific kind of trusting action (see also Golembiewski and McConkie, 1975). Without spoiling the fun, think about the negative bits of trust. Sometimes all that trust is, is the best of two bad choices.

Maybe that did spoil it a little, but there we are. Deutsch examines the choice in detail, and so should we. In fact, the decision may have been influenced by entirely different kinds of trust. That they are different raises a very important point about trust – trust is different in different contexts. In this case Deutsch suggests there may be at least nine different (reasons behind) the trust choices made. So, let's see.

First, and perhaps, hopefully, least, is trust as despair. Sadly, this is something most of us have experienced: that the act of choosing is better than the situation we are in right now (for Carol, it's the purgatory of both not being with Alice and indeed not knowing if Alice meant her to get eaten or not –which might well have ramifications for their relationship).

The second is trust as social conformity. In many situations, trust is expected, and violations lead to severe sanctions. And so, Carol chooses the door because she may end up socially ostracised or labelled a coward.

The third is trust as innocence. The choice of a course of action may be made upon little understanding of the dangers inherent in the choice. This innocence may be rooted in lack of information, cognitive immaturity, or cognitive defect (so-called pathological trust).

The fourth is trust as impulsiveness. Inappropriate weight may be given to the future consequences of a trusting choice. As we'll see in our Prisoners chapter, this is a bit like the shadow of the future that Axelrod talks about in his work (for example, Axelrod, 1984). It is also exhibited in the decision of many children to choose one marshmallow now instead of two later.

Fifth, we have trust as virtue. Here, trust is naturally considered a virtue in social life. By now, hopefully, we have reach an understanding that trust is important to society (and it working!) and so, to be sure, it is virtuous. How does this relate to Carol? It's like this: she trusts Alice because it's a good thing to do, and not trusting Alice would be, well, wrong. I know, I know, her life is at stake, and so on, but read on, because…

The sixth is trust as masochism. Here, Carol opens the door because she expects (or wants) the tiger rather than the beautiful other, since pain and unfulfilled trust may be preferable to pleasure. Sure, it's a thing.

Seventh, we have trust as faith. Carol may have faith in 'the gods' such that she has no doubt that whatever awaits is fated and thus to be welcomed. As Deutsch notes, and what is more, this chapter explores a little in some way, having faith to a large extent removes the negative consequences of a trusting decision.

Eighth, we have trust as risk-taking or gambling. Carol (faithless Carol) jumps because the benefits of the beautiful other are so much more than the risk of the tiger that it's worth risking.

Lastly, ninth we have what most people see trust as — trust as confidence. Here, Carol chooses because she has confidence that she will find what is desired rather than what is feared. Cast your mind back a little to the way we discussed trust earlier in this book: it has to do with risk. And so, to make a trusting decision, first we have to accept that there is a risk involved. This story is the very definition of such a situation.

I put it another way toward the start of the book: trust is the response to a situation of risk — it's one way we manage such situations. It's also optimistic in the confidence sense. Trust is "strongly linked to confidence in, and overall optimism about, desirable events taking place" (Golembiewski & McConkie, 1975, page 133).

The truster always hopes for good things, whilst taking a risk that bad things will happen (Boon & Holmes, 1985).

If you revisit Deutsch's definition above you'll notice that the bad things is required to make the situation worse than the good thing would make it better. Deutsch isn't the only person to think such a thing. As we'll see, Luhmann defines trust much the same way. It makes sense, too. Think, for example, of a couple who want to go out for a movie together. They get a babysitter for their baby so that they can have some 'them' time and off they go. They are clearly expecting the babysitter to look after the baby so that they can get a small reward of a few hours together. This is trust — the cost of the baby not being looked after are incalculably worse than the benefits to be gained from a few hours of peace.

This requirement is not always present in researchers' definitions of trust, but it is seen as important in many cases. Deutsch goes a little further with some basic assumptions about people. Put simply, that they tend to try to do things that they think will be good for them and not to do things that they think won't be. The strength of these 'trying' actions is related to things like perceived utility (we'll get to this in a second); whether they think their actions are more likely to make something good happen, or more likely to make something bad not happen; the probability of the good or bad thing happening; the utility they can get from doing the things and the 'immediacy' of the event happening (how close are we to the good or bad thing happening). What does that all mean? Think about a situation where you eat a marshmallow and lose a year of your life as a result. Now, losing a year of your life at the age of five is not that 'immediate' and so the marshmallow (which has a positive utility–it tastes good!) Is probably going to disappear pretty quickly. But what if you're 60, 70, 80? You may be less willing to risk losing that extra year and so, well, the marshmallow can wait for the grandchildren to come around. To be even more basic: people are likely to act in a way that helps them to survive.

Deutsch goes a bit further and says that people are actually all 'psychological theorists'–not only do they do this stuff for themselves, they also think about how others might be doing it and make assumptions (that people will act to survive, get nice things but not at any cost, and so on).

Actually that last bit is interesting not only because it is almost certainly true and helps us to understand the actions of others. It's also interesting because of one of the problems of doing research on trust: there are an awful lot of definitions of trust. We are all, in our own ways, experts in what trust is. How could we be otherwise? I often start talks about trust by saying that there's probably a different personal definition of trust for everyone in the room. It's not too much of an exaggeration either. Remember at the start of this chapter I mentioned the massive number of articles written about trust. This is a part of the reason why.

Okay, so utility. Put simply it's a measure of what you get out of a situation. Dive a little deeper and you could say that it is the difference between the good bits and the bad bits of the situation. You might measure it with money in some circumstances. For example, you pay the babysitter and you pay for the movie when you go out. It costs you something (forget the trust bit for a second). What do you gain? The pleasure of the movie, the pleasure of being with your significant other for a few hours without interruptions. Are the costs worth the gain? That's the utility bit. Yes, you can take it much further (we will in our Prisoners chapter) but it's kind of like this: you gain something and you lose something in most situations. If there is a risk of losing something much more important to you than what you stand to gain, and you can't control the situation, you put something in place to make that situation bearable.

That something is trust.

This is pretty much where we are with Deutsch at the moment, but consider this: people are not rational and, whilst they do try to maximize the good stuff they're actually not that good at figuring out what the potential costs are — you might say we're pretty poor at risk evaluation — and even if we could do it well, putting our baby in the hands of a (potentially very young) stranger is sheer madness. What is the difference between leaving the stranger in your home with the baby and giving the baby to a stranger on the street to look after while you popped into the shops? That we do it (the former not the latter) is evidence of how irrational trust actually is.

Still, we manage to make it work and we can justify it quite well to ourselves. As an exercise, try asking someone to explain why they trusted another person in some circumstance. The further into the rabbit hole of trust you take them, the more defensive they are likely to become. Why? Because trust isn't rational.

Does this mean that we can trust machines? We'll see.

Niklas Luhmann

Figure 2.3: Niklas Luhmann

Niklas Luhmann was a sociologist who wrote prolifically about subjects as seemingly disparate as intimacy and the law. His social theory is a little too complex to go deeply into here, and undoubtedly theoretical sociologists would be quite unhappy with me trying to anyway. This, of course, is not to mention how they might feel about what I'm going to talk about anyway, which is Luhmann's theory as it relates to trust. Luhmann wrote about trust (*vertauen*) and power (*macht*) in separate but connected volumes in German. These volumes were brought together in an English translation in 1979 (oddly enough called *Trust and Power*).

The trust part of Luhmann's work is what I'll focus on here. It has the added benefit of being to the point in the English version. Apparently Luhmann's writings in German were a little complicated. It is at this point that my limited grasp of German would fail, in other words.

For Luhmann, an awful lot of what his eventually systems theory of society would look at was that a social system (a society, if you like) was sort of a zone of non-complexity. It allows the individuals inside it to exist in an infinitely complex external environment because they get to choose what to communicate and pay attention to, and society is constructed of these communications.

Okay, that was almost certainly not what he would have said at all, but it'll do for now because as far as trust is concerned, this is exactly what was meant. Trust, for Luhmann, was a means of reducing the complexity of life. Indeed, without it we would have trouble getting up in the morning.

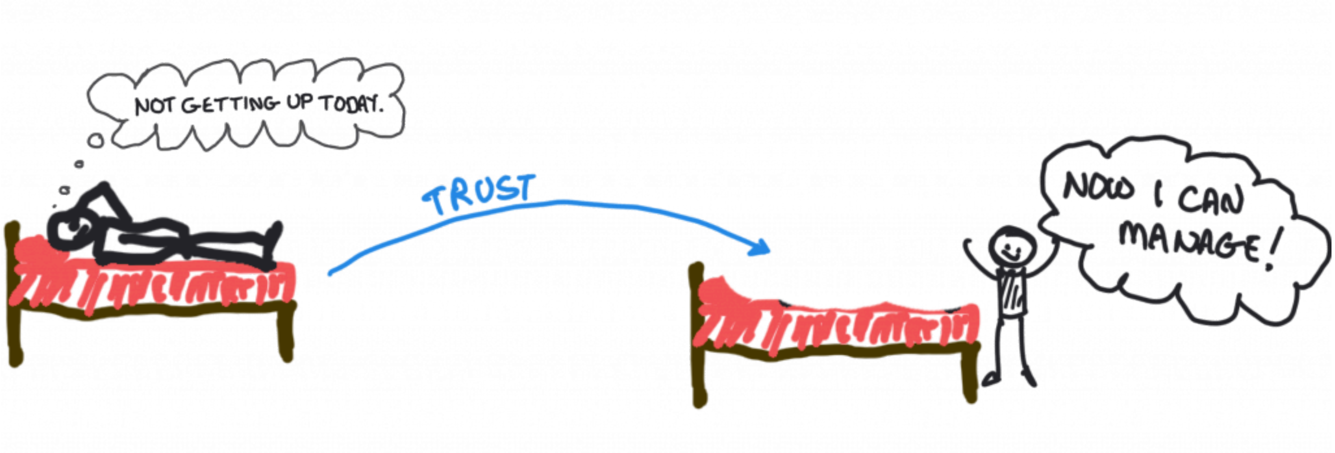


Figure 2.4: Trust means reducing complexity enough that you can think about getting up in the morning.

How does that work? You can think of it in a relatively simple way. When you cross the road, you're implicitly trusting the driver around you not to drive over you. You are implicitly trusting the people around you not to do something bad. You are trusting, in other words, that the society around you will 'behave' so that you can continue to go about your own business. Think about this for a second. If you had to think about all the bad things that could possibly happen in your day, how would you feel? Honestly, if you thought about everything, good or bad, you would probably blow your mind. I'm sure I would.

Trust means we don't have to think about all of those things. It means that we can take some things as given. To put that another way, we can take some things 'on trust'. We don't have to think about those car drivers because we can 'trust' they won't try to run us over. When we look at it like that, we can see exactly how trust reduces complexity in everyday life. This was Luhmann's point, of course. Naturally there is much more. In particular, Luhmann sees trust as a "basic fact of human life" (Luhmann, 1979, page 4).

It's actually a really powerful theory because it explicitly acknowledges a few different things:

- The fact that human beings live in a complex society
- The fact that human beings can't really control other human beings (not really, and this is where that power bit comes in, although we won't go there right now).
- The fact that human beings can therefore make life much more complex of themselves and others because of this agency.

If that all seems a little complicated, think of it like this. We all wander around doing our own thing. Much of the time us doing that has no effect on other people, but sometimes it does. Sometimes people might need us to do something. Sometimes us doing something gets in the way of what others might wasn't to be able to do. And so on. Because of this, we have the means to make things more complicated for ourselves and others simply because of the things we can do. Not only that, because if we do trust people, or even take things on trust, like we talked about earlier, this vastly increases the things that we can in fact do. We can cross the road to look at what is on the other side. We can hire a babysitter, and so on. As Luhmann said:

> "Where there is trust there are increased possibilities for experience and action, there is an increase in the complexity of the social system and also in the number of possibilities which can be reconciled with its structure, because trust constitutes a more effective form than, for example, utility theory of complexity reduction."
>
> (Luhmann, 1979, page 8).

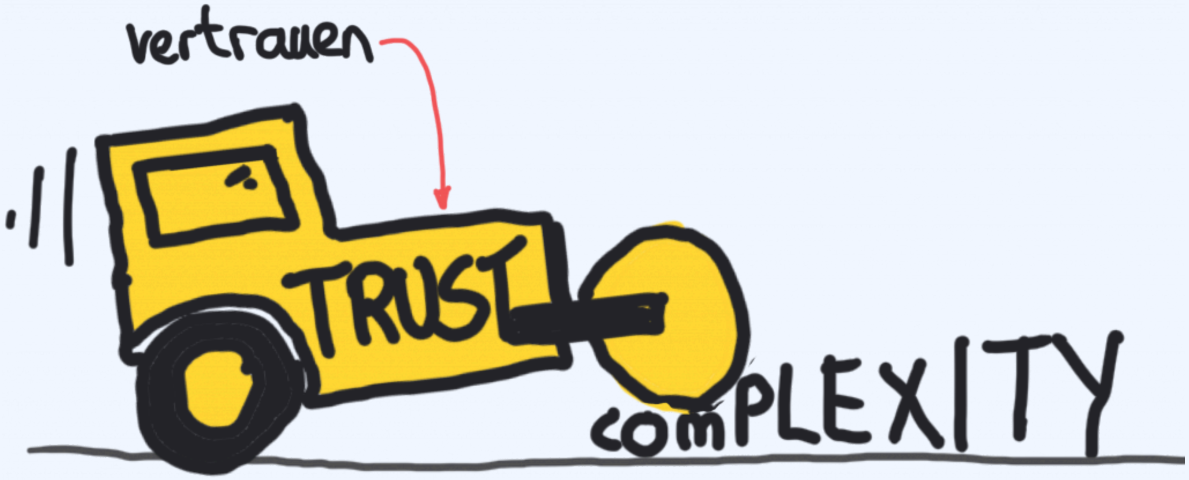


Figure 2.5: "Complexity is nothing to me!"

Pretty powerful stuff, that–not only does trust allow us to do more and new things, it allows us to handle these new things at the same time. One more way of looking at it? Trust is a way of coping with the freedom of others (Luhmann, 1979).

What was that about utility theory? It's like this: in utility theory people, as rational actors, as supposed to act to maintain or increase utility–we'll pick the path which has the best reward (or lowest cost) and so on. That sounds great until you realize that really, we're not very rational at all and in fact we do some rather silly things.

The point Luhmann was making there is that trust isn't an entirely rational thing. But that's okay because it allows us to do things which utility theory might say were stupid (or wrong). Like, for instance, jump off a bridge with an elastic cord tied around our ankles.

In other words, trust allows us to accept the fact that risk exists. We'll come back to this a lot more in the course of the book, because trust and risk are beautifully intertwined. For Luhmann, as for Deutsch if you think about it, trust is needed because risk is there. As Luhmann put it, "Trust . . . presupposes a situation of risk" (Luhmann, 1979, p.97).

Now we are getting somewhere, you think to yourself. Sure. Look at it like this: we know that risk is there is pretty much everything we do, right? There's a risk the babysitter might not be a nice person. There's a risk the person we elect might be a sociopath (more on that one later). There's a risk that the person controlling a huge social network is a sociopath, come to that. There's a risk that Steve might fail to finish this book. Put it another way: making plans means looking at the risk that in the future stuff might happen that makes the plans screw up: “a knowledge of risk and its implications allows us to make plans for the future which take the risks into account, and thus make extensions to those plans which should succeed in the face of problems or perhaps unforeseen events” (Marsh, 1994, page 42).

Sounds good, right? A quick look back at Deutsch will show you something a little similar. Carol chooses the door she does despite the risk of being eaten. The truster, in this sense, hopes for an outcome that is good, but accepts the risk that it might turn out, erm, badly (Boon & Holmes, 1991).

This view is not unusual. Piotr Cofta (2007) also looks at how risk and trust, as well as control, are related, as we’ll see later.

Diego Gambetta

Figure 2.6: Diego Gambetta

Gambetta, another (and another of our pioneers) who has studied trust extensively, defines trust like this in his book:

> “Trust (or, symmetrically, distrust) is a particular level of the with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action.”
>
> (Gambetta, 1990, page 217).

Clearly this has a lot in common with Luhmann, especially the last bit — trust is a way of coping with the freedom of others. It also brings in the idea of subjective probability, which is quite important, especially when we start talking about how to compute trust, but bear in mind that trust is not probabilistic. It's also important that Gambetta talks about using values for trust, again because computers like that sort of thing.

You may have spotted that Gambetta refers to distrust in this definition. Specifically, he talks about distrust being symmetrical to trust. It's not entirely clear what that means, and I'm not sure it's correct, but we'll get to distrust and things like that later in the book. It is worth bearing in mind though that trust and distrust are not two sides of the same coin. As Luhmann notes, they are not merely opposites, they are functional equivalents. You can have trust where distrust has been, and vice versa.

Gambetta's work is interesting, and not just because he studies the mafia. One of the things he talks about is signs (Bacharach & Gambetta, 2001). Basically this is about the signs we give about how trustworthy in a given situation we might be. These are things like businesspeople wearing business attire, or police officers wearing their uniform. The signs can be more or less overt than that, and the basic assumption for this is of course that the person who is about to make a trusting decision has eyes. To put it another way, "the truster sees or otherwise observes the trustee before deciding. She therefore can, and should, use these observations as evidence that the trustee has, or lacks, trustworthy-making qualities" (Bacharach & Gambetta, 2001, page 148). Now this is interesting not just because the signs are acknowledged, but that the word trustworthy is used. It's an important word and we'll discuss it in detail later, but for now remember: trust and trustworthiness are not the same thing.

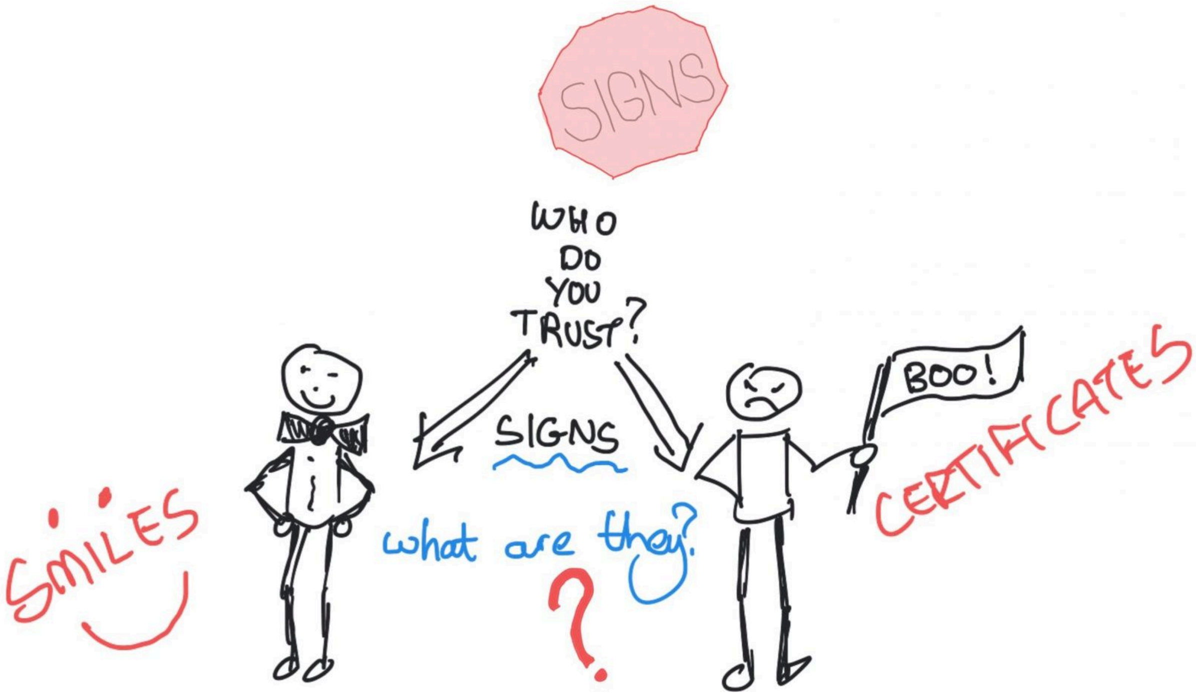


Figure 2.7: Signs!

As I will repeat through the book, Trust is something that you give, Trustworthiness is something that you have. To put it another way, I'm quite free to trust an untrustworthy person...

So far, we've seen that people have defined trust in terms of risk, and in terms of the freedom of others. But what if we look at it from the point of view of how people might behave in a society where they are perceived to be, say, experts in some field or other? This is related in part to what we were just talking about with trustworthiness. Surely you would be more likely to want to trust a car mechanic to fix your car than a computer scientist? I can fix cars, but I'm really not very good at it. The end result is not a great job and it takes ten times as long, I'm afraid. On the other hand, whilst I have the greatest of respect for mechanics, I wouldn't necessarily ask them to fly a passenger jumbo jet across the Atlantic.

Bernard Barber

Do you see where this is going? Bernard Barber did. He's another of our pioneers. He's another sociologist who studied trust. He actually started to look at it because he was a little dissatisfied with how trust was referred to in everyday life. I can't say I blame him. Things have not got any better. Trust is probably one of the most overloaded words out there in the English language. Sometimes we even use it in the exact opposite way that it should be used. But we'll come to that when we talk about security. Let's return to Barber. In his *The Logic and Limits of Trust* (Barber, 1983) he attacks the problem in various ways. As a sociologist he sees trust as an aspect of all social relationships, and "predominantly as a phenomenon of social structural and cultural variables and not . . . as a function of individual personality variables" (Barber, 1983, page 5). That last bit puts him squarely in Luhmann's camp, by the way, and not Deutsch's. That's definitely okay. And it's not just there either because Barber sees trust as a way of looking at the future. An expectation. Barber specifically lists three types of expectations in these social relationships (Barber, 1983, page 9, from Marsh, 1994, page 44)

> "Expectation of the persistence and fulfillment of the natural and moral social orders; expectation of "technically competent role performance" from those we interact with in social relationships and systems; expectation that partners in interaction will "carry out their fiduciary obligations and responsibilities, that is, their duties in certain situations to place others' interests before their own."

There are many reasons why this is important. Firstly, because there is an explicit reference to natural as well moral social orders. Part of this is an expectation that things like physics and biology stay the way they are. This is actually a really neat thing because all of a sudden we can say things like "I trust this chair to hold my weight" and have it mean that I trust the laws of physics will keep on working and I won't fall through the substance of the chair (which is unlikely to happen but you never know). Barber's reference to the moral social order is also interesting and akin to the kind of trust that Bok says society will collapse without (Bok, 1978).

Secondly, it's important because it introduces the idea of competence to us. We trust people because they are technically competent. For instance, when I say I trust the chair it is also likely because I trust that the person who made the chair was competent at making chairs . It's this kind of trust we place in politicians in the sense that they are good at their jobs, for example.

Third, Barber introduces us to the idea of fiduciary trust. This is a vital part of how our society actually works. Remember that car mechanic? How would I even know if the work they did was dangerous and liable to result in a crash? It's about responsibility and obligations: the mechanic is obligated to be responsible when fixing my brakes. Think about it from the point of view of a patient in a hospital talking with a surgeon: at some point the surgeon is the expert who will be cutting into the patient and it would be nice if they knew what they were doing, It would also be nice if they were good at what they did. Further, and probably even more importantly, the patient trusts the surgeon to have their best interests at heart when performing their job. That's fiduciary trust.

It's interesting, isn't it? When the year 2020 started few of us even knew about Wuhan, and probably even fewer knew anything much about COVID-19. When it finally appeared in our midst we trusted the professionals (the health-care experts) to have our best interests at heart as a society, and as a result to give the right advice to the politicians (who we elected to have our best interests at heart, at least in some way). We trusted the politicians to listen and do the right thing. And so when the lockdown happened in various parts of the world, people listened.

Where trust in any of those things broke down, the lockdown either simply didn't happen, to disastrous effect, or its effect was limited by people who perhaps didn't have the best interests of their constituents at heart. And as I write this in 2021 when we are still facing the problem, our trust in the chain may well be broken in various parts of the world because those we trusted to have our best interests at heart failed that test in some way: perhaps they travelled when they asked us not to; perhaps they didn't wear masks when we were asked to; perhaps it was other things. The result? If you're reading this you can probably see it for yourself.

Trust can be a matter of life or death.

It's the fiduciary trust, allowed with the idea of competence-based trust, that is quite powerful in Barber's work. It gets even better though because Barber then thinks about the fact that someone who is good at one thing might be rubbish at another. To put it another way, "trust cannot necessarily be generalized." (Barber, 1983, page 18). As a case in point, I'm actually quite good at writing and I'm not too bad at cooking but, as we will see later, neither of those things say anything about my skills as a brain surgeon.

Onora O'Neill

Trust as non-generalizable is important and something that Dame Onora O'Neill (another of our pioneers!) put quite succinctly in her TEDx Parliament talk in 2013:

> "I might for example say that I certainly trust a certain elementary school teacher I know to teach the reception class to read, but in no way to drive the school minibus." (The talk is here– Stop and watch it right now!).

My own usual example in this regard is that I would trust my brother to drive me to the airport but not to fly the plane.

The thing is, though, if someone has your best interests at heart, they won't actually try to do something they know they can't do because it would let you down. In this sense, we trust people to know their limits and stick within them. There's much more wrong with saying you can do something that you can't and failing than there is of saying you don't know how and asking for help. This of course begs another question: is the person who trusts mistakenly at fault for doing so? Is the person who was trusted mistakenly at fault? I'll leave you to think about that one for a while.

Importantly and certainly poignant at the time of writing, Barber sees fiduciary trust as important as a social control mechanism in a positive sense. Think about it like this: society would probably like to be peaceful and well run by, say, politicians. The thing is, to run society means that someone has to have power. This power is the politicians' because they are granted our trust (Dasgupta, 1990). As Barber notes:

> "… the granting of trust makes powerful social control possible. On the other hand, the acceptance and fulfillment of this trust forestalls abuses by those to whom power is granted."
>
> (Barber, 1983, page 20).

The thing is, if we do want to trust the people we put in power, we have to trust that we can remove them from power, through the ballot box preferably, otherwise if necessary (Dasgupta, 1990). As we have sadly seen, the breakdown of this trust is apt to be violent if the ballot box is seen as an inadequate control mechanism (Marsh, 1994, page 46).

Piotr Cofta

Control is one of the main aspects of the work of Piotr Cofta (for example in 2007), yet another of our pioneers. Cofta sees trust and control as diametrically opposed. Let's see what that means. As we've already seen in this chapter and we'll see much more in the book as we go on, trust is explicitly tied to risk: if risk exists, trust is necessary (Luhmann, 1990). As Dunn says, "however indispensable trust may be as a device for coping with the freedom of others, it is a device with a permanent and built-in possibility off failure." (Dunn, 1990, page 81). When we trust someone (or something) in some situation to do some thing, we accept that they might not do it. We accept that things can go wrong. We accept risk.

This brings us back to the question I left you with earlier: whose fault is it when trust is betrayed? Still thinking about it? Okay, let's carry on then.

If it is the case that trust "presupposes… risk" (Luhmann, 1979, page 97), and I certainly believe that it is, where does that leave us with respect to control?

Control means what it says. If you control something, you know from one instant to the next what it will do. How can it be otherwise? So here's the thing: if you know what will happen, where is the risk?

Exactly.

Cofta's argument is that we are increasingly relying on technology to connect us, and this has problems if we think about those signs I talked about before – after all, how can you tell who is a dog on the Internet? In his work, Cofta examines the idea of trust and control with respect to our inability to tell if someone is trustworthy at the other end of the line.

There's more though: part of the problem isn't that visibility thing, it's complexity. How does the Internet work? How do mobile communications work? Does anyone have a really solid understanding of the whole of the Internet – where the cables are, where the routers are, how they all work together and where a single piece of information might be at any given time?

Of course it's a rhetorical question.

More to the point, some people probably do understand all this, but bear this in mind: our general lack of understanding is a definite barrier to our ability to control the systems we use. And if we can't control the systems we use, what does that leave us? It leaves us with a risk that they might not work the way we want them to. And when there's a risk, there's going to be trust.

Another example might help here, so let's return to my car. Actually, I drive an electric one, so let's return to the truck I owned previous to the car. There's a picture of it (Figure 2.8).

In this picture you will notice that the weather is a little inclement. This happens sometimes in Canadian winters. At such times, if I needed to go someplace, there may well have been fingers crossed. You see, the real value I placed in my vehicle is the ability to start when the weather is like that, and the temperature is around −30 or so Celsius. The conversations I have had in the past with my vehicles are often telling. They include cajoling, begging, and more than once, threatening the scrap dealer (needless to say they don't always work). The thing is, I need my truck to start in all weathers. I trust it it to do that.

Figure 2.8: A lovely winters day in Ontario. I hope the truck will start...

Consciously or not, there is an expectation, perhaps based on all the other −30 days, perhaps based on the fact that I plugged in the vehicle overnight, maybe even that I had it serviced regularly, and so on, that the thing will start, and I won't be left stranded in the cold.

Looked at another way, this is a transactional form of trust In a sense. I give care to be able to trust. The more care I give in the sense of servicing, or plugging in when the weather is going to be cold, etc., the more entitled to not being 'betrayed' by the vehicle I might feel. This is related to the transactional trust explored in Reina and Reina's *Trust and Betrayal in the Workplace* (Reina & Reina, 1999) where, amongst other things, trust is given in order to maintain trustworthiness (see also Stimson's quote earlier in this little book). The difference, as others have pointed out, is that the truck doesn't actually have the capacity to betray me – it's an inanimate object.

And yet, we still talk about inanimate things as if we trust them. We'll get to this when we start talking about the Media Equation later in the book.

This quite nicely brings us to my unanswered question though: if your trust is betrayed, whose fault is it? The picture is perhaps building a little more, especially if we bring transactional trust into the picture. The question begins to turn itself around to something like "Do I have the right to trust you to do this thing?"

Let's leave it there and carry on.

Jeremy Pitt

Of the pioneers here, Pitt is one of the more recent, but his work has left a lasting impression on the field[1], particularly in the **how** to use social agents to investigate theories. In his extensive work Pitt has examined social justice, Ostrom (see here for Ostrom!), fairness, forgiveness, trust, prisoners dilemmas (if you don't know what that is, I write about it in the next chapter) and more. He has invented a particular way of looking at

[1] And continues to do so

these things, which he calls Artificial Social Construction. What, you ask, might this be? Put simply, it's using artificial agents to create artificial societies in which to test the theories he examines. It's actually pretty cool when you think about it.

At the heart of this is that many people have lots of ideas about how societies work. For instance, as you'll find in the next chapter, game theory researchers think in terms of how rational agents behave in specific situations, Ostrom thought about how people come together in resource management contexts, Gambetta uses trust to think about things like cooperation among people in different groups, with different information, and so on. It is not always evident, despite the fact that some of these theories are based on careful observation or experimentation, that they actually might be relevant or useful (or powerful) in other social settings. So, one way to find out is to build one of those social settings and find out what happens. How does one do that? With plenty of coding and plenty of thinking about how to take an abstract theory and make it implementable so that you can encode it into the agents that you want to use the theory.

In turns, Pitt and colleagues have examined trust, forgiveness, justice, fairness, scarcity and much more. His work is far-reaching and his methodology extremely powerful.

I am by no means doing justice to Pitt's work here. In fact, to do that, you'd need a whole book, so it is indeed fortunate that there is one! There are also extensive publications for you to look at in the further reading section! And to make things even more worthwhile, I was able to interview him and he has given permission to use the result here in this book. To watch it (I highly recommend you do…) head to the website for this book.

We're about at the end of our pioneers for now. However, the interesting thing about pioneers in a research field is that they're always popping up. If the research field is living, there will be moments in it where everyone should stand up and take notice. That makes this chapter, perhaps even more than the rest of the book, a living document.

In other words, watch this space.

Meanwhile, it's time to talk about Prisoners, in what could be another chapter but could as easily just belong in this one. It's your call.

WHATEVER HAPPENED TO THE PRISONERS?

Trust is something that you give. It is a gift.

Nobody can make you trust them.

If anyone demands that you trust them, it's probably a good clue that you shouldn't trust them at all.

Introduction

Pay attention[1]. You may need this later.

It's sort of about games. It's also sort of about how people make decisions and how, regardless of what scientists, economists, game theorists (or just plain anything-ists) might like, we're an irrational sort. Actually, that's kind of important because trust, you see, is not always a rational thing. Let's see now, it all started with Chuck (Chuck is the one with the mask!) and his co-conspirator Eve getting caught.

[1] Have you ever wondered why people say "Pay Attention"? It's because attention is a finite resource and it is worth something. When you use it, you are actually paying the person or thing with something you have only a finite amount of...

Figure 3.1: Chuck. You know, the bad one…

Read on…

‘Allo ‘Allo

(Not the TV show, which was good, but what the police officer says…) Chuck and Eve were out on their usual nefarious business being less than good. It so happened that Officer Crabtree, our police officer, was following them. Naturally they got caught. That’s how it works.

Figure 3.2: Officer Crabtree (you know, the one who is not bad).

Now Officer Crabtree chats with his captain, who says, “This pair have done a few heists. Make ‘em squeal” (that’s how police captains speak). They’ve got our pair of criminals bang to rights on this heist so they use it as leverage. Let’s see how *that* works.

Chuck is told, “We know you did other things, tell us what you know about Eve’s involvement and you can go free!” (A plea bargain no less.) Naturally there’s an ‘or else’ which goes like this: “If you stay quiet, Eve will rat on you and you go down for the lot whilst Eve goes free.”

Eve gets the same ultimatums.

To translate a bit: Both get told they can rat or stay quiet and the results will depend on what the other said. Trouble is, they can’t talk to each other and make a pact.

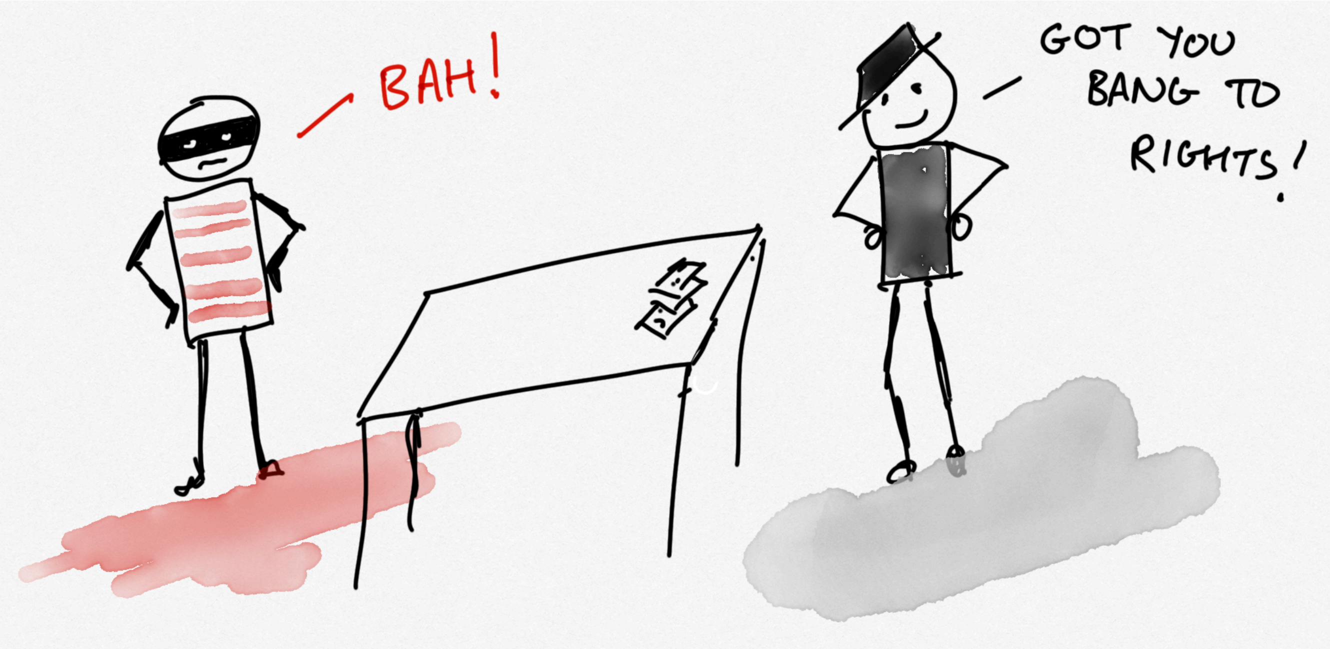


Figure 3.3: Bang to Rights...

It’s a dilemma.

The Prisoner’s Dilemma

[2].

To recap, Chuck and Eve can’t talk to each other and they’re faced with a choice – rat or stay quiet. Each can’t tell what the other will do until it’s done.

[2] Okay, so there are two prisoners, so it probably should be The Prisoners' Dilemma, but if I typed that purists would undoubtedly shout at me, and anyway, the dilemma is that of Chuck, or Eve, who are singular, if you see what I mean.

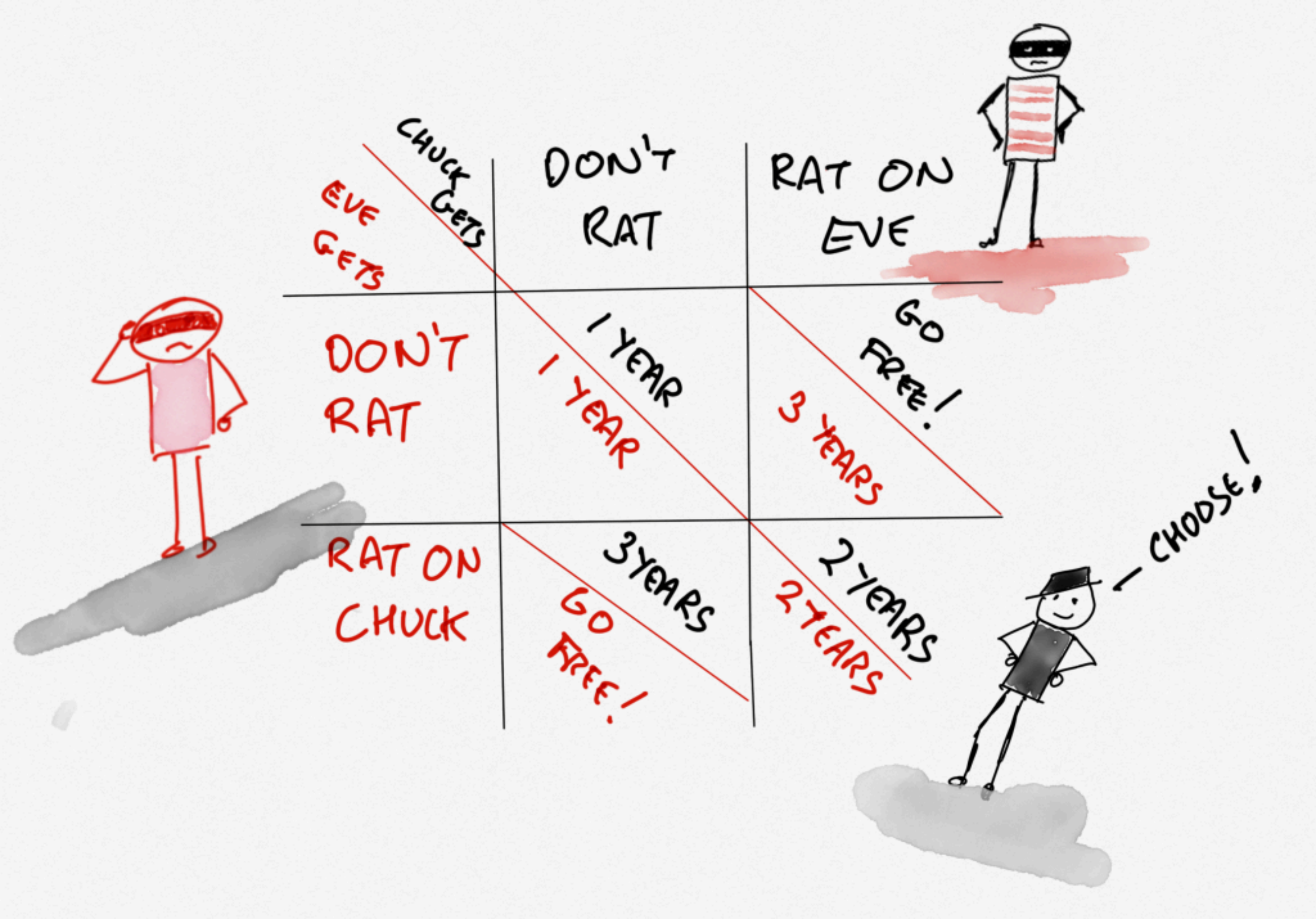
Well, what would *you* do? Be honest, there's no-one here but us. To make it more interesting, let's add a little more nuance.

Figure 3.4: It's simply not right...

If Chuck rats on Eve and Eve stays quiet, Chuck goes free; but Eve, who has now had a bunch more bad stuff pinned on them, has to serve three years. The same would happen to Chuck if Chuck stayed quiet and Eve ratted. There's more. If Chuck and Eve both stay quiet, they'll serve the one year that they would get anyway for the heist they got caught in. And if they both rat on each other, they get to spend two years in jail for admitting to more than the heist they got caught in. I hope that makes sense.

This is a pretty pickle, and can be represented by a 'matrix' of sorts. There it is look![3]

[3] (I knew these drawings would come in handy)



CHUCK GETS EVE GETS	DON'T RAT	RAT ON EVE
DON'T RAT	1 YEAR 1 YEAR	GO FREE! 3 YEARS
RAT ON CHUCK	3YEARS GO FREE!	2 YEARS 2 YEARS

Figure 3.5: The Prisoner's Dilemma Matrix

A little thought will suggest to you that the best both could do was to stay quiet and hope the other does too. If they get ratted on, though, they will get three years! So that's not a good thing, right? So maybe they should just spill the beans! (But then they'd get two years instead of the one they would've got if they'd stayed quiet! Free is good. Jail is not.

Figure 3.6: It is what it is...

Let's Make It a Game!

The thing is, Eve and Chuck just have to make this choice once. What happens if the choice happens again? And again? And again?

It's kind of like this: if they both know how many times they'll have to make the choice, they'll both just rat on each other every time because, well, they're going to rat in the last stage and so they will have to rat on the previous one, and so on. See how that works?

> Men are not prisoners of fate, but only prisoners of their own minds.
>
> Franklin D. Roosevelt

It becomes much more interesting if they don't know how many 'rounds' they will have to 'play'. At that point they can't be sure what's going to happen and so perhaps the best strategy is to stay quiet? Imagine if the choices were for money instead of jail time. For this, let me introduce Alex[4], our game show host. Hello, Alex. Now in our game the competitors get a similar choice to the crooks: cooperate (stay quiet) or defect (squeal). If they both cooperate they get a **Reward** for cooperation, which is, say $3. If they both defect they

[4] Alex Trebek, in memoriam.

get a **Punishment**, still a bit of money but just $1. If one cooperates whilst the other defects then the one who cooperated gets a **Suckers payoff**. Basically nothing. And the other? They get the biggest money – a $5 **Temptation** to defec**t**. See those bold letters? Good.

If you've been paying attention, you'll have noticed that those red, underlined letters are worth something. For this thing to be a Prisoner's Dilemma, they need to be arranged in a particular way: T>R>P>S. You can see that in the picture (I hope!) You can make all kinds of games around this and the dilemma remains. It's pretty neat (and simple)

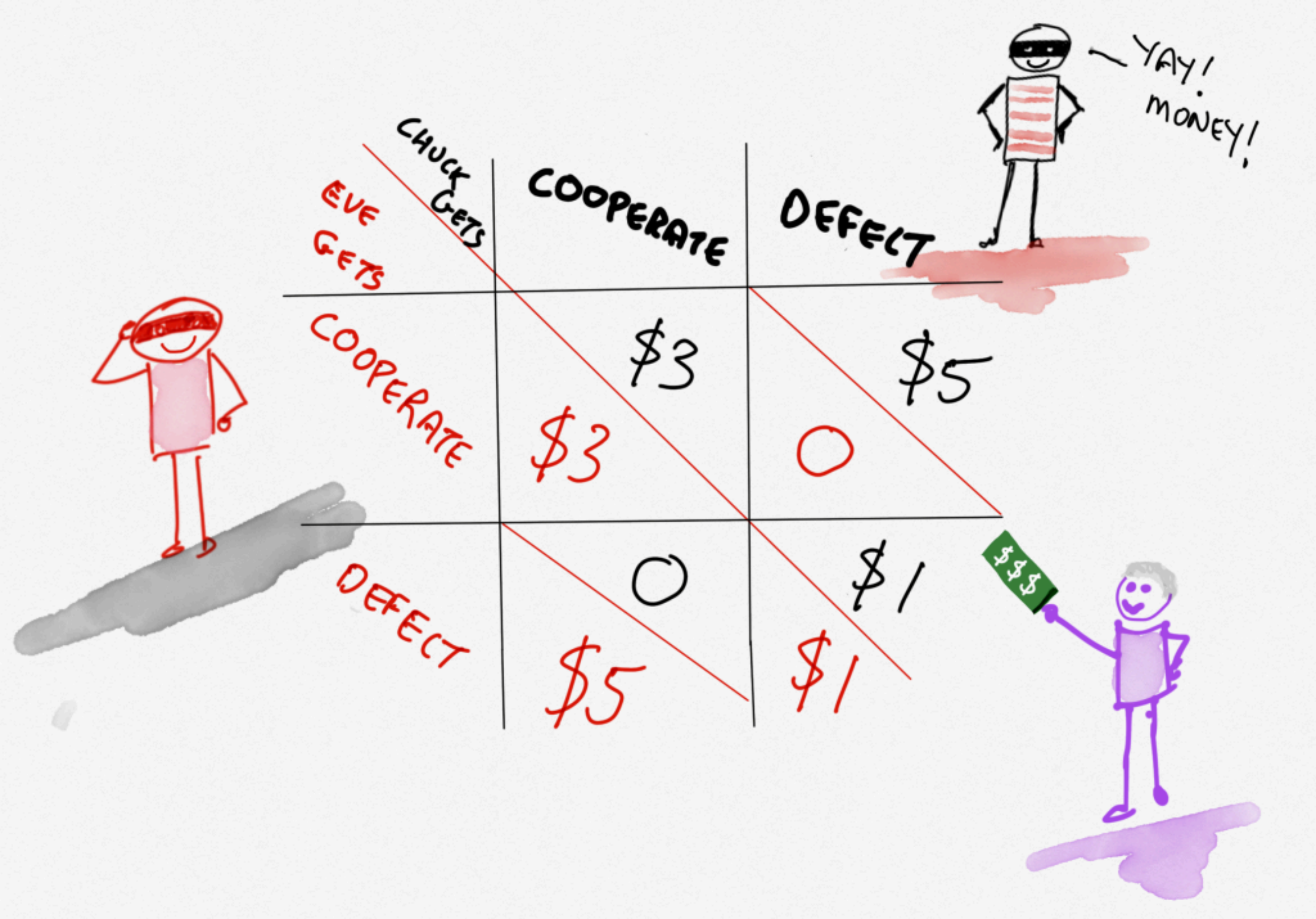
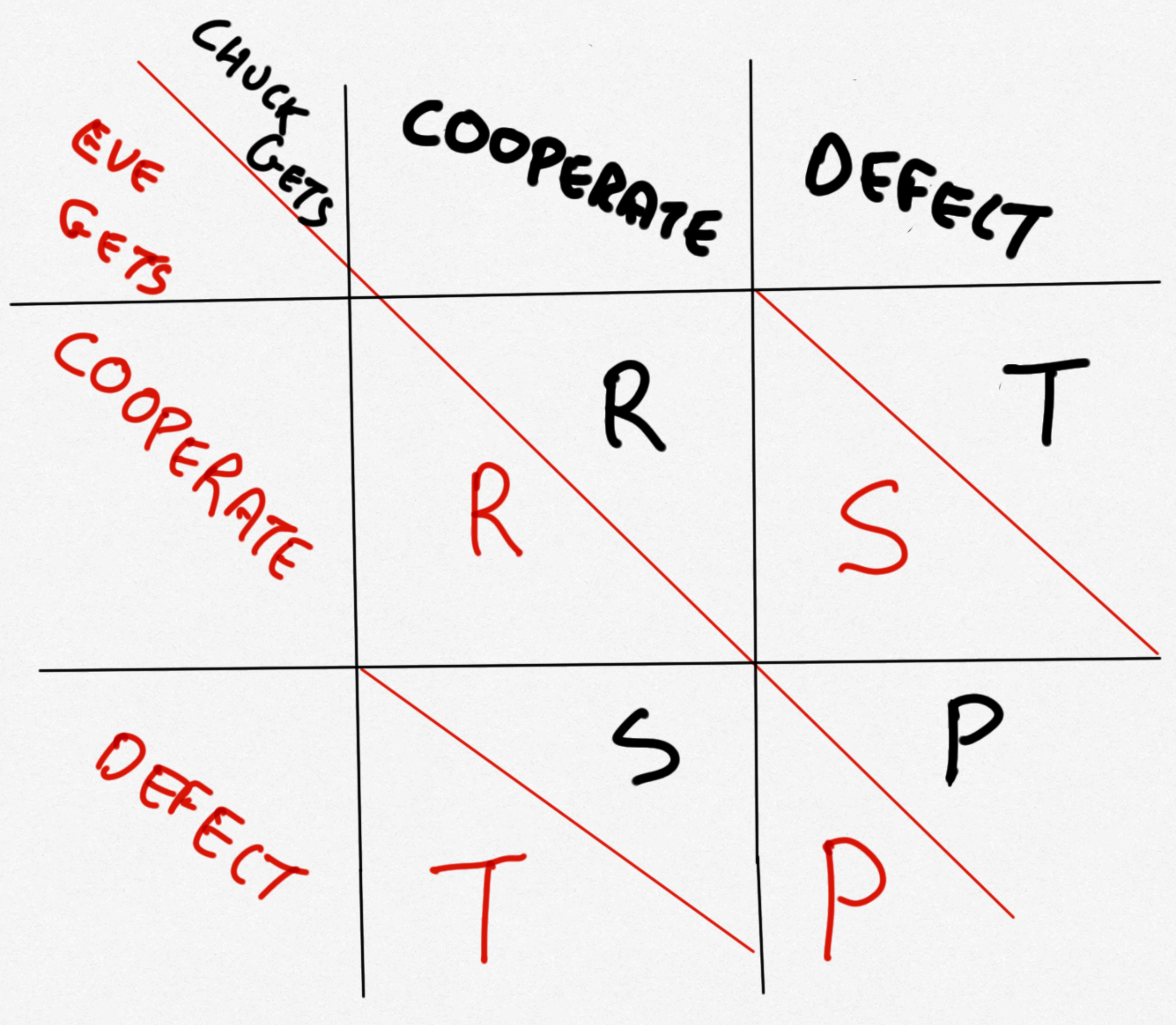


Figure 3.7: And so, it comes to be a game



CHUCK GETS EVE GETS	COOPERATE	DEFECT
COOPERATE	R R	T S
DEFECT	S T	P P

Figure 3.8: The Prisoner's Dilemma Matrix, with letters.

The Prisoner's Dilemma (PD) has been the source of many different studies since its inception in the 1950s by two RAND researchers: Merrill Flood and Melvin Dresher, and its subsequent codification and description (using the prisoners we know today) by Canadian mathematician, Albert Tucker. To bring it to our more recent past, let's talk about a couple of tournaments run by Robert Axelrod in the 1980s (Axelrod, 1984). Axelrod is a political scientist and was examining the dilemma as a kind of lens for looking at how decisions can get made in the context of international relations (it's a long story and has roots in the Cuban Missile Crisis — if you're interested, here is a very nice telling of the story). It's worth noting that Flood and Dresher didn't 'invent' the Dilemma – it's always been there, but they identified how it kind of works (and that T>R>P>S relationship. Actually, it happened to be a really good explanation for mutually assured destruction (MAD) and the Cuban Missile Crisis (CES, 2019).

Before we go too much further and people get all excited about games and Game Theory–yes, it is neat and yes, we can explain or even possibly try to predict behaviours (or maybe shape them the way we want). But people aren't rational and people don't behave according to simple assumptions. Sure, you can definitely use it to generalize and even study over vast quantities of people and economic behaviour, and it has been done. I have always thought of it a little like Asimov's (Seldon's) psychohistory – it works, but there are some assumptions and restrictions. To put it another way, "Applied in the right way— by understanding the motivations of those who participate and the social norms that guide them — game theory can successfully predict how people will behave and, similarly, its techniques can help in economic system design." (Wooldridge, 2012). The trick? understanding[5]

Anyway, back to Axelrod. He built a computer program that ran a tournament for the PD which would face off different strategies against each other for a number of rounds (the number being unknown to the strategies). He asked several notable social scientists to submit code for the strategy they would like to run and pit them all against each other in a kind of round robin tournament.

The results were surprising: the strategy that did the best, submitted by a social scientist and peace studies researcher, Anatol Rapoport[6]. It goes like this: cooperate first round, then do whatever your opponent does on the previous round for your next decision. So, it was nice: it cooperated first, but if you ever defected against it, it would assuredly defect back (which made it provokable). It was also forgiving because, well, if you subsequently cooperated, it would too. Tit for tat. Nice, provokable and forgiving.

Axelrod, being on to something, then put ads in various journals for people to submit their own strategies, telling them which one had won the previous tournament. Lots of very complex and sneaky algorithms were submitted, the tournament went again and... tit for tat won again. Pretty neat eh? What is the take-away from all this? Well, being nice, provokable and forgiving is pretty good.

[5] It may appear at this point as if I am 'against' Game Theory in some way. Nothing could be farther from the truth. A healthy amount of scepticism is not always a bad thing. Do your research, properly. Having undoubtedly upset some people I move on.

[6] So, another peace researcher, like Morton Deutsch. Maybe there is something to thinking about peace, after all...?

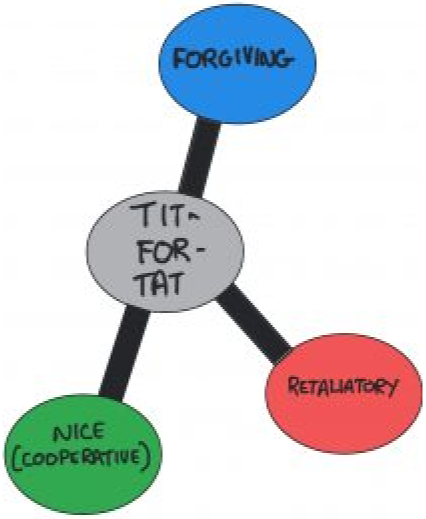


Figure 3.9: Tit for Tat – Nice, Forgiving and Retaliatory (Provokable)

Finally, it became very clear to other strategies exactly what tit for tat would do next: it was predictable and clear in its intentions.

What does this mean for cooperation? What, in fact, does it mean for trust, since this is a book about trust?

It's like this: tit for tat never actually won any of its games, if winning means 'got the most points,' it just pushed every other strategy to do better. Why? Because it basically encouraged cooperation. You can never do better than cooperating against tit for tat (unless of course you came across a complete sucker who always cooperated no matter what you did!) But more to the point, strategies playing against it began to cooperate because it was the sensible thing to do. It's a pretty neat thing and Axelrod went on to do a whole bunch of research into tit for tat in many different fields, from political science to molecular biology.

If Tit for Tat encourages cooperation, what does that tell us about trust? On a simple level, it tells us that if you are clear in your intentions and you always do what you say you will, it becomes pretty easy to trust you in that particular context. If I know you'll poke my eye out if I poke yours, I probably won't try to poke yours out… And so, it's not necessarily trust in the other (it probably goes without saying that the USSR and the US didn't trust each other) but trust that you will do as you say ("Bomb us, and we will bomb you back to the Stone Age").

I mentioned earlier in the book that trust is contextual. It is. It's also very sensitive to the what for question. What is one agent trusting another for? What is it they are trusted to actually do? Here, Tit for tat shows us that we can trust it to do exactly what it says it will. That's an important thing. We can take this into a different direction and think about why cooperation happens. Why do, for example, some animals help each other out? Are they just being kind? Let's dive into evolution and stuff for a little bit.

Hamilton's thought (Hamilton, 1963; 1964) was that natural selection favours genetic survival, not, as you may have thought, a particular species' reproductive success (it's called Hamilton's Rule). So, because the genes 'want' to survive, animals in the same species who are also kin will help others (like vampire bats sharing blood with those who haven't fed that day – for an excellent discussion of which, see Harcourt, 1990). Altruism is about helping the genes survive at some cost to the individual, in other words. It becomes a bit mathematical and has to do with the number of shared genes and stuff but it's interesting explanation for some altruistic behaviour. Richard Dawkins took at look at that in The Selfish Gene (Dawkins, 1976[7]). So, altruism might actually evolve because it is good for the population, to put it (far too) simply.

This brings us to reciprocal altruism – doing something nice because you expect it will come back as something nice for you). It goes along the same kind of lines and is sort of explained by things like kin selection. But how does that all work in humans? After all, we do nice things for other people all the time (and sometimes not expecting anything back – call that 'real altruism'). One thing we might do is something nice in the expectation someone will do something nice back to us (call it 'calculated altruism'). In other words, there's some form of *quid pro quo*.

What has this got to do with trust? Think about it this way: it's entirely possible you could do something really nice for someone at some cost to yourself in the expectation you'll eventually get a reward of some sort from them. There's a risk, of course, that they won't do anything for you at all. So in effect you are trusting them to be nice back. Trivers (1971) suggests that trust and trustworthiness (two distinctly different things) as well as distrust are actually ways of regulating what are called subtle cheaters.

It works sort of like this: trustworthiness is something that is exhibited by a trustworthy person, right? So if someone wants to be seen as trustworthy (and thus, get the nice things) they will behave in a trustworthy manner (by repaying the nice things). If they don't, they will become distrusted (because, well, nobody will do nice things for them if they're just nasty). And so it goes. Cheaters are encouraged to be trustworthy simply because being trustworthy gets nice things. Nevertheless, cheaters do exist, and there are plenty of strategies to manage this kind of regulation for cheaters, which we'll get to later in the book. Still, it sometimes does pay to be 'good.' It also sometimes pays to be able to look like you're good (see Bacharach & Gambetta, 2000).

Okay, so the Prisoner's Dilemma is all nice and everything, and in a while we'll come back to it, because as I've mentioned before Deutsch did a bunch of research with it too, back in the day. But before we go there, let's visit the Commons.

The Tragedy of the Commons

I've already mentioned Hobbes and his famous *Leviathan*. Well, to be fair I've mentioned Hobbes and quoted

[7] Dawkins, R., The Selfish Gene. Oxford: Oxford University Press. 1976. ISBN 978-0-19-286092-7

from *Leviathan* (Hobbes, 1651) (that bit about nasty, brutish and short) which was really arguing that people need governments to be able to be 'good' because governments could punish people who were 'bad'[8]. Well that's great, you think, but I want to take you to another kind of argument, and to do that, we need to think about what is called *The Tragedy of the Commons*. It was brought forward in a seminal paper by Garrett Hardin. Imagine if you will…

> "Picture a pasture open to all. It is to be expected that each herdsman will try to keep as many cattle as possible on the commons… But this is the conclusion reached by each and every rational herdsman sharing the commons. Therein is the tragedy. Each man is locked into a system that compels him to increase his herd without limit—in a world that is limited. Ruin is the destination toward which all men rush, each pursuing his own best interest in a society that believes in the freedom of the commons." (Hardin, 1968, page 1224)

Not clear enough? Try this: in a situation where some kind of resource is shared, it just makes sense for individuals to try to use as much of it as possible. That's a problem because doing that is bad for everyone as a whole. This relates to the Prisoner's Dilemma right? How? Because once again we are asked to believe that people are rational. Read on.

There may well be a way around this problem, and in Hardin's paper the suggestions are privatization or government regulation. Both of these can work because all of a sudden the private individuals want the resource to last in the first case, or the government can punish them if they don't in the second case (remember Leviathan?). To put it another way, humankind is basically screwed because it's too self-interested and overexploits its resources.

Okay then, this of course means we're basically not going to survive. At this point, we can either pack up and go home or turn to Elinor Ostrom.

[8] See also System Trust.

Figure 3.10: Elinor Ostrom.

Ostrom was a political economist, and one of only two women ever to win the Nobel Prize for Economics for the work she did on the Commons, which may well save us all, or at the very least provides a path.

Governing the Commons

It goes like this: understand institutional diversity. “What are you talking about?” I hear you say.

Let me explain. Sure, you can privatize resources, but in the end, there are always people who will try to rob you. You can rely on governments, and sure, that can work if the government itself understands what it is trying to protect. There are examples of where these kinds of solutions just don’t work (fisheries, for example) for various reasons.

But Ostrom in her research found that there are examples of where the ‘Commons’ was a shared (private to the individuals sharing it) resource and was sustainably managed, harvested, or whatever. By the very people who used it. How can this be? Because the people who were using that resource had recognized that the answer to the question of shared resources was not ‘it’s this or that’ but in fact was ‘it’s complex’. That means that they figured out that the resource they were sharing was a complex system within a set of other complex systems and that it needed a really complex set of rules to ‘govern’ (and thus share) it. There are examples of this from lobster fishers off the coast of Maine to Masai tribesmen in the bush grazing cattle.

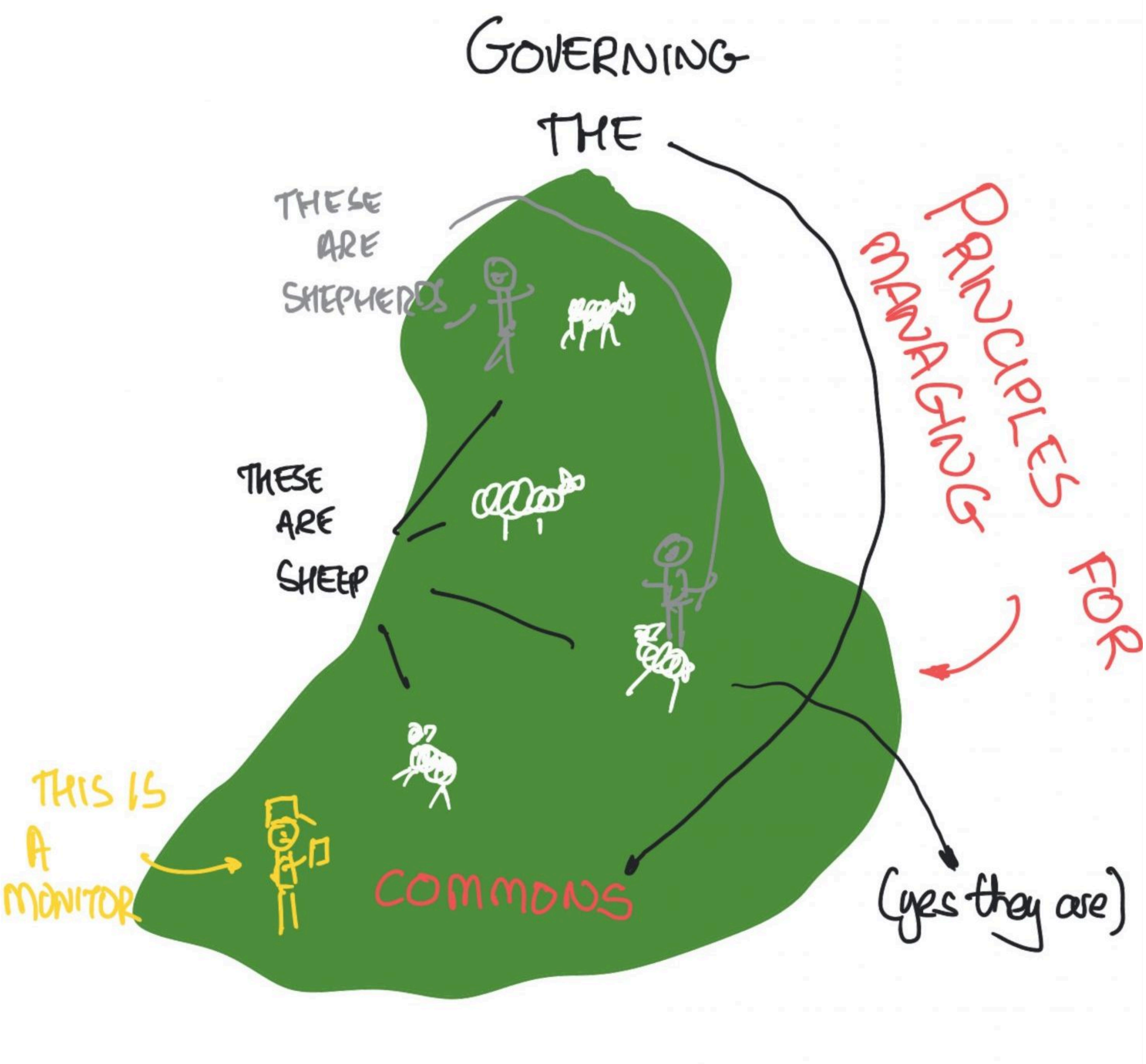


Figure 3.11: The Commons

What have we learned here? That complex problems sometimes involve complex solutions, and, well, simple games aren't always the answer.

To paraphrase what Ostrom found in two words, in one sense we are talking about sustainable development. Or perhaps we are talking about managed commons. And did you notice something else? It comes down to the trust that the people in the local system have in each other, because they need to trust each other to make this work!

Well, *sort of*. In fact, Ostrom introduced a set of 'design principles' for how to manage a commons (technically a Common Pool Resource (Commons), but you get the idea). The book is a little academic, so I'll do my best to do the ideas justice here. I'll also perhaps order them in a way different than you may see them elsewhere. That's okay – as long as the eight are there, you'll do fine.

Perhaps the most important thing to remember is that there is a resource (the Commons) and there is a group of people who want to (or must) share it for their own well-being. Once you realize that, the first of the principles becomes clear: there has to be a clearly defined group and a clearly defined Commons, so that everyone knows within the group that they are in it, that others are not, and what it is they are making rules about for sharing. Because this is all about rules: how to use, who can use when, how much to use, and so on. So the second principle is to match the right rules to match the local conditions that the Commons is found in. It may be, for instance, that even though the Commons is the same between different groups (land, say, or water) the local conditions can be very different. When the rules are matched, things are good.

This brings us to the rules themselves, because the third principle is that the rules are made, and can be changed, by the group of people who are using the Commons.

Why does this matter? Because the people using the Commons are the people who are local enough to be able to understand what works. They're also the ones 'on the ground', as it were, and so the fourth principle comes into play: there needs to be a system for monitoring how the rules are followed – this can be amongst the people using the Commons or some third party – and we'll talk more about monitoring in a bit. But when you monitor, you may well find that some people are breaking the rules and using just that little bit more than others, for instance. In that case, there needs to be some form of punishment (Ostrom calls them sanctions) for breaking the rules. The people who do the punishing may be the people who use the Commons and set the rules, but they could be some other third party again, which is fine as long as they answer to the people who use the Commons. I'm sure you can understand why – after all, if the punishers are not accountable, they could do whatever they wanted with impunity[9]. For the fifth principle, it is important to note that these sanctions should be gradual in some way. Sometimes a slap on the wrist, or diminished reputation, is enough to bring people back in line, especially if the infraction was small. Bigger, or more prolonged bad behaviours likely need gradually bigger punishments. You get the idea.

Now it gets even more interesting because, well, sometimes people disagree on what is happening and that's no good! So, there needs to be some form of conflict resolution system that everyone agrees to. When we say everyone we mean both the people using the Commons and, if there are any, the third parties that do the monitoring and sanctioning – that's because there may well be times when they disagree too. So, the sixth principle talks about how there need to be fair and agreed upon ways of resolving these disagreements. The emphasis is on purpose. If people start thinking that things aren't fair, they're going to stop trying to care.

Of course, we're usually talking about a Commons that exists inside, say, a country, and isn't the whole country. Clearly, the people who use the Commons are probably not the leaders of the country! The thing is, the leaders of the country (or what Ostrom called 'external authorities' – so it may be the local council or

[9] See also here.

the provincial or state government, and so on) need to agree that the people using the Commons know best, ultimately, and are allowed to makes these rules and choose sanctions, and so on. This is the seventh principle, and if they decide that this isn't allowed, usually things go bad because in general they don't understand how the Commons itself is used – and so at least principles 2 and 3 are not followed. Usually this results in the Commons being over-exploited and, ultimately, early lost. A whole chapter in Ostrom's book is devoted to these sadly avoidable instances.

Still, this is important: there are outside authorities, and more often than not, they have their own sets of rules and are themselves part of bigger conglomerates: cities, councils, provinces, countries, and so on. Each of these higher levels may actually use or benefit from things that are associated at least with the Commons in the first level (like streams flow into rivers, or rivers to the sea, and what happens to one can seriously affect the others. And they need their own rules for how these are managed. So the eighth principle is that this is complex and that to be successful you need to have nested levels of complexity from the initial Commons all the way up. Rules that are missing or inappropriate in any level will seriously limit the ability of other levels to be successful.

Well that sounds just fine, and I think we can (mostly) all agree that sustainability is a wonderful thing. But what has this to do with trust?

Okay then, let's see. First, the people need to trust each other enough to come together and make some rules. Then we get to the monitoring and sanctions bits, and it gets really interesting. If the people utilizing the Commons don't trust each other, or think there could be infractions, they need to spend a lot of time and effort monitoring to make sure everyone is behaving. Okay then, sometimes this is too much for everyone concerned – they have their own jobs to do, after all! So, as we noted, third parties can be used to monitor and sanction. "*Now then,*" you may be thinking, "*that's all very well, but you just said monitoring* costs *something. People won't do it for nothing.*"

This is true. There are usually rewards (prestige, goods, money) for being a monitor. Ostrom notes though that "the appropriators tend to keep monitoring the guards, as well as each other" (Ostrom, 1990, p.96). This is an important point in our own narrative, because it implies that there needs to be some form of trust between people and guards and vice versa. The less the trust, the more the monitoring (which costs you!). You have to trust the system to make sure that it will enforce its rules.

And then, there needs to be trust between the external authorities and the nested systems within, all the way down to our original common resource and back again. Trusting the authorities to do the right thing (even if the authorities are those who sanction in our name) or the system to do the job of trust for you (because it ensures the rules are followed) is what we call system trust, and it's pretty important notion that we'll revisit later in the book.

It should be pretty clear by now that trust is a central component of the ways in which these resources are managed. Interestingly enough, trust itself can be seen as a Commons, and exploited in various ways – and people often talk about how trust gets used up or destroyed by things that happen or systems that exploit it. We'll come back to that too.

Can we do this without trust, though? I'm glad you asked, because in another excellent book, which is part of the Russell Sage Foundation series on trust, the authors argue that we can. The book, *Cooperation without Trust?* (Cook et al, 1995), argues that there are institutions, practices, norms and so on that we put in place to be able to manage in a society without having to trust those we are dealing with. This may well be so: we have legal eagles looking out for us all the time, and police monitoring us, courts to impose sanctions, and so on. The question is, what happens when you stop trusting them?

I said earlier that trust was important in the systems Ostrom explored, and I hope I've made it clear where it can be important, whilst (with the help of Cook et al.) it's also possible to look at it from the point of view that trust is at least sidelined by the systems that are put in place to enable people to be able to go through their daily lives, commerce, and so on, without thinking about trust all the time. It's complicated, you see.

What's really neat about all of this is that it has been taken (the principles) and embedded into computational structures to examine, for example, what rules for monitoring might be decided upon in self-organizing systems, which are usually computational systems which are sort of distributed, and sometimes (often) without some form of central control. Sometimes we might have ad hoc networks, for example, or a multi-agent system (MAS). An important body of work leverages Ostrom's principles into these systems.

And that brings us all the way back to Jeremy Pitt. For whom, see here!

But it doesn't stop there. Up to here I've brought you through some of the thoughts of people who have been thinking about this for a long time. However, I did say at the start of the book that it is about **computational trust**. And so it is, which is why we didn't dive *too* deep into the mists of trusty time – don't worry, there's plenty of space left for the second edition! The thing is, as with ice-nine, the things I've talked about so far are the *seed* of the computational side of things: look at different pioneers and you'll likely get a far different form of computational trust.

Because there are far different forms from the computational trust that this book focuses on.

Without further ado then, let's consider: for something to be computational possible, it must be 'reduced'[10] to a state that makes sense to computers. And to do that, it is necessary to figure out whether or not you can actually *see* it enough to reduce it properly. And that's what the next chapter is about.

[10] I say reduced, but I mean 'moulded.'

CAN WE SEE IT?

Introduction

One of the interesting things about trust is that you can't see it. I like to compare it to the wind, or to light. You can only see these things by the effect that they have: trees bend (and if they grow in places where the prevailing wind is pretty constant, they grow sideways). You don't see the light traveling through the air, or a vacuum for that matter. You only see light when it hits something–the ground, a leaf, particles of smoke. Trust is a lot like that.

You can't explicitly see trust between two people. You can't see trust between a person and a computer. You can't see trust between a person and a dog. What you can see is the effect that trust has on the agents in the interaction. There is a tangible difference between a scene with two people who trust each other talking to one another, over one where the people don't trust each other. If you are in the same room you can feel the difference. You can feel when you are trusted by a dog, and the dog feels it back.

Look at this picture:

Figure 4.1: A rainbow over the trees at Lochinvar, Ontario. September 26th, 2019. (Picture: Steve Marsh)

You can see the rainbow (I had to stop to take the picture and it's a little blurry, but I had never seen a rainbow so red before). What's a rainbow? Light, refracted through water particles in the sky. Without the water you wouldn't see the light. Without trust, you don't see much light either.

While I was researching trust a long time ago I faced a quandary: how to represent it in a way that computers could actually use it. It occurred to me that numbers were probably a good thing. I was taken with Deutsch's work, which discussed subjective probability. What if you could represent trust that way, I wondered. Simple answer? Probably – after all, look back at Gambetta's definition.

More complicated answer? Maybe not. Trust *isn't* a probabilistic notion. It's not like the people in a trusting relationship get up every morning and flip a coin every time they want to make a move. So whilst there's a probability that someone might do something, it's not up to chance. Let's put it another way: when I'm in a relationship and considering someone being able to do something, whilst I might say that the person is "more likely than not" to do that thing, it's not like they roll a d100[1] and do it one day and not the next because the d100 rolled differently.

Bear this in mind. It's important, and we will come back to it.

The Continuum of Trust

So, using a 'sort of' probability is a relatively straightforward way to represent the phenomenon until we get all bogged down with the real way it works. For instance, you could represent trust on a scale from, say, 0 to 1, where 0 is no trust and 1 is full trust. More on that one later because, well, full trust isn't really trust. You'll see.

I worked with that for quite some time until it occurred to me that there was something I couldn't account for. What, I wondered, did **distrust** look like? Gambetta sees it as symmetrical to trust in his definition. Others (McKnight especially) see it as distinct from trust.

You see, distrust and trust are related but not the same. Whilst you might be thinking to yourself "Whatever. Stop sweating the small stuff. Make 0 to 0.5 distrust and 0.5 to 1 trust" I respectfully suggest that it doesn't work that way. Distrust is distrust, it's not a lesser value of trust.

But the two are related.

This is important. It took me a long time to work around being comfortable with keeping trust and distrust on the same scale, but I reasoned like this: The two things are related, and you can move from trust to distrust. You can even move from distrust to trust (it's harder, but you can do it). So whilst they are not the same they are connected. In the sense of a computational scale, I eventually settled on −1 to 0 as the range of distrust and 0 to 1 as trust. Yes, 0 is on both. IN effect, it's saying that we haven't got the information we need to make a judgment one way or another. Anyway, this separates the two yet keeps them related; it allowed me to think what I thought were interesting things about how the whole worked.

So that leaves us with a really neat continuum for "trust" with a range between −1 and +1. There's all kinds of stuff you can do with that. Not least, you can show it in a picture. As you might have spotted by now, I like pictures. Not because they say a thousand words, but because they can say the same thing differently. Looking at a problem from a different direction is the most valuable thing you can do with any problem (apart, of course, from solving it!).

[1] It's been pointed out to me that this is a reference that others may find a bit confusing. What is a d100? Well in role playing (you know, like *Dungeons and Dragons*... Where were you when they were screening *Stranger Things*?) we sometimes need a percentage, like in chance, for the probability that something happens. We can roll two dice, one which has faces of 00 to 90, and one which has 0 top 9 faces. Put them together (like a 40 and a 5) and you get 45. Simple. Except this has its own stupid debates too, by the way. Some people have too much time on their hands, or want to lecture too much. As an aside, if you're a computer scientist, starting at zero makes *perfect* sense.

Now I could make all kinds of nice lines to show what it looks like, but the spirit of this book is a little different from that, so Figure 4.2 shows what that looks like.

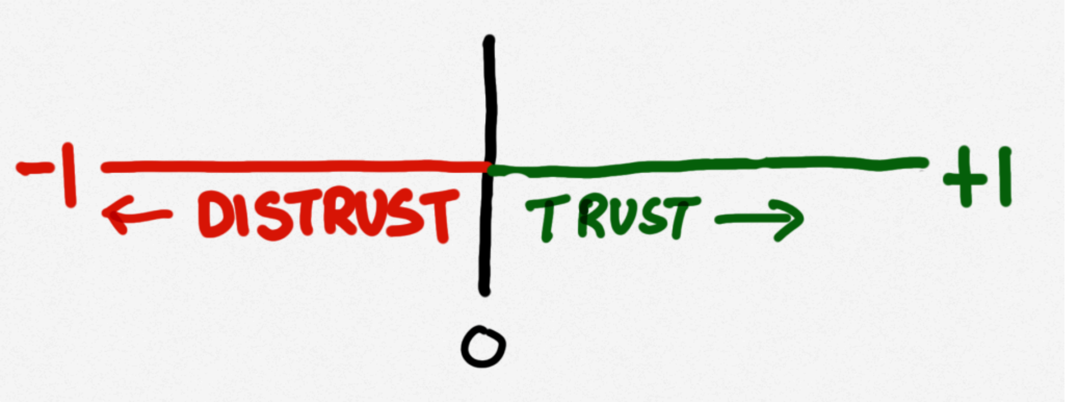


Figure 4.2: A continuum from distrust (-1) to trust (+1)

Pretty good, eh? Now we have a really neat tool to be able to explore trust. Don't believe me? It gets better, as you'll see. Firstly it allowed me to be able to think about that other problem I was having with trust. The first is that you can't see it. We still can't see it, but we can visualize it, and that's good.

The second was something I alluded to earlier. Remember when I started getting *D&D* geek and talking about a d100? Right, I said there was something really important there. This is where we tease that out.

You see, when we decide to trust someone to do something, we don't think in terms of probabilities. We trust them. That's it. When we decide to trust the babysitter, we don't run back from the movie theatre every minute to see if the probability of their behaving properly has changed. When we trust the brain surgeon to do the operation, we don't keep getting off the operating table to see if they are doing it right.

Let me digress a moment. One of the more fascinating things for me whilst at high school[2] was when our physics teacher, Mrs. Pearce, showed us how light behaves when you put a laser beam through prisms and through gratings (very, very, very thin slits in a piece of metal). The results were fantastic for little me (okay, I was 17): light behaved in different ways depending on how you looked at it.

Mind *blown*.

Now, it was a long time ago and I'm no quantum physicist (indeed, no physicist at all, as it turns out), but here's what I got out of of it: the phenomenon is called wave-particle duality and it goes something like this: photons (and electrons and other elementary particles) exhibit the properties of both waves and particles. Light, which is made up of photons, is a wave and a bunch of particles all at the same time. As a really simple explanation for a really complex phenomenon, that worked for me.

I like simple explanations for complex things. As you will see later in the book, I practically demand them.

Excuse the digression, but it's like this: trust behaves like light. Not just because you can't see it until

[2] I'm an Old Hancastrian as it happens, from one of the oldest schools in England

it interacts with something. It behaves like light because you can talk about it in two distinct ways. The continuum helped me to figure this one out better.

In the first instance, we reason with and about trust, trying to work out that 'probability' that the other person or thing will do what they say they are going to. But when we make a trusting decision, this reasoning is no longer valid. When we decide to trust someone in a specific context, trust isn't probabilistic, if it ever was: it's binary. We either do, or we don't.

It's the same trust. It just behaves differently.

How can we show that on a continuum? Well, it's like this: you can see trust as threshold-based. This means that when it reaches a certain value (if we put values on it) it crosses a threshold into the I-trust-you relationship from the I-sort-of-trust-you relationship. You can see the picture in Figure 4.3.

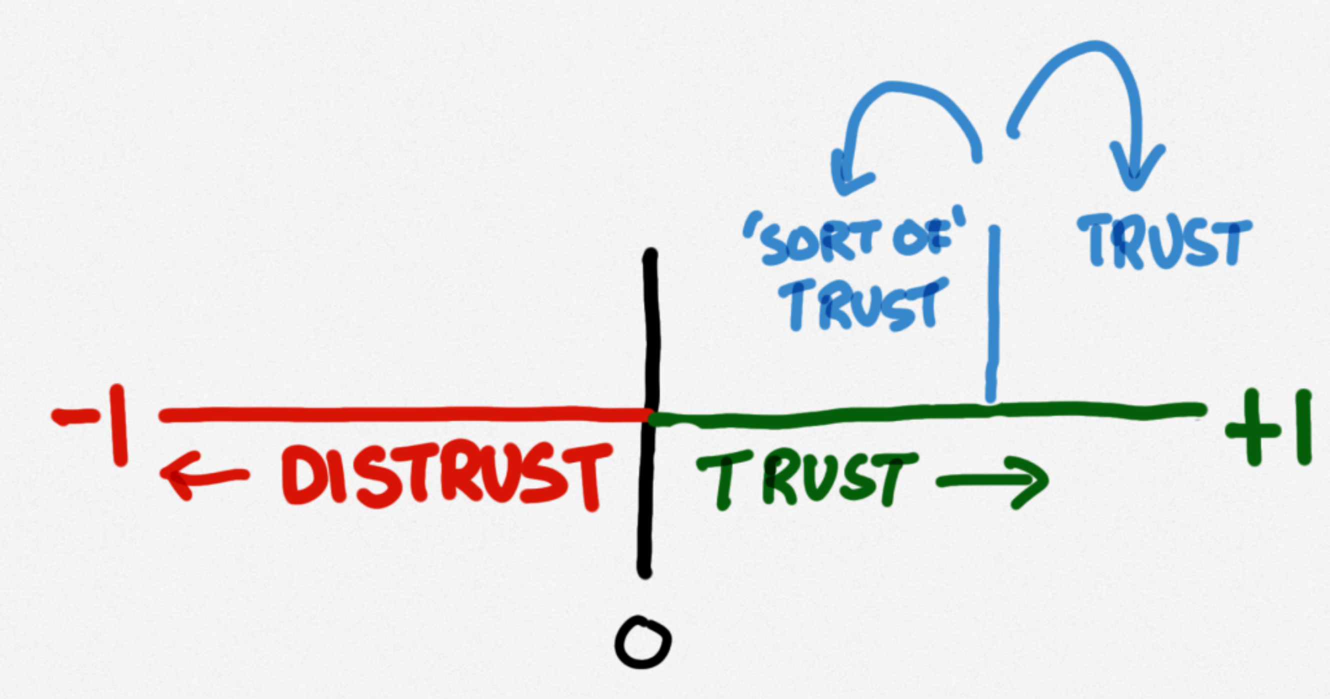


Figure 4.3: A graph depicting the Trust and 'Sort of' Trust sections on the Trust Continuum.

See how that works? Excellent!

Cooperation Threshold

Now, I was back then quite concerned with agents working together (basically, in a distributed artificial intelligence sense, although being a proud European I called it multi-agent systems. If you're not sure what either of those is, DuckDuckGo is always your friend!). This, I decided, was to be called "cooperation" and so that threshold became forever known to me as the "cooperation threshold" and so, the picture becomes what you see in Figure 4.4.

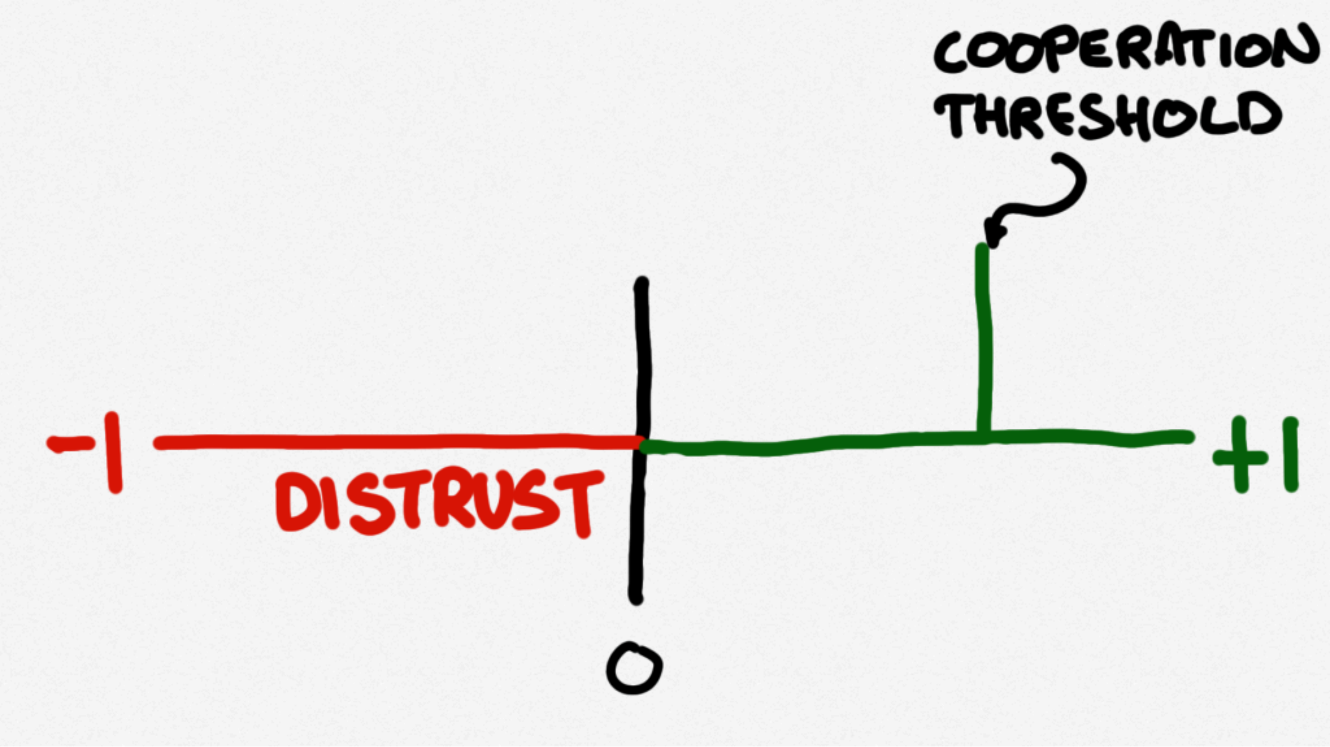


Figure 4.4: A graph depicting the position of The Cooperation Threshold on the Trust Continuum.

Above the cooperation threshold, you can say, “I trust you” (and the ‘what for’ is implied, to be honest). Below it, all sorts of really interesting things happen, which is wonderful.

Take a look at that picture in Figure 4.4 again. What do you see? Below the cooperation threshold we now have two distinct things (actually, we have three, but I will get to that). We have “sort of” trust, and we have distrust. At the time, that was fine – there’s only so much you can do for a PhD and to be honest, starting a new field is pretty cool, even though I didn’t know that at the time. I didn’t completely leave it though, as you’ll see in a wee while. In fact, I saw it as an ideal situation for thinking about things like system trust, the law, contracts, trustless systems, building trust and so on. We’ll get to more of this later in this chapter. Let’s not get sidetracked too much.

I should mention that as I was thinking about the continuum I was also thinking about how to calculate all this stuff: cooperation threshold, trust in context, all that kind of thing. It was a fun time. In the next chapter I’ll show you what I came up with and pair it with the continuum more.

Back to that gap. You know, the one that hangs out between zero and the cooperation threshold. It doesn’t seem right to say that the other person isn’t trusted in this context. After all, there is a positive amount of trust, yes? So it’s, like, trust, yes? That’s what I thought too.

And so, much later, a colleague, Mark Dibben and I tried to tease out what this all meant. When I say “much later,” I mean around 11 years in a paper at the iTrust conference (Marsh & Dibben, 2005). We decided to tease out that little gap and at the same time think about how the language around trust might be used to better define what the continuum was trying to show. I mean, definitely unfinished business.

We tackled distrust first because we had for some time been troubled by the use of "mistrust" and "distrust" as synonymous. After all, firstly why have two words for the same thing and secondly, possibly more importantly, it is not how the "mis" and "dis" prefixes work in other words.

We took our inspiration from information. When you consider that word things start to make much more sense. When we put a "dis" at the front of it, we get "disinformation", which can be defined as information someone gives to you in order to make you believe something that is not true. As I write, there's been a lot of that done recently by people who should know better but clearly don't care. Regardless, disinformation is a purposeful thing. However, if we put a "mis" in front of information we get "misinformation" which is defined more as mistaken information being passed around, sort of like a nasty meme. To put it another way, you can look at misinformation as stuff people believe and tell others with the best of intentions, and not to mislead them (and there's another interesting "mis" word).

So, disinformation is purposeful, misinformation is mistaken, an accident if you like. Disinformation clearly leads to misinformation – in fact, you might say that misinformation is the purpose of disinformation.

Back to trust. We decided that this was a really nice way for us to think about trust. Distrust, which we really already have, is an active decision not to trust someone, in fact to expect them to behave against our best interests. Fair enough, that's pretty much where we already were. However, this allows us to look at mistrust as a mistaken decision to trust or not. In other words, it is something that exists across the whole continuum.

To explain a little more, think about Bob and Chuck, our poor, overworked security personas. Chuck is entirely naughty and untrustworthy, and will always do bad things. Bob does not know this and decides to trust Chuck to post a letter, say. Chuck happily opens it up, reads the content, steals the money that was in it and throws the rest into a ditch. The end. In this instance, we would say that Bob made a mistrustful decision: it was a mistake. But it was still trust.

Let's now take Bob and Derek. Derek is everyone's idea of a wonderful guy, entirely trustworthy and would do anything for anyone. However, Bob takes a dislike to him immediately (I don't know why, perhaps Derek has nicer teeth) and decides to distrust Derek, never giving him a chance to prove his trustworthiness. Guess what? In this instance, Bob also made a mistrustful decision: it was a mistake, but in this case it was to distrust instead of trust.

This frees up distrust to do what it was supposed to do in the first place: represent that belief about someone we perceive as not having our best interests at heart. So Bob definitely distrusts Derek, but this is a mistake, which means he mistrusts Derek too. Similarly Bob trusts Chuck, but this is also a mistake, which means he mistrusts Chuck.

It takes a while, give it time.

Alright, so now we can look at our continuum again! Guess what? Nothing changed! (Except that anyplace on it could be a mistrustful decision). That's a bit boring and it really doesn't answer the question: what's that bit between zero and the cooperation threshold?

Fortunately we thought of that too. There's a clue earlier. I called Chuck untrustworthy which means that it's probably not a good idea for Bob to put his trust in Chuck. That prefix, "un" means something.

If we go back to our contextual thoughts of trust, we start thinking about this: Steve can't do brain surgery, but he can cook a mean saag paneer[3]. In the context of doing brain surgery, I'm untrustworthy[4]. In the context of cooking saag paneer, I'm trustworthy. Now, you were to ask, say, my partner Patricia if she trusts me there would (I hope!) be a positive answer. But she still wouldn't trust me to do that brain surgery, right? Saag paneer now, oh sure, I can make that any time.

There's a couple of really nice pictures around here that show what I mean. They're Figures 4.5 and 4.6. I'm sure you will find them.

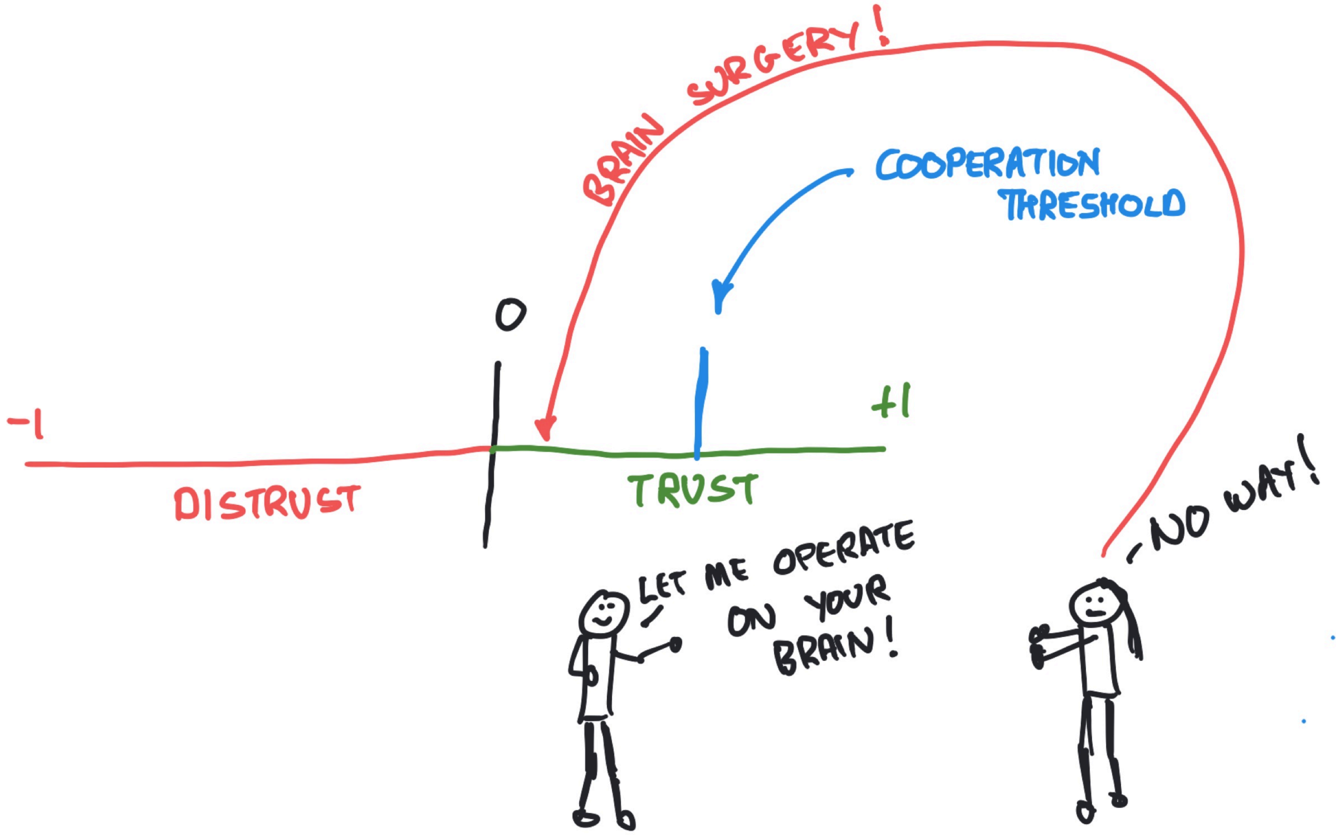


Figure 4.5: A graph depicting an example scenario of what might be below the Cooperation Threshold on the Trust Continuum.

[3] It's actually true. It is also an example you will see overworked in another chapter…

[4] You could say I'm untrustworthy if I wanted to try to do it, anyway. If I don't ask you to trust me to do it, am I still untrustworthy?

Untrust

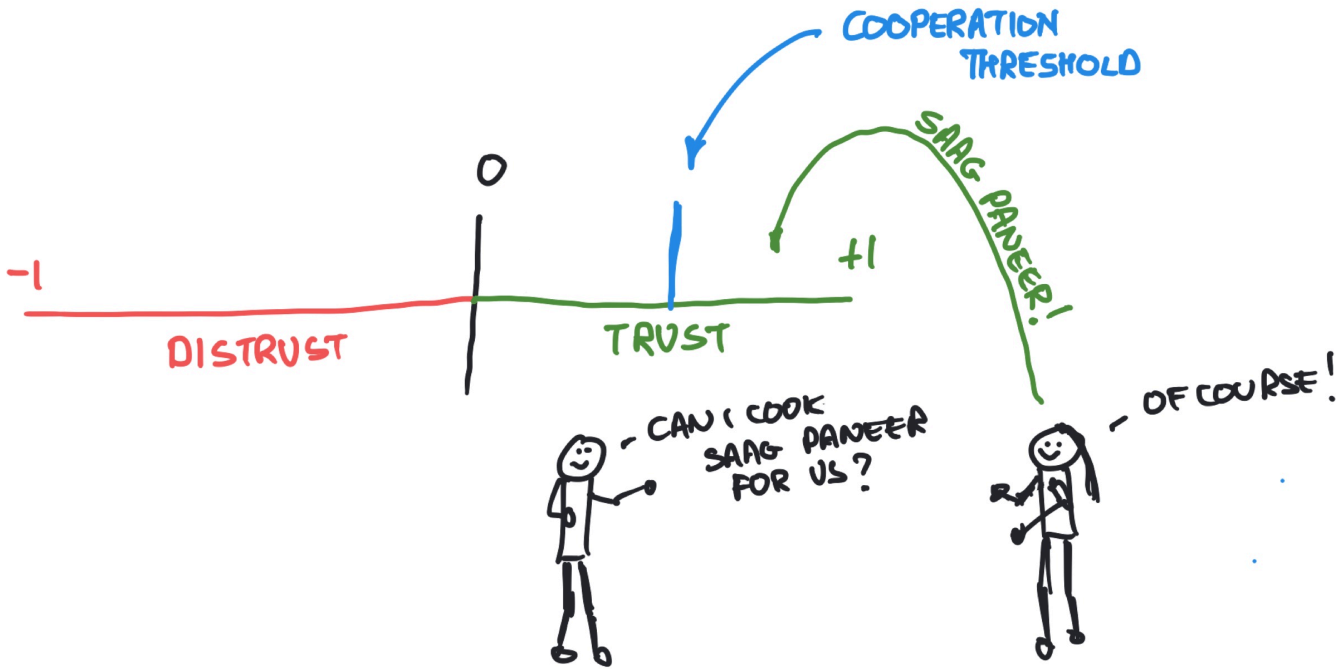


Figure 4.6: A graph depicting an example scenario of when trust is above the Cooperation Threshold on the Trust Continuum.

Alright. See how Patricia still trusts me in the brain surgery thing (I probably wouldn't but there you are, all analogies break down somewhere. Mine is people messing about with my brain). Right, there's trust, but certainly not enough to allow what I'm asking in that context. That is the 'sort of' trust which Mark and I decided to call "untrust," taking a cue from untrustworthy, you see? And Figure 4.7 shows you where it lies.

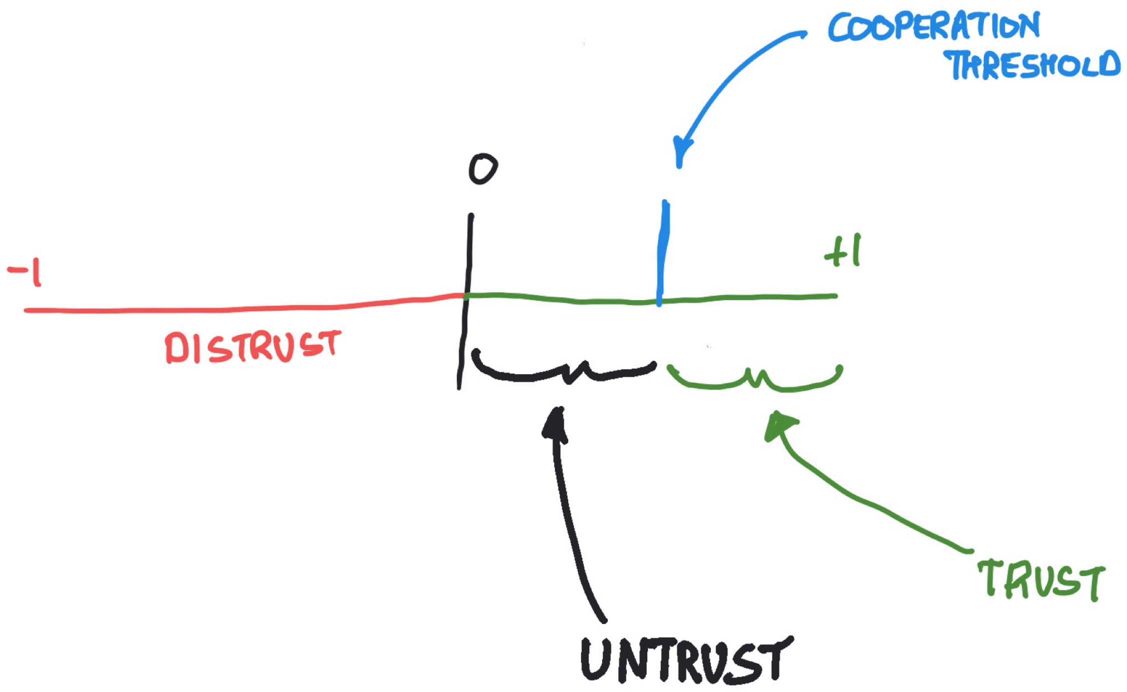


Figure 4.7: A graph depicting Untrust and Trust sections on the Trust Continuum in relation to the Cooperation Threshold.

And right now, we can bring that continuum back into play with a nice new shiny word, which is very exciting.

Why is it exciting? Because it's always nice when things work out.

Let's go back to 1985, or thereabouts. This was around the same time I was watching laser beams whizz around the physics lab, but the world was probably much more interested in what Ronald Reagan and Mikhail Gorbachev were talking about: nuclear disarmament (I like the picture in Figure 4.8… If people can smile properly at each other, we're getting somewhere).

Figure 4.8: Reagan, Gorbachev and many more in Geneva discussing disarmament, 19 November 1985. https://catalog.archives.gov/id/75854467

Now these discussions went on for a while and during them, also at the signing, President Reagan used the words, “trust, but verify”. President Reagan learned the phrase from the American scholar Suzanne Massie. He used it many times. I know this because growing up in the 1980s you heard a lot of Reagan (and Thatcher). It comes from a rhyming Russian proverb. This probably reveals a lot about the Russian psyche.

I had, at the time I was doing my PhD research, been troubled about this for several years. Essentially, “trust, but verify” is the equivalent of popping over to the house every too often to see if the babysitter is drinking the single malt.

In sum: it’s not trust.

I was gratified to hear that President Gorbachev (whom I confess is something of a hero of mine) in response brought up Emerson.

Let’s put it like this: if you have to verify the behaviour of the other person (or evil empire) then you simply don’t have the trust you need to get the job done. Try looking elsewhere.

At the point of figuring this out, my doubts were not laid to rest, but they were at least put on a continuum. This, too, you can use the continuum drawings for (Figure 4.9).

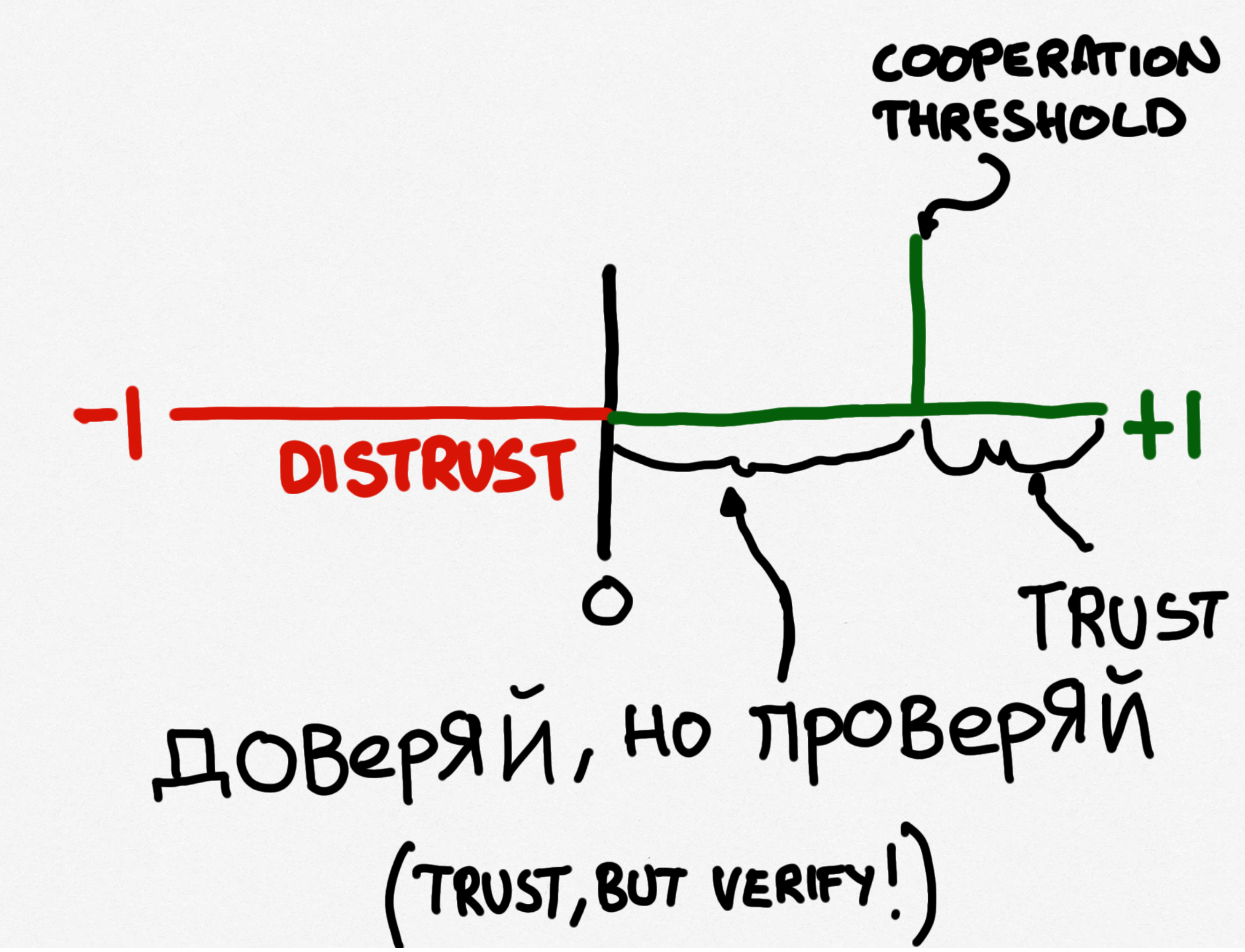


Figure 4.9: A graph depicting an example of what to do when below the Cooperation Threshold on the Trust Continuum – Trust, But Verify!

To drop into politics for a short paragraph, since then, various politicians have co-opted the phrase, most recently Pompeo, then Secretary of State for the USA, who used "distrust and verify." I do not subscribe to this point of view[5]. But then, I never really subscribed to the "trust, but verify" one either. Oddly, the world seems a much less trusting place than it did in the 1980s; yet at that time, trust seemed in short supply.

We are now in a pretty nice position to be able to talk about the behaviour of trust.

Generally speaking, it's accepted that trust is fragile–it goes up (slowly) and down (fast). Certainly this appears to be the case, although as Lewicki & Bunker note (1995), in a relationship there may well be plateaus which are reached and which serve other purposes, which we'll look at it a moment.

[5] To paraphrase Sting.

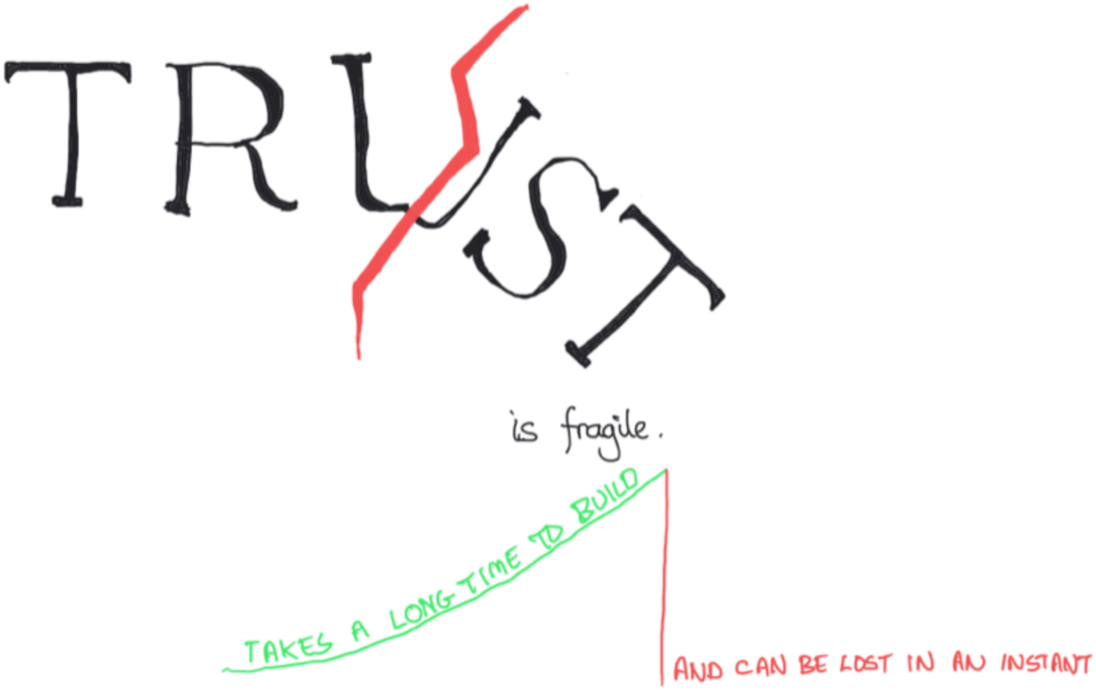


Figure 4.10: Trust isfragile.

In order to illustrate this, we can turn our continuum on its side and add another dimension, that of time. It kind of looks like what you see in Figure 4.11.

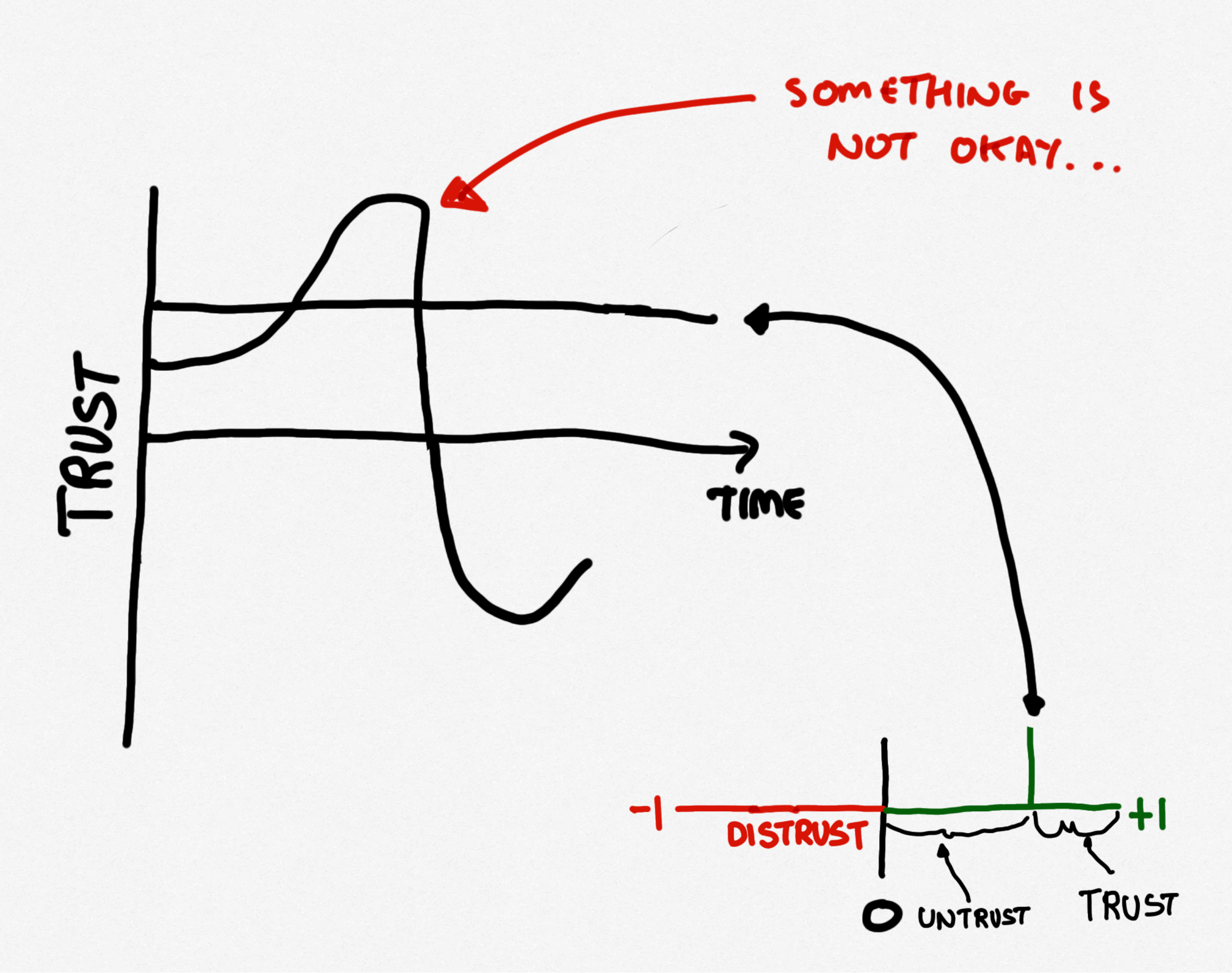


Figure 4.11: If we flip the continuum on its side and add time, we get this.

You can see the continuum is still there, it's just sideways and the y-axis on this graph, and the x-axis as ever is time. A quick look at that graph shows, in this instance, trust growing from some point in time based on, presumably, good behaviour on the part of the trustee. However, at the "something is not okay" point in time, the trustee "betrays" the trust in some way, which means that the truster loses all that nice trust and it heads into the distrust territory. At this point, it may well be hard for the truster to forgive that just happened.

This is fragility.

Remember that "forgiveness" word. It will come in handy shortly.

Let's explore that a bit further now that we have a more interesting graph, with time no less, to build on. First of all, let's talk about plateaus.

Lewicki and Bunker (1994) in their work distinguish between **calculus-based**, **knowledge-based**, and **identification-based trust** (after Shapiro, 1992). In the first, the trust is based on something like a cost-benefit analysis — what would it cost to break the relationship versus what is gained from keeping it, for instance. They also see it as deterrence-based because "punishment for non consistency is more likely to produce this type of trust than rewards or incentives derived from maintaining the trust" (Lewicki and Bunker, 1994, Page 146). Be that as it may, knowledge-based trust follows when predictability is established. Identification-based trust is established when the desires and intentions of the other person are internalized[6]. As you might expect, people probably have lots of calculus-based, many knowledge-based and just a few Identification-Based relationships.

Which reminds me of *wa*, or harmony in a Japanese society (one that is based on mutual trust, but with some special aspects where foreigners are concerned) which I first encountered in my PhD work[7]was written about by Yutaka Yamamoto in 1990. It goes a bit like this:

> Context 1: The presumption of general mutual basic trust is beyond reasonable doubt. The relations between members of close knit families, between intimate friends, and between lovers are examples.
>
> Context 2: The presumption of general mutual basic trust is reasonable, but it is not beyond reasonable doubt� Examples are relationships between neighbours and casual acquaintances.
>
> Context 3: The presumption of (general) mutual basic trust is unreasonable. A meeting with a person for the first time is an example; the first meeting between a Japanese and a foreigner is, for the former, a paradigm.
>
> Yamamoto, 1990, page 463.

If you think about it, things like context one match Lewicki and Bunker's and Shapiro's knowledge-based trust, whilst context two matches identification-based trust and context three more closely matches calculus-based trust. It probably goes without saying that there are subtle differences here, but the similarities are striking. It also likely goes without saying that as you see in the chapter on calculations in this book, it would *seem* to be that all I talk about really is calculus-based trust. It's always been a little bit of a concern to me that this would seem to be so. I *want* my artificial agents to be able to get into more deep relationships, if I'm honest. I'll talk about that a bit more here. This also brings to mind Francis Fukuyama's book – Trust: The Social Virtues and the Creation of Propserity. Fukuyama wrote this in 1996 (so, after Yamamoto). In it, Fukuyama combines culture and economics and suggests that countries with high social trust can build enough social capital (told you it was economics) to create successful large-scale business organizations. I'm not convinced that this is so, but the countries that displayed such trust included Japan and Germany, culturally

[6] There may be some debate as to whether machines ever get to this stage, but we'll leave that for another chapter.

[7] And identified for future work, which this kind of is, but there's more.

strong mutual trust countries you might say. Lower mutual trust (or more rigidly defined) countries like Italy and France don't do so well on the social capital front (although they are highly family-oriented countries). Quite interestingly, he also identified (in 1996) the growth in individualism that would cause problems in the USA

Okay, back to the continuum for a moment. There's not actually much of a difference between these different kinds of trust and the continuum except that I don't really label the different trusts because the continuum takes care of this.

One interesting thing that Lewicki and Bunker identified is that, if a person is in one level (say Identification-Based) then there may be different **plateaus** within that level. Meanwhile, if trust is violated at some level or sub-level, it is not necessarily completely destroyed but may, depending on the situation, be preserved yet lessened — for instance to the next level down. This is because of the large amount of personal capital and effort that is used in the creation of the trust relationship in the first place, not to mention the loss of things like 'face' or self-esteem if a high level trust (like Identification-Based Trust) is betrayed and seen to be a mistake, since no-one likes to admit such costly mistakes[8].

Figure 4.12 shows a possible example using a graph for trust with plateaus… Note that 'good' behaviour results in a slow growth to a plateau and bad behaviour results in a drop to the next plateau down, or possibly worse.

[8] For a little more on this, also see here.

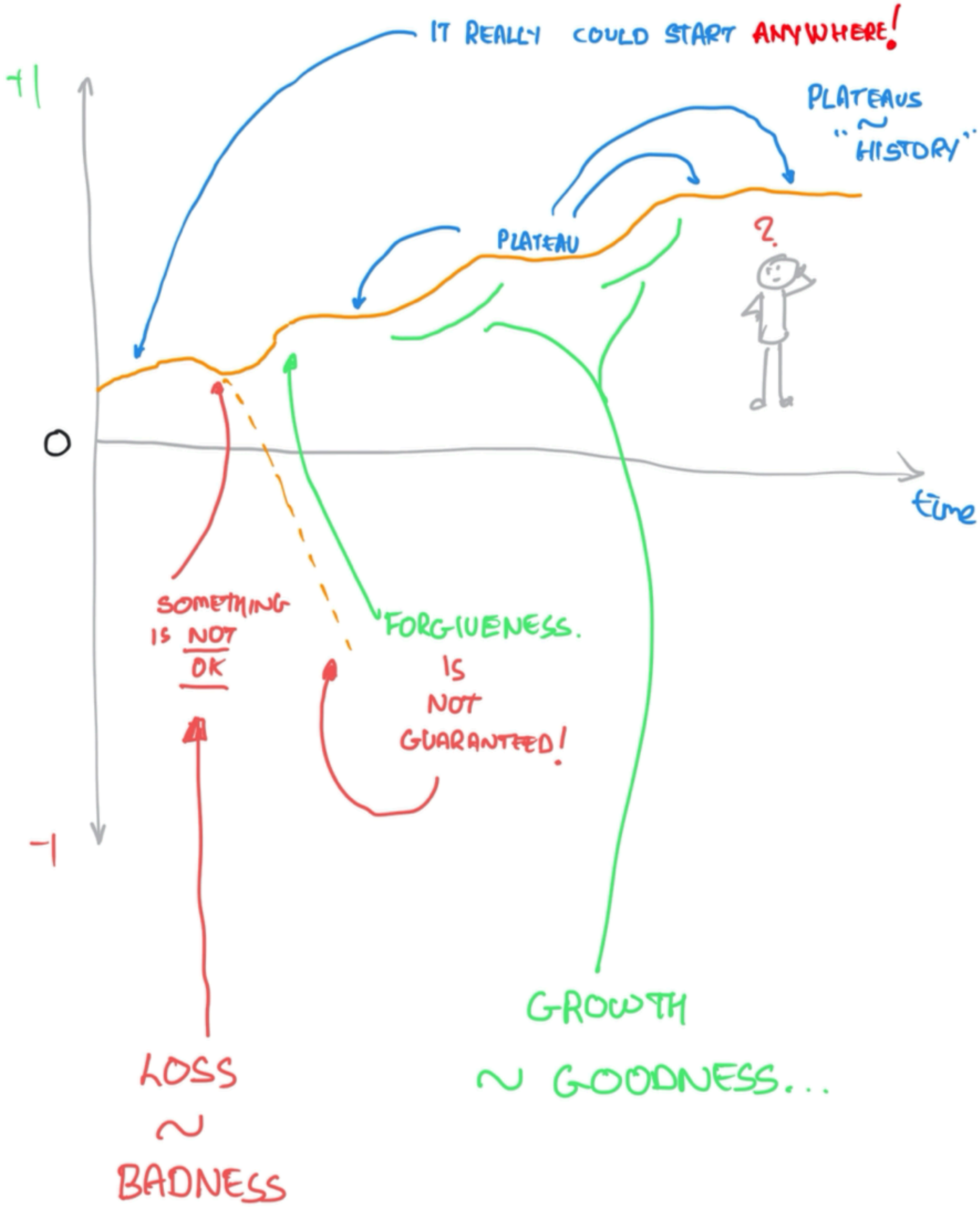


Figure 4.12: This graph depicts trust with plateaus on the Trust Continuum. 'Good' behaviour results in a slow growth to a plateau and bad behaviour results in a drop to the next plateau down, or possibly worse.

In all of these different situations, the *amount* of trust increase is relative to the 'goodness' of what is done as well as the trusting disposition of the truster (which we will come to later). The decrease in trust is relative to the 'badness' of what was done (likewise, we will come to this later).

As an aside, I'm not claiming that trust *actually does* behave like this, but I am saying that *it may well*, and there are various explorations around it that bear it out. However, it is really useful when describing what trust might look like in computational systems, which was really the point. That it in some way clarifies trust in humans is a nice thing to have at this point.

Forgiveness

In Figure 4.12 you see the word forgiveness, and you see that it is not guaranteed. In Figure 4.11 you don't see it, but you can probably figure out where it might happen, if it does. Indeed, that forgiveness is not guaranteed is the case.In Marsh & Briggs (2008) we examined how this all might work in a computational context.

Forgiveness is interesting not just because it repairs trust; forgiveness is interesting because it is a fascinating way to deal with problems you face all the time online, like cheap pseudonyms. What are they, you ask? Well, in 2000, Eric Friedman and Paul Resnick (Friedman & Resnick, 2000) brought up this problem: in a system where you could choose a **pseudonym** (basically a fake name) for yourself you could join, do wonderful things, then do a really bad thing and leave only to join as someone else, with no reputational harm, as it were. For instance, you could be "john1234" on eBay, sell a lot of little things for not much money, build up a nice reputation, and then slowly get more and more expensive so as not to raise eyebrows, and finally charge a big amount for something before heading off into the sunset of anonymity with a fat wallet. A couple of seconds later you could rejoin as "tim1234" and no-one would be any the wiser, so you could do it all over again. Sure, there are some safeguards in place for this kind of behaviour but it's actually a pretty easy attack on reputation systems. It's basically a sort of whitewashing. We'll get to attacks (and defences) in a later chapter.

So, why does forgiveness help? If we accept that people are fallible (and computers too), then we automatically accept the risk that they may make mistakes, and sometimes even try to cheat us. Hang on, this bit is where some mental gymnastics is required[9]. We can also accept that they don't need to dodge away and create a new pseudonym if forgiven. Think about it this way:: if someone screws up, on purpose or by accident, they don't need to try to hide it if forgiveness is something that might happen. More to the point, things like forgiveness allow relationships like that to continue in the face of uncertainty over who is who (as it were: better the devil you know).

This is, it must be added, a rather dysfunctional view of a human relationship: forgiveness isn't a good thing if it means that the forgiver stays in an abusive situation. However, that's not what I'm advocating here

[9] Because I made it up and even *I* sometimes ask "what?!"

for various reasons, and with various safeguards it can be handled pretty well. Generally speaking though, forgiveness in these environments is better applied when mistakes happen, rather than deliberate betrayals.

In systems where humans collaborate across networks, there is a lot of research about how trust builds and is lost, as well as how things like forgiveness help. For instance, Natasha Dwyer's work examines **trust empowerment**, where the potential truster gets to think about what is important to them and is supported by the systems they use to be able to get this information. This is opposed to Trust Enforcement, where reputation systems tell you that you can trust someone or something because their algorithm says so. We'll come back to things like Trust Empowerment when we look at where trust can help people online later in the book.

Forgiveness and apology in human online social systems was extensively studied by Vasalou, Hopfensitz & Pitt (2008). IT goes something like this: in one experiment, after a transgression and subsequent apology, a trustee was forgiven (it's a one step function). That works pretty well when the trustor is human actually because if we bring in Trust Empowerment, the forgiveness step is based on information that the thruster has in order to let it happen. I hasten to add that this is a very short explanation of something very important and that there's an entire section of the Pioneers chapter devoted to Jeremy Pitt and colleagues' work in this book.

In the case of computational systems, there's a lot of grey inbetween the grim sadness of distrust and the "ahh wonderful" of trust, especially where forgiveness might be concerned. So we decided on an approach that took into account **time**. You will have noticed by now that our graphs using the continuum have stealthily introduced the concept of time to this discussion. So, we have it to use, and as Data once remarked, for an android, 0.68 seconds "is nearly an eternity" (Star Trek: First Contact).

In the model we are examining here, forgiveness is something that can happen pretty much after a length of time that is determined by the 'badness' of a transgression, the forgiving nature of the truster and (cf Vasalou et al, 2008) if an apology or expression of regret was forthcoming.

Moreover, in the model, the rate of forgiveness is something that is also determined by the same three considerations. For more information on how this all works in formulae, look here.

Naturally, we can show that on our 'continuum-time' graph, as you can see in Figure S.13.

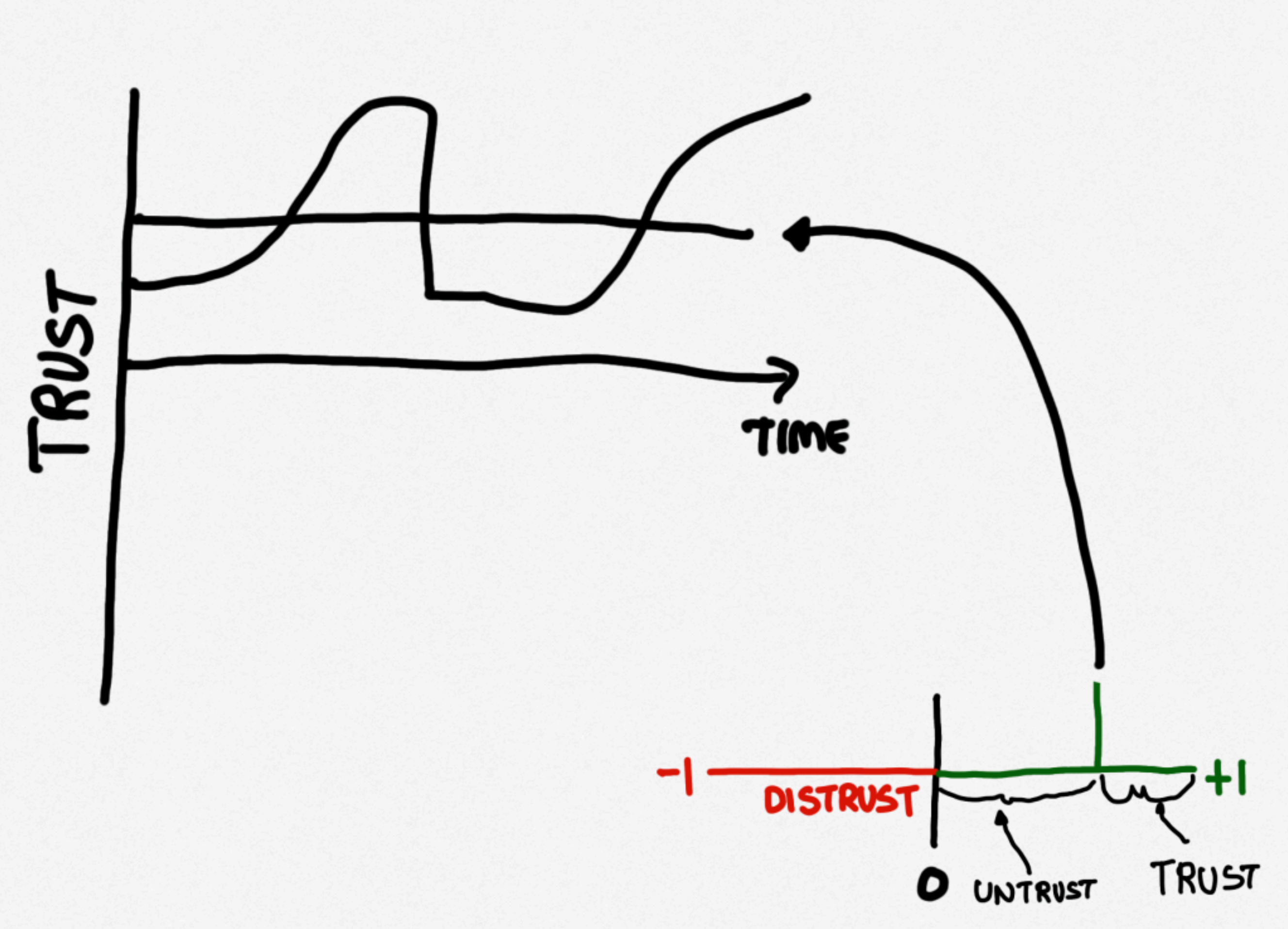


Figure 4.13: This graph depicts trust building as time passes by plateaus at the Cooperation Threshold position on the Trust Continuum.

If you take a look at that you can see that the trust goes slowly up, above the Cooperation Threshold (that blue line) and then something bad happens which makes the truster lose trust (the amount of the loss is related to the 'badness' of what happened, as before). Over time (related to the things I mentioned before) this is allowed to begin rebuilding, the rate of which is dependent on the same 'forgiveness' variables. Neat, isn't it. Naturally it's hard to show on a one dimensional continuum, so we won't.

But there is something we can show

Ever been in a relationship when something so egregious happens that the thought of forgiveness is not even possible? Almost certainly. Lucky, if not. In either case, it's probably fair to say that there are situations like that.

The same can happen for our continuum of computational trust too. In such an instance, the drop in trust is too large for the system to think about forgiveness. Recall that this drop is related to the 'badness' of what happened and it begins to make sense. Figure S.14 shows what that looks like. Take a look at it.

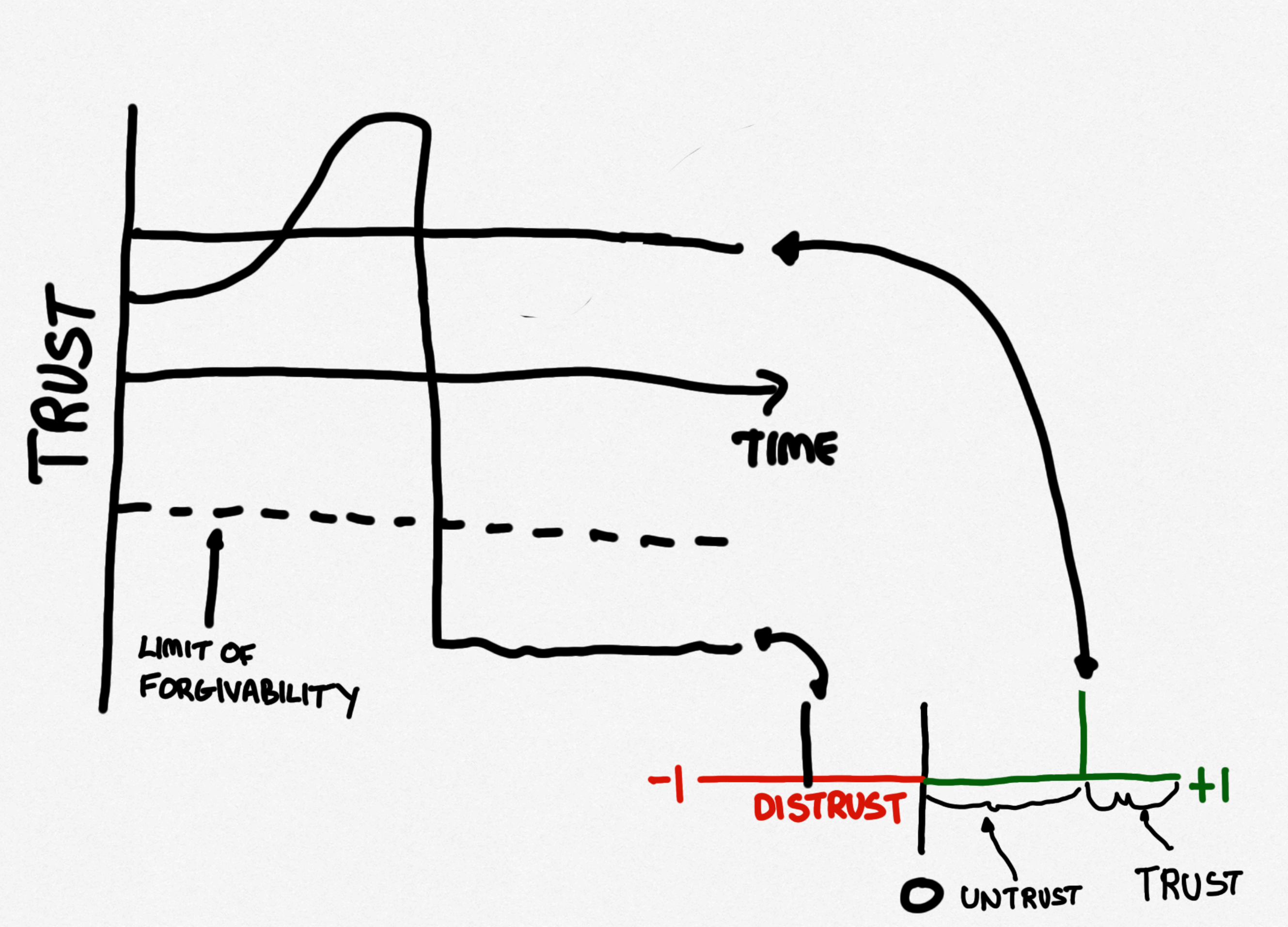


Figure 4.14: This graph depicts trust going below The Limit of Forgivability from above the Cooperation Threshold on the Trust Continuum. This represents that there is no possibility of forgiveness and the level of trust doesn't change afterwards. We call that dashed line The Limit of Forgivability

You will see two horizontal lines, one above the x-axis of time, which is our regular Cooperation Threshold (which you can also see in Figure S.13). The second (dashed) is below the x-axis, so it's in 'Distrust' territory, although it really doesn't have to be. This is the place where the straw breaks the camel's back, as it were: it's the limit at which the truster walks away with no possibility of forgiveness – and as you can see from the graph, the level of trust just doesn't change afterwards.

We call that dashed line **the limit of forgivability**.

It's an important line, and for our model its value is dependent on a couple of things – the agent (or person) being considered (which is something we will come to) and how forgiving the truster is (yes, that too).

Phew.

Let's have a look at that on the continuum. You'll see it in Figure 4.15, which is one of the saddest drawings in the whole book, and for good reason.

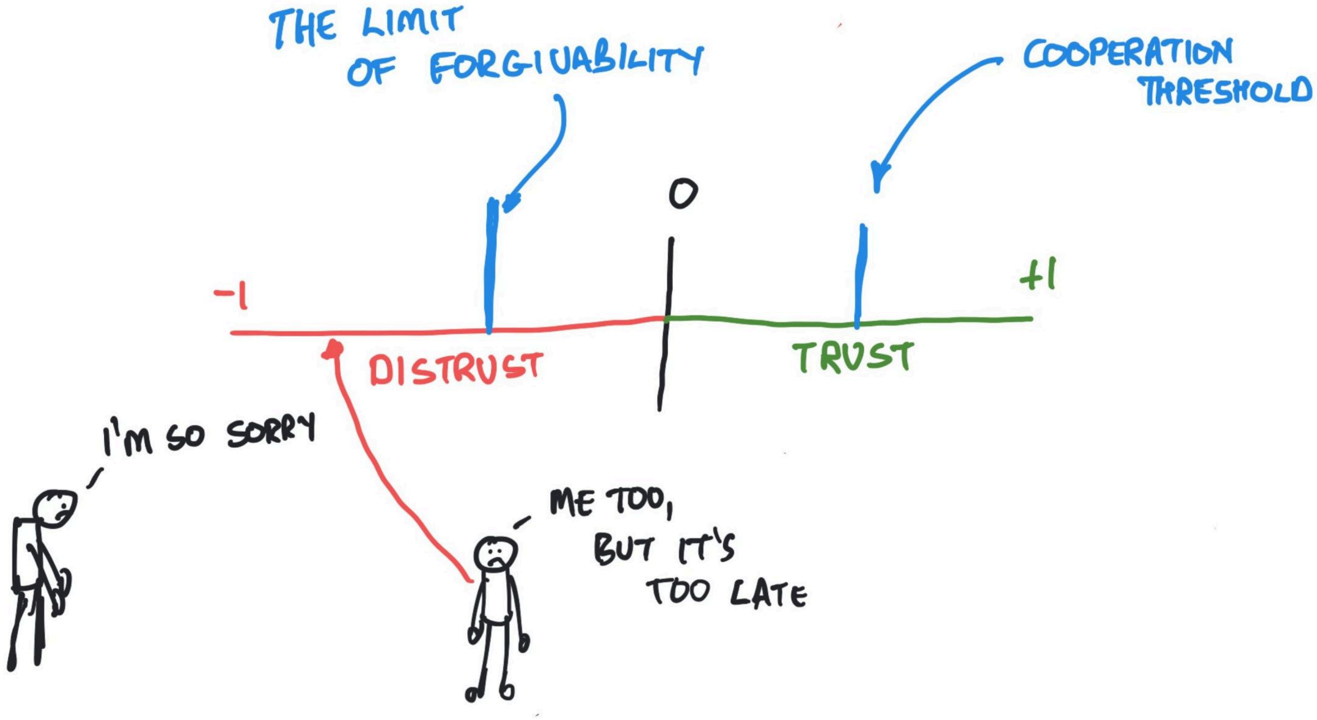


Figure 4.15: A graph depicting an example scenario of when trust falls below The Limit of Forgivability on the Trust Continuum.

I mentioned regret and apology earlier. You can't see them on the continuum (although you can probably see both in Figure S.15). However, they exist, and they are a really nice segue to the next chapter, which takes what we have here and puts mathematics around it. Don't worry, it's not complicated mathematics. Why not? There are several reasons, and in the next chapter you'll find out.

I think this wraps it up for our continuum. If you look back at the start of the chapter it was really about 'can we see trust?'

The answer is yes. Just look around you.

"The reward of a thing well done is to have done it."
President Mikhail Gorbachev quoting Ralph Waldo Emerson at the INF Treaty signing, December 8th 1987.

AND SO, TO CALCULATIONS

Introduction

In the last chapter we looked at a continuum for trust, which allowed us to do some really neat thinking about how it kind of works, especially for computational systems. The behaviour is pretty close to what you would see in humans, which is a nice bonus. I say "kind of" because it doesn't really say how it works, it just shows you what it looks like. For the how it works bit, you need to think about some mathematics.

I also mentioned in the last chapter that the mathematics involved was kind of simple, for various reasons. This is so. The reasons are both personal and human. They have in the past got me into sort of trouble though. Once I was at a job interview where there was a talk required about the research the candidate does. I presented trust. It seemed like a good idea since that's what I think about. All the time. I gave my talk, and was asked, pretty much right away, "Why do you say you use pseudo-maths for your formalization?" (or words to that effect). My answer was that I don't really like maths. I don't think it went down too well. I certainly didn't get the job.

But the point is a good one. Too often we build more and more complex answers for computational systems to be able to handle the minutiae of complex human existence. Trust models are no exception, and this is not the right way to go. Why? This brings me to the second reason.

I don't *really* hate maths. I just think it doesn't have to be hard.

The models we build for trust and reputation have become so complex, in order to handle attacks or different circumstances, or whatever, that they are indecipherable by someone who doesn't have a PhD in maths. This may seem like an exaggeration but consider: if the maths is too difficult for the people for whom the model is supposed to be working, what use is it?

This is a contentious way to start a book chapter, and the obvious retort is "well self-driving cars…" or "well space travel…" or whatever. No arguments from me that these things are complicated and people kind of expect that. I don't for one second pretend to understand the maths involved in getting Armstrong, Collins and Aldrin to the moon, although as a very small less than one year old I did apparently watch. I don't pretend to understand the systems of a self-driving car either. This is fine. In fact, when you think about it, it makes a lot of sense. These things are hard. Not only that, they take skill to even do without the computer, if that was even possible for Apollo 11. Driving a car is a complex task with plenty of responsibility involved; it's not simple even for people. That's why we have a driving test.

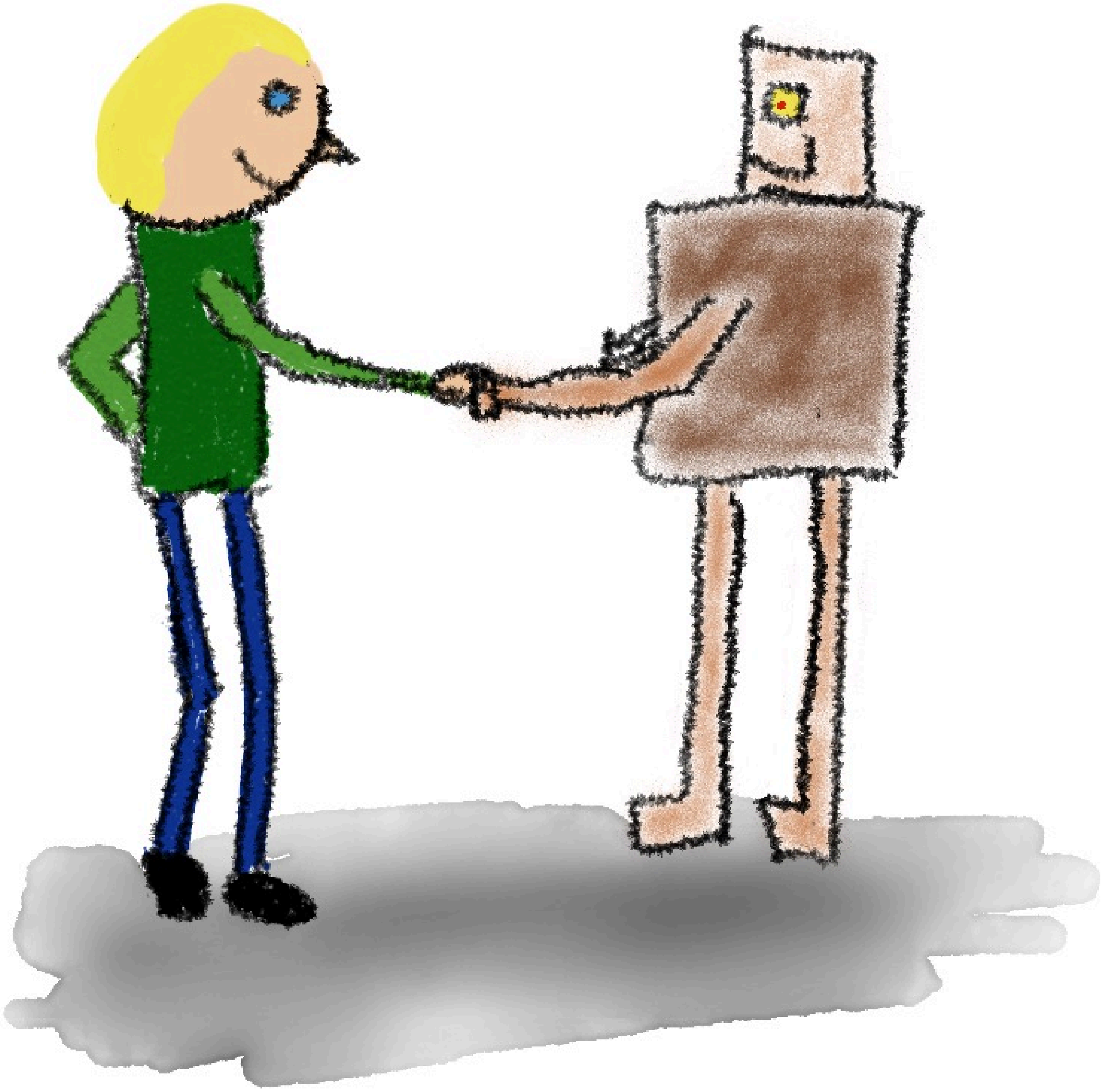


Figure 5.1: People and Computers can Trust Each Other

However, trust is something we do. Every day. We are actually pretty good at it. Sometimes we get taken for a ride (about which, see the end of this chapter) but we really are pretty good at it. So why should the systems that are helping us to make trusting decisions be complicated black boxes that we need an advanced degree in maths to understand? Hint: it's not because we don't know how trust works. Ask a baby.

There are many many models of trust out there. Some of them are less complex, others are more so. It's actually quite hard to keep track of all the models that keep appearing. There's a reason though, that there are so many. It goes like this: there are many different uses you can put trust to — you can use it to decide on that infamous babysitter, you can use it to consider car mechanics, banks, politicians, who to buy from online, whether to follow advice (from online or not). Almost anywhere you have a decision to make where risk is involved you will see trust pop up. It isn't always the primary tool being used, but it'll be there. As we'll see, sometimes it's there considering something that seems completely orthogonal to the thing actually being considered.

Human beings are liminal creatures. We live on the edges of things – and if we are not on the edge we are heading there. Apart from the ubiquitous opposable thumb, one of the reasons why we are so successful at filling the ecosystems we are presented with is that we're so adaptable and innovative. Whatever your thoughts about what humans are doing to the planet, this is hard to deny. Trust is representative of that: it's a completely adaptable phenomenon that we use to help us in lots of ways. But we're good at this stuff. Computers, not so much. Adaptation isn't something that was considered a lot for a long time, and even now computers are a poor relation to the natural world. So, to make a trust model that works in a specific instance means that you need to create a specific trust model. That's a thing.

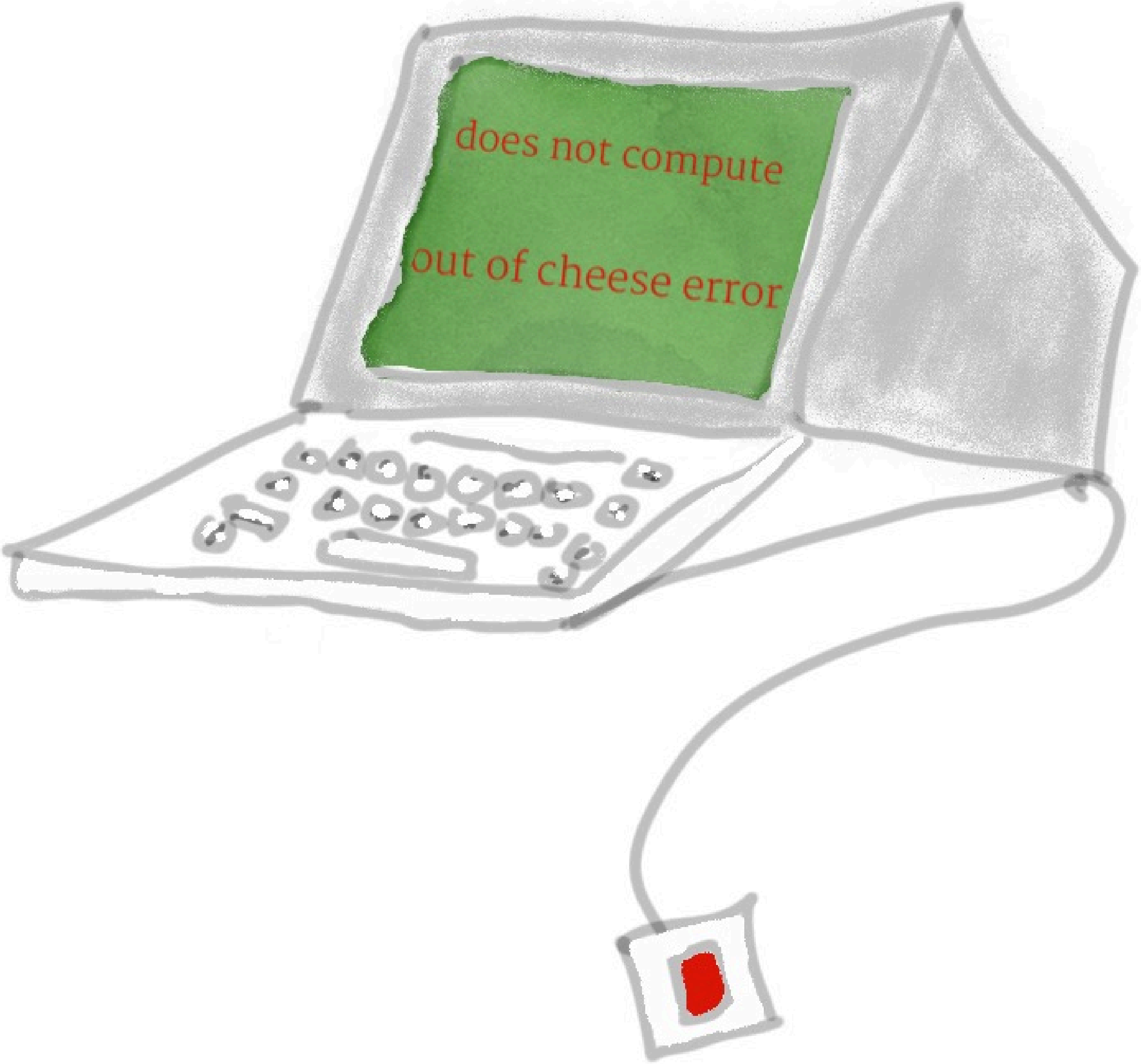


Figure 5.3: Sometimes computers don't know the answer. (Thanks Sir Terry.)

As a result, generic trust models are few and far between. That there are any at all is testament to the belief that it should be possible and the tenacity of various researchers. Still, the general models aren't brilliant. In the case of the one I created back in the early 1990s, it had less to do with tenacity and more to do with the fact that at the time I didn't actually know you couldn't.

The model itself is pretty simple mathematics-wise, but it is the first one that tried to bring together the computational and the social, the birth of **computational trust**. It is also quite generic, but one of its graces is that you can plug just about any inputs into it to make it more specialized. We'll get to that. In this chapter I'm going to describe it for a couple of reasons. The most obvious is that it is mine, but on other levels, it is both simple and descriptive.

I recently came to the realization (as in, I was told) that I am a functionalist when things like this are concerned: it is less important to me the way something works than the fact that it does. More, that it does something that we might consider, functionally, to be what you are expecting. It's easy to get into philosophical discussions about whether or not the systems using this (or other) trust model are actually thinking in trust, as it were, and somewhere in this book we might, but the fact is that they sometimes behave as if they are, and that's enough for right now. We'll come back to this, don't worry.

Basic Trust

In the last chapter we used the continuum as a way of looking at how trust might look – sort of the way it might behave in different circumstances. Let's flesh that out. In this model, there are three different kinds of trust and two questions. We've actually come across the two questions already in the previous chapter. That whole cooperation threshold thing revolves around them. The first is how much trust do you have (in a situation – remember trust is contextual) and the second is how much trust do you need in that situation. The second one is the **cooperation threshold**. I'll show you how we calculate that shortly.

The first question requires us to look at our three different types of trust. We'll do this from the point of view of the truster, which for the sake of simplicity we'll call x (you can call them Bob or Alice or whatever, it's all good).

Our first kind of trust is the one that stays with us from pretty much when we're born, if not before. It's basic, learned from the environment and all our interactions, as well as our predispositions for whatever reason. We call it **basic trust** and, from the point of view of x, we can represent it like this : T_x. Figure 5.4 shows our stick person with that Basic Trust.

Figure 5.4: A Basic Trust Stick Person

It's important, don't forget it. Everything that happens to our wee truster, every experience, every interaction, everything, has an impact on this basic trust. As you might expect, at the start of the truster's existence it might be quite volatile, but the more interactions they have, the less volatile it gets — it takes quite a lot to change the trusting nature of someone. Either quite a lot or something quite memorable.

So, that's our basic trust. It doesn't really help too much when little x is thinking about another being in their existence. Let's, for the sake of argument, call this new thing y, shall we?

General Trust

In this instance, when we say that x is thinking about how much they 'trust' y in general, we really mean is, based on the evidence, the interactions, history and so on that the two have together, x thinks that y is, for example, pretty trustworthy, or untrustworthy, or whatever. We call this **general trust** and we represent it like this: $T_x(y)$.

This is important too. Remember it, there may be a quiz later.

Okay, so if people asked me if I trusted my brother[1], with no context, I'd say, "Yes". My general trust in him is pretty high. Trouble is, I am pretty sure his flying skills are at least as good as mine, which is to say non-existent.

[1] who as I type this has just finished his own PhD! Yay Pete!

So, as I talked about in the previous chapter, in the context of flying the plane I was sitting in, I think my level of trust is basically in the "for goodness sake, no!" range that you can see in Figure 5.5 (Sorry, Pete).

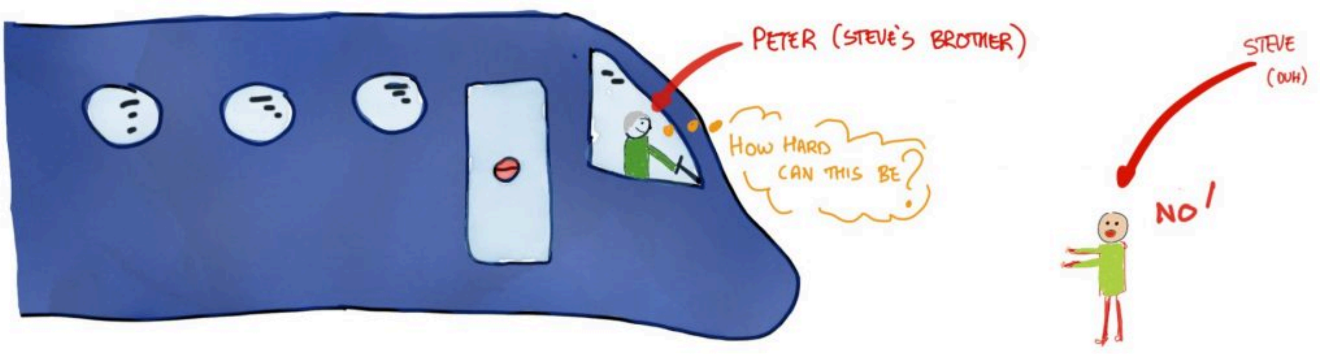


Figure 5.5: Peter wants to fly a plane. Steve isn't so sure...

Situational Trust

What is missing then? I trust him a lot. I just don't trust him to fly a plane (or perhaps to do brain surgery). The missing piece, of course, is context. Remember the trust system: people, process and place? Well, context is part of the place bit. In the formalization I created I eschewed the word 'context' for reasons I cannot fathom this far distant from the decision, and instead used the word 'situation'. So what we have is what I called **situational trust**. It kind of stuck. This is represented for our little x and y as: $T_x(y, \alpha)$.

Hang on! Hold those horses! See that new thing, α? That's the context or situation that the consideration is being done in. For example, it could be, "Do I trust my brother to fly the plane?" or, "Does Patricia trust me to cook saag paneer properly?" Which is wonderfully illustrated in Figure 5.6, if I say so myself.

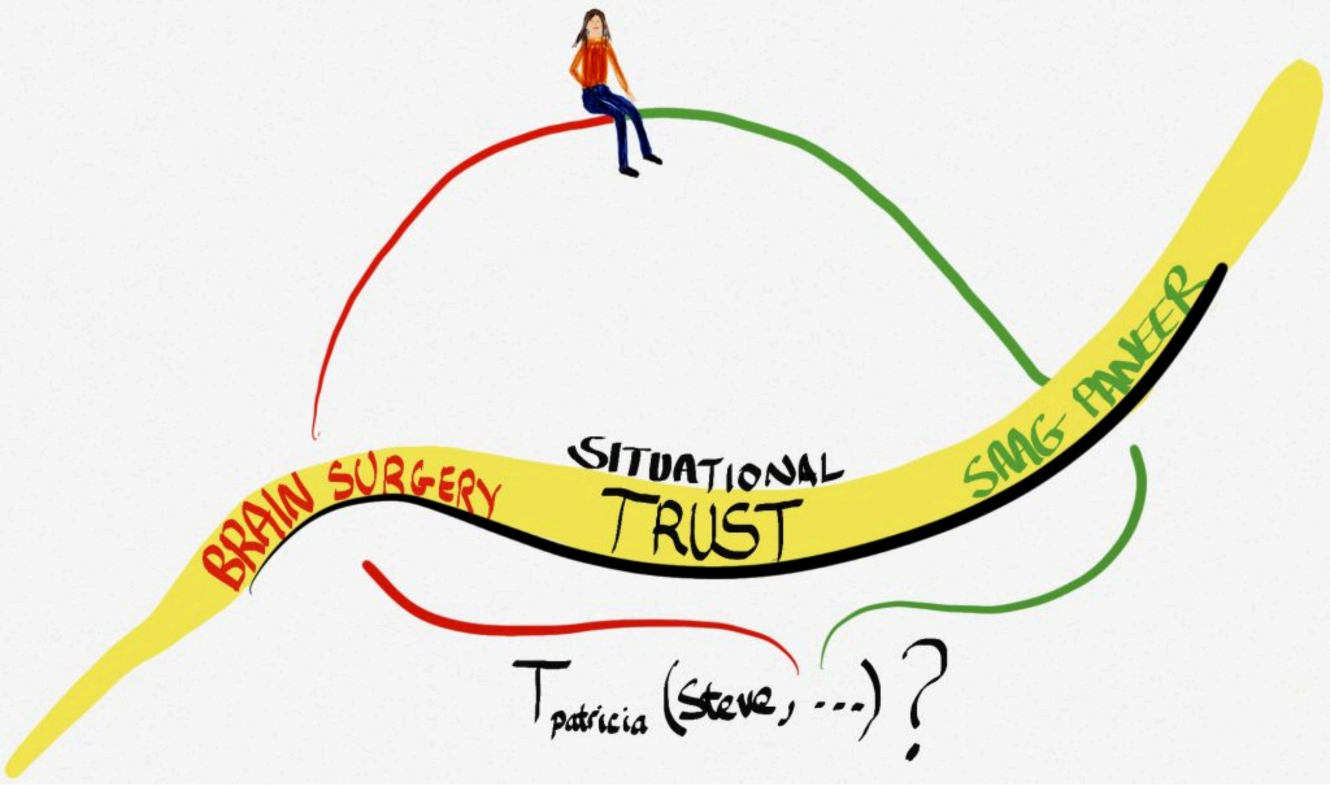


Figure 5.6: Patricia and the “brain surgery vs. saag paneer cooking” conundrum

Situation, or context, can be anything that the truster is thinking about. Writing a letter, posting one, lending or borrowing money, driving, flying, hitting, you name it. As you might expect, it is also changing from instant to instant, because trust is not just contextual, it is contextual. Would I trust my brother to fly the plane if everyone else was incapacitated and he was the only one left? You tell me. Figure 5.7 might help…

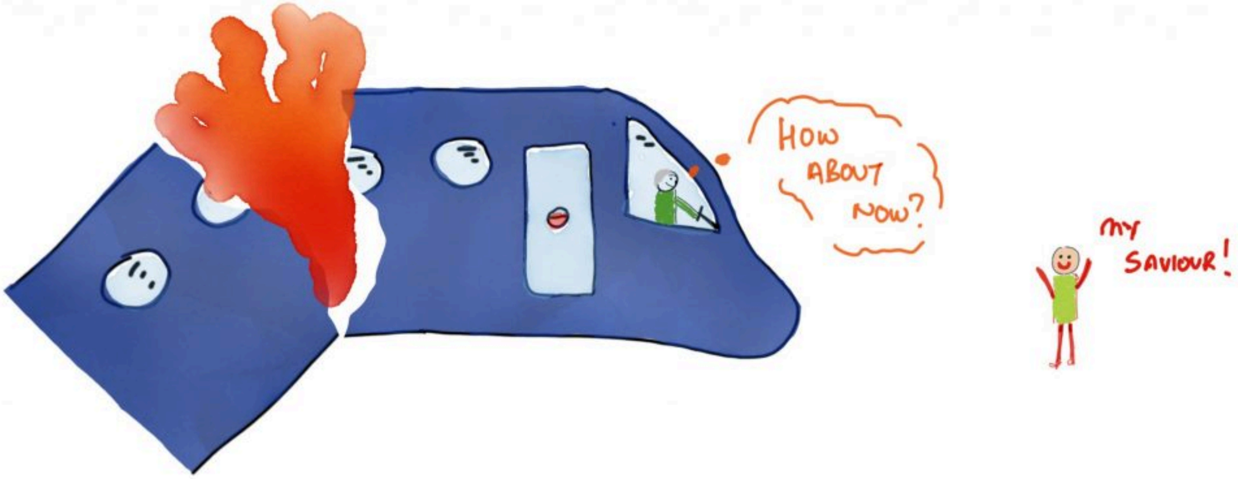


Figure 5.7: Pete is the only one left to fly the plane or we all perish…

You see, context matters. But how do we plug all that context into the situation? Let's see. But first, how did we get the general trust thing again? I mean, how does x figure out how to trust y in general if they've not met before?

It's like this: in the absence of any other information, the value of at the start of a relationship is set to the basic trust T_x. This solves lots of problems and seems to work pretty well — after all, it's kind of how we work. The key words in the last but one sentence are "absence of any other information" because in general for someone you are entering into some kind of relationship with, there will probably be other cues, like referrals from friends or colleagues, reputation from the wider world, and so on — this is something Erving Goffmann pointed out in 1959:

> "When an individual enters the presence of others, they commonly seek to acquire information about him or bring into play information about him already possessed … Information about the individual helps to define the situation, enabling others to know in advance what he will expect of them and what they may expect of him."
>
> (Goffman, 1959, page 13).

Still, it's nice to know we can fall back on something, eh? Back to situational trust. It goes like this. Situational trust is calculated from three distinct functions, each of which is ultimately tailorable to match the kind of trust we want to model (like buying, selling, teaching, romantic relationships and so on – they all have different inputs to trust, right?). The equation looks like this:

$$T_x(y, \alpha) = U_x(\alpha) \times I_x(\alpha) \times \widehat{T_x(y)}$$

Utility

There's a lot to unpack here, so let's get to it.

You already know the left hand side, that's easy. That's just the situational trust, which will become a value (in a range $[-1, +1)$ – bear with me, we'll get to those brackets).

The $U_x(\alpha)$ represents **utility**. Utility is a pretty loaded term when we talk in these kinds of contexts because it has specific meanings, like "what x stands to get from something". It could be counted as money or some other kinds of reward, like for instance the pleasure of a quiet night at the movies with your significant other while the baby is being looked after by the babysitter.

I can almost feel you thinking about the costs bit. Sure, yes, utility can be a cost-benefit analysis. You gain something but it costs you something. You pay the babysitter money. You have to work for your money, and so on. All very good, and you can plug these considerations in here.

(If you are now thinking "but the babysitter could..." then wait for a short while. We will get there.)

The thing about utility here is that you can pretty much plug in anything you want from what can be gained, and what it costs. If you're buying, selling, working and so on it's fairly clear. If you are thinking about the babysitter and the night out, that's pretty straightforward too. It gets a bit more cloudy when you think about having your car maintenance done, especially if you're short on cash — maybe it can wait a wee while, maybe the maintenance doesn't feel like it's giving you much right now (but wait until 3am on the highway...).

Figure 5.8: Serves you right, Steve.

I think you get the point.

Another thing: in Game Theory, the people or agents being considered are expected to be rational utility maximizers, which is to say that they will choose the thing that they get the most reward from. This isn't necessarily the case in this model.

Our second thing? $I_x(\alpha)$ is the **importance** of the situation for our little x. You might think "well surely that's the same thing as the utility," but it isn't, here. Importance is a measure of how much little x wants the thing to happen. Think about the babysitting thing for a moment: what if x is in the military and tomorrow has to head out for a six month mission? How important is a few hours of time alone with their Significant Other compared to, say, a couple who go to the movies every week? It works for money too, although there's a fuzzy 'utility-lite' thing happening here. How much more important is winning $10,000 to someone who has nothing in the bank than it is to your everyday billionaire? There is a utility thing here but I like to separate them because it makes more sense to me. I think of it like this: The utility of $10,000 is $10,000, if we are thinking money value only. But that doesn't say much about how it might just change the life of the badly off compared to the billionaire (I know it would help me. I expect Jeff Bezos wouldn't even notice it).

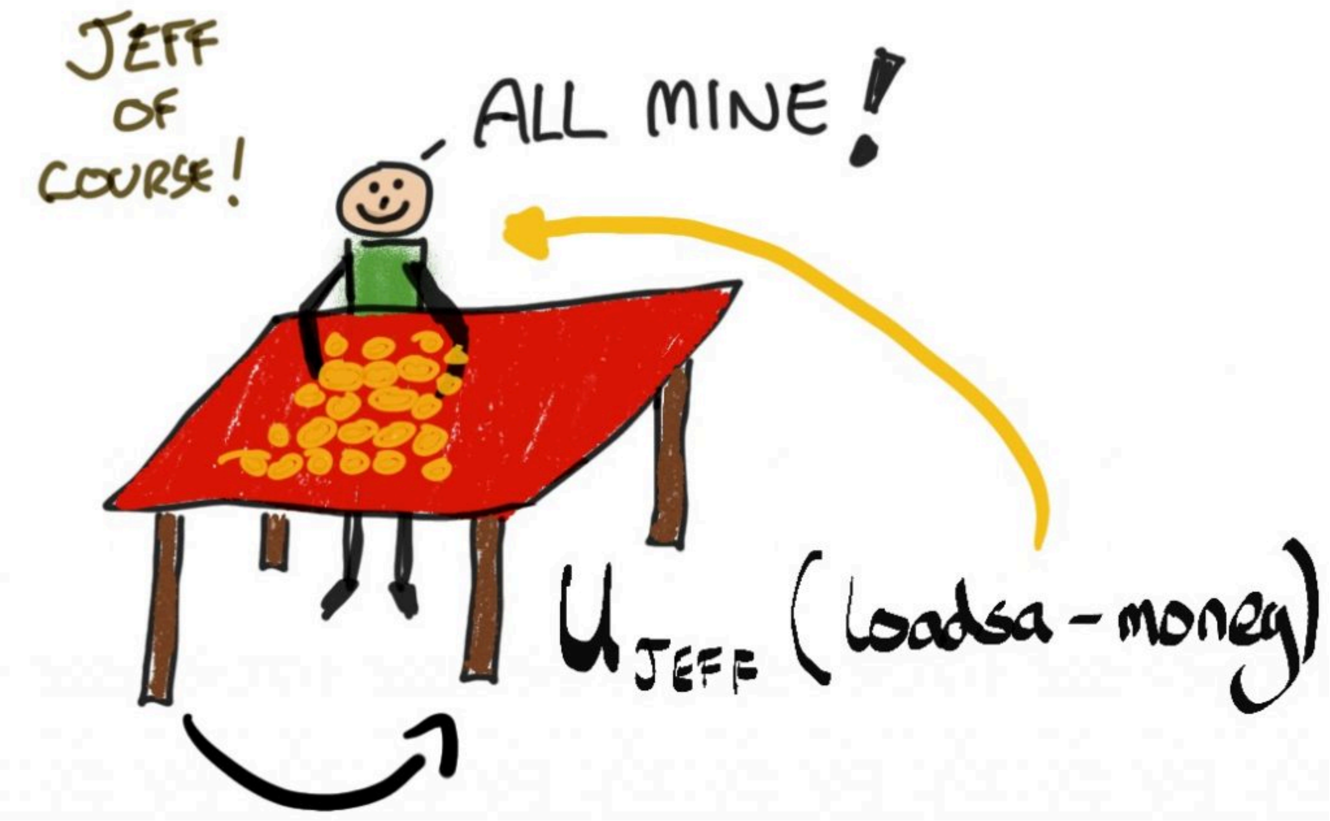


Figure 5.9: There's a lot of money there, but what is the utility of all that money? Kind of depends on who is asking.

As with the utility, you can plug in any kind of consideration into importance. Probably you can be even more wild and wacky with what you plug in — things that matter are wildly different for different people, let alone in whatever different contexts there are. For instance, one of my horses right now needs his hooves trimmed. He's actually a pony and I couldn't ride him if I wanted (I suspect my feet would touch the ground!). The utility of having the hooves trimmed is pretty low (to me!). More trimmed hooves, a few hoof clippings for the dogs to play with. The importance is pretty high, because if not done then he is probably pretty uncomfortable.

I'm sure the discomfort and subsequent lack thereof are actually of some utility to the pony because, hey, discomfort isn't fun. It's also probably of high importance to him, but of course I have no way of knowing.

Here's the thing: I have no idea of knowing for you either (not for having your hooves trimmed, but I think you get the idea). Things like Importance and to a lesser degree utility are personal. That doesn't mean we can't guess (guesstimate?!) them though, and improve our guesses with observation and so on. That's kind of what trust is like.

For the sake of completeness, when I was thinking about these two things I kind of assumed that they would be in the range of $(0, +1)$. So there's no negative utility or importance, for instance. If there was, I'm going to say that our little x wouldn't touch the situation with a bargepole. However, one of the more interesting things about the formalization is that it allows us to think things like "what if the utility was negative but the situation was really important?" – I'll leave that as an exercise, both in terms of thinking up such an example and what it might mean.

The last thing in our situational trust equation is obviously expecting rain, for it has a hat: $\widehat{T_x(y)}$. Clearly there is some magic at work here.

Okay, not *really*.

The hat is there to show us that we're using some form of estimate or even calculation for what is really the general trust in y of x. What do I mean by that? Well, a lot of it has to do with what kind of person x is. Pessimist? Optimist? Forgetful? Spiteful? You name it.

For instance, we might say that x is a pretty rational agent, happy to think about all the experiences they have had with y and basically average them all out, sort of a 'fair comparison' type of thing. In that case, you could say that:

$$\widehat{T_x(y)} = \tfrac{1}{|A|} \textstyle\sum_{\alpha \in A} T_x(y, \alpha)$$

Which is basically saying x will take the average of the results of (what happened in) all the experiences they have had with y in similar situations. Or you might say:

$$\widehat{T_x(y)} = \tfrac{1}{|n|} \textstyle\sum_{t=1..1-n} T_x(y)^t$$

Which basically says, go back in memory for n experiences regardless of whether they are similar or not and average that lot out. We introduce time here (that t) — hope that's okay. Hard to talk about memory without bringing time into it.

You can play with this a lot. x might consider memory and similar experiences. They might consider only positive experiences or just negative ones. Those last two are indicative of optimism or pessimism, by the way. What would an optimistic x think? How about:

$$\widehat{T_x(y)} = \max(T_x(y, \alpha)^{t-1}, \ldots, T_x(y, \alpha)^{t-n})$$

The thing is, as you may have spotted, it's quite customizable to different kinds of people, agents, circumstances and so on.

It's pretty neat.

Cooperation Threshold

Okay, that's our first question (you remember: How much do I have?). The second one is how much trust do I need? This is also, as you might expect, quite tailorable, and it definitely tries to take into account some of the things I've taken about in the previous chapters. Let's just right into it. As you already know; it's called the cooperation threshold. Here's how I calculate it:

$$\text{Cooperation\_Threshold}_x(y, \alpha) = \frac{\text{Perceived\_Risk}_x(\alpha)}{\text{Perceived\_Competence}_x(y,\alpha)} \times I_x(\alpha)$$

Woot, that's pretty interesting! (I would say that.)

Okay, so let's see. We'll start with the fact that a negative cooperation threshold simply doesn't make sense — well, if it was negative we'd cooperate with those we distrust, even. There may well be some examples in international espionage or the old Great Game where this is the case, but let's leave the Machiavellian stuff out for now. So this is going to have a value in the range latex[/latex]. We'll normalize it if we have to, which is to say, if the answer comes out to more than one we'll just make it one, and make it zero for less than zero. Yes, I know that the negative ones are interesting, and indeed the greater than one situation tells us we'd never cooperate, but in the first instance we're ignoring the issue and in the second, since Situational Trust can never be +1, it's a moot point.

Perceived Risk

So $\text{Perceived\_Risk}_x(\alpha)$. is pretty straightforward. How risky is the situation? There are plenty of different risk estimation things out there so I won't go into that here. Risk estimation in general is something humans are pretty rubbish at though, which is at least a point for the computational side!

Figure 5.9: In risk estimation, the agent finally feels valued.

Remember that whole regret and forgiveness thing we got to in the last chapter? If you recall, they were really interesting tools for helping figure out how trust goes up and down in different circumstances. This is indeed true, but I didn't really talk about what that really meant. In other words, how does it work?

Like most things to do with the model I'm presenting here, the answer is "that's up to you" but the general idea comes from Jonker et al (2004) who in a nutshell found that positive experiences increases trust and negative ones decrease it. You probably are thinking "yep, figured that out for myself" but firstly it never hurts to check and secondly they also found out other things. Like for instance that positive (say maximum) trust was reached after two positive experiences and negative after three (or two, depending) negative experiences. Now, take that with whatever amount of salt you wish. The thing is, we can all pretty much agree that trust does go up and down. I showed you a bunch of pretty graphs with absolutely no scales on them (oops) in the previous chapter, so there we are, that proves it.

But how much?

That's the real question, isn't it? If you get into a relationship and things go Pete Tong (look it up, I can't do all the work for you), how much trust do you lose? How much do you gain if things keep going well? Recall from the last chapter that trust is often seen as fragile (slow up, quick down) and there may be plateaus around there someplace.

For this, there are many different formulae, all of them sensible. Cast your mind back to the chapter on Prisoners. Remember tit-for-tat? You could say that its trust went to maximum after a positive experience and minimum after a negative one. You could also say it was very forgiving, so if its opponent went back to being nice, then after one round it was forgiven. Extreme? Maybe. Works? Sure. As you can imagine, it gets more complex from there. In the model I'm showing you in this chapter, we did it in a few different ways. There are reasons for this, much of them have to do with the fact that if you remember we were interested to see how trust worked and so playing around with different behaviours made sense.

Anyway, at its simplest we said that you could increase trust by some value related to how important the situation was after a positive experience, and decrease it similarly if the experience had been negative. That might look something like this:

$$T_x(y)^{t+1} = T_x(y)^t \pm f(I_x(\alpha)^t)$$

This has the merit of being simple but has the fault of showing us pretty much not much. What is this function and how does it work? As ever, it's up to you. For instance, it could look like:

$$f(I_x(\alpha)^t) = \frac{I_x(\alpha)^t}{10}$$

Which means that the trust will go up or down by one tenth of the importance of the situation to x. Doesn't take into account fragility though, so you can figure out that it would probably make sense to say something like "after a negative experience reduce trust by ten times importance, after a positive one, increase it by one tenth of importance". That would do the trick. The result is that trust increases ten times slower than it decreases. Again, this is easy to play with and there are so many different things that can influence it, For example, optimism, pessimism and (ta da!) regret or forgiveness.

Regret

So, if you recall, regret was a couple of things. The first was as a decision tool. In this instance, we use it to say things like "I will regret this if I don't do it" or "I am going to regret this". You get the idea. In that mode, you could plug **anticipatory regret** into the formula for the cooperation threshold above. What might that look like? For the sake of argument let's say that we can calculate anticipatory regret, so we don't have to work that in here. How might we do that? Well, to be sure it's really just a guess in any case, but in a world with perfect information you might be able to do things with utility offing something versus not doing it at all. This isn't the same as doing it and it not happening though. For instance, if I don't buy a lottery ticket one week and my numbers come up, there's quite a bit of difference between not buying one and not having a set of favourite numbers in the first place, or buying one and having the numbers not come up. There's some great research on that too, which right at the end of the book you'll find pointers to.

Anyway, assuming we guessed how much we would regret not doing something it might look like this:

$$\text{Cooperation\_Threshold}_x(y, \alpha) = \frac{\text{Perceived\_Risk}_x(\alpha)}{\text{Perceived\_Competence}_x(y,\alpha)} \times I_x(\alpha) - \text{Regret}_x(\neg\alpha)$$

In case you're not familiar with the notation, that strange neg in the Regret thing means "not". So, as you can see here, this has the effect of lowering the cooperation threshold by how much x thinks they are going to regret not doing this thing (and so increases the likelihood they will do it, as it were). You can do similar things with "regret it if I do it" and so on.

On the other side of the behaviours comes "what happens after bad things got done". In other words, how much does x regret having trusted y because y betrayed x?

As with most things, this kind of depends. It depends on how much x does regret trusting y. It may well also depend on how much regret y feels that they betrayed x. Because it may have been an accident. It may have been a stupid slip. Or it could be quite purposeful and no regret felt. As with most things in this little convoluted world of trust, even an expression of regret may be insincere. Regardless, we do have some thoughts on it.

The first is to think about how much x regrets the betrayal. This is kind of related to how much they regret trusting y in the first place, and how much they lost from the betrayal — these two things are not the same, as a moment's thought will show you. For the sake of simplicity here, we use one formula that brings both of these things together:

$$\text{Regret}_x(\alpha) = (U_x(\alpha) - U_x(\alpha')) \bullet f(\kappa, \lambda, \mu)$$

This takes some unpacking, so let's start. Here, $U_x(\alpha)$ is what x actually expected from the situation (if it had all worked out) and $U_x(\alpha')$ is what x actually got (which might be nothing, or worse. Note that if it's negative, then it actually **cost** x something, which does bad things to regret!).

Okay, but what about that function:$f(\kappa,\ lambda, \mu)$? Well, this tries to take into account some of the more emotional aspects of the situation. Some of these variables are relatively straightforward for our computational friends, others not so much. That's life. Here we go.

First, kappa (that's the one that looks like a κ – have you ever wondered why we use Greek letters in this kind of thing? I mean, like α, $\beta$and stuff like that? There are a few conventions. I've broken a few of these rules in my own equations because the letters are supposed to mean something specific in a lot of cases, but hey, I think I'll survive).

Anyway, kappa (κ)stands in this formula for what was lost. You could look at this in purely utilitarian terms (which is easy for the rational stuff) but there is a reason it is there. The loss of something like a relationship, or innocence, or whatever, is a visceral thing. That kind of loss fits here too.

The next Greek letter is lambda (λ – and definitely this is conventionally to do with things like wavelengths in physics and so on, but here, not so much!). In this instance, lambda (λ) stands for "what it meant" to the agent. Again, this is a bit hard to pin down, but it has to do with things like the importance of the thing that was expected. Sometimes, really small things have a very large importance, they "mean" something to us. That fits here.

And finally, we have mu. Yes, mu. This looks like μ (and yet another convention is crushed – serious mathematicians would say mu like that was for an Average! Which begs a question as to why they wouldn't just say "Average". There are probably lots of reasons for this but certainly one of them is that, well, we all like to feel special).

In this formula, mu (μ) stands for "how it felt". Now this one is really hard to pin down for a computer. However, if you think about it as a human, being betrayed, losing something important, things like that, that feels bad. That's what we are trying to capture here.

So we have… what was lost, what it meant, and how it felt. I think you'll agree that these things pretty much sum up the pain of regret in a lot of ways.

And yes, we can of course plug in various ways of calculating it. For instance (pay attention, this one has something new in it…), let's say that x actually *already knows* y. In this instance, by 'knows' we mean 'has had experiences involving…' which basically means that there is some form of relationship there.

The word relationship conjures up a lot of different things. For instance, it might be just a "oh sure, I know that person, we've talked a couple of times" all the way to "life without them would be like a broken pencil", with of course many points between. The reason that this is important is that it says something about the meaning of the relationship to (in this case) x. And as we've seen above, meaning is something important in our function. So let's say that the importance of the relationship has an impact here.

So far so good.

Now, the situation where x has trusted y also means something. In the formulae that we have seen so far, this is best represented by how we calculate the cooperation threshold: there's competence, there's risk and there's importance. We also started thinking about that regret part too (either "I will regret it if I do it" or "I will regret it if I don't", sort of thing).

Ultimately, what the cooperation threshold becomes is a signal of how much the agent wants to get this 'thing' done in this context. It also wraps into it things like how important it is to the agent that the job is done properly.

There are a couple of asides here. Those two things don't always play well together. I mean, you might want something done really badly and want it done properly, but the "want it done really badly" might also mean that you are prepared to accept a less than perfect job so that it does get done.

I'm sure you can see the problems with all this. Humans are quite complex beasts and to be honest trust and things like that are not the only motivators that we have. The computational systems this model aims to help are no less complex, but they do have the benefit of being to some extent under our control. This means we can choose what we think makes sense.

For the sake of argument in this instance, let's say that what makes sense is that the higher the cooperation threshold the more the important job is. The lower the cooperation threshold the less important it is (because our agent is happy to accept a shoddy job). This means that, the higher the cooperation threshold the more likely it is that things like "what was lost" and "what it meant" are badly affected. Okay, so that would mean we could say that our function looks something like this:

$$f(\lambda, \kappa, \mu) = \text{Cooperation\_Threshold}_x(y, \alpha) + I_x(xy)$$

Smashing, now we are cooking with non-renewable hydrocarbons.

See that last thing? It says $I_x(xy)$. In our notation, this means the importance of the relationship to x. It's similar to the importance of a situation in its makeup, but it says more about the inter-personal (inter-agent) relationship.

One last thing before we go to the next bit. You may have noticed that none of λ, κor indeed μ appear in the calculations we make. This makes it not so much a function as *a way of thinking* about the situation for

x. As I've said many times, trust is personal and the things that matter are likewise personal. In this instance, those three Greek letters are placeholders for anyone looking at the formula to realize what it is really trying to say: that regret is multi-faceted and should take into account much more than an opportunity cost. As such, it may be that you can think of a different formula to put in there, and that would be just fine. Better in fact, in some ways, than copying what is there, because it is yours. The beauty of all of this is its customizability: this is not a set of formulae that say "this is how trust works", it is the start of a conversation about *how trust might look*. Perhaps most of all it is a framework that people can use to make trust-reasoning computational agents that think like they do. And that is worth a lot of complicated equations.

One last example. Perhaps x and y don't know each other, it's a first interaction. We can get really simple here and say something like:

$$f(\lambda, \kappa, \mu) = I_x(\alpha)$$

Which kind of makes sense because all x really wants to think about is how important it was that this thing got done (and what it felt like when it didn't).

Think you can find different ways of thinking about it? Awesome. You have my blessings.

Now, we've arrived at a way to think about how much trust goes up and down. One that takes into account regret, that is. Which was one of the things that we were trying to figure out at the start! It's a long journey, sorry. You may recall, if you're still with me, that we had a pretty simple equation before, like this:

$$T_x(y)^{t+1} = T_x(y)^t \pm f(I_x(\alpha)^t)$$

And we talked about what that function of importance might look like, and so on. Now we can bring in regret, which is cool. Why is it cool? Because it allows us to further enhance the way the model can work.

We now have a way of calculating regret. If you remember, it looked like this:

$$\text{Regret}_x(\alpha) = (U_x(\alpha) - U_x(\alpha')) \bullet f(\kappa, \lambda, \mu)$$

So how does this sound: The more x regrets that the situation went bad, the more trust that they are likely to lose in y. Yes, there are plenty of other ways to think about it, that's the entire point and beauty of the whole thing! But we'll stick with that for now. So, we can now come to a different equation for adjusting trust. This can work in both "that went well" and "that was pretty bad, why did they do that?" situations, and more...

$$T_x(y)^{t+1} = T_x(y)^t \pm f(\alpha, \text{Cooperation\_Threshold}_x(y, \alpha), T_x(y, \alpha), \text{Regret}_x(\alpha), \text{Regret}_y((\alpha))$$

That's a bit big, but if you read it carefully you will see that all it is saying is that the amount of decline or increase in trust that x has in y is dependent on:

- The situation itself (which gives us context);
- The cooperation threshold for the situation (which if you remember was a sign of things like how the situation mattered to x, but in this instance also could be used for things like "it should have been done well" and so on);
- The trust that x had in y for that situation (remember that this takes into account a bunch of things like importance and utility);
- The regret (if any) that x feels about how the situation went (this could be none at all if it went really

well, don't forget); and

- The regret (if any) that y feels about how the situation went (again, this could be none at all if it went really well, but it could be nothing at all if y betrayed x and doesn't care).

Now, does it make sense? And can you see how all of it ties together to make the model work in situations where good and/or bad things happen? I hope so! If it helps, it took me nearly 30 years to get to this stage, and there's a lot of research behind it, which you can find in the last bit of the book. For the sake of completeness, we suggested in Marsh and Briggs (2015) that it might look a bit like this:

$$f = \frac{\text{Cooperation\_Threshold}_x(y,\alpha)+T_x(y,\alpha)^t}{\Xi_x} \times (\text{Regret}_x(\alpha) - \text{Regret}_y(\alpha))$$

That is probably self-explanatory except for one thing, that interesting little Ξ. Firstly, it's the Greek letter Xi, which is pretty neat. Secondly, it is definitely something that x is 'thinking' about (because of the little x that is there. The most important bit is what it means, which is: **how understanding** x is about the whole thing. The more understanding they are, the less effect the whole thing will have on the trust reduction after a betrayal. It is a number and you get to pick it to dictate how understanding your agent is. Yet another way to personalize the agent…

One last thing that we need to get to is that whole thing about forgiveness. As a matter of fact it's a little bit related to that understanding bit, in the sense that it introduces another kind of 'trait' for the agent that explains why some things happen the way they do.

Think back to the last chapter. We looked at forgiveness as a way of rebuilding trust after bad things happened, and we even had nice graphs and that. I also talked about the work Pitt and others had done in the area where forgiveness in online people was concerned (it takes a while, needs apologies, that kind of thing).

Well, consider yourself fortunate because now we get to use our regret things again! We also get to think about what forgiveness might actually look like to a computational agent.

First things first. We are going to assume that:

1. Forgiveness may not happen at all;
2. If forgiveness does happen, it will be after some time that is dependent on the severity of the betrayal; and
3. If and when forgiveness does happen, trust can be rebuilt at a speed that is dependent on the severity of the betrayal.

These are my assumptions. You can make your own if you wish. For example, you might change number 3 to say something like "if and when forgiveness happens it retires trust to its prior level" (which is pretty much what Vasalou et all did post apology in their work). There are lots of ways. You choose. Sound familiar? (It is a bit like a choose your own adventure, I agree.)

By the way, assumptions are a key part of things like this, if not much of science. The first thing we do is say what we believe to be the foundational bit – the things that need to be true for us to be able to carry on with the rest. Sometimes that means we remove some aspect of things like behaviour that we are worried will make things complicated, sometimes we make assumptions like “humans are rational actors”. It helps to clear the path forward. It also allows others to come and say “well, that assumption is a bit rubbish, what happens if you get rid of it or change it?” To which you can say “I don’t know, figure it our for yourself.” And that.

Given those three assumptions then, here’s what we can do with forgiveness. The first thing is say “well, it’s going to take a while” and this looks like:

$$T_x(y)^{t+Ft_x} = T_x(y)^t + Fk_x$$

Let me explain. Firstly on the left hand side we see $T_x(y)^{t+Ft_x}$, which means that the General Trust of x in y after a time equal to now (that’s the t) plus Ft_x something will happen (which we will get to). That is what we decided to call the **forgiveness trait** of (in this instance) x. Remember I told you there was another trait thing going to happen? Here it is. What this means is that there is a sort of “how forgiving” measure for the agent. If you think about it, there’s probably one for you too. If you think about it a bit further, you might realize that it depends on who you might be thinking about and what for and so on, and you will see that this one here is a little too simple to map out humans.

It is, but it’ll do for our model right now. I quoted Lieutenant Commander Data in the previous chapter talking about 0.68 seconds. Ultimately, this will arrive at something like that: a length of time that makes sense in the circumstances to the agents involved.

One caveat to notice here: the higher this number is, the longer it takes to start forgiving. This means that the higher $Ft_x$is the longer it will take to forgive, which means that it is a measure of how unforgiving the agent is. Bear that in mind.

Then on the right-hand side there’s: $T_x(y)^t + Fk_x$. The first item is self explanatory: how much does x trust y **right now**?

Stop there a moment. Consider that something bad just happened. What have we already done? Yes, reduce the trust. So this $T_x(y)^t$means the trust after we’ve taken into account what happened (it might be pretty low, to be honest). Also bear in mind item 1 in our list above: forgiveness might not happen at all. This might be because the agent just doesn’t feel like it, or the trust went below our limit of forgivability, things like that. We’ll come back to that. For now, assume that we will start the forgiveness process. So this leaves us with how. The Fk_xgives us the answer. Remember point 3 in our list. The rate of increase depends on the severity of the betrayal. Bad means it’s going to take a long while, less bad means it might be quicker, and so on. In our equations we also mixed in a bit of regret (which is akin to the apology that Vasalou et al talk about). And so, we arrive here:

$$Fk_x = \frac{(\text{Regret}_y(\alpha) - \text{Regret}_x(\alpha) + I_x(xy))}{Ft_x} \times T_x(y)$$

Which means that how quickly forgiveness happens depends on the apologies, regret, the importance of the relationship, how much trust there was in the first place (when the thing that was bad happened) and that good old forgiveness trait. Remember it was really how unforgiving x was? That’s right: the less forgiving x is, the lower this number is going to be, and the slower forgiveness will be.

One very last thing about this. I have talked about the limit of forgivability before but I haven’t really said how to figure it out. In the work we do here, the limit of forgivability is very closely related to that forgiveness trait. One way you could calculate it would be like this:

$$\text{Limit\_of\_Forgivability}_x = -(1 - Ft_x)$$

Which if you look at it says "the higher that number is, the closer to 0 that limit of forgivability is" which is the same as saying that the higher the number is, the much less likely forgiveness will happen at all after a transgression.

That Pesky Zero, That Even Peskier +1!

Much of what is written here begs a question. We can calculate all kinds of things about trust and distrust, forgiveness and so on, but there's this very special value in the middle which is complicating things. What exactly does 0 (zero) mean in this model? It isn't trust and it isn't distrust. It could represent the fact that the truster has no information to make decisions. It could mean that the truster is ambivalent. It could mean, frankly, pretty much anything. I've glossed over it a little in the text above, but I wanted to point at it and say "this is something that you really need to think about.' As it happens, I have. The concept of having a zero trust value is discussed in my PhD thesis (which is lucky). I decided that it could mean, well, whatever the context wanted it to mean – lack of information, a decision not yet made, a *specific* decision made that, for instance, refuses to consider anyone in a given situation as trusted or distrusted until they are asked (or they ask) to do something – this last one is basically what the concept of Zero Trust in security is about.

It's basically your call. Isn't freedom to choose grand?

Now I did say earlier that I would come back to those interesting brackets for [-1,+1). They basically mean that there is no +1 trust. "But," I hear you say "of course there is, what if you trust someone completely?"

Let me explain. Belief in a deity, or some supreme being is what we're talking about here. It has a name too: faith. It has another name – blind trust. What does this mean actually? It means that the truster has made a decision to trust completely *regardless* of any evidence, in fact. in the absence of any evidence one way or another. Trust is a consideration of risk in a situation (as you hopefully spotted) but in a situation of blind trust there is no consideration, there is just acceptance. This isn't in line with how we (or others like Luhmann or Broadie or Dasgupta) think about trust. In fact, it simply *isn't* trust at all, in the sense that consideration of the other is made. If you enter a relationship without considering the potential pitfalls, this doesn't make you safer, it makes you credulous.

I've nothing against faith, demanded by a deity or simply freely given by someone to someone else, but it's not trust. Do I trust my partner Patricia completely? Of course. But this is tempered by experience, time, behaviours and so on to become *faith* and *belief*. There are, in other words, situations where I don't have to think about trusting her because I already know *I don't need to trust* her – there's no risk.

I imagine that kind of belief is true for people who have faith in their God.

Note carefully that this is not the same as control. As Cofta notes, trust and control are different, but they can result in similar outcomes as faith. That is, if I control completely something else, and there is complete transparency in this control situation, then I don't need to think about trust then either because I *know* what I want to happen will happen. Is this another +1 situation? Well, no, because in the +1 situation you are putting yourself completely in the hands of 'another.' Whereas in the control situation you aren't, you just *know* it will work because *you* are making it so.

Now this is all my opinion of course, based on observation and reading of other works. You may well disagree and that's also okay. As an exercise, consider and try to describe a relationship where that +1 exists as *trust*.

On Other Models and Representations

You may have noticed that the model I presented in the this chapter was my own – that is, the one I have worked on for man-years. This is so, but as you may also have surmised, this is not the only trust model out there. In fact, there are hundreds, possibly even thousands, of them. Some are dedicated to eCommerce environments. Some are used in the context of information retrieval and search. Some are logical, some much more deeply advanced mathematically than that which I presented in this chapter. Many of them work toward attack resistance, whilst others try to more accurately model the ways in which humans consider trust-based decisions.

And all of them are valuable additions to a growing field.

Here's the thing though: presenting them all in this chapter is impractical at best. There are so many, to start with. The fact that many of them are highly contextual (like trust) and applied in a narrow area of use is something slightly more problematic. There are general models of trust (like the one in this chapter) but they aren't hugely popular because they don't answer specific problems particularly elegantly. They are not really meant to, but like general purpose computers, they can almost certainly be 'programmed' to make them more contextually narrow.

One of the more interesting of the general purpose models is that of Christiano Castelfranchi and colleagues (Rino Falcone, Rosaria Conte and others). It's interesting because it looks at trust from a psychological, cognitive point of view – none of the problems of values that you saw earlier in the chapter for this model. The model is goal-based (cognitive agents have goals) and has notions of CoreTrust, Delegation, plausibility and preferences in it that are very powerful and it has been implemented in a computational setting. Like the model in this chapter it also addresses things like Mistrust and Distrust. You can find out more about this model in Castelfranchi and Falcone's excellent book, which certainly holds the mantle of being the first book covering computational trust as a theory. Despite its age (published in 2010) the principles and observations about trust in the book are as current as when the book was published.

Discussion of trust values and calculations would be incomplete without a short look at Audun Jøsang's work. Jøsang created a subjective logic for trust that is based on three values – belief, disbelief and uncertainty, adapting Bayesian mathematics for *opinion-based* reasoning. He provides a set of operators for the model and shows how it can work in this excellent tutorial.

This actually brings us to the problem of how to represent trust. Jøsang does it using subjective logic's tuple of values (which has the added nicety of being representable using a nice triangle of trust reasoning). My own looks at trust as a measure in the range of [-1,+1). Castelfranchi et al's used cognitive models. Others use fuzzy logics or fuzzy notions (for instance PGP, which I talk about in another chapter). Some are simply binary (one trust or doesn't). Does the representation matter? Sort of. Words, and values, have power. They push the way you think about something in particular directions. For example, if a trust model had representations of trust between 0 and 1 may lead you to thinking that the nuances of positive trust are much more important than those of distrust (if it would lead you to think of it at all). If the model suggests that trust is from 0.5 up and distrust (or whatever) is from 0.5 down, it's equivalent to the [-1,+1) in some ways but it points you in the direction of thinking about trust and distrust as ultimately connected in ways that they aren't (well, I don't think so anyway, which is why, although there's a continuum, there's a really special spot in the middle at 0 which means, well, something special in different contexts (see for example Zero Trust in a later chapter).

The problem is: what does 0.5 mean? What does a matrix representing different tuples, or a diagram of a triangle or whatever really mean? We're still in the infancy of understanding trust and modelling it, as this chapter may have said a few times, and to be honest, it's even more complex when you start realizing that *even within the same model*, comparing values is less than an exact science. But then, as Christian Jensen says, perhaps all you really need is a way to measure your own values so that you can compare one option against another and make a sensible (more rational) decision. Sharing trust isn't an exact science in humans either. Try it. Your "I trust them completely" may be my "I trust them a lot" – they're simply not the same things for different people. This is in no way a shortcoming of trust, or the models of it. A long time ago I came up with the idea of *Comfort*, which is a way to abstract trust 'values' to allow people and technologies to share them. Comfort, I suggested, was much more understandable and shareable, so I did some modelling there too. But at the end of the day, emotions (which trust is) are *personal*, that's the point.

Meanwhile, whilst are certainly some literature reviews out there that have looked at many trust, reputation, eCommerce and so on models, as far as I can tell, there haven't really been many published in the recent past, which is something that is worth thinking about doing. Maybe it's something you, dear reader, might want to attempt…

Final Words

This has been a bit of a marathon chapter but at the end we've (hopefully) come to a really strong understanding of how things might work with trust in a computational agent. As you have seen along the journey, there is an awful lot of "you figure out what to put there" here that essentially makes the end result extremely tailorable and therefore really able to match what you want it to look like.

One last note about that, then, because it matters. A problem with many ecosystems is similarity. The more similar things are, the easier they are to attack (which is why the pandemic we experienced in 2020-2021 was so serious – we're all so very nearly the same and none of us could handle this new thing). Trust is like that. The more similar it is across people, the easier it is to attack. We have the potential in a computational system to make it work in lots of different ways, and to think about many different things that matter just to one agent. This makes it harder to figure out which bits to attack in the different agents. It's a relatively simple defence mechanism, but I'll take it.

Of course, one of the reasons people like conmen thrive abusing trust is that we, as people, are so embarrassed by the whole idea of being taken for a ride ("chump", "sucker" and so on are not compliments). Why? Because as I said earlier in this tome, we think we're good at this stuff. We think we're experts. No expert wants someone to come up and say "you're wrong, look at this squiggly bit here, it completely messes up your whole theory". Oh, all of us scientists like to say "science is all about theories that can be disproved" and so on, but we'll still fight like demons to defend those same theories when they start getting looked at closely. As in science, so in trust and the general population. Except that, when we get suckered, we clam up and don't tell anyone, fighting that little battle inside our own heads over and over again. This is of course unhealthy but oh so very human.

What can we learn from this? Computational agents don't have an ego to crush (at least I don't think they do right now, and if they did we'd deal with it then). They don't care about embarrassment. So they can speak out about when they get attacked, if they spot it. And the others around them can make adjustments, and censure the attacker. And so on. But there's more to it than that. If you recall in the previous chapter I started getting a bit philosophical when I discussed *wa*, calculus-based, knowledge-based and identification-based trusts and so on. All of this has to do with reputation, recommendation, honesty, truth and more. And some of that we'll look at in the next chapter. See you there!

HOW POPULAR ARE YOU?

Figure 6.1 Popular Much?

Does being placed #1 mean you're popular?

In this chapter we're going to look at recommendation and reputation. There are similarities, but there are also fundamental differences between the two. In a nutshell, recommender (recommendation) systems tell you you will like something (or someone perhaps) based on some set of measures. Reputation systems, on the other hand, essentially tell you who or what is popular based on, you guessed it, some set of measures.

If you head to someplace like Wikipedia or even did a search for *Recommender Systems* (on DuckDuckGo of course), you might find that the definition goes something like this: the system looks at new stuff and tries to predict how you will rate it based on some set of measures. I've never been a fan of this definition because it implies that the systems are passive, that they just sit there and say to themselves, "Oh, Steve would rate this one 4 out of 5" and there we are. They don't. Recommendation systems are not passive. Quite the opposite in fact. Sometimes they are also quite insidious, guiding you toward things that the system predicts you'll like based on what it already knows about you.

Consider that for a moment. If I happen to like comedies featuring, say, Mike Myers, a recommender system might push me towards Austin Powers (I like Austin Powers), or even a comedy with some other SNL comedian starring in it, or perhaps a movie like Shrek. All pretty innocuous. What if I one day 'like' a page that has something to do with election fraud, or one that has to do with Antifa, or whatever? The system may well quite happily lead me down the rabbit hole that ends in the situation we saw on January 6th, 2021 in the

USA (if I keep coming back to that, it's because it is important). It's a rabbit hole because you don't even know you're going down it and then it's too late. This is not to say that recommender systems don't have positive effects, but it does mean that they are power tools.

What has a power tool got to do with anything, you ask? Let me digress a little to an example I use in class occasionally. Imagine you wanted to put a hole in a concrete wall. You could do it with a screwdriver (I've tried, the result isn't that great but it's a hole and you end up with a useless screwdriver afterwards (needs must)). You could do it with a hand-powered drill type thing. Takes a while. Or you could use a power drill, maybe with a hammer-type action. Much better hole, much quicker.

What if you wanted to take a tree down. Why? Because it's dead of course, or because you are coppicing. You can use an axe. You could use a saw of some kind (there are plenty of different kinds) and build those arm muscles. Or you could use a chainsaw (like in Figure 6.2).

Figure 6.2: Steve and his chainsaw

Much quicker, much cleaner cut, and so on. How about if you wanted to go visit a friend in the town down the road? There's Shanks' pony, of course. It works but takes a while. You could (if you are lucky like what I am) ride a horse or you could resort to the power tool that is your motorcycle. Much faster, a different kind of fun, possibly even cleaner (mine's a Zero).

What about if you wanted to know what was happening in your little world? You could run around town asking people you knew. You could even email people you think like the same kinds of politics as you, or people you think know you well enough to tell you stuff you know would be worthwhile to you. Or you could use Facebook. Or Twitter. Or Parler. Okay, maybe not Parler, we live in hope.

What happens if your hand slips when you are using a drill? I've had a hole in my leg once. No, really. Don't want that again. Could be worse. What if your attention slips as you are rounding a corner on that bike, or if someone else's does in the car coming round the corner the opposite direction? What if you get distracted by your 5 year old running towards you when you are using that chainsaw? What if you aren't thinking enough when you look at your Facebook (or whatever) feed? The results can be damaging, if not disastrous.

Tools like recommender systems are power tools for your imagination, power tools for your mind. As with any power tool, use them right and you will get what you want more cleanly and more accurately and almost certainly much faster than if you didn't use them; use them wrong, or use the wrong one, or simply drift in the wrong direction some day when you weren't paying attention, and you will find yourself in a metaphorical corner with no real happy way out.

You have a brain. Use it.

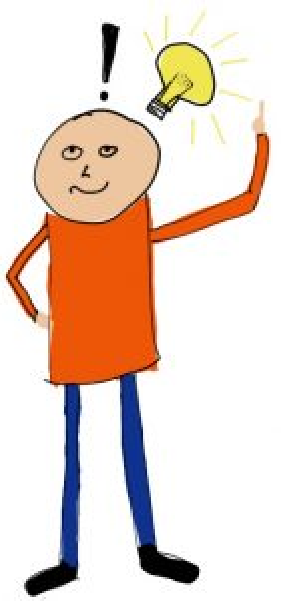


Figure 6.3: Use your head!

The old adage goes, just because you can do something does not mean that you should. We can make marvels, algorithms and tools that will perfectly predict what you will want to see or do or read next, or lead you down that rabbit hole faster than you can say, "election". There is a way to do one without doing the other, but we (and the people who use us as products) seem to have become rather lazy. Treat the things that come to you from these systems with a healthy respect. It will benefit you in the long run.

Back to the original programme.

It's entirely possible for a recommender system to use some form of reputation as a means to determine what to recommend – after all, it kind of makes sense that recommending something or someone popular is probably a safe bet.

However, if you take a step back and think for a minute you'll also see that it makes sense to recommend something based on the fact that you will like it. That's not necessarily popular, but it can be *similar*. Similarity is interesting because it works in pretty much two ways:

- Similarities with something you like; and
- Similarities with something you are.

On the other hand, reputation systems deal with popularity in a few different ways. Ultimately popularity depends on who is looking and who is asking the question. For instance, the popularity of a person may be based on the music they make, or the pictures they draw. It may depend on the specific kind of work that they do, or the ethic they have toward it. For a node in a network, popularity might be measured in speed of throughput, uptime, things like that.

Before we go more deeply into what all of that means, let's address one question: are these Trust Systems? After all, that's what this book is about. The short answer is, no, they're not. They don't calculate trustworthiness, trust, anything like that. They don't reason about trust or trustworthiness (although a reputation system may revolve around perceived trustworthiness – that is, trustworthiness can be a popularity measure – and recommendation may be based on trustworthiness in a specific context). So, what they are then is ways of helping trust reasoning to actually happen. They give information to the trust reasoner. For instance, if I was trying to figure out which of two different airlines to fly with, I might look online for a system which rates airlines to help me (like Yelp or Travel Advisor, or something like that). These systems provide reputation measures and/or recommendations to help me make the 'right' choice. Why 'right'? This isn't actually a normative thing. Because it's all contextual, yes? I might be concerned about the comfort of the seats whilst someone else might worry more about the on-board food (this is really not something to get excited about, but you see the point). The systems we use in this context provide us with information to help us make our decision. You might say 'choosing an airline is not a trust decision' and you'd be right. Giving them your money for a flight 8 months into the future might be. Ask people who bought tickets at the start of 2020 for travel in, say, June. Refunds are, well, not so easy to get. Coupons in return for money. Is this legit or a scam?I digress. Moving on.

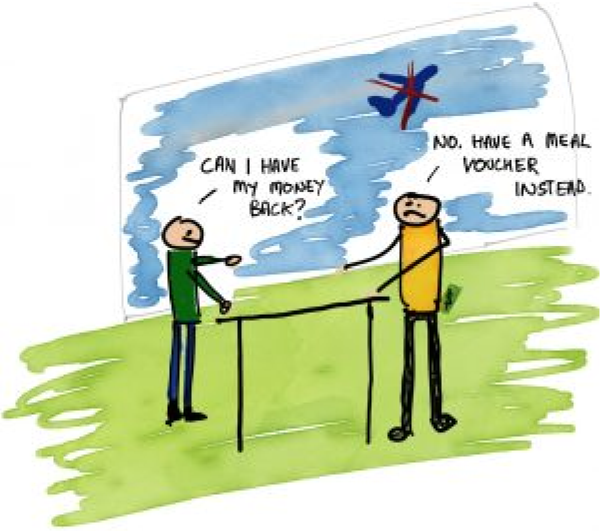


Figure 6.4: Fancy that.

Actually, the previous discussion brings up that trust empowerment versus trust enforcement thing I talked about a little in the Can We See It? chapter. It's like this: recommender systems could be seen as Trust

Enforcement tools: they kind of tell you what to pick. That's a fairly strict reading of what they do, but in some instances they might not even tell you why they are recommending the thing they are. Much of the time these calculations are secret – after all, they are the basic means by which a company makes money – good recommendations mean you will keep coming back for more (at which point they can do things like track you, advertise based on your likes, which are based on your searches, and so on).

If that wasn't enough of a clue, think about **PageRank**.

That's right. PageRank is essentially an algorithm that works out which page to recommend to you based on the search you are doing and the popularity of the pages out there on the web. There's a paper from a while ago describing it, but the real trick is that Google tweaks the model all the time (they use many different algorithms) to beat people who try to game it (we'll talk about attacks and such later). In a nutshell, PageRank calculates the reputation (I know I said recommendation, bear with me) of the pages on the web in terms of popularity, like this: each page's popularity is a function of the number of pages that link to it and the popularity of the pages that link to it. If that sounds a little tautological, don't worry. Think of it like this: if I'm a person who everyone thinks is really awesome and important, clearly other people will want to, you know, be my friend. Ask Kevin Bacon (we will get back to him). If lots of important people are my friends, clearly this is a sign that I am even more important than you previously thought.

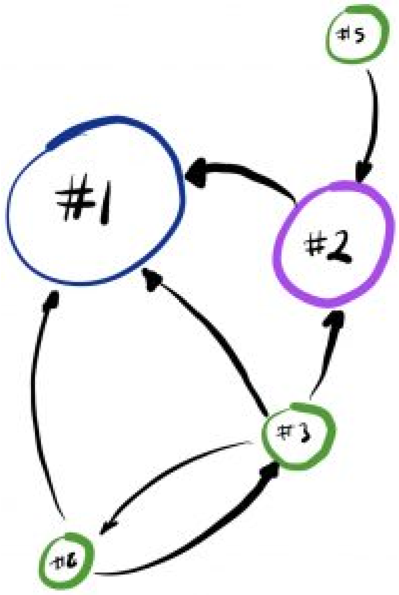


Figure 6.5: PageRank (more links = more popularity) [thanks to lots of links, #1 is the most popular, and so on...]

More popular means more visited, more linked, more trusted? In web page terms, these friends are links. Google's home page is hugely popular – it's certain that there are a huge number of pages that link to it. These can be individual pages or sites, right up to sites like Facebook or Instagram or whatever (doesn't matter – the thing is that there will be some hugely popular sites that point at Google). It doesn't link to that many itself (at all) though. But that doesn't matter either because its popularity is high because of who links to it. If the Google page pointed toward another page, it would be quite important simply as a result of that link. But if the Google page was the only page that linked to it, then that page wouldn't be as important as the Google page because it's about the number as well as the importance of the links.

Alright, so in this particular example, we were looking at Google recommending pages to you. The list of results it gives you is a rank of the pages from most to least important based on the search you do. The top ones are the recommendations. Not that many people go to the second and third pages or beyond of search results. Why would you? Google has you covered (I use DuckDuckGo, for full disclosure). But that recommendation is based on the importance of pages, which ultimately maps to a sort of reputation. Pages don't link to other pages at random, right? They link because the page they are linking to has something relevant to say about the thing they are linking for (talking about). However, as in all things, this might be a negative thing. Imagine if all those pages pointing to the Google homepage were actually saying, "this one sucks". Absent some form of intelligence you may well get the same important result. This begs a question: is a hugely unpopular page/person/whatever important or not?

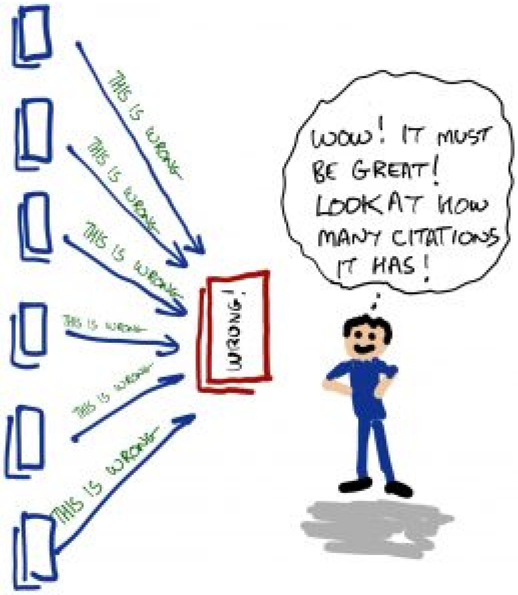


Figure 6.6: Wrong, but good, or bad?

Here Steve assumes that if a website has multiple citations, it must be a great resource. Having many of something does not mean it must be good, or right. It is better to have one, but accurate.

There are a lot of philosophical questions wrapped up in all that which you can feel free to explore in your own time. It is the case that there are different kinds of recommender/reputation systems that look at this all quite differently, as you will see.

Is there mathematics involved in the Google model? Sure, like pretty much any reputation or recommendation system. I'm not going to go into it here. Remember the start of the trust model chapter? Mathematics? Yeah. Seriously though, much of the work revolves around things called eigenvectors and eigenvalues. One day I may explain those here.

Phew, that's the 800-pound gorilla dealt with. I'm sure it'll pop back into the scene someplace though. Watch this space.

The discussion did bring one important thing up though: reputation and recommender systems sometimes go hand in hand. That is to say, recommender systems sometimes use some form of reputation in the way they figure out what to recommend. Bear that in mind as we move forward. In the meantime let's go right back to where we started this chapter: the two systems: recommender and reputation. For the sake of simplicity, let's have a look at recommender systems first. When we verge into reputation territory, I'll let you know.

So, recommender systems can be broadly split into a couple of different types. We have recommendations based on content-based filtering and recommendations based on collaborative filtering. This is not quite the way some people look at it, but trust me on this one.

All filtering works like this: the system 'knows' things about what you like (we'll get to that) and when something new pops up it looks at that thing to see if it matches the stuff you like. Bear with me, we'll get to the matching stuff too. If it does then you will get that thing recommended to you. *Simple*.

There are two key things here. The first is that the system knows something about what you like, and the second is the way it matches those likings with the content of something new. Let's explore them in turn and we'll get back to the content/collaborative distinction.

The content-based filtering does matching by looking at something you tell the system, either explicitly or implicitly, as we will see, and then figures out if the content of the thing matches. This can work through a couple of different relationships. The first is pretty simple – it's based on the relationship that you have with the kind of thing that is being recommended. Imagine for instance that you love Dungeons and Dragons, like my son Owain does. Knowing this means that if a new D&D book or game or movie or whatever comes along, he may well get matched with it. Think of it like your favourite aunt who knows you like cats and so gets you a pair of cat slippers for Christmas.

Figure 6.7: Owain, Dungeons and Dragons, and Socks. Ever wonder how different sites start giving you recommendations for products and brands you have previously searched for?

The great thing about this is actually that content is not as much an issue as our next kind of match. However, the system still needs to 'know' something about you and something specific about the thing it is recommending to you as a result. If Owain was more or less specific in his likes he might get recommendations only for new D&D adventures on D&D Beyond or recommendations that included stuff about Esper Genesis. The system needs to know what these things are about, and so there's your content stuff. The relationship is between the person and the content of the product. To put it another way, the system makes a link between what Owain likes and all kinds of different things he *might* like as a result.

It gets more complex though when we start talking about things like the content of different kinds of things and building recommendations based on that. For instance, say Anika (Steve's daughter) loves Harry Potter books, a good book recommender system might bring forward books in the fantasy genre, or the magic genre specifically, or schooled wizards and witches (it's a thing). There's little point in recommending Harry Potter books. She has those. You want to sell her other books she might like based on the ones she has already liked.

I LOVE HARRY POTTER BOOKS
TRY READING MAGICAL CREATURES ACADEMY. CLICK ON 'BUY NOW'!

Figure 6.8: Anika and the Mysterious Book Recommendation.

The filtering is solely based on similar past products/brands and linkings. Did I mention this was a selling thing? Oh yes, recommender systems are all over eCommerce.

The relationship in Anika's example is pretty simple too – it's between books she liked and similar books that she might like as a result. The products are similar, which they aren't in Owain's case. The filtering and recommendation is still based on the content – the recommendation is based on what the thing being recommended is about in some way.

Here's a different one. My other son, Layne, is very much into his fish. He also likes Marvel movies and TV shows. And The Tick. He's quite fond of chess too. If he was to be online looking for something fun to buy or watch, a recommender system based on collaborative filtering would basically try to find someone else (Person X) in the system who likes fish, chess, Marvel and The Tick and also something else, like a new movie that has just come out and they watched last night. Because they liked that something else and they also like the other things Maybe likes, well, there's a fair chance Layne will like that something else too, right? Right.

In fact, it doesn't matter what the content of that something else actually is to the system at all. All that matters is the similarity between Layne and Person X. This is pretty neat because (as with the first example with Owain) all kinds of different things can be recommended, from games to socks, but also because they can be recommended not just because they are D&D related, but might come completely out of completely left field and recommend something Layne had never thought of: a Marvel movie about fish socks, for instance.

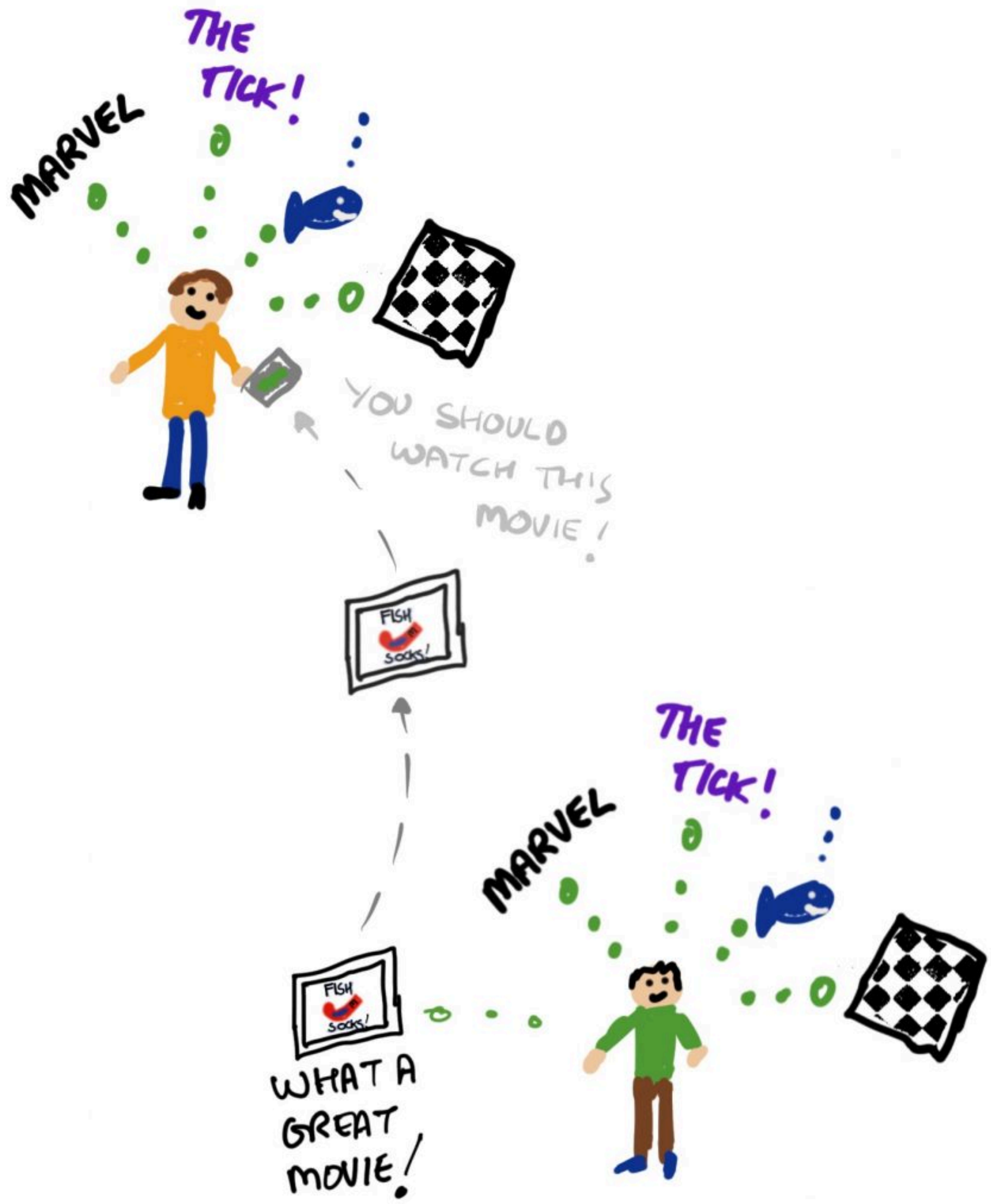


Figure 6.9: Layne and the Marvel Fish Sock Movie (I couldn't fit chess in as well, sorry).

In the first example, with Owain, the system needed to know something about Owain (the user) to be able to recommend something new (which had to do with a specific thing). In the second example (with Anika) the system needed to know something about Anika's preferences, the topic (the genre) as well as the content of the things it was recommending. In the third example (with Layne) the system needed to know a bunch about Layne – and not just that he was or was not interested in D&D or Harry Potter books (although perhaps the Marvel Fish Sock Movie is a bit of a stretch). By now you will probably have realized something about recommender systems. They need data! If they don't have it, they can't do anything. The more the data, the better the recommendations.

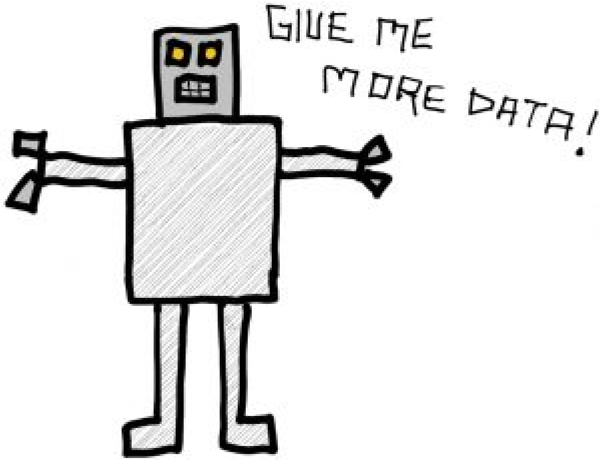


Figure 6.10: Your basic recommender system (sort of)

Technology knows how to get around with knowing more about you! All it wants is data**.**

Think about Netflix for a second. What differentiates them? What does the company do? Stream videos to you? Not really – they use Amazon's Web Services for that, and anyway, anybody really could do that with enough money (as Amazon Prime TV, Disney Plus and a host of other services show).

No, what Netflix does is recommend movies and TV shows to you *based on what it knows about you.* The better it knows you, the better the recommendations get (just don't let your Gran watch her stuff with your profile!). The longer you stay with them, the harder it is to leave (I mean, it's a pain having rubbish shows recommended to you as the next system learns stuff about you).

Just saying. These services need data.

How do they get it? Ratings? Well, that would be a fine thing. I mean, who really rates stuff anyway? Not that many people actually. Netflix can use a bunch of analytics to determine if you watch a movie or TV show to the end, if you binge watch a show, and so on, which is almost as good as ratings because if you watch it, that kind of means you enjoyed it (or were too tired to change the movie). So Netflix gets that stuff there. It also gets it if you rate the movies you watch. Noticed how *easy* it is to rate them? That's because they would really like you to do it. Some services ask questions when you sign up. Amazon Prime almost certainly uses the purchases you made from Amazon itself to help guide its recommendations. There's a lot of rich data there.

Other places also make it really easy to give ratings: click on stars, for instance. In the case of social media sites, you may 'like' stuff or you may follow someone (and the system knows what they like, and so on). All of these things are things that you do. They are explicit ratings.

But hey, let's be honest, none of us are really that excited about clicking on a bunch of stars when we have the thing we wanted. There's a reason why well-designed bank auto-tellers give you your card back before they give you the money. How, then, to get the data from more implicit means – things you do that are associated with the data but don't require you to do extra work (or thinking)? Remember that Netflix thing? Watching a movie is a 'like' that costs you, well, the time to enjoy a movie. Clicking on links to get places, or ads when shown on a web site, or purchasing things are all of them rich sources of data for the recommender systems. Because you don't explicitly go out of your way to rate them, these are implicit ratings. They're still pretty good though: after all, spending a couple of hours of your time watching a movie is a commitment of sorts. There's also a reason why we say 'pay' attention. It's valuable. Buying something, actually spending your money instead of just your time, is also valuable: you put your money where your likes are.

Figure 6.11: Well-designed ATMs do not let this happen. Bet it happened to you!

What are the downsides? Well, the real one for the recommender systems that do the person to person matching (Layne and his socks, remember?) is that you need a bunch of people with their interests to join in before you can give really good recommendations. That's what's known as a **cold-start problem**. It's something which trust systems other than recommender systems suffer from too, which we will get to shortly. It's also something that the other two schemes don't suffer from. In Anika's case all the system needs to know is her preferences and the (already available) books and genres around. Likewise with Owain's D&D, he could be the only person in the system and still get good recommendations about all kinds of D&D products. There's also the problem of **sparsity**, which is where there just aren't enough people to ensure granularity (people who like the same stuff), which other recommendation systems don't suffer from.

But there's something else, too. Okay, so recommender systems are an interesting topic and, as I said at the start of the chapter, can help in trust-based decisions (all data is good for a trust-reasoner, right?). What I didn't talk much about is privacy. It's important, so I will now. There are obvious issues with personal privacy here. It's possibly okay that companies like Netflix have your data – after all, you gave it to them – but if companies start to sell it in order to make money, this is problematic. It's entirely possible that according to the contracts you have with the organizations this is quite legal. This does not make it moral.

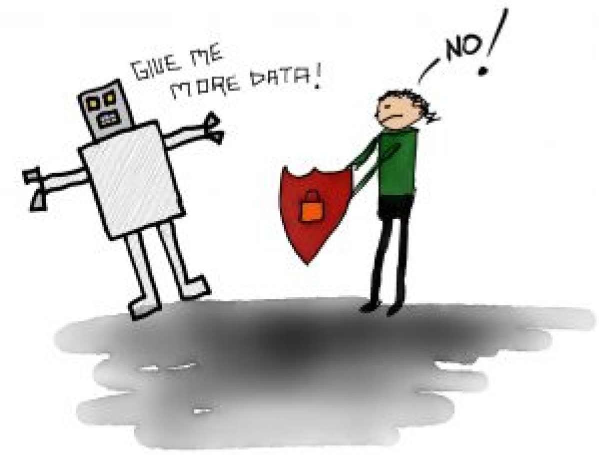


Figure 6.12: Privacy matters. Don't let it go easily.

These magical recommendations are nothing but a breach of your personal information, data, and SECURITY.

How do all these recommendations actually happen though? How are the matches actually made? There's a fair amount of interesting mathematics in this kind of thing. What the systems are trying to do is to find how similar the person's interests are to the aspects of the specific item being compared. For various reasons this isn't the place to get into mathematics but there are some really nice explanations (and pictures) here: https://builtin.com/data-science/recommender-systems.

Reputation (management) systems are a special breed. The basic premise of these systems is that one's reputation can be calculated. Indeed, everyone's reputation can be calculated. There are obvious questions around that, and let's start with the first one to get it out of the way. Can it actually be done? The most obvious answer is "yes" and there are plenty of different examples that you bump into in many walks of life. For example, your credit rating (if you have one, and if you haven't you will) is a measure of your reputation in the context of paying the bills. It's a big deal too, and it's big money. After all, banks use credit agencies every time you want to do something like open an account, or borrow money, or get a credit card. At least in Canada, mobile phone service providers use it when you want to get a contract with them. Private lenders use it. When I needed to get secret clearance for a prior job, the process of figuring out if I was 'trustworthy' enough to get the clearance used it (that was interesting! I mean, the rating was a little off for various reasons!). As you might expect there are quite a few credit rating agencies around. Some even go as far as rating countries (after all, it simply wouldn't lend money to a country that had a poor credit rating).

What other examples are there, you ask? I've already talked about PageRank, which is kind of a reputation system used to rank recommendations for web searches. There are also quite a few 'trust' companies out there that use a whole host of data, from likes and follows to that credit rating, to determine if you are trustworthy (enough for a new job, or to lend to, or to be my friend…). Full disclosure: I once did some work for one of them.

Collaborative filtering systems for recommendations, which I talked about above, are all about people giving 'likes' and so on to things so that similar things can be recommended to them. Like the PageRank thing, they're not far removed from reputation systems: reputation is how many 'likes' something has (literally in the case of a lot of social media, which we will discuss in a bit).

More prosaically, perhaps, Amazon uses a reputation system for its marketplace sellers (go take a look). eBay also uses one for sellers and buyers. In fact, a whole bunch of such sites do – it supposedly removes some of the risk involved when transacting with people you don't know.

There's the thing, right there: reputation systems are good in situations where you don't know someone. They give you additional information that you can base a decision on (see Figure 6.13).

Figure 6.13: Reputation Systems can make people you don't know more clear and reduce the risks of interacting with them.

Think back once more to our trust enforcement vs. trust empowerment discussion. Where are reputation systems? Used properly, and made sufficiently transparent, they can be empowerment tools. Far too often they are leveraged into enforcement tools, as you will see.

They are used to make sure that the person on the other side of the line is not, to coin a phrase, a dog. Because on the Internet, nobody knows if you're a dog.

A slightly more complex answer to the question "do they work?" can be arrived at if you start thinking about the past few chapters that you have hopefully enjoyed. In many cases I've done quite a lot of hammering home the point that trust is subjective – you can have all kinds of views around it, calculate it in lots of ways, apply it in many places according to your own needs, but the point is that it is a decision you make. This is just as important a point when we consider reputation systems. What is important when it comes to someone's reputation? Quite simply, it's up to you. I, for instance, quite like dogs. The systems may have their own views, which we'll come to shortly, but yours may differ. So how do you reconcile these differences? We'll talk about that too.

So the more complex answer is "it depends". It depends on who is asking, who is calculating, what they are calculating for and how the answer is used, because when you start looking at if people pay their bills on time and determining if they are good gardeners, or good security risks, there are obvious problems with the result you're going to get.

Just like trust.

Now we've got that out of the way, let's explore a few of them quickly. Why quickly? Because there's some juicy problems to discuss, that's why!

Firstly it is important to know that reputation systems are actually pretty simple. Their purpose (the dog thing) is to help people build 'trust' where there isn't any, by listening to what society thinks about the trustworthiness of a specific other person in a specific context. If you've been paying attention at all, you'll know that there are quite a few problems with that sentence.

To do this, they count stuff. That's pretty much it. They can count follows, or likes, or links to, or number of days past due for however many bills, or court convictions (and what for), or they can count stars or some other form of feedback from people who others transact with. But that's what they do: they count stuff.

And then they put all that stuff into some algorithm or other in order to come out with an answer that says "yep, you can trust this person" (naturally in context, how could you assume otherwise? Yes, that was sarcasm).

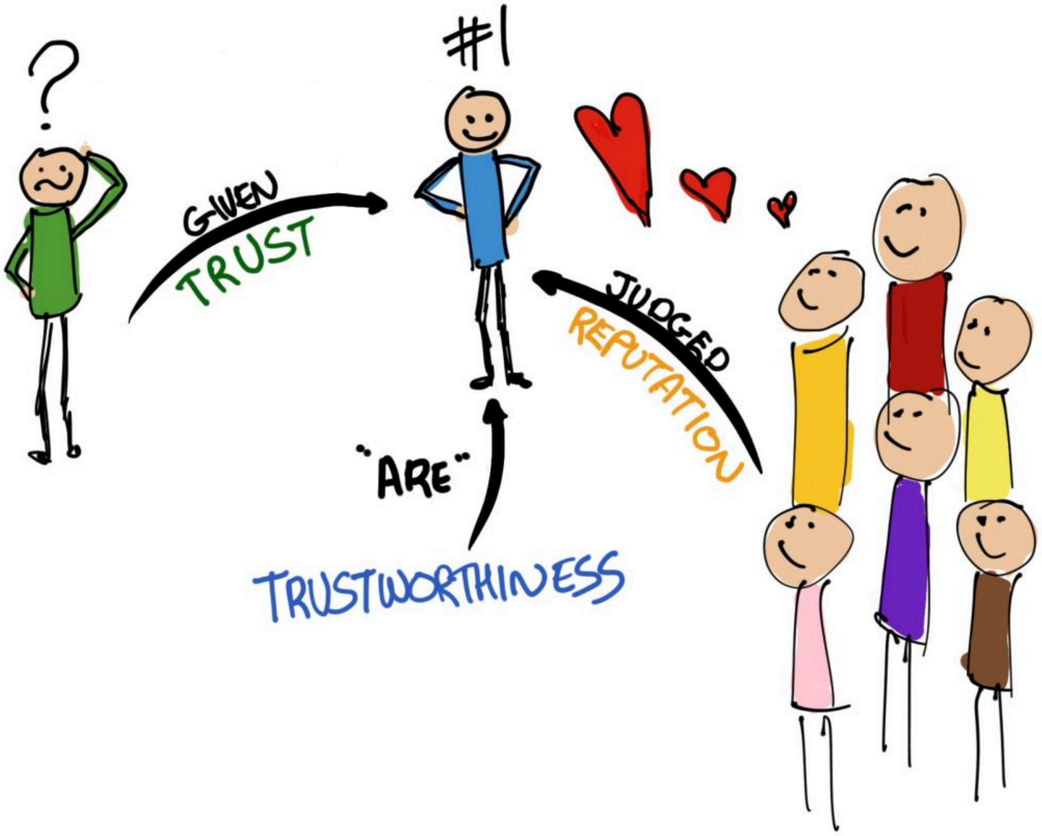


Figure 6.14: Trust is given to you; Trustworthiness is something you are; Reputation is what people judge you as.

There are indeed many examples, and some of them I've talked about just now. Here are some more… Reddit uses a form of reputation called 'karma'. Good karma opens up all kinds of possibilities and different abilities. How to get karma? By posting stuff and having it upvoted (liked) and not downvoted (which loses karma). You can be funny, or sad, or poetic or a painter or whatever, just be upvoted. Slashdot uses karma too, in a slightly different way, but good karma still gets you better abilities (and a greater likelihood of things you write about being seen). As an aside, just because you have good karma on these sites does not mean you are a source of the truth. Unfortunately it is often seen that way, with obvious problems. Various sites, such as Advogato (an open-source community social network which pretty much no longer exists, although a bunch of data from it does) used trust metrics based on peer reviews (in fact, Advogato was set up as a way of testing a specific trust metric for its attack-proof-ness by Ralph Levien. Attack proof trust metrics are sort of like a holy grail for social networks, by the way. Can it be done? Levien thought so). The thing is though, we still keep coming back to that old question: what for? What question are we actually asking when we are asking "how much is that person trusted"?

I could go on. There are many. Aside from eBay in eCommerce there are sites like Epinions. Fancy figuring out if the prof you're about to get is any good? ratemyprofessors.com may have some answers for you. For travel, you might think about asking TripAdvisor. Medical problems? ratemds.com or medicalproblems.com. Truly the world is your oyster (just ask Yelp). Speaking of profs (which we were) we can go even further because there are naturally ratings for things like how popular their research is. Based on citations this time, there is Google''s own i10 index, which counts the number of articles with more than ten citations. The h-index (invented by the physicist J. E. Hirsch) is something pretty much all profs think about if they do research, it goes like this: a h-index of n means that the researcher has written n papers with at least n citations. As you might imagine, a high h-index not only means you write a lot but that what you write gets cited by others (which is good) or yourself (not quite so good – I mean, self-promotion and all that, right?). See a problem with this? I do (see Figure 6.15). I mean, if you're just starting out, when people really look at this stuff and care, what is your h-index going to be? If you only published one paper the maximum it could be is, well, one.

YOU CANNOT HARM ME! MY H-INDEX IS MUCH LARGER THAN YOURS!
H
BUT I ONLY JUST STARTED MY RESEARCH...

Figure 6.15: h-Index. My money is on the young researcher. Eventually.

Everyone, experienced or interns, all deserve a fair chance. Having a larger h-index does not mean success, does it?

To address this problem, Johannes Koelman introduced the Einstein index. You see, Einstein published stuff but the ones he published in his early career were groundbreaking. They are so massively cited that they're probably off the scale. Trouble is, if that scale is the h-index, Einstein as a 'starting' scientist sits at, well, 3. Koelman's Einstein index counts the citations of the scientists' top three cited articles (not books) and adds them up. High Einstein index means, well, good things. Koelman talks more about indices here and it's worth a read. Notably it talks a bit about the problems behind these indices, and the reasons they won't go away even if we wanted them to.

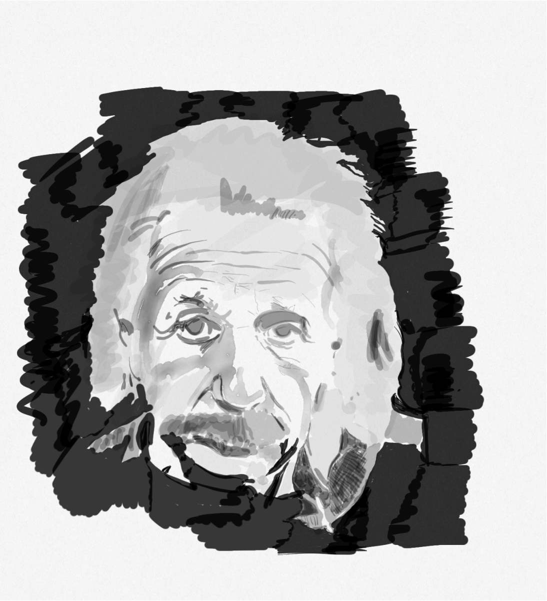


Figure 6.16: Looks like Einstein to me.

Why did I focus on academic indices for reputation? Because I am one. Talk about what you know, eh?

There's actually another one which is quite close to the h-index, called the E number (for cycling – there's also one for astrophysics, which isn't this one), after Sir Arthur Eddington, who was something of a rockstar astrophysicist at the start of the 20th century, and who played a large part in popularizing science, which is a noble endeavour. He was also quite the avid cyclist. The E number for cycling is a number E such that the cyclist has ridden at least E miles on at least E occasions (days). So, to get an E number for cycling of, say 80, you'd have to cycle 80 miles or more in at least 80 days (fortunately they don't need to be consecutive, which would make things quite challenging!). Apparently Eddington's own E number for cycling was 84, which is pretty good and better than I will ever manage.

There's another reason for telling you this stuff. Both the h-index and the E number for cycling have a couple of things in common. The first is that it takes a lot of time to get to a large number. Unless you pump out loads of highly cited articles every year, your h-index will only grow by a maximum of the number of articles you do write (and only then if people actually cite them – which is dependent on them being any good!). So early career researchers are likely to have a low one. The same applies to the E number for cycling – not only does it take time to build it up (you have to cycle for at least a certain number of days to get a number that large, it takes effort, since you have to cycle that number of miles in each of those days. If you're just starting out, you're not going to have much on an E number – my own is around 10, and I don't like cycling that much so it'll probably stay there.

Figure 6.17: Sir Arthur, out for a ride.

Why is Sir Arthur thinking of the number 82 while riding a bike?

This is important because it's a behaviour reputation systems exhibit. The longer you are in the system, the more your reputation can grow (or not, depending on the kind of person you are in that context!). For instance, if you are new to borrowing money, the advice is to get a small credit card (small as in low limit, which is all you're likely to get anyway with no history), use it and pay it off each month. After a time, you build up a history of being a good payer, and so it goes, you have a credit history which is good. It takes time.

On the other hand, it can also be skewed strongly in the wrong direction too. Imagine a site which allows you to rate your farrier, for instance. We'll call it ratemyfarrier.com (if there is a site like this when you are reading, remember I said it first). Like pretty much all these 'rate my' sites, the farrier is put in the system by a happy (or angry) customer to rate them if they are not already there. If it's been around for a while, there will likely be a bunch of older farriers on it, with a history of being kind to horse's feet, or of being quite mean and expensive. You get the idea. Now if I was a new farrier (are you kidding me – that's hard work and skilled too!) And went to a farm and did an amazing job, this would happen: the farm owner would go to rate farrier, login and search for me.

Figure 6.18: Steve the Farrier. Horse not included. Get your own.

Not finding me they would enter my name, a date, probably, and a rating ("5/5 – Steve was really kind and gentle with the horses and helped a lot with advice on a pony who has foundered."). I got 5/5 stars! I am the best farrier on the site! Because, let's be honest, even the best farrier will have someone who doesn't like them, right?

Does that seem fair? Does it seem fair that a potentially really good researcher with few but amazing publications has a lower h-index than a mediocre one who strategically placed a few poorly written papers in highly cited journals and piggybacked off them, perhaps even with lots of self-citation?

OF COURSE IT'S NOT FAIR! (Sorry for shouting).

I can't do much about the h-index thing (I quite like the Einstein index idea though, and if you combine metrics like this you find some good evidence for quality as well). And my career as a farrier is pretty much over now anyway.

So how to manage this stuff? One obvious way you may have already spotted is to use time as a mitigator, or perhaps number of ratings. If the person looking at rate farrier gets to see the number of people who have ranked each farrier they will get much more context. The site could even show them the number of good and bad reviews they got as well as the average stars. Which is exactly the kind of thing you see on eBay for instance. People can leave positive, neutral or negative reviews, and the rating of the seller goes up by 1, stays the same, or goes down by one depending on which of these is left. The result is shown as a percentage (the percentage is the number of positive versus negative reviews, so if a seller has a 90% rating, they will have had 90% positive reviews). You can dig further and see comments, things like that. Amazon's marketplace is similar. There is also information about the length of time people have been using the site.

Uber has a reputation system too. You take a ride, and then you can rate the ride with up to 5 stars. If you rate with less than 5, you get asked why. Drivers can rate riders too. Riders can see the ratings of drivers around and choose one they want based on that. The rating shown is an average of the last 500 ratings. This means that it's pretty volatile until you get a couple of hundred drives under your belt, should you choose to be an Uber driver. It's your call, that's all I'm saying. Things out of your control (traffic jams and so on) are not counted.

Here's a thought: Why is it that we are seeking such perfection? It's not like every student I ever had gets 100% in every course they took. Why not? Because we're all learning. Every day. We all make mistakes. A very good rating to me is around 4 out of 5 stars (80%, it's in the 'A' range, right? That's pretty good, right?). So why does Uber imply it isn't? Why do we all as users of technologies like this insist that the only thing that's good enough is 5 out of 5? How does life work that way? The system results in a couple of potential things happening:

- Either everyone gets (and gives) 5 stars, or
- Everyone is a failure.

Figure 6.19: Only 4 and a half stars? What a failure!

Josh in Figure 6.19 is sad. Does success mean 5/5? This actually brings up another problem with trust and reputation representations: mine is not yours. My 4 stars may be equivalent to your 3 stars. My 0.5 trust may be equivalent to your 0.7 trust. My "Pretty good" may be equivalent to your "Wow! Amazing!" It's subjective, right? Bear this in mind.

The creators of such systems would suggest that the crowd is wise enough to be able to even out the rough and the smooth. But if everyone is either 5 stars or a failure, where exactly *is* the smooth? More to the point, if everyone is either 5 stars or a failure, how am I supposed to choose which 5 star driver is best for me? When reputation systems have insufficient granularity they are practically useless.

No, I don't have the answer. I'm just the one who asks the questions.

I ONLY WANT A FIVE STAR DRIVER!
THAT'S EVERY DRIVER IN THE CITY... BE MORE SPECIFIC!

Figure 6.20: Insufficient Granularity Error! Do not compute!

Is reputation always correct? Always deserved?

There's a saying, which may or may not have originated from Voltaire: don't let the perfect be the enemy of the good. Although many people would rather it didn't, I believe it applies here. Reputation systems are at such a pass that it's almost impossible to tell what they really mean because, well, you're a 5 star or you're not.

I'm not. How exactly does that make me a failure?

Reputation systems abound, and they're not going away. In fact they show every sign of becoming more prevalent. There's a lot to tease out here, and it takes a long time (so if anything this particular chapter will never be finished to perfection (sic))! In the interests of your sanity and mine, let's move to a couple of salient examples, one fictional and the other sadly not.

The fictional one is whuffie. In Cory Doctorow's 'Down and Out in the Magic Kingdom' whuffie is basically the reputation capital in the Bitchun society that everyone gets to see (on their Heads up Displays) at will. Do something good for someone, they'll give you some whuffie, want a nice room for the night? It'll cost you some whuffie. If your whuffie gets low enough, even elevators don't stop for you (ask Julius). whuffie is described as a measure of esteemed respect. What it **is** is reputation. The book (which you should read, it's even free online in lots of different versions, some of which are amazingly creative) describes how it is earned, lost, regained and so on throughout the existence of the characters (who never die anyway). As you might expect, it's dystopian but disguised as 'good'.

This brings us to dystopian disguised as 'good for society'. In China, you may have heard, there is a social credit system. What is this? Well, it's not (totally) Douglas' idea of social credit, that's for sure. Not sure what that means? Look at http://socialcredit.com.au.

The Social Credit System in China is still in development as of the time of writing, which means that there are many questions about how it works and how it might not. There are also different kinds of representation in the system itself (like numerical values for credit or black/whitelisting for credit). It basically works like this: do things that are socially acceptable or correct — like donate blood or volunteer — and you get credit. Do things that people don't like — like playing loud music, jaywalking, bribery and so on — and you lose credit . How is it enforced? By citizen participation (oh, those crowds again, we'll get back to crowds, don't worry), facial recognition systems (and we know how perfect those are, right?), a whole lot of AI. Things like that. There's also evidence that you can buy credit, but of course, that would be wrong, so it never happens (yes, that was sarcasm too).

And so we reach the point of all that: control.

The Social Credit System in China is a control mechanism. It's possible to see it as a form of reputation, and the behaviour is not far from whuffie: if you have bad credit you will be blacklisted, and you won't be allowed to travel (already happened), or stand for politics, or get your children into universities. To become un-blacklisted? Do good things for society.

You get the idea.

Doesn't it sound wonderful? Sure. Until you start asking questions like 'who gets to decide what is good and what isn't?' Is posting on a social network that isn't Chinese good or not? What about reading a certain kind of book? How about running for political office?

Like many things which seem interesting, promising, and plausible in first light, there are huge issues here.

What could possibly go wrong? Blackmail, coercion, corruption, mistaken identity. The list is quite long.

And just in case you think it couldn't happen here, wherever here is, consider: a good (financial) credit score gets you a long way. Moreover, see those reputation scores you're building on all those nice sites you use? Who controls them? Who decides what is 'good'?

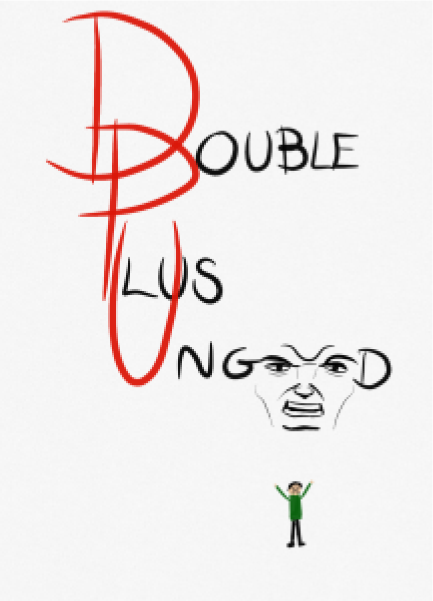


Figure 6.21: War is Peace. Freedom is Slavery. Ignorance is Strength. Social Credit is Truth.

In fact, the concept of social capital is closely linked to this. Social capital is basically the idea that positive connections with people around us mean that we are somehow happier, more trusting, less lonely and so on... Social capital, like reputation, can be used in situations where we are in trouble (ask for help) or need a little extra push (getting your child into that next best school) or a small recognition (like getting your coffee paid for by a co-worker every so often). You can gain social capital, and you can lose it. And if you lose it, then you don't get the good things. It isn't a specific number but the crowd you are part of calculates and implicitly shares it — by showing that you are accepted, by valuing your presence, things like that. It's about shared values and making sure that you share them in order to move groups, society, companies, people who look like you, forward.

Does that sound at all familiar?

Political capital is a similar thing, you could see it as an extension of social capital.

It's all reputation.

It has come to the attention of some thinkers (Like Rogers and Botsman, 2010) that all of this stuff hanging around is pretty useful. I mean, if you have a great reputation in one context, why is it that this isn't used in other contexts? This has led to the idea of "Reputation Banks" where you can manage reputation capital to be able to use it in different contexts. Good reputation capital means you get to choose your passengers as an Uber driver, or get nice seats at restaurants, and so on.

How familiar does that sound? By the way, I think it's an absolutely awful idea. So, why do I sound so down about all of this? Reputation systems, especially when pushed to the limits we see in China's Social Credit System or even the concept of social capital, are a means to control the behaviour of others. This is where the whole nudge theory stuff comes from. That's fine when we think of some of the behaviour that we don't like. I'm sure I don't need to tell you what that might be. And there's one of the problems, because your opinion and mine will almost certainly differ. I might happen to think that certain behaviours are okay — perhaps I have a more profound insight into why they happen than you do. Whereas you just see them as a nuisance. In the superb "This is Water" David Foster Wallace talks about putting yourself aside for just a moment and trying to figure out why things are happening, or why people are behaving the way they are. Like trust, this stuff is very personal and subjective. It's also information-driven in a way that we haven't figured out properly yet. If someone is exhibiting a certain kind of (for the sake of simplicity, let's call it 'anti-social') behaviour, why are they doing it? Do you know? I am sure I don't. But the crowd believes that it does (I told you we'd get back there).

What is the crowd? Well, it's usually not the people who are exhibiting behaviour that challenges it in some way (I'm sorry, that was a difficult sentence to parse). Let's imagine it's neurotypical, probably Caucasian, depending on where you are, almost certainly what you might call 'middle to upper class', possibly male-dominated. None of that has worked out particularly well for the planet so far, so why would we expect it to work out now in systems that expand its power exponentially?

It's also stupid. Crowds are not 'wise'. Their behaviour may be explainable by some statistical measures, but that doesn't mean that what the crowd thinks is good for everyone actually is.

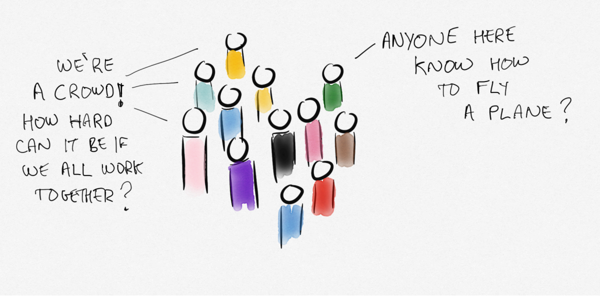


Figure 6.22: Sure, crowds are wise...

A crowd does not guarantee work done. It doesn't mean that what anyone might think is good for us actually is.

To argue the point a little, consider the flying of a plane. If you put enough people together in a crowd who don't know how to fly it, they're not going to get any better at it (thanks to Jeremy Pitt for this example). You need an expert. Some problems are expert problems. Some problems (and their solutions) are domain-specific. I would venture to suggest that figuring out who is more skilled than anyone else, or which is the correct switch to flick on an airplane controls, are certainly both expert and domain-specific problems.

What if the behaviour you see in someone — for example a woman shouting at her son — is the result of a personal tragedy that she is dealing with that leaves her emotionally and physically exhausted (I got this example from *This is Water*)? None of this information fits into a Social Credit System. Even if a credit agency is supposed to let you put a note someplace to explain a discrepancy, that doesn't change the score. If you have missed some payments because you had to pay for your child's medication, the score doesn't care. It's a score. It makes people money, and it helps people who have money decide how to treat those who may not.

If you use a reputation system to decide whether to buy something from someone, then use it as a tool to help, not a tool to tell. A tool to inform, not to dictate. Or, in the language we've used up to now, a tool to empower, not to enforce. Enforcement goes two ways — it enforces behaviour the crowd sees as correct, and it enforces the 'right' choice (the choice that the system wants you to make).

Reputation systems are fine. They give you information that can help you make decisions. Just don't use them to judge people or things. Or to believe one kind of thing over another. Or to choose friends or people to date. Use your head, that's what it's for.

As it happens, they are also open to quite a few different attacks, which is what we'll talk about in the next chapter because the attacks on reputation systems are also amongst the attacks that can be made on trust.

As I said earlier, this chapter will never be finished, so the next time you read it (if there is one) it will likely be different.

But you've reached the end of it here, at this time.

In order then that the social contract may not be an empty formula, it tacitly includes the undertaking, which alone can give force to the rest, that whoever refuses to obey the general will shall be compelled to do so by the whole body. This means nothing less than that he will be forced to be free...

(Rousseau 1762, p. 18)

ON COMPLEXITY

Introduction

In a previous chapter I suggested that there are problems associated with having models that are too complex for people to understand. Most importantly, I argued, if a model is too complex a human being would not actually wish use it to help them. There are some aspects of this argument that are perhaps a little problematic. Firstly, in many places in the book I have suggested that complexity actually necessitates trust: if something is too complex to understand or predict (and thus control) what else is there to do but think in terms of trust? I think this is a fair argument, and it's one that Cofta's work supports. So why wouldn't a complex trust model be worthwhile, since it puts us, ironically, into the 'need to trust' zone? It's a fair question but the irony is a problem. Trust is (if we follow Luhmann's arguments) a means to reduce complexity in the society we find ourselves. How then is it a good thing to make it more complex by giving us tools that are indecipherable?

A counter-argument is of course that the model need not be decipherable, it just needs to be followed — suggestions need to be seen as, as it were, commands. This is the very antithesis of trust empowerment (indeed, by definition it is **trust enforcement**). One of the things that we should be trying to do is to design systems, including those that help us with trust decisions, that are more understandable and by extension more empowering. It should go without saying that helping people understand better is a good thing.

Clearly, there are situations where the autonomous systems we create will use their own models of trust. In these cases, the opaqueness of the model itself — either because it is extremely complex or because it is obfuscated in some way — is of some use. At least as a means to enable more security and strength against trust attacks (which we will talk about later in the book). However, even in these circumstances there remains a need to explain.

Steve's First Law of Computing is that, sooner or later every computational system impacts humans[1]. This is important to remember because at some point a human being will either be affected or want to know what is happening, and often both.

It is at this point that the whole explainability bit becomes extremely important. There is a lot of literature around things like explainable AI, explainable systems and so forth (much of it is in the further reading chapter at the end of the book) but what it comes down to is this: the complex tools that we create for people to use

[1] Actually, if I was more honest, it is that "computing systems are for humans." Regardless of the place, time, space, purpose, how far removed from humans the computer is, computers do impact us, every day. To make a first law that demands that this impact be positive would seem to be a reasonable thing.

have to be able to explain why they have chosen, or recommended, or acted the way they have. They may need to do this after the fact (let's call those **justification systems**) or before (in which case **explainable systems** is probably more *apropos*). It's probably fair to say that we as humans don't always know why we did something, or why we trusted someone when we did, for example. It's a source of great amusement to me to sometimes ask, and a source of some consternation for someone who is asked, why things were done the way they were ("why *did* you trust Bob to deliver that letter?").

A long time ago[2] when expert systems were all the rage, explainability was recognized as important too. If an expert system came up with some diagnosis (there are lots of medical ones) or prediction it was possible to ask it why or more correctly how it came to the conclusions it did. Through a process of backtracking the system was able to show which rules were triggered by which data all the way back to the first evidence or question it got. It's pretty neat actually, and as a form of justification it is impeccable.

The thing is, complex systems like neural networks or genetic algorithms or black box artificial intelligences, or even complex mathematical trust models in an eCommerce setting, can't backtrack like that. They can however maybe *pretend* to, or perhaps lead the human through a path that makes sense to them. This may actually be enough — remember, humans are notoriously bad at explaining themselves too. Holding up other systems to a higher standard, especially in these uncertain circumstances, does rather seem a little demanding.

But this: The systems that we use have to be able to explain or justify in some reasonable way. The more complex, the more imperative this is.

Ten Commandments

This brings us to a paper I wrote with colleagues in 2012, which saw the potential pitfalls of complexity in trust models and tried to address them, or at least to make a start. It's called *Rendering unto Cæsar the Things That Are Cæsar's: Complex Trust Models and Human Understanding* which is a little bit of a pretentious title[3] but the contents are important. The paper was written about trust models, like the one I presented (my own) in a previous chapter. I could see at the time these models becoming increasingly complex and far too specialized to be reasonably called trust models at all. In the past ten or so years I have looked back at the paper and forward to what is happening in AI and human-centred systems and felt that the commandments are rather important there too.

The argument I have made at the start of this chapter was made in that paper and, in the almost ten years

[2] In computing terms, that could mean anything from 70 years to the last few minutes, but in this case it means the 70s, 80s and probably 90s, to be honest.

[3] I can say that because I co-wrote it! My long-suffering colleagues Natasha Dwyer and Anirban Basu probably rolled their eyes when I suggested the title...

since, not much has changed for the better. This is a *little bit* of a shame[4]. As we will see in the Trustworthy AI chapter in this book, there is a huge number of important questions that exist around the systems we are deploying in the world. Many of them are urgent, and not all of them are being addressed all that well. Barring the obvious racism, sexism and other bias in the tools we are creating, complexity and understanding are, to be honest, amongst the most urgent.

To put it another way, and bearing in mind that I am a computer scientist and not entirely a Luddite, AI and its cousins pose an existential threat to humanity. This is quite a claim to make. Indeed, a reviewer of a paper I co-wrote where we postulated that this was the case was instantly dismissive. But it is a needful claim. That they present an existential threat to *humans* is self-evident when one considers the antics of self-driving cars. However, more and more we are seeing the deployment of unmanned aerial vehicles (UAVs), drones capable of delivering death and mayhem thousands of miles from the humans that ostensibly control them (tragically), and even mobile agents to patrol contested borders. This is not a particularly healthy direction to be going, *especially* if we combine the problems of autonomous vehicles with the capabilities of deadly weapons systems. What is more, if the systems we are creating are all but indecipherable to even smart people, it is to our shame to not accept that there is a problem here. This elephant in the room is something I address in more detail in the Trustworthy AI chapter of the book.

Meanwhile, back to that ostentatiously titled paper. The paper presented ten commandments[5]. I repeat them here, along with the extra two commandments we published later in this paper. Where possible I explain some of the reasoning I had behind these particular commandments.

The first eight are from pages 197-198:

- The model is for people.
    - At this point, it's probably reasonable to quote Einstein: "Concern for man and his fate must always form the chief interest of all technical endeavors. Never forget this in the midst of your diagrams and equations." This probably says all that needs to be said but, just to belabour the point a little, we forget why we are doing this fascinating science and engineering at the peril of other human beings (at the very least).

[4] As ever, the master of understatement.

[5] actually, it presented eight, but in a later paper we introduced a couple more because one should always have ten commandments.

- The model should be understandable, not just by mathematics professors, but by the people who are expected to use and make decisions with or from it.
    - I have nothing against mathematics professors; indeed, some of them are very good friends. However, if we design complex models then they need to explain themselves. Consider: it makes no difference if a system *is* better than a human at driving, or decoding, or diagnosing a particular disease or condition if the person using it doesn't trust it. Indeed, it simply adds another layer of doubt and obfuscation to a system that includes people but is designed by scientists and engineers who are often little trusted in the first place. *Even if* the doctor assures the patient that the system has it right (after checking themselves *of course*, and if not why not?) there is little gained if the patient doesn't believe them.
- Allow for monitoring and intervention.
    - It is necessary to "understand that a human's conception of trust and risk is difficult to conceptualise. Many mathematical and economic models of trust assume (or hope for) a 'rational man' who makes judgments based on self-interest. However, in reality, humans weigh trust and risk in ways that cannot be fully predicted. A human needs to be able to make the judgment." (page 197).
    - This isn't actually something new. Kahneman and Tversky's Prospect Theory is a perfect example of how humans don't always make rational decisions. We (humans) are really *rubbish* at this stuff. But that in many ways is exactly the point. If a system making decisions *for* us makes them in ways we *wouldn't*, does this make it right or wrong *from our point of view*? And if it can't explain the decisions made in ways we can understand, what does this mean for how we might feel about it?
- The model should not fail silently[6], but should prompt for and expect input on 'failure' or uncertainty.
    - Again, humans aren't really very good at things like risk estimation, things like that.

[6] I did in an original draft put "fail silent" but was told by a reviewer to make it silent'ly'. I get why, but I also think 'fail silent' conveys more about what I was trying to think at the time.

But what human beings do well is be liminal. Human beings live the edges of things, and make decisions around those edges. To put it another way, it's exactly when there is not enough information to make a decision that humans shine. If this seems to be a problem (because we often make wrong decisions!) it is worth pointing out that in the absence of data the artificial system may well do no better and referring the reader to the first of the commandments; which is to say that at least the human's decision is one that they made for *themselves*.

- The model should allow for a deep level of configuration. Trust models should not assume what is 'best' for the user. Often design tends to guide users towards what the owner or developer of a site or application thinks that people should be doing. However, only the user can make that call in context.
    - Many of us 'know what's best' for the people for whom we design systems. This of course is dangerous. Why is this? It takes very little imagination to look at the ways in which 'persuasion' or 'nudging' or the often downright deceitful use of our understanding of how brains work and see them as problematic. If we stretch this kind of behaviour to allow complex systems that make decisions for us we do ourselves no favours (although we could be doing a large favour to some tech billionaires). In other words, the systems that we deploy should not only be able to explain themselves, they should be *tailorable* whenever possible to better match the way *we* as individuals see are (normatively) good behaviours.
    - What does this mean for a trust model? A look back at the chapter describing a simple model will show you that there are numerous variables that can change or be changed to better align the way in which autonomous trust-reasoning agents behave and reason with and about trust. For example, determination of risks, how utility might be calculated (and what it means to an individual) and from considerations if important to an individual all the way through to the different aspects of 'what it meant' and 'how it feels' in forgiveness calculations. As I noted in that chapter, a great deal of power in trust models lies in their heterogeneity. Much of this comes from a person's ability to personalize the models.
- The model should allow for querying: a user may want to know more about a system or a context. A trust interface working in the interest of the user should gather and present data the user regards as relevant. Some of the questions will be difficult for a system to predict and a developer to pre prepare, so a level of dynamic information exchange is

necessary.

  - We already know that trust is highly contextual. This means that one situation may have different results for trust decisions and calculations, even if much of what exists is similar to or the same as a situation that occurred before. At the very least, the experience (memory) of the truster has changed, but it may be that the location (place) is different, or that the people involved are different in some subtle ways. Regardless of the aspect that resulted in the different decision, and in any given situation, having an autonomous system explain its reasoning costs little (if it is designed properly) and grants much (in terms of confidence and ultimately then, trust – see for example Google's People and AI book).
  - Are explanations always a good idea? Well, yes, but not always in the moment. I leave you with a thought experiment: do you want your autonomous vehicle to tell you why it is about to put the brakes on to avoid the collision (and the reasoning process that got it there) or just to put the brakes on and maybe explain later? Let's see what the next point has to say.

- The model should cater for different time priorities. In some cases, a trust decision does need to be made quickly. But in other cases, a speedy response is not necessary, and it is possible to take advantage of new information as it comes to hand. A trust model working for humans needs to be able to respond to different timelines and not always seek a short-cut.

  - This is probably self-explanatory. There are times when speed is essential – after all, it would be good for an autonomous car to not hit a child who ran into the middle of the road, for instance – but there are also times when decisions should be made at more 'human' speeds, primarily so that the human can take part in them and understand them better. If this seems to be channeling Kahneman (again) it's not by accident.

- The model should allow for incompleteness. Many models aim to provide a definitive answer. Human life is rarely like that. A more appropriate approach is to keep the case open; allowing for new developments, users to change their minds, and for situations to be re-visited.

  - I've noted above that humans are liminal creatures. We also live in a constant 'now'

> which doesn't really ever become a 'then'. We are constantly able to learn from what has happened, to take what has been learned and apply it to different (or even the same) circumstances. Our trust models and the ways they consider and explain themselves should do no less.

And the extra two? They are from this paper, and go like this:

- Trust (and security) is an ongoing relationship that changes over time. Do not assume that the context in which the user is situated today will be identical tomorrow.
    - Already, above, we've talked about different but similar situations, the ability to learn and change behaviours accordingly, living in the 'now' and more. That the relationship between agents (including humans or humans and autonomous systems0 is ever-changing should be no surprise. Context is also every changing and so, when we combine these things, every new moment is a new context to be considered.
- It is important to acknowledge risk up front (which is what all trust, including foreground trust, does).
    - There's not much else to do here except perhaps allow you ask a question…

Foreground Trust

What is foreground trust, you ask? I'm glad you did. I'll present it much more deeply in the Research chapter, but it goes like this: give people the information they need (their own needs, which they can articulate) to make trusting decisions, and they will do so. In other words, don't tell them what is important, let them tell you.

So why was it called "Render unto Cæsar…"? Because there are times when the complex models serve important purposes — the world is a complex place after all. But there are also times when humans (Cæsar, if you will) need to be acknowledged. The commandments aim to satisfy that need. The most important thing to bear in mind: put people first (remember Steve's First Law!).

Let's put it all another way, one that has to do with system trust, mentions of which you'll find scattered around this book (see for instance here for a more full explanation). When we make things too complex, or at least so complex people don't get it, what we are doing is shifting the trust away from where its was (the other person, perhaps?) to something else – the model or the mathematics or whatever. At some point, I'd suggest, this ceases to be reasonable or even worthwhile.

In the next chapter I'll explore reputation and recommendation systems. These are obvious cases in point where complexity does not serve us well: they are specifically trying to help humans make decisions. But are we doing it right? Are we empowering or are we enforcing? And when we look at how they are deployed in things like social networks, are they helping or harming us? Let's take a look.

> Technology is nothing. What's important is that you have a faith in people, that they're basically good and smart, and if you give them tools, they'll do wonderful things with them.

Steve Jobs, Rolling Stone Interview, 1994.

ATTACKS AND DEFENCES

Or, There's Always a Handsome Prince...

Introduction

So there's this story. A man and his pregnant wife who lived next to a large, walled garden were looking for something that would sate the woman's cravings, which of course they found in the form of a leafy thing called Rapunzel[1] in the garden, which of course belonged to a not-too-nice witch. Naturally, the witch is a woman[2]. Anyway, the wife will eat nothing else and so one night the husband pops over the wall to nick[3] some. Of course his wife likes it a lot and wants more, so he goes to get some and this time is caught by the witch.

As you might expect, his behaviour doesn't go down too well with our witch[4]. She allows the man all the Rapunzel he wants, but insists that the child, when born, should be given to her, and the man agrees[5]. The child is born, the witch calls her Rapunzel, and when the girl grows old enough (with long golden hair to boot) our witch locks her in a tall tower with no door and only a window high up.

You've likely heard the rest ("Let down your hair," climb hair, visit Rapunzel) and even if you haven't you can almost certainly predict that a handsome prince pops along, hears Rapunzel singing, naturally falls in love and asks Rapunzel to let down her hair (which she does) which he climbs, they both fall in love, eventually he asks her to marry him. Now, it's almost certain she's pregnant herself by this time, but anyway, she forgetfully mentions the prince to the witch and the witch cuts off her hair in a rage. Prince arrives, climbs up, finds a witch, is thrown (or jumps off) and falls into thorny bushes which make him blind.

Sure, there's a happy ending. You can find it it for yourself. Why does this matter in a book about trust?

Because there's always a handsome prince. Read on, it'll become more clear.

It should come as no surprise that trust systems are subject to attack. After all, trust *itself* is subject to attack: how many times have people been fooled by con artists, or social engineers, to their detriment? Indeed, anything that has at its root an acceptance of risk by definition is subject to an attack, even if the attack is unintentional. I mean, that's the point, right?

Artificial trust systems like reputation and recommendation systems, blockchains (okay, trustless systems

[1] Bet you know the story now! Darn, the secret is out!

[2] It's a thing – traditionally it's a pretty powerful way to keep intelligent women in their place.

[3] Steal. Slang...

[4] Neither should it – after all, yon husband was stealing her stuff, but nobody thinks about that.

[5] There is so much wrong with this story, but it is what it is.

too), and autonomous systems have powerful capabilities. They also have cracks in their armour that allow nefarious others to take advantage. Sometimes this can be as simple as suggesting or displaying ads on a web page that are offensive or that lead to content that isn't what they say they are, but sometimes it can be quite serious.

One of the interesting things about trust, as we have learned, is that it allows things to work when there is no central control possible – decentralized systems, in other words. The use of trust allows elements (nodes) in such systems to reason about each other in order to better make decisions about which node to send information to, or which node to accept it from, which path to route sensitive data through, and so on. Other systems may monitor the environment and send information back to a central reasoning system that allows it to provide a situation report to scientists or, in many cases, military forces. If such information was compromised – for example by taking over one node and having it falsify data, or not route data to the control server, the results could be quite serious. Likewise, as infeasible as you might find it, if someone or something were to take over more than 50% of the miners in a blockchain, the "truth" can be managed, if not simply, then at least possibly[6].

Remember: there's always a handsome prince.

Attacks

In this chapter I'll have a look at some of the potential attacks that exist for trust systems, and I'll go over some of the defences that can be put in place to protect them. Like the chapter on reputation and recommendation, it's a living piece of text. There are always more potential attacks and, to be honest, there will always be people willing to use them, whilst other people try to defend against them. Nobody ever said that it was a good world filled with nice people, but it is the one we have. As it happens, trust is a pretty good tool most of the time to help us get through it. Anyway, we'll start with attacks that are pretty straightforward and go from there as time progresses. There is an excellent report from ENISA that looks at attacks on Reputation Systems that you might like[7].

Trust systems can be attacked in such a way that renders them much less useful to the people who want to use them. For instance, it's perfectly possible in online auction sites that rely on pseudonyms to enter as a new person with no penalty. "Of course," you might say, "because we want new people to join." However, this

[6] And as you already found out in the Trustless Systems chapter, the more time goes on, the more likely it is that a blockchain using Proof of Work can indeed be compromised in this way because mining becomes, well, not worth it.

[7] Yes it is from 2007, but the general attacks havent changed much in the sense that they still happen and we still haven't managed to make things muich better.

also means it's possible to build a (bad) reputation by reneging on promises, not delivering and so on and then leaving before rejoining as someone new.

Reputation models such as those used by eBay, Amazon and other online sites seek to remove some of the mystery about with whom you are dealing by taking a societal viewpoint. Reputation, ultimately, is a societal measure of the potential for trustworthiness of some person (or autonomous agent) in a very specific context.

To put that more succinctly, your reputation as a seller on eBay is a conglomeration of the ratings from everyone with whom you have dealt as a seller of things. If you have dealt with lots of people fairly and got good ratings, your reputation will be high. Do good things, people notice, you get to be known as a person who does good things. At this point it might help to revisit the trust models chapter to clear up what the difference between trustworthiness and reputation is.

The attacks on reputation systems, and indeed trust models that rely on such niceties, are aimed at subverting these scores in various ways.

In a whitewashing attack, attackers repair their reputation in some way after behaving badly to others in the system. How does this work? It's really a cost of having cheap pseudonyms (as Friedman and Resnick would have it).

In a previous chapter, I talked about how a cheap or free pseudonym-based system could allow that to happen. It's worth repeating here. The attacker can build a lovely reputation by lots of small transactions where they live up to (and exceed) expectations. Of course they're going to get a good reputation score… Then comes the trick: make one huge transaction, grab the money, as it were, and run (leave the system). Too bad for the sucker who you messed about with. Now, because the system you are using has a pseudonymous name system (you can call yourself "HappyOtter" or "AquaticLynx" or whatever) the next time you join you can be someone new, with no links to the previous person you were. And you can start all over again.

As the cheap pseudonyms theory tells us, whitewashing attacks work mostly in reputation systems where the cost of entering is very low. It helps to have a system where, when you enter, your reputation is effectively the same as, or close to, that of someone who has been in the system a long time and been good (especially, then, in systems that focus on negative feedback).

It's possible to defend against such attacks by removing these traits – for example by requiring some form of registration, a named user, emails or other semi-unique IDs attached to each user, and so on.

This is not the only way to enhance your reputation. In fact, it is possible to do this with another kind of attack (which has many many uses, sadly) to get what you want. It works like this: imagine that you are quite powerful, or a good coder, and have managed to secure for yourself a bunch of 'drones' that you can make behave any way you like. In particular, you can make them all give you a 5-star rating (remember, anything less than 5 is a fail). The result? You can pretty much do what you like because you're always going to have a preponderance of great ratings. This is a form of what is called ballot-stuffing.

Now, imagine if you took advantage of someone in the system. The person you make into a sucker might not take this lying down, and this could get messy. So what could you do? Well, you could get your drones to vote this person down (give them bad ratings, whatever) so that their word counts, basically, for nothing. You can use drones to just make someone look bad who you don't like, it's all possible. This form of attack is known as bad-mouthing.

Those drones I was talking about? They are a special kind of thing called a Sybil and the use of them is called a Sybil attack. The Sybil attack occurs when multiple actors under the control of a single actor flood the system in some way as to influence the reputation of the controller or someone else. In one setting, it is a form of self-promotion (ballot-stuffing). As we have seen, it can easily, however, also be used to decrease another person's reputation (where it is known as bad-mouthing).

This attack has potential in systems that are distributed, where there is little to no cost to join, and where interactions are not properly or adequately registered – after all, if nobody knows we didn't interact, then you could say anything and it would be believed by the system.

To defend against such an attack, the system could set up some kind of cost for creating new users. For instance, you could charge a small fee for each new user registered – although this might have an impact on getting anyone to register, of course. You might perhaps have some form of a time delay between creating a new user and being able to do anything. Or you might just give new users a zero reputation. A combination is of course possible – imagine for example charging a fee for reputation points that is held in escrow and returned when the user has shown that they behave, and if there is no fee paid starting at zero reputation. You might also want to put in place a way to ensure verifiability, where each transaction is verifiable to ensure that it definitely took place – with some form of certification, for instance.

As you might have spotted by now, there are problems with defending systems like this.

In the eCommerce world, a sort of Sybil attack happens when the manufacturer or marketer of some product pays a fee to have fake reviews posted for it. Amazon has suffered from this in the past (still does) but it isn't the only one. Pretty much any site that allows things like ratings and reviews has this problem.

It's a constant battle, and it's not getting any easier to win. The past year (when I write this) has driven a huge increase in online sales and thus, naturally, a similar increase in things like fake reviews. So can you still trust the reviews? It depends.

The on-off attack manipulates how trust systems remember in order to maintain the ability to benefit from being bad by alternately being good, thus never dropping below a certain trust score. How does this work? Simply by ensuring that, as an attacker, you never behave so badly that the system remembers. Cast your mind back if you will to our discussion of Robert Axelrod and the Prisoner's Dilemma. You will recall that tit-for-tat did quite well in Axelrod's experiments for various reasons (it was provokable, forgiving, and so on). That forgiving thing, which we've talked about too, is akin to forgetting if designed that way. This is to say that if you forget what happened to you, you are likely doomed to repeat the same thing happening – which is why it's sometimes good to forgive whilst not forgetting. Many trust systems have a forgetting factor built in. If an attacker can learn what this length of time, or mechanism, might be, they can behave badly for a time, then well, and then wait long enough for the system to forget before starting again.

As Sun et al. note, it's possible to mimic real life and the way society is seen to evaluate trust in a way to mitigate this attack. As we have learned, it is commonly believed that trust is fragile, that is, hard to build, easy to lose. Thus, it takes some time, and a lot of good behaviour, to be seen as trustworthy, but just a few instances of bad behaviour to become seen as untrustworthy.

Having a trust system behave in this way could certainly have the effect of mitigating an on-off attack, but it also has the effect of making newcomers, if they join with a low rating (since they are untrusted as yet) seem no different from those who have just been 'demoted'. That is, of course, unless you can count the number of interactions someone has had.

As with all defences, there are nuances here. The problem is figuring out what they might be and not allowing an attacker to do the same thing!

Another attack, which mostly affects systems that use a secondary indirect trust (recommendation trust), since it decreases the recommendation trust you have in the other party, is called the Conflicting Behaviour attack. In this attack, the attacker behaves differently to different entities, or groups of entities. Imagine how that might work: if I behave nicely to you, and badly to someone else, there will be conflicting information in the system. If you should go to those others looking for recommendations and they speak badly of me, it is quite possible that you will cease to believe what they have to say about anyone else, or at least discount it. Nefarious, eh?

This is a direct attack on the system itself. It doesn't necessarily benefit me (although it might, somehow – there's very likely a Machiavellian thing here somewhere) but it does mean that trust in the system is eroded, and this is a very bad thing – as we have already noted, sometimes system trust is the thing that we turn to in order to manage a difficult situation.

How do we deal with it? Well, one way is to not use recommendation trust at all, if you suspect that such an attack might be happening. As you might have just thought to yourself, this does rather negate the potential good of recommender systems, whilst also assuming we could spot such an attack in the first place. I cannot but agree.

People are, in the vernacular of the security professional, the weakest link in the security chain. Regardless of the truth or fairness of this belief, it is true that people, the human part of the trust system, are targets of attacks just as the technical parts are. Let's have a look at some of them.

Don't forget that darned handsome prince, by the way, we're getting to him.

We've all heard of phishing attacks, and spear-phishing as a means of precisely targeting them. Quite apart from being more or less sophisticated interface attacks, they, and their relations in social engineering, are also attacks based on trust: you click on links because you trust the systems that you are using to be telling you the right things. See, I told you that system trust was a bit of a challenge! I've already mentioned the problem with the way we design security systems in the Trustless Systems chapter so I won't harp on about it here, but we really ought to do better.

Anyway, the attacks work because either you are not paying attention or because they are extremely sophisticated (possibly both). This brings us to the observation that no-one can pay attention all the time. Moreover, since we are able to spot some attacks quite readily with security tools, the really difficult ones to spot are left to the fallible people to manage for themselves. Oh dear, I did harp on after all.

Defences

So, how can we defend against things like phishing attacks? It's difficult – again there's an arms race between the good and the bad folks, and when the good folks get better at spotting things automatically, the bad folks jump over the new walls that have been created. In other words, we probably won't get rid of them – there will always be something that slips through and isn't noticed because we are focused on our sick child, or the car crash we saw on the way to work, or whatever. People are people, to paraphrase Dickens: that they are what they are, do not blame them.

Whilst we're talking about people, let's look at the most difficult thing to fix, because people just don't like to be mean. Yes, we're talking about social engineering. Social engineering, a pithy term for lying, is a set of techniques used through millennia by con-artists to convince people of their authenticity, and often relies on the inability (or discomfort) of people to say 'no' in social situations.

As a tool to infiltrate physical locations (and hijack digital ones) it is usually extremely successful, and is often enabled by the over-sharing of its targets online. Examples abound of the successful use of the various techniques in the social engineering toolkit but, at the heart of its success is abuse of trust and social expectation.

How do you defend against things like this? Well, one way that organizations do it is by putting policies in place to ensure that people can say no. After all, if a policy exists that says you can't leave an unauthorized person alone in the building then it's (at least in theory) easier to say I'm sorry, I have to stay with you. Other policies that might help are to insist everyone wears an ID badge which is visible at all times and colour-coded, with certain colours indicating that the bearer has to be accompanied. Easy peasy.

The point? If we give people a good reason to say no, perhaps this makes it easier.

I've already noted in the chapter on reputation that we find ourselves in a situation where reputation is an important currency. Reputation can be used to better promote things, sell things, get things done online, and so on. It is also, therefore, a target. If someone's reputation can be hijacked, the results could be very problematic. It could be used to vouch for someone in order to insert them into a social network to be malicious. It could be used to promote malware, and so forth. It could, then, be "spent" on bad things.

Worse, it could be used to blackmail the person to whom it belongs, since it is such an important online tool in the 21st century, much as it was an important societal tool in previous centuries. Locking someone out of their reputation account, or social network account, could indeed be a highly lucrative business. How might this work? There are many ways. Consider for example stealing the account and asking for money to get it back. Or blackmailing the owner in order to get them to do something for you.

In case you didn't think it mattered too much there are companies who specialize in rebuilding the reputation of people who have done something that in retrospect seems a little foolish. The Internet is a terribly harsh place – so many people are quite happy to judge others by some measure which makes the judge look good and the judged look bad, and they are quite happy to be vocal about it, often with disastrous results for the judged whilst the judge feels righteous (I say feels – rarely are they ever such a thing). Consider the case of the person who took a prank picture of themselves at a cemetery. I'm not going to mention the name or the place, you can find it for yourself. The result of this person posting the photo was a Fire X Facebook page, death threats, rape threats, the works. And yes, they lost their job. Still today you can find the person's name and what happened along with the opprobrium attached to the action, almost ten years later.

Frankly, I don't really care much about whether what was done was right or wrong: the reaction was disproportionate, and quite probably based on self-righteous self-importance and hate of others who are not us regardless of what they might do. It's been said that this is possibly part of the 'culture war' that has been stoked by the likes of Facebook – sorry, Meta – and others. Regardless, there are plenty of these kinds of people around and sadly the Internet provides several venues for them to be hateful in. It undoubtedly makes them feel better about their own rather shallow lives.

Don't be those people.

On the other hand, don't be the person who has to get their picture out there right away either. A little forethought might be a good thing in our rather judgmental world. Remember: there is a problem in using power tools when you are inattentive.

It's said the internet doesn't forget (regardless of right to be forgotten laws) but there are reputation management companies such as ReputationDefender which try to work the algorithms used by search engines to make the 'bad' pages appear lower in the results list when a person's name is searched for. It's a good thing such companies exist: they are the handsome princes that we are lucky to have. If you're interested, you can find a good article in the Guardian newspaper about the whole process here.

The consequences of lies and anger on sites which encourage 'friending' and instant reaction have become ever more sorrowful, for instance in the killing of Samuel Paty). And before you think this is not about trust, it most certainly is, you just have to look at it properly. The father of the teen who lied and began this sorry chain of events reportedly said, "It's hard to imagine how we got here."

It really isn't. That's the problem.

To be clear, things like attacking and defending reputation are sometimes deliberate and sometimes accidents. The way in which the system itself reacts is disproportionate (usually negatively) because of what it is: a collection of anonymous fools who should know better. The problem of course is that much of what I just wrote is normative. Your version of the truth or what matters will always be different from mine or anyone else's. The trick with systems like this is to be your own person and not follow the crowd, even if it is noisy. I've talked about crowds before in this book but it is good to remind you: the crowd is pretty much as smart as the stupidest person in it.

I've said before in various settings that one of the things that makes human beings so very easy to con is that they get embarrassed rather easily. It is embarrassing to be taken for a sucker.

It's also something you don't forget easily, I rather like Rhys Rhysson's little line in Terry Pratchett's *Raising Steam*: "I see embarrassment among all of you. That's good. The thing about being embarrassed is that sooner or later you aren't, but you remember that you were."

The other thing about being embarrassed is that it is akin to being ashamed: you really don't want anyone else to know how it happened. So, human beings, well, we hide our shame and our embarrassment and let the con-artists and blackmailers carry on with what they are doing, not letting other people know, even to warn them, because to do so would reveal we are suckers.

One of the best things about artificial systems is that they can compartmentalize what they should be feeling and turn it into action. Instead of being ashamed or embarrassed about being taken for a sucker, an artificial entity can broadcast it far and wide. The end result? The con-artist or blackmailer is less likely to succeed with other such systems (or even people). This is, in fact, partly why things like the Cyber Information Sharing and Collaboration Program could work.

Whilst I'm talking about this kind of thing, I would be remiss if I didn't talk a little more about how people are problematic just because they are people. And people live in a society which can be harsh. Society can also attack the people who are in it, and trust systems like reputation or recommendation systems can amplify this. One way this might happen is by stamping down on differences, or on new ways of doing things – someone proposing such new things might suffer from a particularly bad reputation if those with power and a vested interest in the status quo are threatened. Likewise, there is always the problem of a vocal minority: that small subset of a population that actually does leave reviews and ratings might skew the findings of a person looking for accurate information.

There's no real end to these things.

I promised that we'd return to our handsome prince. As you may have spotted, the prince was able to get into Rapunzel's tower and seduce her regardless of the fact that there was no door and a window many many feet up in the air. To put it simply, there is never a totally secure system. There is never a system that can't be successfully attacked. If there was such a thing, there wouldn't be so many security consultants. Since we're paraphrasing books and plays, let's do one more – as William Goldman's Dread Pirate Roberts says, "Life is pain, Highness. Anyone who says differently is selling something."

There's always a handsome prince, and they can always get where you don't want them to and do what you don't want them to.

A final word or two for this chapter is in order. Trust systems can be attacked. There are also some defences. Unfortunately the defences, like most security procedures, somewhat get in the way of the very tasks you might want to accomplish with the system. Let's go back to our example at the start of the chapter – entering the system as a new person to whitewash your reputation. To defend against this, as we've learned, you can require references from friends before being allowed to enter, or deposits, or any number of other things that affect the cost of entering the system. All of these are valid, and every one of them has a side-effect: to make the system less accessible to the bona fide new members.

We can't get around this: defending systems makes them more challenging to use for the people who want to use them legitimately. Our task is to determine the best defence in any circumstance that has the least impact on the people or computational systems that want to use them legitimately.

WEBS OF TRUST AND PUBLIC KEYS. OH MY!

Introduction

Unless you haven't been paying attention, you might have heard of things like https or SSL, and if you use a web browser at all will undoubtedly have seen https at the front of pretty much all web addresses these days. It wasn't always the case. We have the Electronic Frontier Foundation to thank for the fact that it is so widespread (take a look at that link. It has an https in front of it!) Let's explore this, and while we're at it, some attacks and potential defences.

Let's start with what http stands for: basically it's the hypertext transfer protocol, which thanks to Sir Tim Berners-Lee became the way in which the World Wide Web shared information from servers to our nice little web browsers on the many devices that we use today. The various different ways in which it has changed over the years since it first appeared in 1989 and made the Internet the wonderful place it now is are not important to this book. Just think of it as a way to share information on the web. Now, you will have hopefully noticed that I mentioned http, not https.

Smashing.

The "s" is important – it basically stands for "secure" and it's important. What it does is enable secure communication between those pesky web sites and our beautiful browsers. The things sent using https are encrypted between the site and browser which means that it is hard (I won't say impossible) to eavesdrop on what is being shared, or indeed to change it. We'll get back to the possible attacks on it in a moment, but before we do, consider that it enables secure communication so that things like your Google Mail, or DuckDuckGo searches, or online banking and so on to actually happen.

Encryption

It's a big deal.

How does it work? Well, that SSL I mentioned just now is basically it (although these days it's called TLS). SSL stands for secure socket layer (and TLS stands for transport layer security – basically, meet the new boss, same as the old boss). SSL/TLS actually enable a bunch more than just secure http – for example secure oice over IP telephony – but what is important for us here is basically how the whole thing works.

It's all to do with cryptography, of course. I talked a bit about cryptography in the Trustless Systems chapter, so I won't go too much into it here, but there is one thing I didn't mention, and that was using keys and certificates. Okay, two things (cue Monty Python sketch).

So, on the assumption you know how hashes work, and if you don't, head to the Trustless Systems chapter to find out, let's dive into **public** and **private keys** and such first. Technically, you'd call the field **public key cryptography** and it's actually a pretty neat thing because it allows even small people like us to get some privacy.

There's a couple of ways of doing this, so let's jump into the first, old way, and you will be able to see the issues. It goes like this. Alice wants to send Bob a message. She uses a shared key to encrypt the message. Did I hear you say, "Wait, stop there! What is this magic of which you speak?"

Okay, so **encryption** in four paragraphs or less. Actually this is nothing new, people have been using encryption for many many years, probably as long as there have been people. You might remember your own childhood when you used a bit of trickery to, for example, shift characters up the alphabet so that E meant A, F meant B and so on, and then used this little cypher (okay, in North America you'd probably say cipher, but whatever) to encrypt your message. Let's see now. If Alice and Bob were using this, their shared key would be the number of letter shifts (in this case, five, because the fifth letter is the first), Alice would use this little shift algorithm to encrypt her message to E HKRA UKQ, send it to Bob, who shifted the letters back to decrypt it. I'll leave you to figure out what Alice's message to Bob was.

As it happens, this kind of thing was used by the Ancient Greeks with what are called Scytales. It's pretty neat. Easily broken? Sure. But that's basically what shared secret (**symmetric**) cryptography looks like. Oh, there are lots of fancy bells and whistles that involve rather complicated mathematics and proofs, but they aren't really too important for the purposes of this chapter (or this book), which is about trust, not security, and anyway I'm not cryptographer so I'd probably get it wrong.

So there are lots of examples of this symmetric key cryptography around, from those Scytales to the Enigma machine the Germans used in World War Two and beyond. It has the advantage of being quite quick but the disadvantage of needing to keep the keys secret. There are some answers to that shared secret thing that involve their own bit of cryptography though (look up Diffie-Hellman key exchange or RSA).

Now, **aysymmetric encryption** is the basis of things like PGP (Pretty Good Privacy), digital signatures, and things like that. It's also the backbone of that https I was talking about at the start of this little chapter. It's also called **public key cryptography**. That word "public" is important, as you will see. Bear with me, it's actually not too complicated, but it did take me ages to be able to figure it out when I first saw it – I am clearly quite slow. Actually, symmetry and all of its associated words are one of my kryptonite spelling words – I always get them wrong. Asymmetric encryption basically relies on two different keys which are cryptographically generated (which basically means "clever mathematics").

Let's imagine Alice and Bob again. Alice uses an encryption algorithm. There are plenty around – RSA is one of them, and you've heard of that. ECC (elliptic curve cryptography) is another, and it's often used in cellular communication. Alice gives the algorithm a random number (the bigger the number, the better) and the algorithm gives her two keys. The first is a **private** key, which Alice has to keep secret. The second is a **public** key which she can share. Meanwhile, Bob does the same thing.

In order to send a message to Bob that is private, Alice will need his public key. Skipping over the details of that for a moment, let's imagine for instance that Bob includes it in his email signature (it's just a jumbled mess of text, so that's possible). Alice grabs Bob's public key and uses her encryption tool to encrypt the message she wants to send with Bob's public key. The result will be a big midge-modge of text because it's been encrypted. She can now send this to Bob.

Bob receives a message from Alice which is, well, a midge-modge of characters and thinks to himself "*Ahh, Alice sent me an encrypted message! Cool! I wonder what it says!*". He heads to his encryption tool[1] and pastes the text into it. He also[2] puts in his private key and if he is sensible the password he has used to secure it[3]. He clicks on the decrypt button and gets the message Alice sent.

Okay, so that's pretty cool, because now people like you and me can do really secure messaging.

Curious about doing this for yourself? Well there are various tools around that allow you to, for instance, encrypt your email or other messages. Some are online (like https://smartninja-pgp.appspot.com/#) and some are tools to download and use on your computer[4].

Right, back to the asymmetric bit. It's called asymmetric because there are two keys for each person – a public one that is shared and allows people to encrypt messages for you, and a private one that you keep secret and use to decrypt messages to you. You can also use it for things like signatures, so that you know the message came from the person who says sent it. In this case, let's pick on Alice and Bob again. Bob needs to be sure that the message he gets was actually sent by Alice – after all, it could have some pretty personal stuff in there and he doesn't want to get all mixed up. He asks Alice to sign her messages from now on. Alice is a little confused so looks up how it all works.

Here goes!

First, Alice has to create the message she wants to send. "*That's not so hard*," she thinks. The next step is something you've seen before, it's a hash! So, Alice has to put her message through a hash function (like the SHA-256 one we saw in the Trustless Systems chapter). It's a one-way hash, which basically means, as you already know, that the hash can't be used to get back to the data that was used. "But that means that Bob won't be able to see the message!" says Alice. This is indeed true. We'll get there.

Anyway, the hashed message, plus information about the algorithm used to hash it (like SHA-256 for

[1] W, in this instance it's for decrypting!

[2] because he might have more than one

[3] Ut never hurts to have a password – if the key is lost then at least you have a little more insurance that even if someone finds it or steals it or whatever the password will mean they can't use it to decrypt stuff

[4] . A quick search for PGP tools for your favourite computer will bring you some, but be careful to make sure you are getting things from a reputable provider.

instance), is encrypted using Alice's private key. This bit is important. The only person who should have a copy of Alice's private key is Alice, right? Right. So if it is signed (encrypted) with this private key, Bob will know that Alice must have sent it. What Alice just created is the digital signature of the message. At this stage, if Alice sent the message as regular text along with this signature, what could Bob do?

Right—he can use Alice's public key to decrypt the signature. How this works isn't too important, but as long as Bob has Alice's public key, it'll work. This of course means two things: first, everyone with Alice's public key could do this and second, so can Bob.

When Bob decrypts the signature, what does he get? He gets the hash of the message plus details of the hashing algorithm used. It's a one-way hash! What good is that?! Well, at this stage, what Bob could do is to run the message (which Alice sent as plain text) through the same hashing algorithm. He would get a hash, right? The thing is, if the hash didn't match the one Alice sent, he would know that the message has been altered. Cool eh? On the other hand, if the hashes do match, he would know that Alice's message is the one she sent.

But there is of course the problem that the message Alice sent was sent in plain text. This kind of destroys the purpose of having privacy in the first place! It does give us **non-repudiation** though. If you get a message and the digital signature of the message, it doesn't necessarily matter if the message is in plain text. If you can decrypt the signature with the sender's public key and run the plain text message through there same hash function, you will know that the message you got was the message that was sent, and you can prove it. This itself is quite valuable. But it doesn't make Alice's message private, which is what she wants to do!

What should Alice do?

We already know the answer to this. She can use the regular encryption stuff she already knows! We can skip one of the steps she just did actually because she doesn't really need it.

Here we go. *Again.*

She takes her message and puts it through the hash function. Then, she takes the hash, the message she wants to send to Bob, and the hashing algorithm used, and then encrypts the lot with her own private key. She then encrypts that with Bob's public key. She can now send this all to Bob. *Phew.*

One thing: did you see the step that was missed out? The regular digital signature step was to hash the message and then encrypt this hash (and what the algorithm used was) using Alice's private key. In the steps take above, she doesn't create a digital signature of the message like that. Instead, she takes the message, the hash of the message *and* the algorithm used and hashes all that together with her private key. The end result is pretty much the same: Bob knows (and can prove) the message came from Alice and also that it wasn't changed en route.

How? When Bob gets the message, he can decrypt using his private key. This will get him the message digitally signed by Alice (basically, more useless text). He can take this and decrypt it using Alice's public key. If this works he knows that Alice sent the message. Bob then has the original of the message, the hashed version of the message, and the algorithm Alice used. Smashing – he can run the original of the message through the same algorithm and get himself a hash of it. If the two hashes match, he knows that Alice not only sent the message but also that it wasn't changed. Bob and Alice now go and take a headache pill because this is all quite confusing.

If Bob really wanted to, he could get Alice to authenticate herself like this: Create a random message, encrypt using Alice's public key and send to Alice. Alice is the only one who can decrypt this because she has her private key. She does this and sends the random message back to Bob who, if it matches, knows that it was decrypted using Alice's key and so, well, this must be Alice!

Back to https and TLS (SSL). Oh yes, you'd forgotten we were there hadn't you! It kind of works the same way as what I just talked about, but there are a couple of other steps to think about. It's all to do with certificates! In practice, the public key of a person is a bit unwieldy and anyway, whilst Bob might trust Alice to be who she says she is, that digital signature likely won't hold up in court. We need a way for Alice to prove beyond a doubt that the public key she is passing out actually does belong to her. If we had this, whenever she signed a document everyone would know that she had signed it even if they didn't know her (and indeed even if they didn't really trust her).

This is managed through the use of **certificates**. A certificate is basically a a credential that is given to you by what is known as a **certificate authority**, or CA. Certificates follow a standard format called X.509. Basically they contain that the public key of the CA plus proof you are the owner of that key. Basically this is in the form of the public key of the person who is being authenticated, hashed and encrypted by the CA's private key, which is pretty much the digital signature of the CA who issued the certificate. There are actually lots of CAs around that can issue certificates. How does that work? Each CA is part of a hierarchy of CAs all the way back to what is known as a root. Each step through this hierarchy is basically a trust step. That's why the CA is known as a **trusted third party** (TTP).

The root CA issues certificates to CAs underneath it who it *trusts*, they issue CAs below them with credentials, and so on, until you get to the one that you can get a certificate from if you can prove you are who you say you are to the CA in question. In other words, the CA basically says, "Yep, this is Steve" and you can be sure that the public key in that certificate belongs to me! Did you spot that *trust* word?

Lots of root certificates are stored in browsers (or for browsers) on every machine around the world, so that when the browser goes to a website it should be able to be sure that the website is who it says it is, because if you remember, the signature on the certificate is made because the private key of the CA was used to encrypt the hash of the public key of the website. So, if you have the public key of the CA[5] you can decrypt the signature in the certificate, which gives you the hash, and then take the public key the server actually sent you and hash this. If the hashes are the same, you basically know that the CA signed the public key the server just sent you, right? And you know nobody could have copied it because all you got was the hash of the public key from the certificate authority, **not** the public key itself (so the server couldn't know what the key was, unless it owned it).

[5] You do, it's already on your machine.

I think that kind of answers the question about proving things, but if you really think about it, it's potentially open to all kinds of problems. Most importantly, the CA signs the certificate and you trust the CA. Can you see where this is going? If you haven't read the bit about system trust in the Trustless Systems chapter it might be a good time to do that now. We'll come back to this too.

Now that we know how the whole symmetric and asymmetric thing works, and we have a decent enough grasp of certificates, we can jump right into https[6]. It works, as I have already said, by allowing websites to prove that they are who they say they are, as well as allowing your browser and the website to be able to encrypt the information that is sent between them.

Why is this useful? Imagine online banking without it. Or writing a Google email that was sensitive for whatever reason[7]. It's a big deal. It works in a couple of stages, and it's all to do with that TLS, or SSL protocol.

Here goes! There are actually a few ways this is done, but let's pick one, it's the RSA key exchange algorithm and it's the most popular one. Also, bear in mind this is quite simplified. This isn't a security book, it's about trust. If you want security classes, there are many around.

First, you ask your browser to go to someplace[8]. You could click on a link, or a bookmark, or even type in the address, it's all the same to me (and the browser). Let's call the browser a client from now on. It'll make more sense (to me anyway).

Second, the client basically says, "Hello!" to the server. It's polite to say hello, but in this case the hello contains information like, "This is the version of TLS I support," "These are the cipher suites I support"[9], "Here is a random number." The random number is important, as you will see. We'll call it the **client random number**. The server gets all this stuff and so the third step is for it to reply (it would seem churlish not to). The reply contains a, "Hello back, small computer!" (Well, not really but close) as well as, "Here is my certificate," the certificate that was given to it by a CA, "This is the cipher suite we will use," and another random number, which we will call the **server random number**.

With this information, the client is now able to authenticate the server. It does this by checking the certificate with the certificate authority. In practice it has all those root certificates stored on your computer someplace. The certificate the server sent the client is signed with the CA's private key. To check it, the client grabs the public key of the CA from its store of root certificates and checks the signature against the one that the server sent in the certificate.

How?

By decrypting the signature in the certificate sent by the server using the CA's public key, and checking the

[6] Finally, Steve!

[7] Perhaps because you are a dissident in an authoritarian country, or a journalist protecting a source.

[8] If the address starts with https it's a clue that what follows will happen.

[9] Like, you know, which encryption algorithms I can use.

hash of the server's public key against the hash of the public key that is stored in the digital signature that was signed by the CA. All being well, the hashes will agree.

What does this give us? It gives us the fact that the public key the server sent us is the same as the public key that the CA signed. Nothing else. The server is still not authenticated at this point. Whoa, now there's a trusted third party.

Right, now the client can create a random number and encrypt it using the public key of the server (which the client knows the server has because it sent it to us, and anyway its hash is in the digital signature of the certificate). The client sends this encrypted final random number to the server. The server, if it knows the private key, can decrypt the random number. If it was trying to pretend it was something it wasn't, it wouldn't have the private key that was associated with the public key it just sent out (which the client knows belongs to a specific server because, well, the CA signed it and it all matched). If this is the case, the client would know that not only is the server the owner of a certificate signed by the CA, it is also the owner of the private key associated with the public key that the CA signed in that certificate.

Phew.

The final random number is not sent back to the client at this time. What would be the point? In fact, the final random number that was decrypted, the client random number the client sent in the first step, and the server random number the server sent back in the second, are used together to create a **session key**. Because everyone has the same numbers, the session keys should be the same on both client and server. If they're not, we'll know someone is trying to listen in (a *Man in the Middle*).

The session keys are then used to encrypt a, "Finished!" message from the client and one from the server. Since they are both the same key, we've got **symmetric encryption** happening now (which is faster) and the session can continue with all traffic between the clients and the server encrypted by the session key. Ta-da!

It's a pretty interesting thing, but it is not without its problems. One of these has to do with that trusted third party idea. All it takes for a fake CA to spring up and give fake, yet valid, certificates, is to grab the private key of one of the real CAs. This is in fact exactly what happened to DigiNotar several years ago, with the result that fake Gmail certificates were created and used to allow spying (basically) on the emails being sent using Gmail in Iran (for some users). That bit about being in an authoritarian country and so on that I mentioned earlier? There you go.

Sure, there is a way to revoke certificates but in this particular case the fragility of the system was exposed, because it takes time (and money) to revoke all the certificates in all the browsers and operating systems all over the world (to be safe). Quite frankly, it's a nightmare. Not to mention, there are various other vulnerabilities that can (and indeed have) spring up. We needn't go into them here.

The thing is, https gives a sense of trust in the websites that you are visiting. At the very least it is supposed to give you the assurance that the website you are talking to is the one it says it is, and that nobody can listen in the middle, amongst a few other things. It is clear (I hope) that it does a fair job at all of this. After all, it's so much better than nothing at all. But it relies on a hierarchy of trusted third parties. And where we talk about trust in this setting, what we really mean is that you have no choice.

Somewhere along the line someone (the people who make browsers, for instance) has decided which certificates are good and which won't be used. Plus, you're assuming that the certificates you have are still good. Certificates have a shelf life. If a certificate expires your web browser should tell you when you go to visit a site that uses it. Actually it does, but then it gives you the option of carrying on regardless, sometimes easily and for other browsers with quite a lot of difficulty. Fair enough.

Anyway, there's one thing that rears its head that should be remembered, and that is that this hierarchy is pretty fragile. In response to this, and the rather more distributed nature of the Internet and associated networks that exist, something called the **web of trust** (WoT) has grown. It's like this: anyone can issue a certificate to anyone else that they know to say, "Sure, this is Steve".

"What madness is this?" you might be asking. After all, if Charlie issues me a certificate, what on earth does that tell you? Not much. But what if you knew Charlie and trusted him? And what if not just Charlie, but Dave and Ernest and Samantha all vouched for me? The more people who vouch for me, the better, right? The thing about WoT certificates is that they can be vouched for (verified) by a number of people. Each one can basically sign the certificate to say, "Yep, this is Steve." These signatures can come from people you know and trust, which makes it better, and from people who are known and trusted across the planet (and trusted by other strongly trusted people, and so on). Sharing the trust can often come from things like key signing parties, where people get together physically (so they know they exist, right?) to share codes that will allow them to verify the signatures[10].

The decision to trust a certificate (and thus the person giving it to you) is your own, and could be based on your own requirements. For example, you may say that you won't trust a certificate unless it is at least partially trusted by more than 6 others, or at least fully trusted by one fully trusted other (or more).

Actually, if you bothered to use that PGP stuff I put up earlier in this chapter, you already used WoT stuff. PGP and the web of trust are very closely linked. In fact, they were both created by Phil Zimmerman, a computer scientist. In his words (from the PGP 2.0 manual):

> "As time goes on, you will accumulate keys from other people that you may want to designate as trusted introducers. Everyone else will each choose their own trusted introducers. And everyone will gradually accumulate and distribute with their key a collection of certifying signatures from other people, with the expectation that anyone receiving it will trust at least one or two of the signatures. This will cause the emergence of a decentralized fault-tolerant web of confidence for all public keys."
> –Phil Zimmerman, PGP 2.0 Manual

[10] If you are wondering how that stood up in COVID lockdowns, so am I.

Okay, so this all sounds very nice and utopian (Thomas More would indeed be proud) but it does indeed have its advantages: you get to choose who you will trust, for one thing, and how much, so it's not a million miles away from how trust really works, right? Sure, it has its problems and some of them are actually social, which is pretty neat. After that, it basically allows certain things to happen, according to how much, or if, you trust the other person (like sharing encrypted information knowing that Alice is Alice and Bob is Bob).

Just don't lose your keychain, or your own private key. Why? Because if you lost your private key and someone else got it, they could basically sign certificates as if they were you (which brings us back to that DigiNotar example). Whilst this might not be quite as horribly dangerous as the PKI (centralized) one when keys are lost, it can still be very problematic. More, if someone sends you a message that is encrypted using your public key and you don't have the private one, well, you can't decrypt it. But someone who is not you could if they stole it, and they could pretend to be you too.

Could you use it in real life to help you trust someone? Well, if by trust you mean ensure that this person is who they say they are and make sure the messages I'm sending and receiving are private and authenticated then sure. If you mean other stuff like lend them money or take investment advice or whatever, perhaps you need to look a little further? And if you are using it for that, bear in mind that this is not what it was intended for.

Security and trust are difficult bedfellows. In the Trustless Systems chapter I talk a bit about zero trust, which in fact is really just trust as it should be. Most of the time when security talks about things like trustworthy or trusted what it really means is you don't get a choice. Use it. Or, to be more fair, this is secure and you can use it without worrying. This is in fact the opposite of what trust really means. It's also incorrect because, as we have already learned, there is always a handsome prince.

THE ELEPHANT IN THE ROOM

Or, What Does "Trustworthy AI" Mean Anyway? Or, Can We Trust (Autonomous) Technology? And, Can Technology Trust Us?

Admittedly, one can have too many titles to a chapter, but in this instance there are many reasons. Let's unpack some of them together.

This is not a technical chapter. There are no formulae, there is no cryptography, there are no recommendation or reputation systems. There are no answers, really.

There are plenty of places where trustworthy AI is discussed — what it means, how it might work, why it is important. There's a bunch about things like the Trolley Problem, moral choices, transparency and so on. We will probably get to some of that in this chapter. But we won't cover it all — the further reading chapter at the back of the book (there are always answers in the back of the book!) will provide pointers to some of these other places. But recently it has become clear to me (and a few other people I know) that we may not actually be asking the right questions. So this chapter is about that. The thoughts here have been developed in collaboration with and many thoughtful and urgent conversations with my colleagues and friends Peter Lewis and Jeremy Pitt. Indeed, Peter and I have written a paper that seeks to tease out much of what you will read here. It's all about trusting rocks (seriously). Naturally I highly recommend that you read it. But in the interim, I hope you read this chapter too.

Let's begin with once upon a time.

Once upon a time, there was no AI[1]. Not now. Now we have AI around every street corner — watching us on street corners in fact. Sure, it may be a marketing gimmick — "We have AI! Buy our stuff!" — after all, the history of Artificial Intelligence is sadly polluted by snake oil. But let's imagine we are now in the era where the computers we are using are actually capable of supporting an AI. It's not exactly hard to believe: I wear on my wrist more computing power than got Armstrong, Aldrin and Collins to the Moon. Let's imagine for a while that AI has truly arrived, whatever that might mean, and that we live in an age where we have spawned another intelligence. Just as an aside, if this is true (I leave it to you to decide), this is truly something special and we should as a species be humbly proud of what we have done[2]. The consequences are rather massive, though. So, let's work on that.

Let's continue, then, with a question: what exactly is AI? Margaret Boden, who probably knows better than anyone else alive, describes it as computers doing the things that minds can do. Imagine, for a second or two

[1] I'm well aware this is a specious observation. It really doesn't matter.

[2] Even if it wasn't us specifically, personally, that did it.

(or the remainder of this chapter) that minds and humans are pretty much connected (except where we are talking about AI). That sounds about right. It doesn't say that AI is better than humans, or that there are things humans can do that AI cannot. Indeed, it doesn't actually say that there are things an AI can do that a human cannot.

There is something important here. But there's much, much more. Whether or not an AI can think better than us, or drive a vehicle better than us, or whatever, it is something that resides on a machine (for now) that can be turned off (for now). This sounds profound until you realize that the same can be said for any of us. The only thing that stands in the way of other humans doing it is that it's just, well, wrong to turn off humans. But I digress, although I would like to point out that the usual but you can turn it off argument about AI being lesser than humans isn't actually that valid. We generally get to pick the hill we want to die on.

So why is this in a book about trust? There are two things here. The first is the traditional, "Can we trust (an) AI?" (which I call the Frankenstein question). The second is, "Can (an) AI trust us?" (which I think of as the Prometheus question, for want of a better word). Both of these are huge questions. Let's start with the first.

Can We Trust (an) Artificial Intelligence?

For starters, if you've been reading to now and haven't just jumped here, that's something of a silly question. If you didn't think, "What for?" when you read that, may I humbly suggest you head back to the Pioneers chapter and look for Onora O'Neil? To put it slightly more succinctly: asking the question, "Can you trust X?" is pointless. It doesn't mean anything.

But I digress. The question, however silly, has been asked. Let's see what we might think of can we trust (an) AI to do something?

Have you noticed those (an) things hanging around? They might get a bit annoying, so let's do this: An AI is an agent, some instance of a thing that is artificially intelligent in some way. AI, without the (an), is Artificial Intelligence in general. I sometimes slip between the toward I think, right now, that's ok. It's a bit like, "Can I trust a human?" versus "Can I trust humans?" You could take it further and do the, "Can I trust Steve?" in which case the corresponding question for AI would be, "Can I trust this specific instantiation of a particular form of AI?". I may do away with the brackets now that you catch the drift.

Anyway, trusting AI, or even an AI, to do something, It's really quite a contentious topic, and it has been for as long as I have been thinking about trust (around 30 years now!) and before: is it sensible to actually think about technology in terms of such a human notion as trust? Unpacking that sentence is challenging. The first problem is that of human exceptionalism, which is probably not something you would expect to find in a book about technology. Then again, perhaps it would be sensible to have discussions around human exceptionalism in technology books.

There's often an implicit and sometimes explicit assumption that trust is a human trait. A careful observation of our non-human friends reveals the inaccuracy of this assumption. Whilst trust is a human trait, it's not exclusively human. Which is to say, animals can and do trust us too. And it probably goes without saying that we often consider our animals in terms of trust. The story of Gelert the Faithful Hound, and various similar derivative or coincidental stories from around the world, serve to remind us that when we place trust in a 'lesser' being we need to accept that the trust has been placed for a reason. As you have likely realized if you have come this far in the book, trust is an acceptance of the potential for bad things to happen in any given situation. This isn't just an academic statement and it bears a deeper exploration.

When we place trust we accept the fact that there are things we cannot control in the situation. We also accept that the outcome of the situation may well be in the hands of another. That is, the other (the trusted) may well have the power or capability (or willingness, desire, intention, etc.) to do something that we don't want to happen. The thing about this is that that trusted other need not even know that they are trusted. Castelfranchi notes that it is possible to use the fact that you trust someone as a form of power over them — because moral creatures don't want to let others down. This in no way compels us to tell the other that they are trusted. In fact, it doesn't even say that the other knows we exist.

We often talk about trusting government or trusting some company or other to do something. We often talk about trusting animals to do (or in many case not to do) something. That government may know in principle that we exist, but not usually as individuals. The same goes for the company (absent surveillance capitalism) or the animal. In fact, the same goes for anyone or anything that is trusted. The knowledge of our existence is not a pre-requisite for us being able to trust someone or something.

Sure, sometimes there is some form of acknowledgement that we exist. Of course, there may be a moral or fiduciary order that we can appeal to when trust is placed: for instance, walking across the road in front of traffic is a pretty large display of trust in the drivers of the oncoming vehicles. Not being run over is one of those things that it is probably fair to expect in a society. Likewise, we might trust our children's teachers to behave morally and to educate our children to their best capability. If we go back to Bernard Barber we are looking basically at fiduciary trust in that case and in the case of many professional relationships: people in professional positions are expected to put their own interests aside, or at least have the best interests of the other at heart when engaged in those situations.

We (Patricia and I) own a few horses. We have in the past sent a couple of them to be trained in some way. All that we know is that the animal is being trained. We place a great deal of trust in the trainer because the animal can't tell us how it was treated. We trust our horses to behave in a certain way. Standing calmly when being groomed by a stranger, for instance. Or not going out of control after getting scared at a paper bag on the side of the road (you would be surprised at what scares otherwise calm horses).

But here's the thing: trust means placing yourself in someone or something else's 'hands' (or hooves). It does not give us the right to expect that they will honour that trust. After all, they may not even know they are being trusted! Where we may have the right to expect the trust to be honoured is in the situations where fiduciary trust[3], or standards of care, are known and acknowledged. The babysitter knows they are trusted and what is expected as a result. Likewise the surgeon or the lawyer. But does the horse? Or my dogs? Perhaps more to the point, how can I tell them that I trust them, and if I could, does it really matter? How would it change what they might do?

As a matter of fact, I have a service dog. Her name is Jessie and she and I went pretty much everywhere together before the pandemic (planes, trains and automobiles, for sure). She trusts me and I trust her. Nothing more really needs to be said. Except this: she will do whatever I ask her to because she trusts me. And because I know she would, I don't ask her to do something I know would be dangerous. Why are we diving into this particular rabbit hole? Because this is about an equivalence. When we talk about trusting machines, what exactly is the difference between that and trusting, for example, one of my dogs?

Yes, there's the second (Prometheus) question. I haven't forgotten. We'll get there.

Trust is about accepting the potential for things to go wrong. Sure, we can get clever and talk about the thing we are trusting understanding their obligations, which is usually what gets brought up. But I think (hope!) we just established that the thing that is being trusted doesn't even have to know that it (they) are being trusted. Where is the moral imperative to behave in a trustworthy fashion in such a case? And before we dive into the 'moral' problem there is no requirement in any of the trust definitions I have given in this book that have a moral imperative for the trustee. Bear this in mind.

Clifford Nass and Byron Reeves, from Stanford University, back in the 1990s, did some research about how people (humans) perceive technology (media). The experiments had to do with the way in which people interacted with a computer – the details of the experiments aren't that important, but very quickly, it was like this:

- Take a bunch of people and tell them they are about to work with a computer for some problem.
- Ask them how well they thought the computer did (most of them basically said "pretty good").
- Then split them up and have them do the evaluation again, but half of the people use the same computer and the other half use a different one (there were some pen and paper ones too).

What happened? The people who were taken away from the computer and asked in another room, by another computer, basically had answers that were much more varied and even negative. It was as if they were talking behind the first computer's back, and in front of it they were positive because they didn't want to hurt its feelings. Yes, they knew it was a computer.

The experiment was repeated with a computer able to talk (had a speaker) and the results were basically the same. What does this actually tell us? Well, the experiments were very vigorous for one thing, and I basically summed the whole thing up in no time at all, but in general we can see that humans basically treat technology

[3] If you haven't read about Bernard Barber yet in this respect, you can now...

as a social actor. Indeed, one of the principles that came out of the so called Media Equation is that when systems are designed they should be designed in such a way that acknowledges that people already know how to interact in a social way, and so to give them what makes sense in this context.

To put it another way: when people interact with technology, they basically see it as a social actor.

This is not a small thing. People already see technology as a social actor. This was in the 1990s, when AI was a twinkle in the eye of many a grant-seeking professor. A small digression is in order here. I know the history of AI. It is far longer than you might think — probably as long as people have been thinking there has been a goal of AI. My point here is that, taking a closer look at AI in the last 70 or so years since Turing's paper, the field has constantly re-invented itself. The reasons are many and unimportant here, except that one of them is to retain favour with those who would grant money for research. But then, I am a cynic in this respect.

What would the result be with a technology that actually behaved in context and (seemed to) understand the rules? Let me repeat: people already saw technology as a social actor. It's automatic. It's not like we even had to try to make them. There is always a thing around here that talks about anthropomorphism, which is basically assigning human aspects to, for example, an animal ("Oh look, she is smiling at me!"). I'm not talking about anthropomorphism. The people didn't ascribe human aspects to the computer. They already knew that they were interacting with a machine. They didn't give it a smile or a sense of humour, they just treated it as a social actor in its own right.

And so, let's repeat this thought: if we are already able (indeed liable) to see really quite simple technology as a social actor, where does that leave us with a responsive, adaptive technology?

That's what I thought too.

If we already see technology as a social actor, and we behave towards it as if it were so, it's an extremely small step from there toward trust. A couple of things:

This morning was March 17th, 2021. For fun I picked up my phone and said, "Hey Siri, Happy Saint Patrick's Day". The response? "*Erin go bragh*" (which basically means "Ireland forever", give or take the odd bastardization of Gaelic). When we could travel back in the distant days of 2019 at a "Trusting Intelligent Machines" workshop in Germany at the wonderful Schloss Raischoltzhausen[4]. I was having a conversation and referred to Siri as "she". It was (rightly) pointed out that Siri doesn't have a gender — Siri is just a machine, really. My response? "It doesn't matter."

If people want to ascribe personality to the things they use, who are we to tell them they shouldn't? I'm not going to pick on anyone here, this is only the middle of the very start of a discussion that is ongoing, and it is a moral imperative that the people who I challenge in some way have the right and ability to respond. That's hard to do when something is in print, as it were. So, let's say this: there is a debate about the trustworthiness of AI. There is an argument that we shouldn't actually think of trusting AI at all because, well, for one thing

[4] There are some perks to being an academic — another, also in Germany, is Dagstuhl — look them up, you can then be envious and I can gloat.

it's a machine and so the whole moral imperative question doesn't arise. For another thing, it was made. This is important because if the AI does something wrong and people get hurt we can basically ascribe blame to the programmers or the company that made it.

Before we go further with the other stuff, let's address this, because it's important. There are two things in this position that need to be addressed. The first is that one needs to ascribe blame when things go wrong. This is a cultural position. Other cultures believe in different approaches. For example, Indigenous peoples in North America have great faith in restorative justice. The point? Seeking to blame and hold responsible isn't representative of the entirety of the human race anyway, not to mention that it has had some rather embarrassing and cruel results — pigs and other animals are not culpable. Fortunately we appreciate this, now. How long will it take until both revenge and retributive justice are seen as outdated[5]? The argument is that AI cannot be culpable either, of course, and so we need to seek others to blame.

Consider this: when an AI is released into the real world, every experience it has changes it. It is almost instantly no longer the thing that was released. Who is to blame if it fails? When Tay was released to Twitter in 2016, she was innocent in the sense that she didn't know any different (although some things she did know not to touch). Her subsequent descent into racism and homophobia was perfectly understandable. Have you seen what gets posted on 'social' media? Much more to the point, she wasn't the agent that was released onto Twitter as soon as she was released. There really is no-one to blame. Truly. Sure, Microsoft apologized, but most importantly, Microsoft apologized like this: We are deeply sorry for the unintended offensive and hurtful tweets from Tay. It is easy to say that Microsoft was at fault, but Tay posted the tweets.

Did you notice something just then? I'll let you think about it.

There is a great deal of airtime devoted to making AI more trustworthy by, for example, increasing transparency, or predictability, or whatever, in the hope that people will trust it. The goal is to get people to trust AI, of course, so that all its beneficence will be showered upon us, and we will be, as it were, "All watched over by machines of loving grace."[6]

Sure, that was sarcasm, but the point is this: some people want us to trust AI. So the answer of course is to make it more trustworthy. This is answering the wrong question. Trustworthiness is something that, if you have got this far, you know is the provenance of the thing or person you are thinking of trusting. That is to say, we don't give trustworthiness to something, it either is or is not trustworthy to some extent. What we give is trust. More to the point, we can choose to trust even if the thing we are trusting is untrustworthy. Even if we know it is untrustworthy.

To labour the point a little more, let's return to the media equation. As a reminder, because it's been a few words since then: people treat technology as a social actor (they are even polite to technology). The argument that we shouldn't trust technology because it is basically just an inanimate, manufactured 'thing' is moot. I'm

[5] Yes, I know that is naïve.

[6] Which, if you don't know it, was the last line of a poem by Richard Braughtigan, as well as a rock band!

not going to argue one way or another about whether or not we should trust an AI. That cat is already out of the proverbial bag. If you haven't seen that yet, let me spell it out for you: that people already see their technology as a social actor means that they almost certainly also think of it in terms of trust. It truly doesn't matter if they should or not, they just do.

This leaves us with only one option, which is what Reeves and Nass told us all along: design technology on a path of least resistance. Accept that people will be doing what people do and make it easier for them to do so. Even if you don't, they will anyway, so why make it hard?

Let's briefly return to the trustworthiness of AI. I've already said it's pretty much a done deal anyway – we will see AI in terms of trust regardless of what might happen. The argument that we should make AI more trustworthy so that people will trust it is pointless. What is not pointless is thinking about what "trustworthy" actually means. It doesn't mean "more transparent", for instance. Consider: the more we know about something, the more we can control (or predict) its actions, and so the less we need to even consider trust. Transparency doesn't increase trustworthiness, it just removes the need to trust in the first place.

But of course, AI, autonomous vehicles, robot surgeons and the like are not transparent. As I already talked about in the Calculations chapter (and in the paper about Caesar, actually), we've already crossed the line of making things too hard for mere mortals to understand. Coupled with the rather obvious fact that there is no way you can make a neural network transparent to even its creator after it has learned something that wasn't controlled, we are left only the option to consider trust. There is not another choice. Transparency is a red herring.

That given, what can we do? We are already in a situation where people will be thinking about trust, one way or another. What is it that we can do to make them be more positive? Again: this is not the right question.

Really.

If you want someone to trust you, be trustworthy. It's actually quite simple. Behave in a trustworthy fashion. Be seen as trustworthy. Don't steal information. Don't make stupid predictions. Don't accuse people with different skin colours of being more likely to re-offend. Don't treat women different from men. Don't flag black or brown people as cheating in exams simply because of the colour of their skin. Just don't. It's honestly not that hard. It's actually not rocket science (which is good, because I am not a rocket scientist). If the systems we create behave in a way that people see is untrustworthy they will not trust them. And of course, with very good reason.

We are applying AI in all kinds of places where we shouldn't, because the AI can't do it properly yet. And we expect people will want to trust it in a positive fashion? Let me ask one question: if you saw a human being behaving the way much of the AI we have experienced does toward different kinds of people, what would you do?

Before we finish this chapter, let us, for the sake of thinking, climb out of one hole and burrow into a vastly different one. Let's revisit the Prometheus question.

Can An AI Trust Us?

It is possible that at this point that you begin to think that Steve has gone slightly bonkers. Bear with me. Let's think about it for a second. We are developing, indeed have developed, black boxes. For those who don't know what that means, a black box is something we can't actually see inside to see how it works. There are such things in the world today: Artificial Intelligences that are opaque to all of us; they have learned from the information given them (training data) and yet more from the experiences that they have had in the 'real' world. If the experiences that they have had impact the data that they reason with, then it goes without saying that each one is unique. Just like us. Just like my dogs.

I'm not saying that the systems we have developed are as complex or indeed as capable as humans, or even the animals we live with. But is that the point? If we are able to rank a very advanced (for us) AI, relative to the fauna of the world, at what point on the spectrum of living things would we put them? I said above that we can turn off the machines that these AIs exist on. It's true. It has been asked before if we have the moral right to turn off a machine on which resides some sentient AI that does not want to be turned off (it is the very thing Arthur C. Clarke's 2001 is about, or Sarah Zettel's excellent "Fool's War"). It's a fair question, but it sort of misses the point. Here's the thing: one of the things about owning animals (for example on our farm) is that in general we get to decide when they are born and when they die. And (as Terry Pratchett would have it) in-between we have a duty of care. It's important.

How does the animal feel about dying? I'd venture to suggest that my dogs don't actually think about it at all, they live in a long moment. But we have laws about treating them badly. We have laws about how they should be cared for. Moral expectations are very high with respect to their care. And their deaths. If I asked one of my dogs if they didn't want to die, what kind of answer would I get?

Have you seen the point yet?

The dogs trust me to care for them, just as much as I trust them to be gentle, not chew up my furniture[7], things like that. They probably class as sentient because they react to stimuli, and so forth. But they react because they can. A disembodied autonomous system that resides on a machine doesn't have that luxury. So, can it trust us?

If you have read this far in the book you will have seen that trust models exist for artificial agents. I even showed you one. Given the right signals, the right information, and in the right context, the agent will behave as if it trusts in some way. It will change (adapt) its trust based on experience. It will recognize different contexts

[7] Some hope!

(if given the right tools) and think about trust accordingly. Just like my dogs. The trust model I made was developed so that agents in different situations could reason about how and when (and what for) they trusted each other, but it has always begged the question.

Can a machine trust a human?

One last thing. Let me return to Tay and the question I asked after I had talked about her. Did you notice that I had called her "her" or "she"? Whether you did or not probably has a lot to say about what you thought of the chapter.

TRUSTLESS SYSTEMS

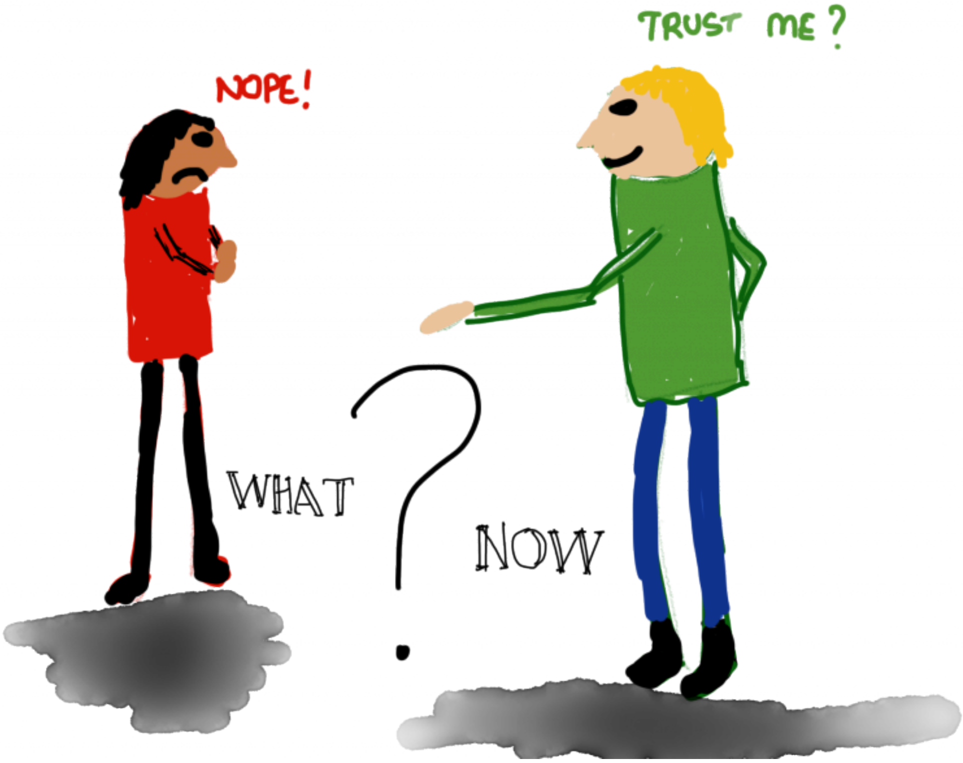


Figure 10.1: When this happens…

This might seem like a bit of an odd thing to put in to a book about trust systems, but it's a bit of an odd book, so that's one thing. Actually, trustless systems are, despite the idea that they might be trust-free, taking trust to a different kind of consideration. When we talk about trustless systems, we're mostly thinking about blockchains, and this chapter will look at those. After which, we'll have a little journey into things like zero trust security. It'll be fun.

Trustless Systems and Blockchain

But before I go any further in this chapter, it is important to give a huge thank you to Professor Nitin Kale who over the course of two wonderful and engaging SAP bootcamps explained more about blockchains than I could ever do, and graciously agreed to read and comment on a draft of this chapter for me.

Back in the financial crisis days of 2008, a paper was released by a certain Satoshi Nakamoto. It was entitled A Peer-to-Peer Electronic Cash System and it described in quite a lot of detail a process by which one could create a system of electronic money. Money is odd stuff – it requires certain things before it can actually be considered to be money – things like exchangeability, ownership, non-repudiation, and value (perhaps through scarcity, perhaps because it takes an effort to make or get it, and so on. Nakamoto's idea was to have a consensus-based hashed linked list to allow for these things to be achieved. The money was called Bitcoin, but the technology that it is based on is called a blockchain.

Figure 10.2: Recognize this?

Plenty of things have been written about blockchains, Bitcoin, Ethereum and so on, and indeed many many different electronic money systems exist. As well, many different blockchains exist, both public and private, to do all kinds of interesting things beyond money: smart contracts, real estate, international shipping and more. I don't need to add much more to that collection of literature, so this chapter is going to very quickly describe how blockchains work and a few use cases that are interesting from the point of view of trust (and no trust) and not go all in on how the idea might change the world and the way we work, live and play (If you're looking for utopianism you may have spotted by now that I'm not that into it).

A blockchain is a hashed linked list. To a computer scientist (me!) this is not entirely new – I learned about linked lists when I was an undergrad (at Stirling University) in the 1980s – but the way in which the blockchain works is quite intriguing because of the way it brings in cryptography and things like proof of work (PoW). Let's go through this from first-ish principles, starting with the concept of linked lists and going from there.

Imagine a scavenger hunt. You have to solve a clue to find the next clue, and so on, through a big 'chain' of clues until you get to the final prize. Each clue in the chain has a small prize associated with it so that getting there first is fun. It's a race, but when you find a clue you leave it where it is so that the other people playing have a chance to solve it too (but take the prize!). Each clue points to the next clue, and so on.

This chain of clues is akin to a linked list. Except that a linked list is stored in a computer. And you don't need to solve clues (usually – we'll get back to this later though). The way it works is very similar: each item in the list has some data associated with it (the data can be anything you like, just like the prizes in the treasure hunt – names, addresses, pictures, whatever, it's just data). It also has a pointer to the next item in the list. A pointer in computing terms is just like an address. It tells the computer where the next item is in memory or storage or wherever, much like the clue in the scavenger hunt 'points' to the next clue.

And that's it. We have in each item two distinct things (see figure 10.3):

1. The data;
2. The pointer to the next item in the list.

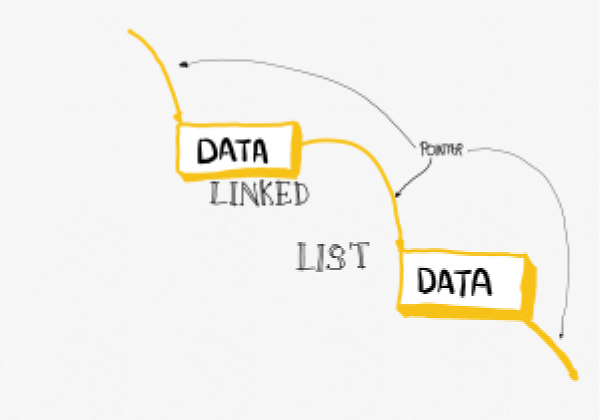


Figure 10.3: Your Basic Linked List.

You can get a bit more clever because sometimes it would be nice to be able to go backwards and forwards in the list (say, when you're searching for stuff, or re-arranging the list in some way, which we won't talk about because it can get interesting quite quickly!). In that case the items in the list contain:

1. A pointer to the previous item in the list;
2. The data;
3. A pointer to the next item in the list.

It's pretty neat actually, there's all kinds of different structures you can make out of it if you add more pointers and so on, but I think this is good enough for now.

So that's a linked list, but what about a hashed linked list? It's pretty straightforward, it just brings in a bit of cryptography for a few different purposes. The first of these is that you'd know if something changed. To get to understanding that, let's quickly talk about Hashes.

A hash function is a function that takes data of any length and reduces it to data of a fixed length. Any length is not an exaggeration. You can hash single words and you can hash entire encyclopedias and the end result is a single fixed length piece of data. It should also be **deterministic**, which means that the same input will always give you the same output.

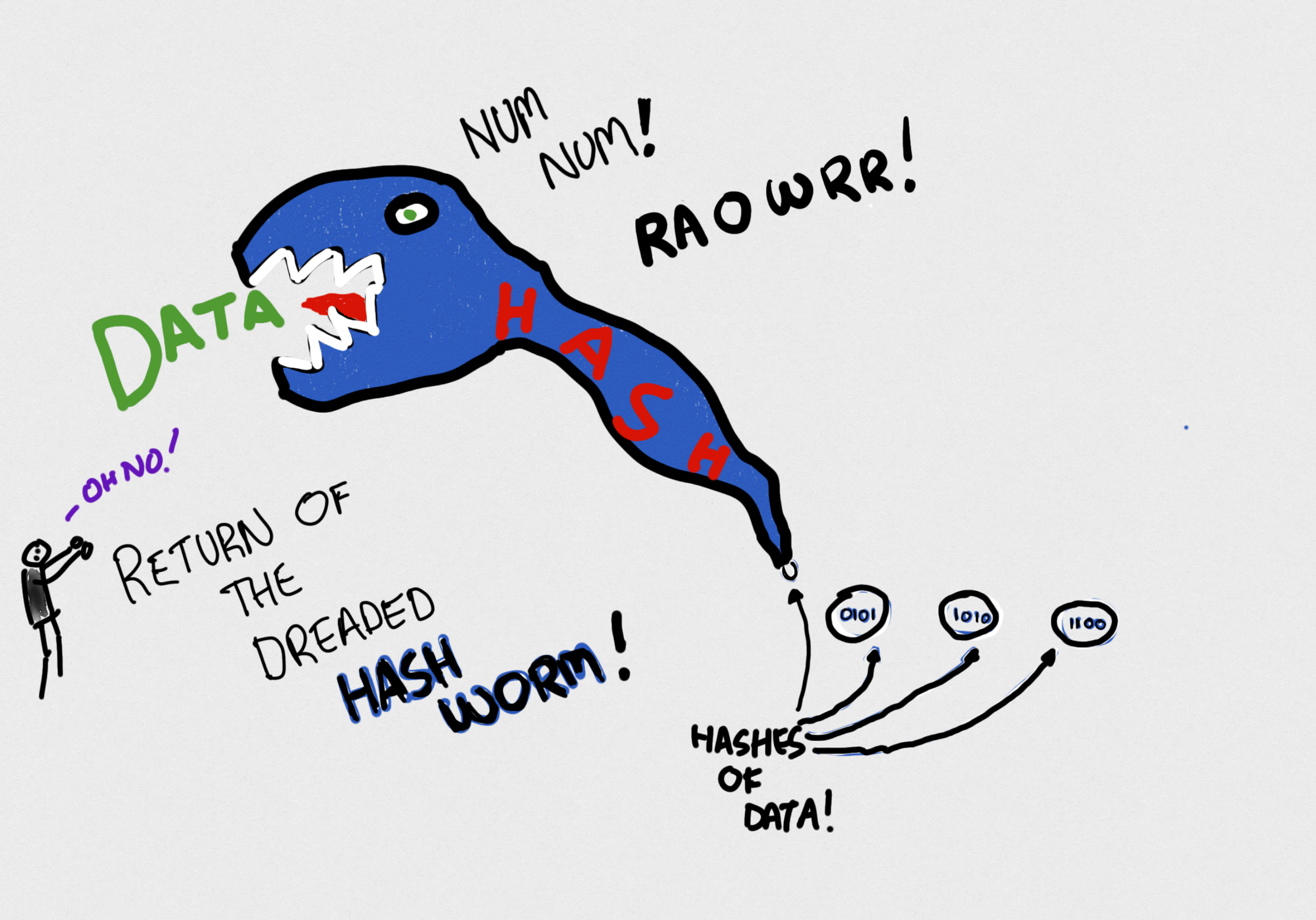


Figure 10.4: The Hash Worm eats data and basically poops it out in identical sizes. The thing is that each of these poops is unique based on the data (however much) the Hash Worm eats. Hashes are not actually remotely like this. Hash Worms do not exist.

But blockchains use **cryptographic** hashes. So what does this mean? Basically it adds a few requirements to that fixed length, deterministic one.

The first of these is called **pre-image resistance** (I know, it's funny how different fields all have their own little ways of talking). Pre-image resistance goes like this: If you've got a hash (call it *h*) it's difficult to find a piece of data (which in the hash world is called a message, so we'll call it *m*) that will give you that hash value when you pass it through the hash function. This means that the function is basically one-way – you can get a hash from a message, but you can't get the message from the hash easily.

The second quality we're looking for is called **second pre-image resistance**. This basically means that if you have a message, which we will call *m1*, then it should be *difficult* to find another message (call it *m2*) that gives you the same hash. This is also called **weak collision resistance** because in hash world, a collision is two messages having the same hash (which then we have to do something about, like create a bucket of messages that have the same hash).

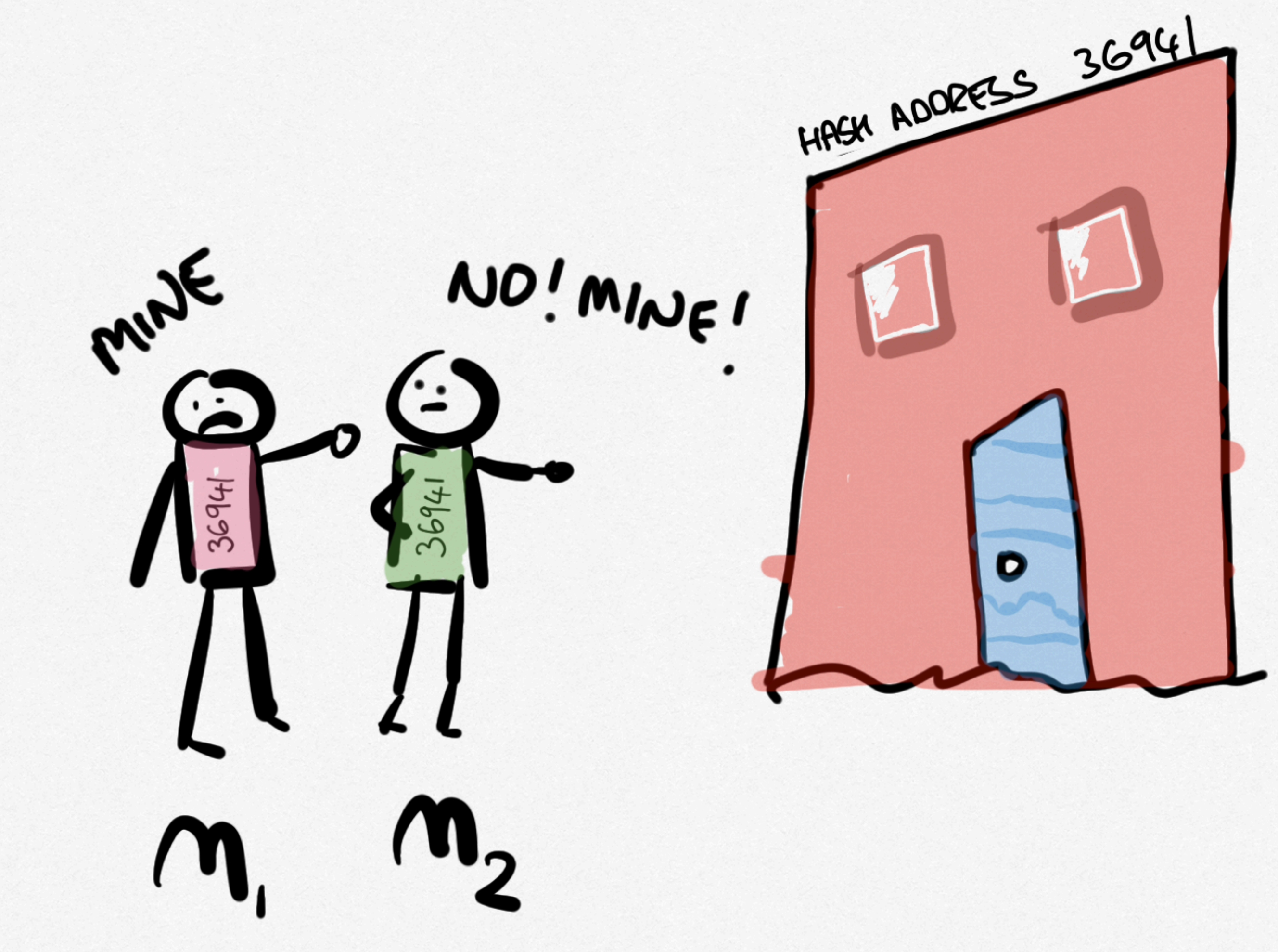


Figure 10.5: A Weak Collision. Sort of. Basically, M1 and M2 are trying to fit into a house big enough for only one of them. The Hash Function has given both of them the same address (hash) and these collisions are not what we want.

Why is it called weak collision resistance? Because we have a strong collision resistance too! This is quite similar but it says that for two messages (we'll call them *m1* and *m2* again) that are different, then it should be difficult to find a hash for *m1* that is the same as the hash for *m2*. This sounds the same as the weak one but isn't quite, if you think about it carefully.

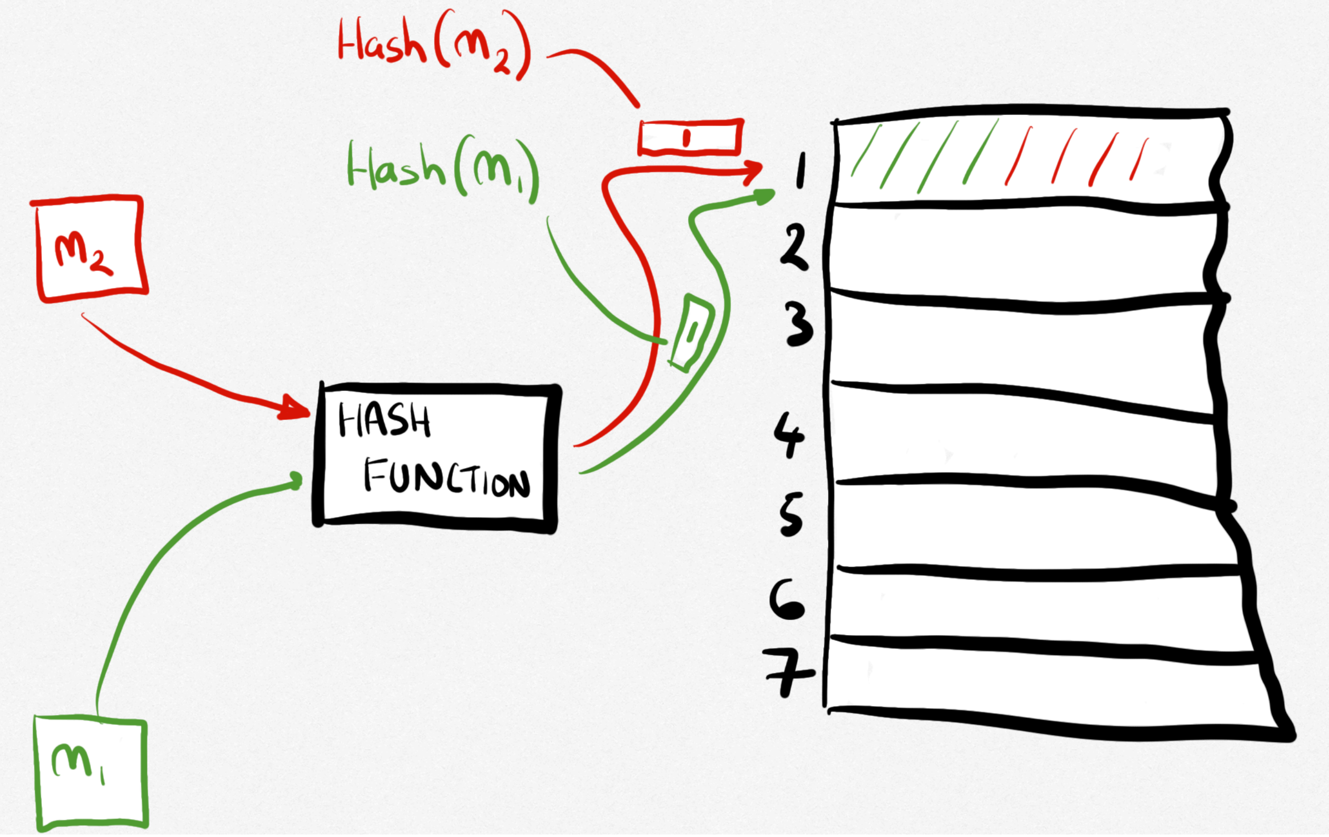


Figure 10.6: A collision actually looks more like this: two different pieces of data, after hashing, arrive at the same address. We don't want Hash Functions that do this – that is, each unique piece of data should ideally have a unique hash (address).

Those things all say difficult which basically means very hard but also in more technical terms means 'would take an awfully long time'.

There's also something called the **avalanche effect** which is nice to have. It basically says that a small change in message (say from "Steve is wonderful!" to "Steve is wonderful?") would result in wildly different hashes from the hash function[1]. Finally, the function has to be **uniform**, which means that the probability of one hash number is the same as all others. This will come in useful in a moment.

Why are all these things important? Because when the function meets them, there's very little chance that an attacker can get a hash and find out what it is the hash of, as well it would be really easy for anyone to know if the message was somehow changed (because the hash would be very different).

Bitcoin (and lots of other blockchains) uses the SHA-256 hash function, which is from a family of functions called SHA-2, which was developed by the US National Security Agency. It's pretty secure. The number of

[1] Sure you can see the difference!

possible hashes is 2 to the power 256 (2^{256} – see Figure 10.5) which is quite a large number. The hashes are all 64 hexadecimal characters long. It meets the properties we talked about just now.

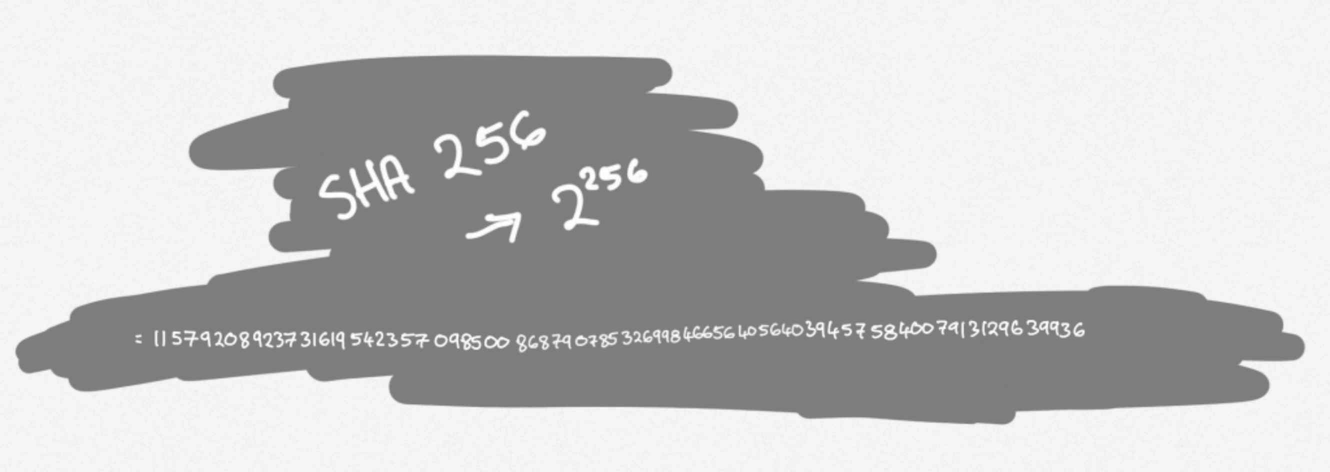


Figure 10.5: Quite a Large Number then

Smashing, now you know all about hashing!

Bitcoin, if you are interested, uses this SHA-256 function to encrypt usernames (so that there's privacy) as well as for something called proof of work. Which we'll get to in a bit.

Okay, so now that we're okay with cryptographic hashing, how does this work with the linked list thing? The first thing to bear in mind is that we want to be able to know if someone changes stuff. If they can do it without us spotting it, then we have a problem. In an electronic cash example, you could send money to someone and then go back and change the fact that you had sent it after you got the thing you were paying for. This is clearly not a good thing. So we're looking for some guarantees that if someone tries to change stuff, we'll know. Secondly, we're looking for guarantees that what has happened will stay happened. You may have spotted that our cryptographic hash stuff helps there.

As far as blockchains are concerned, the links in the linked list point backwards. That means that every block of data has a pointer to the block that came before it, not after it like in the scavenger hunt. All the way back to the first block, which is called the genesis block. Imagine a murder mystery where you work backward from what happened (the victim is dead) to how it happened (the actual murder). Okay, so that wasn't a brilliant example but you get the idea.

Why do the links point backwards? Because we want to be able to look back through the history of all the things that happened so that we know that they did. If we see a transaction, say, and the hashes for that data are correct (it hasn't been changed) they we 'know' that the transaction happened.

Now, if you look at pictures of blockchains on the web you will likely see the pointer going forwards. This is not really how it works (they basically point backwards – the pointer is a hash of the block header from the previous block), but there you are.

Here's a fifty-thousand foot view of what happens in practice. Every so often a block gets created (we'll come to that in a second). That block contains a set of transactions (in Bitcoin they would be someone sending money to someone else, for instance). We know the block that came before (everyone can have a copy of the entire chain if they want). We don't need to actually point to it like we would in a regular linked list. What we do is hash the header of that block (you can see a header in Figure 10.8). This gives us a hash that we can include in the new block as part of its data. The rest of the data is pretty much transactions, which are stored in the form of what is called a Merkle Tree (the details of which aren't really that important here). This is itself hashed (so that we can know what length it will be, you see, as well as knowing if something gets changed, because if it did, the hash would change). The hash of that gets put into the header of the block. There are a few bits of metadata that help the chain figure out what version it is, when it was made, and something called nBits, which relates to proof of work, and these are also stored in the block's header.

And then there is one small thing called a nonce (fun fact: nonce means number used once. Oh, the fun computer scientists have).

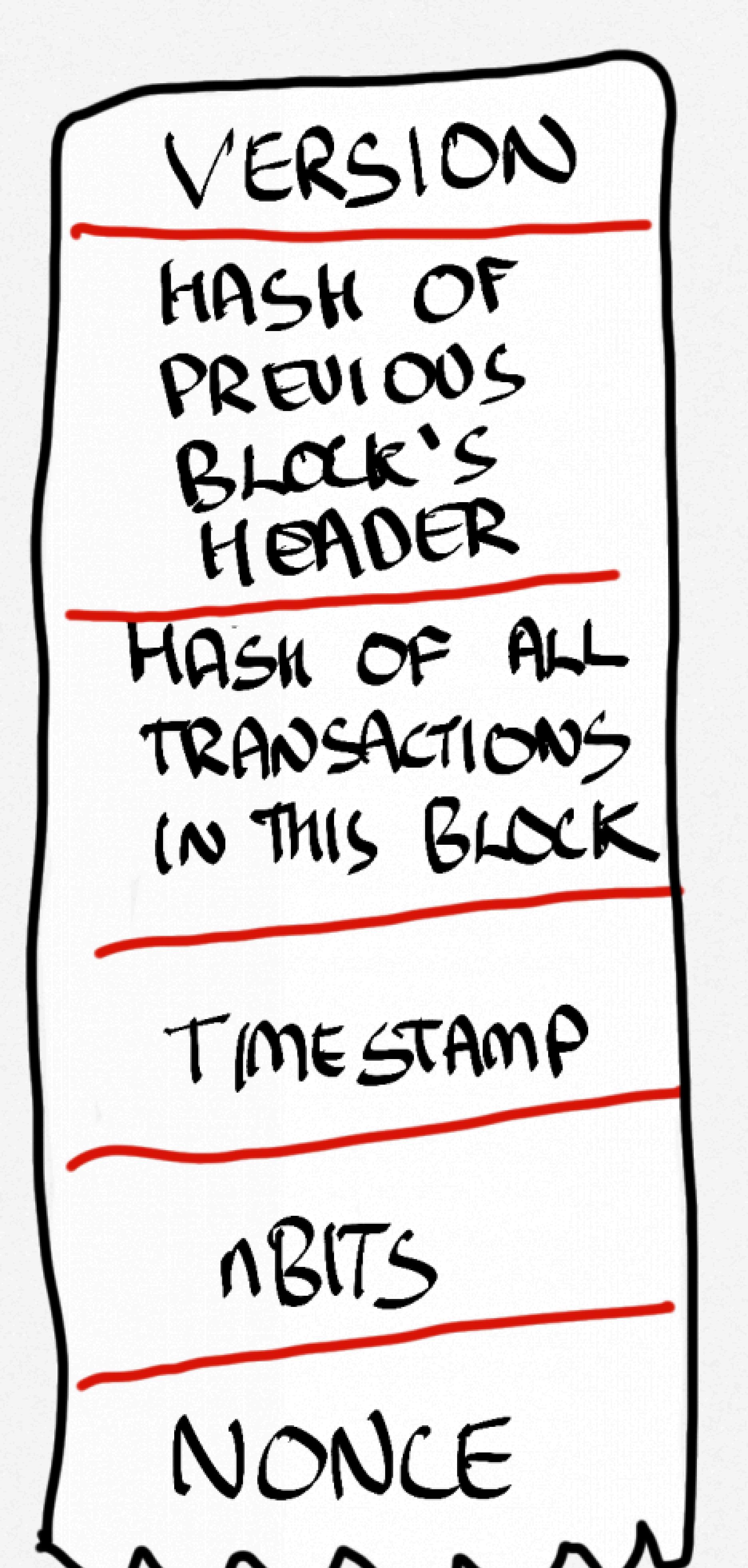

VERSION
HASH OF PREVIOUS BLOCK'S HEADER
HASH OF ALL TRANSACTIONS IN THIS BLOCK
TIMESTAMP
nBITS
NONCE

Figure 10.8: A Block Header.

Here's where the proof of work thing comes in. You see, there is a reward (in Bitcoins if we look at that blockchain) for actually creating blocks, it's called mining. If it was really easy to make blocks, everyone would do it and the cash would be effectively worthless: neither rare nor hard to get.

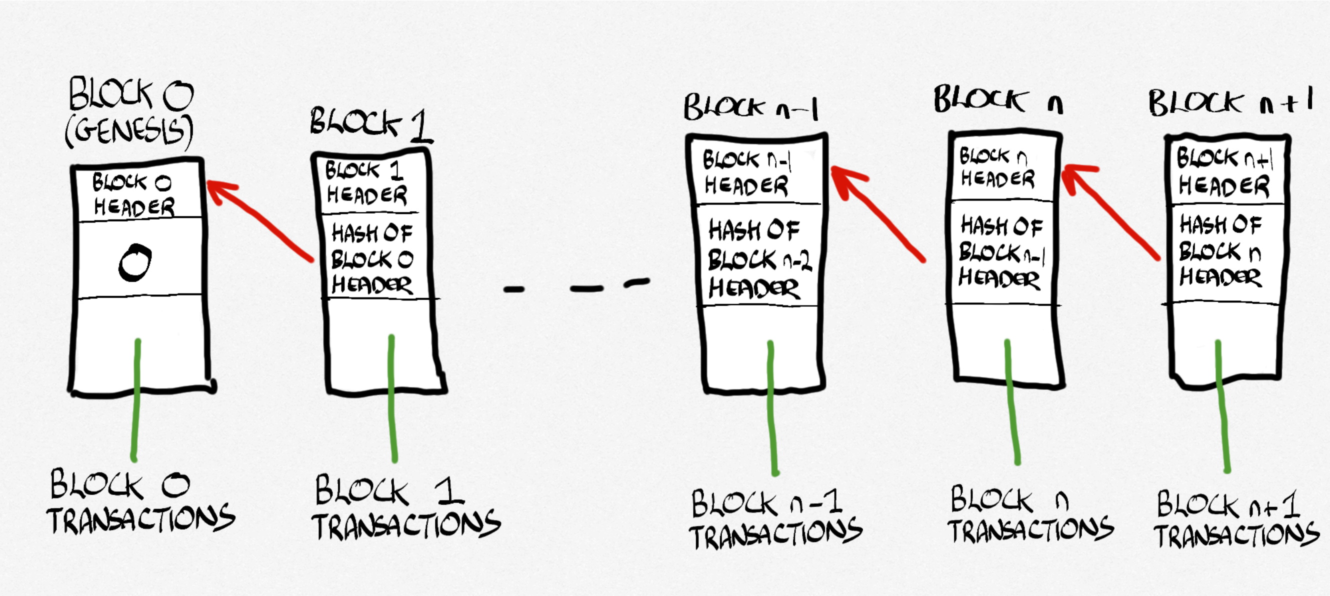


Figure 10.9: A Blockchain...

So, to complete the block, the miner has to find a number (the Nonce) that would mean that when the block data (which is the transactions, the hash of the previous block and the Nonce) is hashed it has a set of, for example, 4 or 5 or whatever zeros in front of it (or some other kind of pattern). Remember that nBits in the header? In a nutshell, that is the number of zeros we're looking for. It changes as the blockchain gets older.

Remember that the hash function is uniform: the probability of one hash is the same as for any other hash. So how do you find a hash with lots of leading zeroes? Start with the Nonce being 0, hash the whole thing, and see what you get. If you don't get what you are looking for, add one to the Nonce (now we have a Nonce of 1) and hash it again. And so forth.

Basically, count up from 0 until you find the first hash that has several preceding zeros. You can (and miners do) split up that counting, so for instance you could have ten machines, one of which starts with a Nonce of zero, the next with a Nonce of, say, 1,000,000, the next with a Nonce of 2,000,000 and so on. Which (if the hash space was ten million) would effectively do the search in around a tenth of the time. But the hash space is actually , which I am reliably told is larger than the number of grains of sand on the planet. This led me to wonder who counted them, but no-one has been able to answer that question. Anyway, it's huge, but there's no way to find the thing you're looking for other than starting at the bottom and going up.

This, naturally, can take time and it takes huge quantities of computing power (seriously, these miners have massive numbers of highly optimized machines basically searching for the right Nonce.

And when they have found the right Nonce, they push the block, the Nonce and the hash out to the community. It gets checked and, if they all agree that it is correct, and it is the first one there, it gets put on the chain ready for the process to start all over again.

Figure 10.10: Proof of Work! Well, not really, but you get the idea. Miners have to show that they actually did the mining.

That search is called **proof of work** and it's how miners prove they didn't cheat and mess about with the data in the transactions, for one thing. It is rewarded with new coins (which is why it is called mining) as well as promised transaction fees from the people whose data is in the transactions for the block. In effect you can offer nothing, but it's unlikely that any miner will choose to take that transaction and put it in a block because it's worthless to them. Which brings us to the final aspect of all this, the mempool, which is a big collection of all the transactions that have been put in play by people (like writing a cheque, really, for instance) that haven't been mined (put into a block) yet. If they're not in a block on the chain, they haven't actually happened. Miners pick the juiciest (best paying) transactions and put them in blocks to mine (find the Nonce) and get rewarded for doing so.

If you have been paying attention you will notice that I said "the first one there" when I talked about the community verifying the work done. That's because there are lots of miners, all of whom are beavering away at finding those happy Nonces. It's a race. First to get there wins everything. And if you're not first, the block you were working so hard on empties back into the mempool and you start again.

Phew. Blockchains in something like ten minutes.

I should probably talk about **proof of stake** whilst I am here. You see, proof of work has some serious shortcomings. For one thing it takes huge amounts of power to keep all those miners' computers going. Seriously. Bitcoin, for instance, has an annual carbon footprint around the same size as New Zealand (if you want to see the statistics, here is a good place: https://digiconomist.net/bitcoin-energy-consumption). For a single transaction, the carbon footprint is equivalent to almost 700,000 VISA transactions. We're not talking small change here.

At this point, take a thought as to the environmental impact of Bitcoin alone. To a certain extent, this is in the realm of "just because we can do something doesn't mean that we should".

Naturally this is very expensive. Which means that miners sell the bitcoins they are rewarded with to pay for it all with money in the 'real world' (technically, it's FIAT money). This has a few different effects. The first is that it devalues the worth of the currency itself (in this case Bitcoin). The second, more serious, goes a bit like this…

There is a limit on the amount of Bitcoin out there. It's built in to the system itself. Eventually, the miners will run out of coins to mine. This means that the only reward they will get for creating blocks is that which the transacting parties are willing to pay. This will decrease over time rather than increase (at this point the transacting parties are the ones with the power) and so, eventually, will the number of miners (it's not financially viable).

Here's the thing. Bitcoin, indeed all blockchains, are only secure against attack from anyone who has less than 51% of the miners. If someone is in that position, they can create their own transactions, invalidating anyone else's, and no-one would be able to stop them. Naturally this is something of an issue if, for instance, you're only left with a couple of miners doing all the work.

Sooner or later, it's always all about trust. Bet you knew I would get to that. Moving on.

Proof of stake is an idea that says: instead of miners proving that they did a bunch of work, which is basically unsustainable in the long term (I would venture to suggest the whole thing was unsustainable in the beginning, but that's what happens when you get computer nerds to try to solve a puzzle: they throw power at it), let's make it so that miners can only mine based on a percentage of the coins that they themselves hold. They have a Stake in the success of the currency. This means that even if they did get to 51%, they wouldn't want to screw around with the currency because they would actually have 51% of the coins and they'd be hurting themselves quite a lot. It also means that relatively small miners can do some work if they have a small stake.

Bitcoin uses proof of work. Ethereum (another blockchain which I'll come to later) is moving to proof of stake. There are innumerable blockchain cryptocurrencies out there[2] and some use either whilst some use hybrids of both. But at least now you know what it all means.

To get back to the point quickly: if someone wanted to go back and change a block, say by putting an additional 0 on the money that they were paid, a few things will happen:

[2] Seriously. For a while there it was like - turn around and there's a new one!

1. The hash of the block will be incorrect
2. So they will have to re-mine the block (remember, it's expensive!)
3. But if they do that, it will invalidate the hash of the next block because the next block uses the previous block's hash and, well, it's different now).
4. This would mean they'd have to re-mine that next block.
5. And so on, until they get back to the top of the chain, which, because time does not stand still, has been moving forward anyway, meaning that they'd have to re-mine a whole bunch more blocks and still fall behind.

The further back the block, the more infeasible it becomes to change it. That's why some people wait until transactions are a few blocks back before they finish the transaction. That is, if I paid you a million bitcoins for something in a transaction, you might well wait for ten or more blocks to be created on top of the one with that transaction in it before you, say, sent me the super yacht.

Why would you send it, you ask? Because you can't at this point say that you weren't paid: it's easy to go back and prove it, because it's on the chain. And I can't get the yacht and grab the money back by changing the block the transaction is in because it would be far too expensive, I may as well let you keep the darn yacht.

If you‘re interested there's really neat example of blockchains (and hashing) here, courtesy of Anders Brownworth. It shows a bunch to the stuff I've just been writing about quite nicely and there are even some videos. And unlike other blockchain demos it doesn't require you to give it your email address so that it can spam you forever.

Now, blockchains are sometimes called **trustless technologies**. This doesn't actually mean what you think it might mean. In fact, trust is very much a part of the picture. Did you spot why?

It's like this. Blockchain technology allows for interactions and transactions in situations where there is either no trust or not enough trust. It does this by taking the thing that people are concerned about and replacing it with something that they are not. To be more precise, it allows people to trust the technology in the place of trusting the person. If you wanted to be more pedantic you might say that it allows people to trust a society that ratifies a certain transaction rather than having to trust the person they are transacting with.

This is actually very important. Things like this allow for collaboration and cooperation where trust is in short supply. And that helps the world go round.

System Trust

But here's the thing: blockchains don't do away with trust, they just move it someplace else. I've talked about **system trust** a couple of times in the book, and it's a good time to look at it here because of how it applies to blockchains. It doesn't just apply to blockchains though, as we'll see.

System trust isn't a new concept. In fact it was called 'institution-based trust' in a paper by McKnight, Cummings & Chervany (1998), itself referring back to Shapiro's (1987) "The Social Control of Impersonal

Trust". What is interesting about the Shapiro paper is that it finds that the various social trustees that we put in place to help protect us against the need to trust are themselves trusted (naturally). This leads to problems because it's rather paradoxical. The problem? We don't get rid of the need to trust when we rely on other systems, we just shift it someplace else. Sound familiar?

Sure, it's more complicated than that, so let's explore it a little further. It's like this: there are many many times every day when we engage with agents over whom we have no control – insurance companies, pharmacists, specialists, the government and so on. The reasons are many, but to quote Shapiro (page 630):

> "unwitting principals, blinded by distance, organizational structure, secrecy, and lack of expertise, idly await the future dividends of symbolic promises made by faceless strangers."

The only *real* control we might have is to simply not engage, and that's rather problematic if you want to continue to exist in a complex society. The problem, you see, is that all of the little things we put in place to help us move around this complexity are founded on trust (they certainly hold up to a definition of putting oneself in the control of others) but the systems in place actually do little to help enforce the correct behaviour of the others involved. Shapiro calls them 'Guardians of Trust", and they can be seen as enforcing things like fiduciary trust (which you've seen before when we looked at Barber earlier).

So who watches these watchers[3]? Who, in the final analysis, ensures that the "Guardians of Trust" are trustworthy? Well, naturally people put in place to watch them. Perhaps you can see the problem, since at this point it probably makes sense to put people in to make sure that those watchers are themselves trustworthy… As Shapiro says (page 649):

> "One of the ironies of trust is that we frequently protect it and respond to its failures by bestowing even more trust."

And that's basically system trust. As problematic as it is, we put trust in the systems around us to ensure that the people around us that we can't control behave properly when we are not in a position to know if they are. And so forth. It's at this point where you might be asking "well, are there any other trustless technologies than blockchains?" This is a pretty good question.

Most importantly, there is actually something called zero trust in the realm of Enterprise solutions (not the space-ship). Let's take a look.

[3] Yes, the old Qui custodiet ipsos custodies issue!

Zero Trust

Figure 10.11: Zero...

Before we start, the basic premise of **zero trust** is that your systems are already being attacked. I've often started talks about trust and security with the basic statement that if you have a system connected to the Internet, either your system is compromised now or it will be soon To put it another way, zero trust assumes that there is definitely an attacker in the system. This is not an unreasonable assumption. It also means that, regardless of whether you are "inside" your own "protected" system or not, you can't assume that it is trustworthy. I first identified zero trust as a special thing in my PhD thesis back in 1994, but as a security practice, zero trust was advanced by a certain John Kindervag at Forrester. It is worth noting here that in general zero trust applies to things (computers and such) in a network, and people have coined the term "next generation access" to take this to people in a network, but for now we'll mix them all up into the single term of zero trust.

So how does this all work?

Traditionally, security goes something like this: put up a bunch of firewall, use defence in depth, make sure that the perimeters of your systems are protected and "impenetrable". As well, is people need to be off site and away from the network, use things like VPNs (Virtual Private Networks) to ensure that, although they are away, they are still 'within the perimeter'. Much of this uses things like encryption, packet monitoring, access control tools, some AI perhaps to spot traffic that doesn't seem 'right', and so one. In other words, a whole bunch of tools to ensure that your internal network is 'secure.' And then we call it 'trustworthy.' This sounds like a great thing until you realize a few different issues.

The first is that there are people in your network and, like it or not, people are special. They will write passwords down on Post-It Notes, they will forget to change the network their computer has access to when they are at home, they will be susceptible to things like phishing attacks and so on. People are often said to be the weakest link in a security chain. When you look at all the things that might go wrong because of them, you might think that this is a reasonable description. We'll get back to that!

The second issue is that generally, if something gets 'behind' all your protections, the protections themselves don't tell you. In other words, when they fail, they fail silently. The end result is that anyone or anything who is in the system with nefarious intent is pretty much in the clear unless you do some hard looking yourself. Some logs might help (and indeed do) but you need to look at them and see the issues.

And third? When something is inside, the permissions are quite open, especially if you have been relying on the perimeter to protect you. Christian Jensen, a colleague of mine from DTU in Denmark, likens this to a medieval castle. It has strong walls but the odd chink in the mortar that might just let a person inside if your sentries are not paying attention. Once inside, that person could dress like a soldier, or a peasant, or whatever, and get places you don't want them to with impunity. Remember that Handsome Prince story? Well, when the Prince got into the tower he'd got through the defences and could do as he wished, basically. Those issues are evident when you look at things like foreign access to US infrastructure that has been reported recently here or the SolarWinds debacle. It's not getting any better, because…

Fourth, **you're in an arms race**. You put protections in, the bad folks try to break them, developing bigger and better tools to do so and selling them to the people who want to break in. And so on. The thing about arms races is that they never end. Meanwhile, the bad people only have to succeed once whilst the security folks have to succeed every single time there is an attack. Not great odds.

Finally, this: we are great at making tools that take away some of the issues that we associate with people: we have spam filters, phishing attack filters, firewalls, different kinds of authentication and so on (we'll get back to the authentication thing. We'll also talk more about attacks later in the book). The thing is, many of these tools spot and deal with attacks that would be obvious to the human anyway, and let through the things they can't spot to allow the human to figure them out. Most organizations will train their people in security but hardly ever enough[4]. Thus, the person doesn't see the attack because:

1. They were never trained to see it; and anyway
2. It's hard to spot. Because if it wasn't the automatic systems would have spotted it!

At which point this becomes a human fault (or as some smart security types say PBKAC – Problem Between Keyboard And Chair – truly helpful) which in fact it quite clearly is not. Regardless of the idea of Zero Trust, we can do much better by the people we are trying to serve, I would venture to suggest. Add all this up and we can see that 'trustworthy' it most certainly isn't.

Anyway, zero trust does away with these assumptions. The real assumption that it makes is that there is an imposter inside your network. It could be a piece of hardware with spyware on it, or an Internet of Things device with poor security (that would be most of them then). It could be a BYOD (Bring Your Own Device) tool like someone's iPhone or Android device which you can't examine to see if it is compromised or not. It

[4] Regardless of the fact that people are in fact the last line of defence in a secure system.

could be a bad actor (human). Many of these things could happen and more, if you have been looking at the things I wrote just now. And of course, spotting them is hard.

So what to do?

Trust nothing until you have to, and even then assign least privileges. This means, figure out who is asking to be able to do something and whether or not they say they are who they say they are (identity), then figure out if they are allowed to access the thing (asset) they are asking for (access). If they are, ensure that this is all they get to do (monitor) and that the person who owns the asset knows what is going on (report).

It's not entirely new – least privilege and 'need to know' and so on have been around for a very long time. What is new is the acknowledgement that the systems we think are trustworthy actually are not, and that everything inside your own network has to be treated the same as everything outside it – with zero trust – until has proved it is who it says it is, and that it has access to the things it asks for.

That authentication thing? You can use MFA — Multi-Factor Authentication — to ensure that identity is what it is claimed to be. MFA is about asking the person or thing who wants access to prove that they are who they say they are in different ways — a password, a PIN, a swipe card, biometrics and so on all mixed up into a set of things needed for that proof. Often you hear "something you have" (like a swipe card), "something you know" (like a PIN or a password) and "something you are" (like a fingerprint or iris scan). There are many many different ways to combine these, and the more you have, the more secure the identity proof is (in theory).

You can combine these in zero trustwith things like periodic re-challenging — that means asking the thing or person to go through that proof again. Or you could monitor what is happening and react to things that are not or should not be allowed based on the access privileges assigned to that person or thing. Basically, you create a model of everything in the network, assign privileges to these things but only when asked for and monitor to make sure those privileges are being adhered to. In addition you could monitor to ensure that, even when they are being adhered to, the person or thing doing them is the same person or thing that was granted the privileges in an ongoing way.

Phew.

Interestingly, this page says "Put Your Trust in Zero Trust" which basically says everything you need to know about if there is no trust present or not (remember that System Trust thing?).

One more thing: the way this works works is pretty much exactly how we have been saying trust works in this little book. This is a significant departure from regular information security, which talks in terms of being able to "trust" something that is supposedly secure. Which would mean that there's "no risk, we've got you covered." As you may recall, the whole point of thinking about trust is because it tries to get things to work when there are no such guarantees. In other words, that nobody can bet trusted until you've thought about what you are trusting them for. Which means doing things like determining the trustworthiness of people or things in a given context, continually monitoring if needed, putting safeguards in place and so on. If this doesn't sound familiar in the case of Zero Trust and Next Generation Access it's because you haven't really been paying attention...

As it happens, back in 2012 I happened to be giving an invited talk at the Information Security for South

Africa conference (ISSA) in which I basically said things like “your systems are already compromised, or if not will be soon. The thing that will work is trust,” before heading into a description of how which looked remarkably like zero trust[5]. Whilst I wasn’t booed out of the room, I was told by a professor in the audience that (more or less) I had just told him that basically the things he had been doing for his entire career were wrong. This may be true, but I don’t think he meant it as a compliment. Such is life.

If you’re interested, a good overview of zero trust is here. It may not change your life, but it will show you how being suspicious and using trust properly can actually help security (honestly, it took a while for people to realize that…).

Is it trustless? It is, in the same way that blockchains are trustless. There is of course trust, but at the same time it is trust that has been transferred to the system or managed in a way that ensures that it is placed extremely carefully (and not for great lengths of time). It’s probably as close as you’re going to get to a trustless system because, at the end of the day, trust is always somewhere. If it wasn’t, nobody could actually do their jobs, secure or not.

It’s an old observation that the only really secure system is one that isn’t connected to anything else and has all its ports glued shut so that nobody can plug anything in. It’s also a pretty useless system in the final analysis because a lot of the power of the technology we use comes from things like network effects – being connected is a huge boost to potential.

When you’re in that kind of place, somewhere or other you’re going to have to trust someone who needs access to a file, or something that needs to forward a packet. You can put in place things like zero trust, and indeed probably should, but in the final analysis if you want something useful to happen, well, you know where to look.

[5] Yet another example of me being far too early with an idea!

RESEARCH AND ONGOING CONCERNS

Introduction

Not being one to miss out on a chance for self-promotion, like most scientists, I'm going to use another chapter to talk about some of the work that I and my colleagues have done. In a career of around thirty years (as I write this), I have been lucky enough to be able to do this in many cases with people whom I respect and admire, and their names will be shown in this chapter.

Some of what we've done has explored how trust systems might be used to defend people and systems from attacks like the ones I talked about in the earlier in the book. Some has been exploring Social Adeptness in technology — so that the technology can exist alongside us in a functioning society. In other work we've explored how information can flow around networks when trust is used as a tool. There's more, too.

Helping People

As I already noted, attacks on people often rely on their expectations for social behaviour (phishing and social engineering, in particular). I've already mentioned that you can put policies in place in organizations to help manage this, but sometimes we're not in an organization — we might just be on the street, or at home, commuting, looking after a sick child, or whatever the case may be.

In one sense, there is a need to armour people against the worry of saying, "no". In another sense, though, there is a need to armour people against situations where they just aren't able to focus. Which is most of the time, but in some circumstances the focus is much less than in others. This is natural – that sick child takes precedence. As does a car accident you were in, or witnessed. You can probably think of many more examples.

In any case, we surmised a while ago that the technical (process) part of the trust system may well be a powerful defence in both cases. Let's focus on the first. As it stands, the technology that is deployed has has no concept of what is proper social behaviour. The supposition is then that it doesn't really get embarrassed when having to say, "no" for instance. This is the premise behind what we call foreground trust and its associated technology, device comfort. Let's have a closer look at how this all comes together.

Foreground Trust and Device Comfort

I will start with a reminder of what you already know (if you've been reading till now!). We care about empowering people. Part of the concept of **trust empowerment** is giving people the tools and information they need to make a trust-based decision without actually making it for them. This is a subtle problem and a political position. Giving people information that matters to them and is not disinformation is a political position. There is little doubt that there are people or organizations that benefit from an uninformed, or worse, misinformed, public. Unfortunately, as we have seen, it doesn't take a great deal to make misinformation (or indeed disinformation) happen in the social networks that exist in the start of the second decade of the 21st Century. This is a major problem and taking a stand against it is inherently making a statement about what is important to you. This is political, and that's good.

In order to give people information it pays to know what context the information is needed in. If this sounds a little like "trust is contextual", you are indeed correct. Information is contextual, in the sense that different information is naturally more or less important than other information at specific times and in specific contexts. For example, the fact that Steve has no hair on his head is of no importance if you are asking him for help with a mathematics problem. But if you were looking for him in a crowd of hirsute men, it might make a difference. But, you might think it matters in the mathematics problem too.

The point? It's up to you.

I mean, sure, it makes little sense, but perhaps there is something about being bald that means you're less likely to ask for advice. You have your own reasons. Foreground trust is a way of acknowledging both the contextuality of information as well as the individual information needs (and reasons). Its basic premise is that the person (or agent) making the trust decision has specific requirements that matter to them in that context. It becomes much more important when you consider that online there is a dearth of the signals that we normally use to make trusting decisions. Zoom can only do so much to help you see the body language we have used basically forever to assist us. It is worse when there is no Zoom. This brings us to **trust enablement** and **foreground trust**.

Trust enablement is a theory discussed in Natasha Dwyer's PhD thesis, which examined how trust worked in digital environments amongst strangers. The point is to allow users of digital environments to, as Dwyer would say, "proceed on their own terms". Which should sound an awful lot like the past few paragraphs tried to say, just in a few less words. The idea? Create a tool to help people gain the information that they needed from the systems that they were using, in order for them to make trust-based decisions around other people in an online environment (where normal social cues are often unavailable).

Trust enablement leads us quite nicely to allowing systems to be able to figure out what that information might be, or at least ask and (and learn). This is the premise of foreground trust. It's not really so different from Trust Enablement (and both are empowerment tools). The idea is to help people with technological prompts about the trustworthiness of the information or systems (or indeed people) that they are dealing with. If you remember the chapter on complexity, it touched on Foreground Trust a little when the ten commandments were discussed. A similar concept, *anshin* is presented in this paper and this paper. *Anshin* (a sense of security) is a Japanese concept that the authors examine the context of security, for instance in online tools.

Let's bring this back to trust enablement for a moment. The point is that the systems we create can use social contexts and cues to help people do what people do. Especially where there are problems for the person getting the information needed.

Let's take this a little further. What would it be like if we could take these things and allow systems to be able to reason with them in context? To that end, our own work, similar to *anshin* but applied in a different context, is called device comfort. Device comfort extends foreground trust by allowing personal (and ultrapersonal, like wearable) devices to be able to determine context and tasks preferences and requirements for their owners, and use this as a tool to represent their individual level of "comfort" to the user based on what is happening to them, or the user is doing.

Why does this matter? Remember the thing about about focus? Right. A computational system can be designed to be focused specifically on the important security (or other) and behavioural cues of the owner as well as others in the situation. Imagine: a device that specifically flags when something happens to affect your focus — like a 911 (or 999) call following that accident, or a different route taken to work — and lets you know when you are about to do something that has the potential to be dangerous (or uncomfortable). Like clicking on a phishing link — indeed any link unless you have been paying attention to what you are doing. Or, say, tweeting when it's late at night. Or unlocking your device when crossing a border, opening yourself up to different kinds of espionage. As an aside, we did develop the idea for a tool that took the data off your device(s) and stored them securely in the cloud with a password you didn't know. Cross the border (whichever, I'm not fussy or prejudiced) and restore them to the device on arrival wherever you were going (say, the hotel you are staying in). Sort of a digitally locked plausibly deniability tool. After all, if the stuff is locked and you don't know the password. Imagine the potential!

As a tool to combat social trust based attacks, comfort and foreground trust are formally untested, but show great promise in informal settings, and we leave it as a thought exercise to see in which kinds of settings a little comfort enabled device might work.

Forgetting and Forgiving (or Not)

At the risk of repeating some of what has come before in the book, let's return to forgetfulness. We'll get to forgiveness again shortly. As we've learned so far, trust is a dynamic and contextual measurement based on things like observed behaviours, risks, importance and so on. It's reasonable to say that good behaviour can increase trust, whilst bad behaviour can decrease it. How that increase and decrease works is a different matter, but in general that's a fair assumption. As you already know, trust is often seen as 'fragile', too —hard to gain and easy to lose.

But in order to make decisions, things need to be remembered. Trust systems need to remember what has happened in order to be able to calculate trust(worthiness). However, if they remember everything, what is gained? (And what is lost?) More to the point, if events that happened a long time ago have the same weight as those that were recent in our models, we hit a problem: people change. And so can other actors in the system. What was "bad" could become "good", or at least not bad, and vice versa.

In order to manage this problem, a "forgetting factor" can be introduced which either weights temporally distinct behaviours or observations differently or simply forgets what happened after a time (there are also adaptive forgetting mechanisms). Jösang's Beta Reputation System has such a factor built in. In my thesis, I call it a "memory span". It's possible, then, to allow trust systems to forget, or to behave as if they forget, things from the past. Does this represent how people do things? Let's think about that for a moment. For one thing, it's probably fair to say that different events have different memorability. Some events or behaviours are so egregious or so incredible that forgetting them is out of the question, and the effect that they have on the trust value is constant (particularly the bad things, which is why, perhaps, the fragility of trust is so well entrenched).

Here's a thing to think about: What would a forgetting function that took all this into account look like?

Unforgiveness and Regret Management

In previous chapters I talked about forgiveness and how it might work for an artificial trust system. I did, at the time, mention that there are problems with some of the rationale there. In particular, forgiving some(one/thing) just because there don't appear to be any other viable things out there to work with works sometimes but is pathologically problematic at many others. Abusive relationships are a case in point. However, In the defence of forgiveness, think about this: forgiveness is a powerful expression of self control and acceptance that is recognised as being positive for both forgiver and forgivee. More importantly, in human relationships, forgiving does not necessarily mean remaining in uncomfortable situations, and we wouldn't recommend it for online situations either, where there was a potential for harm.

However, we do argue that trust systems being in a position to forgive transgressions and stay in interactions can bring benefits. As well, knowing the interaction can be challenging is information the trust model can use to strengthen itself (which is where the "Trust but Verify" thing can come in). Yes, of course there are situations where this stuff needs to be managed differently. In the place of this, we considered regret management. More specifically, we considered making someone regret something they did.

Regret is a powerful tool in the mechanisms of trust systems, *a priori* as a means of determining whether to act or not (anticipatory regret) and as a means of determining the difference between potential actions , but a posteriori as a means of determining the cost of what happened when things went wrong (which applies both to the trustee and the truster). In the latter instance, regret can be used as a tool to determine the amount by which trust is adapted following a transgression (for instance, a high regret on the part of the truster may reduce trust by a larger amount, as well, regret expressed by the trustee for their actions may mitigate this reduction somehow).

Taking this further, in this paper we proposed a system that centralizes the management of regret in a distributed system. From the paper:

"there exists a potential to use trust management systems as a means of enforcing the accountability of others in a transaction or relationship. In this sense, both parties in a transaction need to know that there is a recourse to as system that will lead to the punishment of a transgressor."

In a way, this is a resort to the idea of system Trust, which we've already talked about (in the TrustLess Systems chapter). Basically, if both truster and trustee know that the system will punish transgression, enforcing the regret of the transgressor, in other words, the likelihood of transgression may well be lowered. If the system is trusted to do the job properly.

I leave it as an exercise to think about how you would represent the transfer of trust to the system, how this affects the two foundational questions, and more, whether or not, if the regret management system works, trust is even being considered at all.

Endings

Like all of the chapters in this book, this one is an evolving piece of work. In fact, considering that research moves forward and ideas and questions keep happening, it is probably a little more changeable than the others. The beauty of a digital book like this (unless you printed it!) is that it can grow. That last word is important. We'll discuss it more in the ending of the book.

ON PERMANENCE: AN EPILOGUE OF SORTS

In a previous chapter, I used the word "grow" for the evolution of a chapter. The choice of word was important and carefully made. It also applies to the book as a whole.

The written word, once written, has some permanence. This should not be any different for a book that is not "printed". What is written, barring new editions and new discoveries, is written. It can be corrected but it shouldn't be seen as so malleable that it can change between today and tomorrow. That's one of the reasons why Wikipedia is often used as an example of a place not to get answers from when you are researching a paper for university or some other school. It's not that the information is necessarily always bad, it's just that from one moment to the next you can't tell if what you are looking at is good. It may have just been added, it may be an edit which has introduced an error, and so on.

A serious scientific textbook type of thing, which this work has aspirations of being, is expected to be a resource that is at least *correct* – for some value of correct – which means that when you look at it, you expect that what you read is not written in error. In fact, of course, errors can and do creep in and can be corrected or expanded on in revisions and addenda. No 'science' book is ever 'finished,' just like no science endeavour is really ever complete. It's the nature of the thing.

All of which means this: the 'book' you have reached the end of is not finished and (until I shuffle off this mortal coil and join the bleeding' choir invisible) I would like to think that I'll potter about in it, adding bits and not taking bits out. Yes, you read that right: not taking bits out because then you won't know what was written before and why it changed. I'll just ~~strike it out~~ and explain why, which seems to me to be a much more sensible way to manage and understand the evolution of information in a digital document and gives you, the reader, the context with which to learn more. Bear in mind that Hypothesis is also enabled for the book, at least on the Pressbooks site, so you can annotate to your hearts content, and share the same with others.

Which bits am I working on now? (A list that will likely change!). Further reading sections, much more in the Pioneers chapter and a chapter on security and trust, to link with the trustless systems and webs of trust chapters as well as more on attacks and defences. But right now, you have reached the end. Except for a list I am curating of further reading for the interested which, as the Pioneers chapter states, is based on a few decades of thought but is also a tad subjective.

Thanks for getting to the end. I would love to hear from you if you liked, looked, hated, didn't find what you needed, or just plain didn't care for what you did find. It will mean a lot to me as I potter about in it.

REFERENCES AND FURTHER READING

Standing on the shoulders of giants, as Newton would have it, means acknowledging them for the immense contribution they make to human knowledge. I refer to several different publications throughout the book and they are listed below, but there are more because also in this list are works I believe are vital to the field I'm in. As such, much as with the rest of the book, the content is fluid because there is *always* something new and wonderful.

Wherever possible I have put a hyperlink to the document.

Axelrod, R. (1984) *The Evolution of Cooperation*. Basic Books.

Bacharach, M. and Gambetta, D. (2001) Trust in Signs. In Cook, K (Ed.) *Trust in Society*. Russel Sage Foundation.

Barber, B. (1983) *The Logic and Limits of Trust.* Rutgers University Press.

Boon, S. and Holmes, J. (1985) The Dynamics of Interpersonal Trust: Resolving Uncertainity in the Face of Risk. In: Hinde, R. and Gorebel, J., Eds., Cooperation and Prosocial Behaviour, Cambridge University Press, Cambridge, 190-211.

Bok, S. (1978) Lying: Moral Choice in Public and Private Life. Penguin Random House.

CES (2019) Power and Politics – The Prisoner's Dilemma. Centre for European Studies.

Cofta, P. (2007) *Trust, Complexity and Control: Confidence in a Convergent World*. Wiley.

Dasgupta, P. (1990) Trust as a Commodity in Gambetta, Diego (ed.) *Trust: Making and Breaking Cooperative Relations*, chapter 4, pp. 49-72.

Deutsch, M. (1962) Cooperation and Trust: Some Theoretical Notes. In Jones, M.R., (ed) *Nebraska Symposium on Motivation*. Nebraska University Press.

Deutsch, M. (1973) *The Resolution of Conflict*. New Haven and London: Yale University Press

Dunn, J. Trust and Political Agency in Gambetta, Diego (ed.) *Trust: Making and Breaking Cooperative Relations*, chapter 5, pp. 78-93.

Gambetta, D. (1990) *Trust: Making and Breaking Cooperative Relation*s. Basil Blackwell.

Hardin, R. (2004) *Distrust*. Russel Sage Foundation Series on Trust.

Lamott, A (2019) Bird by Bird: Some Instructions on the Art of Writing and Life. (25th Anniversary Edition). Penguin Random House.

Luhmann, N. (1979) *Trust and Po*wer. (English translation). Wiley.

Marsh, S. (1994) *Formalizing Trust as a Computational Concept.* PhD Thesis, Department of Computing Science and Mathematics, University of Stirling.

Pitt, J. (2021) *Self-Organizing Multi Agent Systems: Algorithmic Foundations of Cyber-Anarcho-Socialism*. World Scientific.

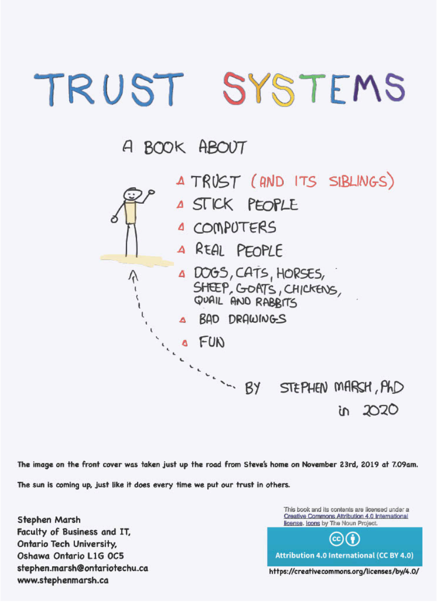
Reina, D. & Reina, M. (2015) *Trust and Betrayal in the Workplace.* Penguin Random House.

APPENDIX: INFOGRAPHICS

The following hand-drawn infographics were part of the original Trust Systems novel. It's exciting to see how the project has changed -and will continue to change- over time!

You can click on the images to view them at full resolution. Long-form image descriptions are available as well!

Introductions



TRUST SYSTEMS

A BOOK ABOUT

- TRUST (AND ITS SIBLINGS)
- STICK PEOPLE
- COMPUTERS
- REAL PEOPLE
- DOGS, CATS, HORSES, SHEEP, GOATS, CHICKENS, QUAIL AND RABBITS
- BAD DRAWINGS
- FUN

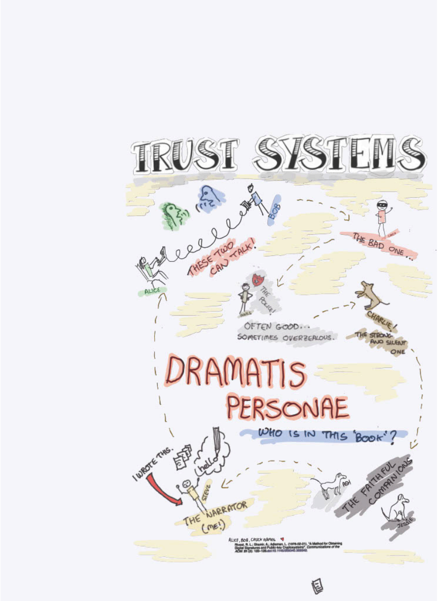
BY STEPHEN MARSH, PhD
in 2020

The image on the front cover was taken just up the road from Steve's home on November 23rd, 2019 at 7.09am.

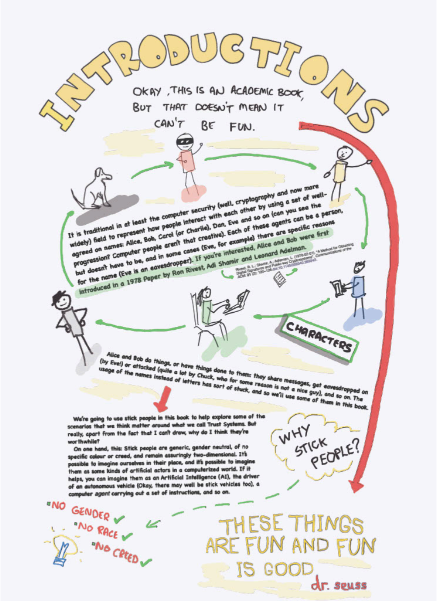
The sun is coming up, just like it does every time we put our trust in others.

Stephen Marsh
Faculty of Business and IT,
Ontario Tech University,
Oshawa Ontario L1G 0C5
stephen.marsh@ontariotechu.ca
www.stephenmarsh.ca

Table of Contents (image description)



Dramatis Personae (image description)


INTRODUCTIONS
OKAY, THIS IS AN ACADEMIC BOOK, BUT THAT DOESN'T MEAN IT CAN'T BE FUN.
Rivest, R. L.; Shamir, A.; Adleman, L. (1978-02-01). "A Method for Obtaining Digital Signatures and Public-key Cryptosystems". Communications of the ACM 21 (2): 120–126. doi:10.1145/359340.359342.
CHARACTERS
WHY STICK PEOPLE?
NO GENDER ✓
NO RACE ✓
NO CREED ✓
THESE THINGS ARE FUN AND FUN IS GOOD
dr. seuss

It is traditional in at least the computer security (well, cryptography and now more widely) field to represent how people interact with each other by using a set of well-agreed on names: Alice, Bob, Carol (or Charlie), Dan, Eve and so on (can you see the progression? Computer people aren't that creative). Each of these agents can be a person, but doesn't have to be, and in some cases (Eve, for example) there are specific reasons for the name (Eve is an eavesdropper). If you're interested, Alice and Bob were first introduced in a 1978 Paper by Ron Rivest, Adi Shamir and Leonard Adelman.

Alice and Bob do things, or have things done to them: they share messages, get eavesdropped on (by Eve!) or attacked (quite a lot by Chuck, who for some reason is not a nice guy), and so on. The usage of the names instead of letters has sort of stuck, and so we'll use some of them in this book.

We're going to use stick people in this book to help explore some of the scenarios that we think matter around what we call Trust Systems. But really, apart from the fact that I can't draw, why do I think they're worthwhile?

On one hand, this: Stick people are generic, gender neutral, of no specific colour or creed, and remain assuringly two-dimensional. It's possible to imagine ourselves in their place, and it's possible to imagine them as some kinds of artificial actors in a computerized world. If it helps, you can imagine them as an Artificial Intelligence (AI), the driver of an autonomous vehicle (Okay, there may well be stick vehicles too), a computer *agent* carrying out a set of instructions, and so on.

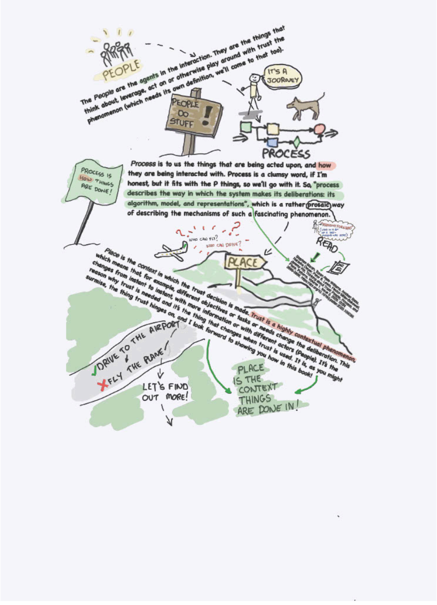
Introduction (image description)

NOT TOO WELL
NOT TOO BAD!
?
A BIT CROSS
YAY STICK PEOPLE!
Why are all Steve's stick people bald?
SO?
What is this book about anyway?
TRUST SYSTEMS
PEOPLE
PROCESS
PLACE

On the other hand, stick people are tremendously expressive. They can have smiles, frowns, be angry or sad or happy or sick. Just a quick glance can tell you an awful lot about how the stick person feels. It's even possible to imagine what they might do next. In other words, and very important, it's possible to ascribe beliefs, desires and intentions to our stick people. Those three words were not chosen randomly. We'll get to that in a moment.

If you have read the title, you will have noticed that this book is about Trust. A little further reading will have reached *Trust Systems*. A Trust System is a collection of three vital things: *People*, *Process*, and *Place* (our three P's!). These three make up the way in which we can examine, or even create, interactions and occurrences and relate them to trust. As far as we're concerned, anything to do with trust that matters has these three things making it up (hence, a system). We'll be exploring Trust Systems quite a lot in this book, so by the end of it, this should make sense (if not, we fire the stick people).

Introduction Continued (image description)

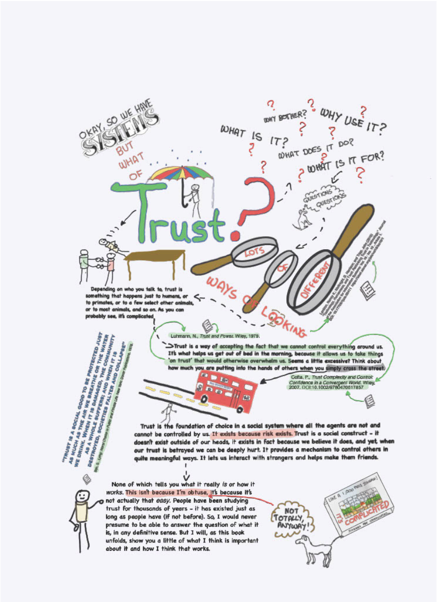


The *People* are the agents in the interaction. They are the things that think about, leverage, act on or otherwise play around with trust the phenomenon (which needs its own definition, we'll come to that too).

Process is to us the things that are being acted upon, and how they are being interacted with. Process is a clumsy word, if I'm honest, but it fits with the P things, so we'll go with it. So, "process describes the way in which the system makes its deliberations: its algorithm, model, and representations", which is a rather prosaic way of describing the mechanisms of such a fascinating phenomenon.

Place is the context in which the trust decision is made. Trust is a highly contextual phenomenon, which means that, for example, different objectives or tasks or needs change the deliberation. This changes from instant to instant, with more information or with different actors (People). It's the reason why trust is needed and it's the thing that changes when trust is used. It is, as you might surmise, the thing trust hinges on, and I look forward to showing you how in this book!

People, Process, and Place (image description)



Depending on who you talk to, trust is something that happens just to humans, or to primates, or to a few select other animals, or to most animals, and so on. As you can probably see, it's complicated

Lankton, Nancy K.; McKnight, D. Harrison; and Tripp, John (2015) "Technology, Humanness, and Trust: Rethinking Trust in Technology," Journal of the Association for Information Systems: Vol. 16 : Iss. 10 , Article 1. DOI: 10.17705/1jais.00411 https://aisel.aisnet.org/jais/vol16/iss10/1

Luhmann, N., *Trust and Power*. Wiley, 1979.

Trust is a way of accepting the fact that we cannot control everything around us. It's what helps us get out of bed in the morning, because it allows us to take things 'on trust' that would otherwise overwhelm us. Seems a little excessive? Think about how much you are putting into the hands of others when you simply cross the street.

Cofta, P., *Trust Complexity and Control: Confidence in a Convergent World*. Wiley, 2007. DOI:10.1002/9780470517857

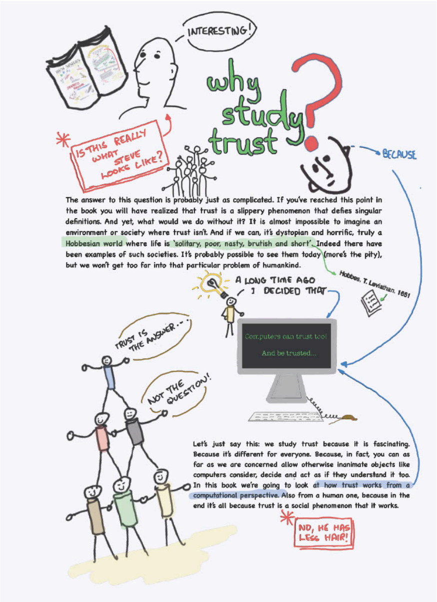
"TRUST IS A SOCIAL GOOD TO BE PROTECTED JUST AS MUCH AS THE AIR WE BREATHE OR THE WATER WE DRINK. WHEN IT IS DAMAGED, THE COMMUNITY AS A WHOLE SUFFERS; AND WHEN IT IS DESTROYED, SOCIETIES FALTER AND COLLAPSE"

Bok, S., Lying: Moral Choice in Public and Private Life. New York: Dover Publications. 1978.

Trust is the foundation of choice in a social system where all the agents are not and cannot be controlled by us. It exists because risk exists. Trust is a social construct – it doesn't exist outside of our heads, it exists in fact because we believe it does, and yet, when our trust is betrayed we can be deeply hurt. It provides a mechanism to control others in quite meaningful ways. It lets us interact with strangers and helps make them friends.

None of which tells you what it really *is* or how it *works*. This isn't because I'm obtuse, it's because it's not actually that *easy*. People have been studying trust for thousands of years – it has existed just as long as people have (if not before). So, I would never presume to be able to answer the question of what it is, in any definitive sense. But I will, as this book unfolds, show you a little of what I think is important about it and how I think that works.

What Of Trust? (image description)



The answer to this question is probably just as complicated. If you've reached this point in the book you will have realized that trust is a slippery phenomenon that defies singular definitions. And yet, what would we do without it? It is almost impossible to imagine an environment or society where trust isn't. And if we can, it's dystopian and horrific, truly a Hobbesian world where life is 'solitary, poor, nasty, brutish and short'. Indeed there have been examples of such societies. It's probably possible to see them today (more's the pity), but we won't get too far into that particular problem of humankind.

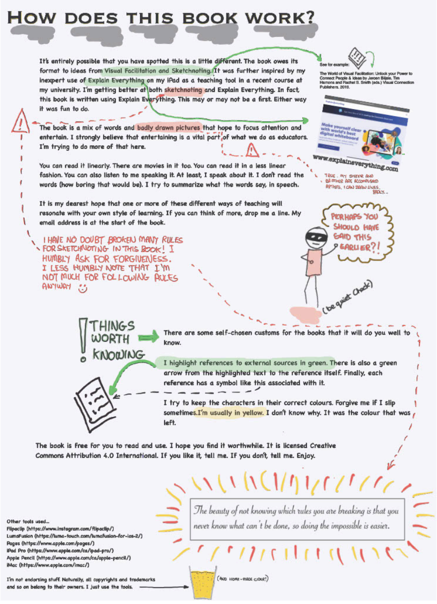
Hobbes, T. Leviathan. 1651

Computers can trust too!

And be trusted...

Let's just say this: we study trust because it is fascinating. Because it's different for everyone. Because, in fact, you can as far as we are concerned allow otherwise inanimate objects like computers consider, decide and act as if they understand it too. In this book we're going to look at how trust works from a computational perspective. Also from a human one, because in the end it's all because trust is a social phenomenon that it works.

Why Study Trust? (image description)



HOW DOES THIS BOOK WORK?

It's entirely possible that you have spotted this is a little different. The book owes its format to ideas from Visual Facilitation and Sketchnoting. It was further inspired by my inexpert use of Explain Everything on my iPad as a teaching tool in a recent course at my university. I'm getting better at both sketchnoting and Explain Everything. In fact, this book is written using Explain Everything. This may or may not be a first. Either way it was fun to do.

See for example:

The World of Visual Facilitation: Unlock your Power to Connect People & Ideas by Jeroen Bijlsie, Tim Hamons and Rachel S. Smith (eds.) Visual Connection Publishers. 2018.

www.explaineverything.com

The book is a mix of words and badly drawn pictures that hope to focus attention and entertain. I strongly believe that entertaining is a vital part of what we do as educators. I'm trying to do more of that here.

TRUE. MY SISTER AND BROTHER ARE ACCOMPLISHED ARTISTS, I CAN DRAW LINES, BADLY...

You can read it linearly. There are movies in it too. You can read it in a less linear fashion. You can also listen to me speaking it. At least, I speak about it. I don't read the words (how boring that would be). I try to summarize what the words say, in speech.

It is my dearest hope that one or more of these different ways of teaching will resonate with your own style of learning. If you can think of more, drop me a line. My email address is at the start of the book.

I HAVE NO DOUBT BROKEN MANY RULES FOR SKETCHNOTING IN THIS BOOK! I HUMBLY ASK FOR FORGIVENESS. I LESS HUMBLY NOTE THAT I'M NOT MUCH FOR FOLLOWING RULES ANYWAY ☺

PERHAPS YOU SHOULD HAVE SAID THIS EARLIER?!

(be quiet Chuck)

THINGS WORTH KNOWING

There are some self-chosen customs for the books that it will do you well to know.

I highlight references to external sources in green. There is also a green arrow from the highlighted text to the reference itself. Finally, each reference has a symbol like this associated with it.

I try to keep the characters in their correct colours. Forgive me if I slip sometimes. I'm usually in yellow. I don't know why. It was the colour that was left.

The book is free for you to read and use. I hope you find it worthwhile. It is licensed Creative Commons Attribution 4.0 International. If you like it, tell me. If you don't, tell me. Enjoy.

The beauty of not knowing which rules you are breaking is that you never know what can't be done, so doing the impossible is easier.

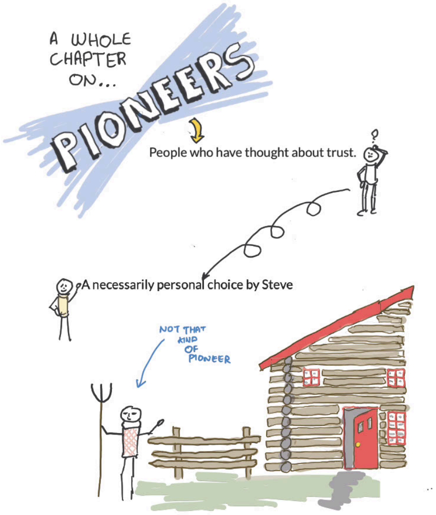
Other tools used...
Flipaclip (https://www.instagram.com/flipaclip/)
LumaFusion (https://luma-touch.com/lumafusion-for-ios-2/)
Pages (https://www.apple.com/pages/)
iPad Pro (https://www.apple.com/ca/ipad-pro/)
Apple Pencil (https://www.apple.com/ca/apple-pencil/)
iMac (https://www.apple.com/imac/)

I'm not endorsing stuff. Naturally, all copyrights and trademarks and so on belong to their owners. I just use the tools.

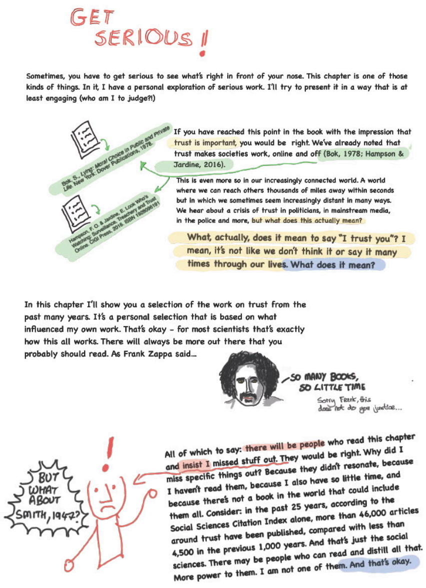
(AND HOME-MADE CIDER!)

How Does This Book Work? (image description)

Pioneers


A WHOLE CHAPTER ON...
PIONEERS
People who have thought about trust.
A necessarily personal choice by Steve
NOT THAT KIND OF PIONEER

A Whole Chapter on Pioneers (image description)



GET SERIOUS!

Sometimes, you have to get serious to see what's right in front of your nose. This chapter is one of those kinds of things. In it, I have a personal exploration of serious work. I'll try to present it in a way that is at least engaging (who am I to judge?!)

If you have reached this point in the book with the impression that trust is important, you would be right. We've already noted that trust makes societies work, online and off (Bok, 1978; Hampson & Jardine, 2016).

Bok, S., Lying: Moral Choice in Public and Private Life. New York: Dover Publications, 1978.

Hampson, F. O. & Jardine, E. Look Who's Watching: Surveillance, Treachery and Trust Online. CIGI Press, 2016. ISBN 1928096191

This is even more so in our increasingly connected world. A world where we can reach others thousands of miles away within seconds but in which we sometimes seem increasingly distant in many ways. We hear about a crisis of trust in politicians, in mainstream media, in the police and more, but what does this actually mean?

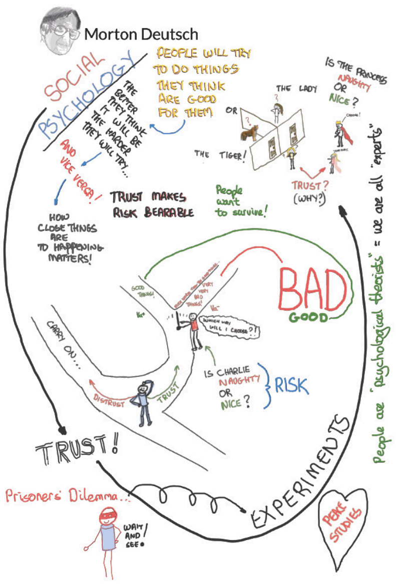
What, actually, does it mean to say "I trust you"? I mean, it's not like we don't think it or say it many times through our lives. What does it mean?

In this chapter I'll show you a selection of the work on trust from the past many years. It's a personal selection that is based on what influenced my own work. That's okay – for most scientists that's exactly how this all works. There will always be more out there that you probably should read. As Frank Zappa said...

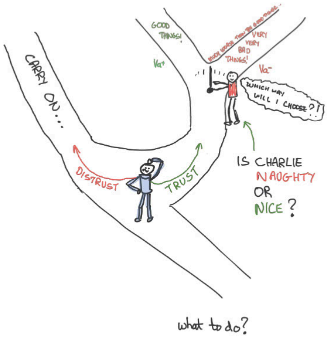
SO MANY BOOKS,
SO LITTLE TIME
Sorry Frank, this does not do you justice...
BUT WHAT ABOUT SMITH, 1947?

All of which to say: there will be people who read this chapter and insist I missed stuff out. They would be right. Why did I miss specific things out? Because they didn't resonate, because I haven't read them, because I also have so little time, and because there's not a book in the world that could include them all. Consider: in the past 25 years, according to the Social Sciences Citation Index alone, more than 46,000 articles around trust have been published, compared with less than 4,500 in the previous 1,000 years. And that's just the social sciences. There may be people who can read and distill all that. More power to them. I am not one of them. And that's okay.

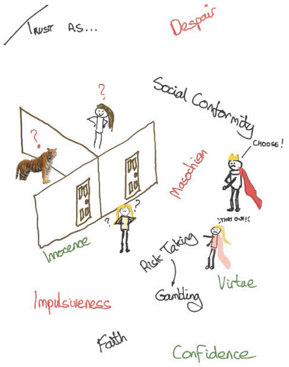
Get Serious (image description)


Morton Deutsch
SOCIAL PSYCHOLOGY
PEOPLE WILL TRY TO DO THINGS THEY THINK ARE GOOD FOR THEM
THE BETTER THEY THINK IT WILL BE THE HARDER THEY WILL TRY...
AND VICE VERSA!
HOW CLOSE THINGS ARE TO HAPPENING MATTERS!
TRUST MAKES RISK BEARABLE
People want to survive!
THE LADY
OR
THE TIGER!
IS THE PRINCESS NAUGHTY OR NICE?
CHOOSE!
TRUST? (WHY?)
GOOD THINGS!
VERY VERY BAD THINGS!
Va+
Va-
WHICH WAY WILL I CHOOSE?!
BAD
GOOD
CARRY ON...
DISTRUST
TRUST
IS CHARLIE NAUGHTY OR NICE?
RISK
TRUST!
Prisoners' Dilemma...
WAIT AND SEE.
EXPERIMENTS
PEACE STUDIES
People are "psychological theorists" = we are all "experts"

Morton Deutsch (image description)


CARRY ON...
GOOD THINGS!
Va+
MUCH WORSE THAN THE GOOD THINGS...
VERY VERY BAD THINGS!
Va-
WHICH WAY WILL I CHOOSE?!
DISTRUST
TRUST
IS CHARLIE NAUGHTY OR NICE?
what to do?

Choosing a Path (image description)



Choosing a Door (image description)

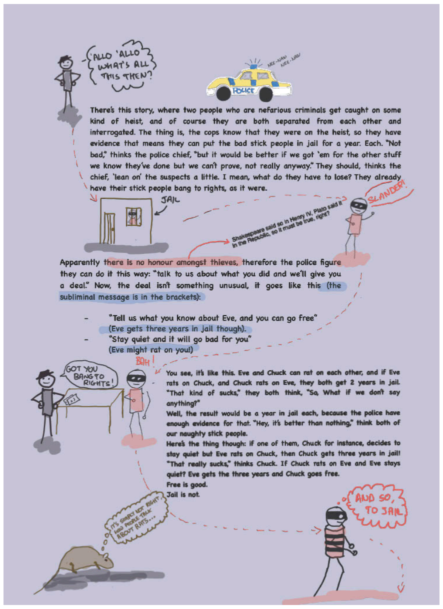
Prisoners

Whatever Happened to the Prisoners?

It's all about games...

In which Innes the police officer puts Chuck and their co-naughty-person in a dilemma.

Whatever Happened to the Prisoners? (image description)



There's this story, where two people who are nefarious criminals get caught on some kind of heist, and of course they are both separated from each other and interrogated. The thing is, the cops know that they were on the heist, so they have evidence that means they can put the bad stick people in jail for a year. Each. "Not bad," thinks the police chief, "but it would be better if we got 'em for the other stuff we know they've done but we can't prove, not really anyway." They should, thinks the chief, 'lean on' the suspects a little. I mean, what do they have to lose? They already have their stick people bang to rights, as it were.

Shakespeare said so in Henry IV, Plato said it in the Republic, so it must be true, right?

Apparently there is no honour amongst thieves, therefore the police figure they can do it this way: "talk to us about what you did and we'll give you a deal." Now, the deal isn't something unusual, it goes like this (the subliminal message is in the brackets):

- "Tell us what you know about Eve, and you can go free" (Eve gets three years in jail though).
- "Stay quiet and it will go bad for you" (Eve might rat on you!)

You see, it's like this. Eve and Chuck can rat on each other, and if Eve rats on Chuck, and Chuck rats on Eve, they both get 2 years in jail. "That kind of sucks," they both think, "So, What if we don't say anything?"

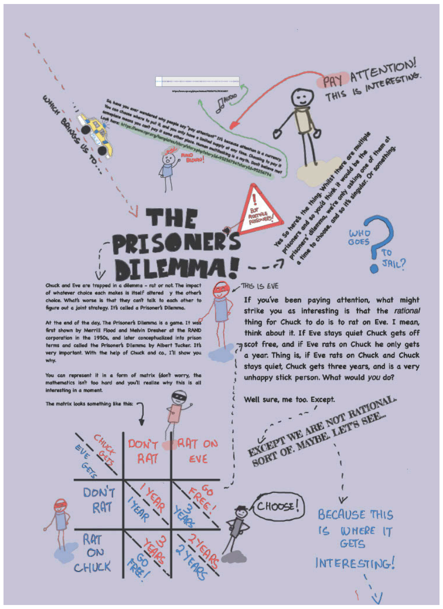
Well, the result would be a year in jail each, because the police have enough evidence for that. "Hey, it's better than nothing," think both of our naughty stick people.

Here's the thing though: if one of them, Chuck for instance, decides to stay quiet but Eve rats on Chuck, then Chuck gets three years in jail! "That really sucks," thinks Chuck. If Chuck rats on Eve and Eve stays quiet? Eve gets the three years and Chuck goes free.

Free is good.

Jail is not.

The Prisoner's Dilemma (image description)



So, have you ever wondered why people say "pay attention?" It's because attention is a currency. You can choose where to put it, and you only have a limited supply at any time. Choosing to pay it someplace means you can't pay it some other place. Human multitasking is a myth. Don't believe me? Look here: https://www.npr.org/templates/story/story.php?storyId=95256794?storyId=95256794

Yes So here's the thing. Whilst there are multiple prisoners and so you'd think it would be the prisoners' dilemma, we're only asking one of them at a time to choose, and so it's singular. Or something.

Chuck and Eve are trapped in a dilemma – rat or not. The impact of whatever choice each makes is itself altered y the other's choice. What's worse is that they can't talk to each other to figure out a joint strategy. It's called a Prisoner's Dilemma.

At the end of the day, The Prisoner's Dilemma is a game. It was first shown by Merrill Flood and Melvin Dresher at the RAND corporation in the 1950s, and later conceptualized into prison terms and called the Prisoner's Dilemma by Albert Tucker. It's very important. With the help of Chuck and co., I'll show you why.

You can represent it in a form of matrix (don't worry, the mathematics isn't too hard and you'll realize why this is all interesting in a moment.

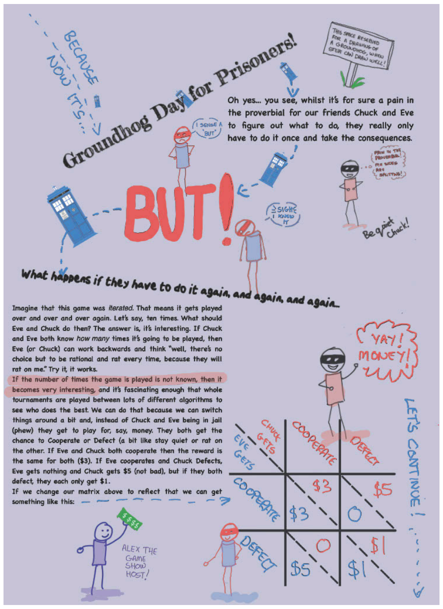
The matrix looks something like this:

CHUCK GETS EVE GETS	DON'T RAT	RAT ON EVE
DON'T RAT	1 YEAR 1 YEAR	GO FREE! 3 YEARS
RAT ON CHUCK	3 YEARS GO FREE!	2 YEARS 2 YEARS

If you've been paying attention, what might strike you as interesting is that the *rational* thing for Chuck to do is to rat on Eve. I mean, think about it. If Eve stays quiet Chuck gets off scot free, and if Eve rats on Chuck he only gets a year. Thing is, if Eve rats on Chuck and Chuck stays quiet, Chuck gets three years, and is a very unhappy stick person. What would *you* do?

Well sure, me too. Except.

The Prisoner's Dilemma Continued (image description)


BECAUSE NOW IT'S...
Groundhog Day for Prisoners!
THIS SPACE RESERVED FOR A DRAWING OF A GROUNDHOG, WHEN STEVE CAN DRAW WELL!
I SENSE A 'BUT'
PAIN IN THE PROVERBIAL! ME SIDES ARE SPLITTING!
BUT!
SIGH I KNEW IT
Be quiet Chuck!
YAY! MONEY!
$$$
ALEX THE GAME SHOW HOST!
LET'S CONTINUE!

Oh yes... you see, whilst it's for sure a pain in the proverbial for our friends Chuck and Eve to figure out what to do, they really only have to do it once and take the consequences.

What happens if they have to do it again, and again, and again...

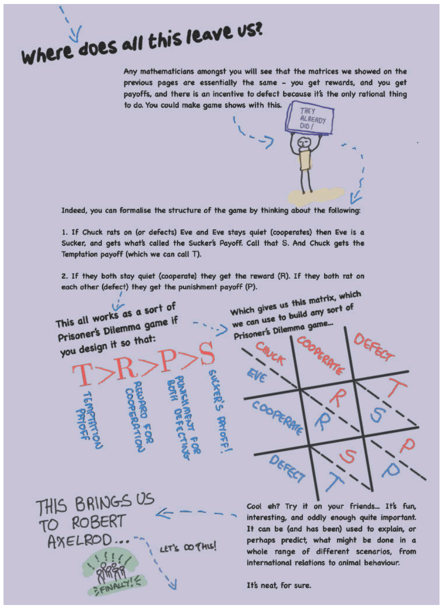
Imagine that this game was *iterated*. That means it gets played over and over and over again. Let's say, ten times. What should Eve and Chuck do then? The answer is, it's interesting. If Chuck and Eve both know *how many* times it's going to be played, then Eve (or Chuck) can work backwards and think "well, there's no choice but to be rational and rat every time, because they will rat on me." Try it, it works.

If the number of times the game is played is not known, then it becomes very interesting, and it's fascinating enough that whole tournaments are played between lots of different algorithms to see who does the best. We can do that because we can switch things around a bit and, instead of Chuck and Eve being in jail (phew) they get to play for, say, money. They both get the chance to Cooperate or Defect (a bit like stay quiet or rat on the other. If Eve and Chuck both cooperate then the reward is the same for both ($3). If Eve cooperates and Chuck Defects, Eve gets nothing and Chuck gets $5 (not bad), but if they both defect, they each only get $1.

If we change our matrix above to reflect that we can get something like this:

CHUCK GETS EVE GETS	COOPERATE	DEFECT
COOPERATE	$3 $3	$5 0
DEFECT	0 $5	$1 $1

Groundhog Day for Prisoners (image description)



Where does all this leave us?

Any mathematicians amongst you will see that the matrices we showed on the previous pages are essentially the same – you get rewards, and you get payoffs, and there is an incentive to defect because it's the only rational thing to do. You could make game shows with this.

Indeed, you can formalise the structure of the game by thinking about the following:

1. If Chuck rats on (or defects) Eve and Eve stays quiet (cooperates) then Eve is a Sucker, and gets what's called the Sucker's Payoff. Call that S. And Chuck gets the Temptation payoff (which we can call T).

2. If they both stay quiet (cooperate) they get the reward (R). If they both rat on each other (defect) they get the punishment payoff (P).

This all works as a sort of Prisoner's Dilemma game if you design it so that:

T>R>P>S

Which gives us this matrix, which we can use to build any sort of Prisoner's Dilemma game...

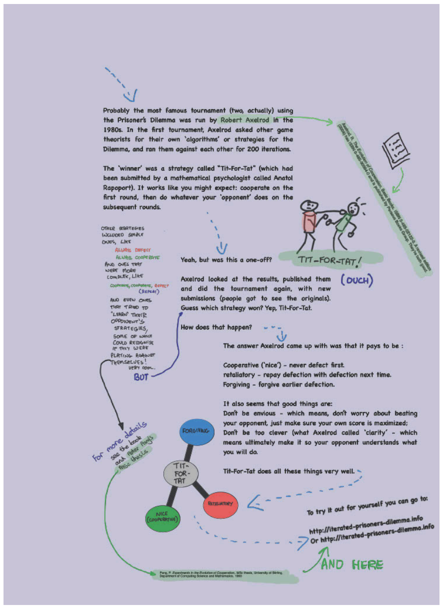
EVE \ CHUCK	COOPERATE	DEFECT
COOPERATE	R R	T S
DEFECT	S T	P P

THIS BRINGS US TO ROBERT AXELROD...

Cool eh? Try it on your friends... It's fun, interesting, and oddly enough quite important. It can be (and has been) used to explain, or perhaps predict, what might be done in a whole range of different scenarios, from international relations to animal behaviour.

It's neat, for sure.

Where Does All This Leave Us? (image description)



Probably the most famous tournament (two, actually) using the Prisoner's Dilemma was run by Robert Axelrod in the 1980s. In the first tournament, Axelrod asked other game theorists for their own 'algorithms' or strategies for the Dilemma, and ran them against each other for 200 iterations.

Axelrod, R. The Evolution of Cooperation. Basic Books. ISBN 0-465-02122-0. A revised edition (2006) has ISBN 0-465-00564-0 and is published by Perseus Books Group. They're both good.

The 'winner' was a strategy called "Tit-For-Tat" (which had been submitted by a mathematical psychologist called Anatol Rapoport). It works like you might expect: cooperate on the first round, then do whatever your 'opponent' does on the subsequent rounds.

OTHER STRATEGIES INCLUDED SIMPLE ONES, LIKE
ALWAYS DEFECT
ALWAYS COOPERATE
AND ONES THAT WERE MORE COMPLEX, LIKE
COOPERATE, COOPERATE, DEFECT (REPEAT)
AND EVEN ONES THAT TRIED TO 'LEARN' THEIR OPPONENT'S STRATEGIES, SOME OF WHICH COULD RECOGNISE IF THEY WERE PLAYING AGAINST THEMSELVES! VERY COOL.
BUT

Yeah, but was this a one-off?

Axelrod looked at the results, published them and did the tournament again, with new submissions (people got to see the originals). Guess which strategy won? Yep, Tit-For-Tat.

How does that happen?

The answer Axelrod came up with was that it pays to be :

Cooperative ('nice') – never defect first.
retaliatory – repay defection with defection next time.
Forgiving – forgive earlier defection.

It also seems that good things are:
Don't be envious – which means, don't worry about beating your opponent, just make sure your own score is maximized;
Don't be too clever (what Axelrod called 'clarity' – which means ultimately make it so your opponent understands what you will do.

Tit-For-Tat does all these things very well.

For more details see the book and Peter Pang's MSc thesis

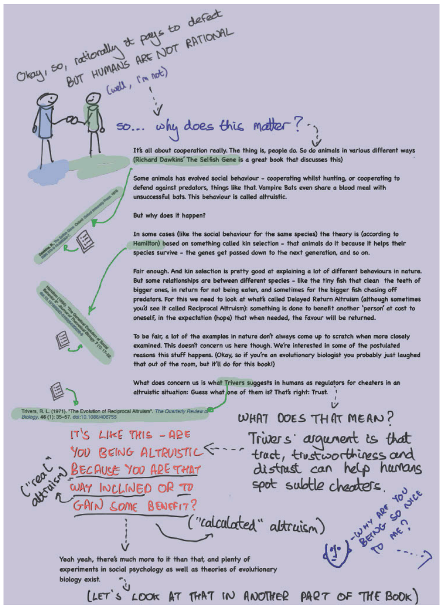
To try it out for yourself you can go to:

http://iterated-prisoners-dilemma.info
Or http://iterated-prisoners-dilemma.info

AND HERE

Pang, P. Experiments in the Evolution of Cooperation. MSc thesis, University of Stirling, Department of Computing Science and Mathematics. 1990

Where Does All This Leave Us Continued (image description)



Okay, so, rationally it pays to defect
BUT HUMANS ARE NOT RATIONAL
(well, I'm not)

so... why does this matter?

It's all about cooperation really. The thing is, people do. So do animals in various different ways (Richard Dawkins' The Selfish Gene is a great book that discusses this)

Some animals has evolved social behaviour – cooperating whilst hunting, or cooperating to defend against predators, things like that. Vampire Bats even share a blood meal with unsuccessful bats. This behaviour is called altruistic.

But why does it happen?

In some cases (like the social behaviour for the same species) the theory is (according to Hamilton) based on something called kin selection – that animals do it because it helps their species survive – the genes get passed down to the next generation, and so on.

Fair enough. And kin selection is pretty good at explaining a lot of different behaviours in nature. But some relationships are between different species – like the tiny fish that clean the teeth of bigger ones, in return for not being eaten, and sometimes for the bigger fish chasing off predators. For this we need to look at what's called Delayed Return Altruism (although sometimes you'd see it called Reciprocal Altruism): something is done to benefit another 'person' at cost to oneself, in the expectation (hope) that when needed, the favour will be returned.

To be fair, a lot of the examples in nature don't always come up to scratch when more closely examined. This doesn't concern us here though. We're interested in some of the postulated reasons this stuff happens. (Okay, so if you're an evolutionary biologist you probably just laughed that out of the room, but it'll do for this book!)

What does concern us is what Trivers suggests in humans as regulators for cheaters in an altruistic situation: Guess what one of them is? That's right: Trust.

Dawkins, R. The Selfish Gene. Oxford: Oxford University Press. 1976.

Hamilton (1964). "The Genetical Evolution of Social Behavior II". Journal of Theoretical Biology. 7 (1): 17–52.

Trivers, R. L. (1971). "The Evolution of Reciprocal Altruism". *The Quarterly Review of Biology*. 46 (1): 35–57. doi:10.1086/406755

WHAT DOES THAT MEAN?

Trivers' argument is that tract, trustworthiness and distrust can help humans spot subtle cheaters.

IT'S LIKE THIS – ARE YOU BEING ALTRUISTIC BECAUSE YOU ARE THAT WAY INCLINED OR TO GAIN SOME BENEFIT?

("real" altruism)

("calculated" altruism)

–WHY ARE YOU BEING SO NICE TO ME?

Yeah yeah, there's much more to it than that, and plenty of experiments in social psychology as well as theories of evolutionary biology exist.

(LET'S LOOK AT THAT IN ANOTHER PART OF THE BOOK)

Why Does This Matter? (image description)

What does all this mean?
Remember – tit-for-tat is nice
This means it will not defect first
But it is also provokable

The Prisoner's Dilemma Games (image description)

TO OUR MOTHER, FOR THE GIFT OF BREATH