1 Proofs by Induction

Induction is a method for proving statements that have the form: $\forall n : P(n)$, where n ranges over the positive integers. It consists of two steps.

- First, you prove that $P(1)$ is true. This is called the *basis* of the proof.
- Then you show: for all $n \geq 0$, if $P(0), P(1), P(2), \ldots, P(n)$ are all true, then $P(n+1)$ must be true. Or symbolically, you show

  $$[P(0) \wedge P(1) \wedge \cdots \wedge P(n)] \rightarrow P(n+1)$$

  for all n. This is called the *induction step*.

  A crucial ingredient in the induction step is this assumption that $P(0), P(1), P(2), \ldots, P(n)$ are all true. We call this the *induction hypothesis*.

Why is the above argument enough to establish $P(n)$ for all n? You're directly showing $P(0)$, and from this the induction step implies $P(1)$. From that the induction step then implies $P(2)$, then $P(3)$, and so on. Each $P(n)$ follows from the previous, like a long of dominoes toppling over.

Induction also works if you want to prove a statement for all n starting at some point $n_0 > 0$. All you do is adapt the proof strategy so that the basis is n_0:

- First, you prove that $P(n_0)$ is true. (The basis.)
- Then you show: for all $n \geq n_0$, if $P(n_0), P(n_0+1), \ldots, P(n)$ are all true, then $P(n+1)$ must be true. (The induction step.)

The same intuitive picture applies, except now we're only pushing over the dominoes starting at n_0 rather than 0.

Also, it's fine (and sometimes useful) to prove a few base cases. For example, if you're trying to prove $\forall n : P(n)$, where n ranges over the positive integers, it's fine to prove $P(1)$ and $P(2)$ separately before starting the induction step.

2 Fibonacci Numbers

There is a close connection between induction and recursive definitions: induction is perhaps the most natural way to reason about recursive processes.

Let's see an example of this, using the *Fibonacci numbers*. These were introduced as a simple model of population growth by Leonardo of Pisa in the 12th century. Here's how he described it. Suppose you start with a newborn pair of rabbits, one of each gender. Starting when they're two months old, they produce one pair of rabbits per month as offspring. Each subsequent pair of offspring behaves the same way: starting when they're two months old, they too produce one pair of rabbits per month.

How many pairs of rabbits in total do you have after n months? Let f_n denote this quantity.

- To begin with, $f_1 = 1$ and $f_2 = 1$.
- In month 3, a new pair of rabbits is born, so $f_3 = 2$.
- In month 4, the original pair has more offspring, whereas the just-born pair doesn't produce any yet. So $f_4 = 3$.
- In month 5, the two pairs of rabbits who were already around in month 3 have offspring, so $f_5 = 5$.
- Clearly, we could keep going for f_6, f_7, and beyond if we wanted to.

So what's the pattern? Rather trying to write a formula for f_n directly in terms of n, which is not so easy, we can instead write it directly in terms of previous entries in the sequence. All the rabbit pairs in month n were either there in month $n-1$ (that's f_{n-1} of them), or they're new offspring from rabbit pairs who were there in month $n-2$ (that's f_{n-2} more). So we have a simple formula that defines f_n:

$$f_n = f_{n-1} + f_{n-2}.$$

We call this a *recurrence* since it defines one entry in the sequence in terms of earlier entries. And it gives the Fibonacci numbers a very simple interpretation: they're the sequence of numbers that starts $1, 1$ and in which every subsequent term in the sum of the previous two.

Exponential growth. Since the Fibonacci numbers are designed to be a simple model of population growth, it is natural to ask how quickly they grow with n. We'll say they grow *exponentially* if we can find some real number $r > 1$ so that $f_n \geq r^n$ for all n.

The following claim shows that they indeed grow exponentially. We'll first present this with the value of r chosen as if "by magic," then come back to suggest how one might have come up with it.

- Claim: Let $r = \frac{1+\sqrt{5}}{2} \approx 1.62$, so that r satisfies $r^2 = r + 1$. Then $f_n \geq r^{n-2}$.

Given the fact that each Fibonacci number is defined in terms of smaller ones, it's a situation ideally designed for induction.

Proof of Claim: First, the statement is saying $\forall n \geq 1 : P(n)$, where $P(n)$ denotes "$f_n > r^{n-2}$." As with all uses of induction, our proof will have two parts.

- First, the basis. $P(1)$ is true because $f_1 = 1$ while $r^{1-2} = r^{-1} \le 1$. While we're at it, it turns out be convenient to handle Actually, we notice that f_2 is defined directly to be equal to 1, so it's natural to handle $P(2)$ directly here. $P(2)$ is true because $f_2 = 1$ and $r^{2-2} = r^0 = 1$.

- Next, the induction step, for a fixed $n > 1$. (Actually, since we've already done $n = 2$, we can assume $n > 2$ from here on.) The induction hypothesis is that $P(1), P(2), \ldots, P(n)$ are all true. We assume this and try to show $P(n+1)$. That is, we want to show $f_{n+1} \ge r^{n-1}$.

  So consider f_{n+1} and write
  $$f_{n+1} = f_n + f_{n-1}. \tag{1}$$
  We now use the induction hypothesis, and particularly $f_n \ge r^{n-2}$ and $f_{n-1} \ge r^{n-3}$. Substituting these inequalities into line (1), we get
  $$f_{n+1} \ge r^{n-2} + r^{n-3} \tag{2}$$
  Factoring out a common term of r^{n-3} from line (2), we get
  $$f_{n+1} \ge r^{n-3}(r+1). \tag{3}$$
  Now we use the the fact that we've chosen r so that
  $$r^2 = r + 1. \tag{4}$$
  Plugging this into line (3), we get
  $$f_{n+1} \ge r^{n-3}(r+1) = r^{n-3} \cdot r^2 = r^{n-1}, \tag{5}$$
  which is exactly the statement of $P(n+1)$ that we wanted to prove. This concludes the proof.

Looking at the proof suggests how we might have guessed which value of r to use. Note that if we were trying to write the proof, not yet knowing how to define r, we could get all the way to Inequality (3) without having specified it. From here, we can look ahead to where we'd like to be — Inequality (5) — and notice that we'd be all set if only we could replace $r + 1$ with r^2. This suggests that we should choose r to be a solution to $r^2 = r + 1$, which is what we did.

3 The Structure of an Induction Proof

Beyond the specific ideas needed to go into analyzing the Fibonacci numbers, the proof above is a good example of the structure of an induction proof.

In writing out an induction proof, it helps to be very clear on where all the parts shows up. So what you write out a complete induction proof as part of on homework, you should make sure to include the following parts.

- A clear statement of what you're trying to prove in the form $\forall n : P(n)$. You should say explicitly what $P(n)$ is.
- A proof of the basis, specifying what $P(1)$ is and how you're proving it. (Also note any additional basis statements you choose to prove directly, like $P(2)$, $P(3)$, and so forth.)
- A statement of the induction hypothesis.
- A proof of the induction step, starting with the induction hypothesis and showing all the steps you use. This part of the proof should include an explicit statement of where you use the induction hypothesis. (If you find that you're not using the induction hypothesis at all, it's generally a warning that there something is going wrong with the proof itself.)

4 An Exact Formula for the Fibonacci Numbers

Here's something that's a little more complicated, but it shows how reasoning induction can lead to some non-obvious discoveries. Specifically, we will use it to come up with an exact formula for the Fibonacci numbers, writing f_n directly in terms of n.

An incorrect proof. Let's start by asking what's wrong with the following attempted proof that, in fact, $f_n = r^{n-2}$. (Not just that $f_n \geq r^{n-2}$.)

- Incorrect proof (sketch): We proceed by induction as before, but we strengthen $P(n)$ to say "$f_n = r^{n-2}$." The induction hypothesis is that $P(1), P(2), \ldots, P(n)$ are all true. We assume this and try to show $P(n+1)$. That is, we want to show $f_{n+1} = r^{n-1}$.

  Proceeding as before, but replacing inequalities with equalities, we have

$$\begin{aligned} f_{n+1} &= f_n + f_{n-1} \\ &= r^{n-2} + r^{n-3} \\ &= r^{n-3}(r+1) \\ &= r^{n-3}r^2 \\ &= r^{n-1}, \end{aligned}$$

  where we used the induction hypothesis to go from the first line to the second, and we used the property of r that $r^2 = r + 1$ to go from the third line to the fourth. The last line is exactly the statement of $P(n+1)$.

The funny thing is: there's nothing wrong with the parts of this "proof" that we wrote out explicitly. The problem came earlier: we don't have a correct base case. That is, $f_1 = 1 \neq r^{1-2}$. In fact, the induction would have been fine if only the base case had been correct; but instead, we have a proof that starts out with an incorrect statement (the wrong base case), and so it fails completely.

An exact formula. Still, so much of this wrong argument seems to work that we're tempted to try salvaging it. If we could just modify the formula a bit so it worked on the base cases of 1 and 2, everything else would be fine.

So suppose instead of $f_n = r^{n-2}$ (which is false), we tried proving $f_n = ar^n$ for some value of a yet to be determined. (Note that r^{n-2} is just ar^n for the particular choice $a = r^{-2}$.) Could there be a value of a that works?

Unfortunately, no. We'd need to have $1 = f_1 = ar$ and $1 = f_2 = ar^2$. But by the definining property of r, we have $1 = f_2 = ar^2 = a(r+1) = ar + a$. Thus we have

$$\begin{aligned} 1 &= ar \\ 1 &= ar + a \end{aligned}$$

which cannot be satisfed by any value of a.

But there's one more trick we can still deploy. $x^2 = x + 1$ is a quadratic equation, and it has *two* solutions: $r = \frac{1+\sqrt{5}}{2}$ and $s = \frac{1-\sqrt{5}}{2}$. Any statement $P(n)$ of the form $f_n = ar^n + bs^n$ (for any choices of a and b) would work just fine in the induction step: once we assume $P(1), \ldots, P(n)$ are all true, we can write

$$\begin{aligned} f_{n+1} &= f_n + f_{n-1} \\ &= ar^n + bs^n + ar^{n-1} + bs^{n-1} \\ &= ar^n + ar^{n-1} + bs^n + bs^{n-1} \\ &= ar^{n-1}(r+1) + bs^{n-1}(s+1) \\ &= ar^{n-1} \cdot r^2 + bs^{n-1} \cdot s^2 \\ &= ar^{n+1} + bs^{n+1}, \end{aligned}$$

where we used the induction hypothesis to go from the first line to the second, and we used the property of r and s that $r^2 = r + 1$ and $s^2 = s + 1$ to go from the fourth line to the fifth. The last line is exactly the statement of $P(n+1)$.

So now we just need to see if the as-yet undetermined constants a and b can be chosen so that the base cases $P(1)$ and $P(2)$ work. To make these work, we need

$$\begin{aligned} 1 &= f_1 = ar + bs \\ 1 &= f_2 = ar^2 + bs^2 = a(r+1) + b(s+1) \end{aligned}$$

Solving these two equations in the two unknowns a and b, we find that there is a solution: $a = 1/\sqrt{5}$ and $b = -1/\sqrt{5}$. Thus, we arrive at a formula for the Fibonacci numbers:

$$f_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n.$$

Not a formula that you would necessarily have guessed in advance ...

5 Induction and Recursive Algorithms

We saw last time, via the example of the Fibonacci numbers, how induction is very natural for understanding recursive definitions; here we extend this theme by using induction to analyze recursive algorithms.

In the coming lectures, we'll be looking at algorithms in cryptography, and we'll see that one basic routine needed there is to compute $a^x \bmod n$ for very large values of x. The naive approach to this would involve $x - 1$ multiplications, but since we'll be dealing with enormously large values of x (for reasons we'll see later), this could be prohibitive. So it's natural to ask: can we compute $a^x \bmod n$ using significantly fewer multiplications?

It turns out that we can; the idea is based on the following observation. If x were a power of 2 (say $x = 2^j$), then we could simply square $a \bmod n$ for j times in succession, thereby computing $a^x \bmod n$ using only $\log x$ multiplications. (For purposes of discussing this problem, all logarithms will be base 2.)

And this is not specific just to powers of two. For example, to compute $a^{1025} \bmod n$, we could first compute $a^{1024} \bmod n$ using ten multiplications, and then perform one final multiplication by $a \bmod n$. Or to compute $a^{768} \bmod n$, we could first compute $a^3 \bmod n$, which is easy, and then square it mod n eight times.

In fact, we now show that these ideas can be extended to work for *all* exponents x. The big savings in multiplication is in being able to square rather than multiply by a, and one can do this (at least once) whenever the exponent x is even. So the recursive algorithm we consider takes advantage of this by squaring the intermediate result whenever possible.

```
Function exp(a, x, n)
  If x = 1 then return a mod n.
  Else if x is odd then
    let b = exp(a, x − 1, n).
    return a ∗ b mod n.
  Else (x is even)
    let b = exp(a, x/2, n).
    return b^2 mod n.
```

Note the recursive nature of the algorithm; until x gets down to 1, it calls itself on a smaller value of x. As a result, induction naturally suggests itself: claims about the behavior of the algorithm with exponent x can be reduced to claims about the behavior with smaller exponents, which can be handled using the right induction hypothesis.

In fact, one needs induction to formally prove that this algorithm is computing the right thing, but we omit this proof. (It's provided in the book.) Instead, we give a bound on the number of multiplications it uses.

For the rest of this discussion, we fix values for a and n. We now claim that for any exponent x, the function $exp(a, x, n)$ uses at most $2 \log x$ multiplications. Note that for large values of x, this is enormously smaller than the $x - 1$ multiplications needed by the naive method. (For example, if $x = 10^{12}$, it's the difference between a trillion and 80.)

Here is a proof.

- What we are trying to prove has the form $\forall x \geq 1 : P(x)$, where $P(x)$ denotes the statement, "$exp(a, x, n)$ uses at most $2 \log x$ multiplications." So it is natural to try a proof by induction.

- The basis, $P(1)$, says that we use $2 \log 1 = 0$ multiplications for $x = 1$. This is true, since the algorithm simply returns a mod n in this case.

- Next, we perform the induction step. The induction hypothesis is that $P(1), P(2), \ldots, P(x)$ are all true, for some $x \geq 1$. We assume this and try to show $P(x + 1)$.

- Notice that the algorithm has two cases, depending on whether $x + 1$ is even or odd, and so it is useful to have two cases in the induction step as well.

  - If x is even, then we invoke $exp(a, \frac{x+1}{2}, n)$ to get the value b. $\frac{x+1}{2}$ is smaller than x, so by the induction hypothesis (specifically $P(\frac{x+1}{2})$), we use at most $2 \log((x + 1)/2) = 2(\log(x + 1) - 1) = 2 \log(x + 1) - 2$ multiplications to get the value of $exp(a, \frac{x+1}{2}, n)$. We then perform one more multiplication to get the square, for a total of at most $2 \log(x+1) - 1 \leq 2 \log(x+1)$. This proves $P(x+1)$ in this case.

  - If x is odd, then we invoke $exp(a, x, n)$ to get the value b. If we just think in terms of this one recursive call, we will not get a strong enough bound: the induction hypothesis says that we use at most $2 \log x$ multiplications to get this, plus one more to get $a * b$ mod n, for a total of at most $2 \log x + 1$. But $2 \log x + 1$ is in general larger than $2 \log(x+1)$, the bound we're hoping for. This doesn't say that what we're trying to prove is false, simply that we need a better argument.

    The problem is that when x is odd, the initial recursive call only reduces x by 1. But the good news is that $x-1$ is even (since x was odd), so the *next* recursive call divides it by 2, giving us the kind of big progress that we need. Let b' be the value of $exp(a, x/2, n)$ after this second recursive call; we use two more multiplications after this to get $b = (b')^2$ mod n and then the final value $a * b$ mod n. And by the induction hypothesis (specifically $P(x/2)$), the number of multiplications used to get $exp(a, x/2, n)$ is at most $2 \log(x/2) = 2(logx - 1) = 2 \log x - 2$. Adding in the two extra multiplications, we get $2 \log x \leq 2 \log(x + 1)$, which is the bound we need for establishing $P(x + 1)$.

    Since this handles both the case in which x is even and in which x is odd, this concludes the proof.

6 Induction vs. Common Sense

Induction can be hard on one's intuition, since it leads to conclusions that are the consequences of enormous numbers of tiny steps (from 1 to 2 to 3 to ... n to $n + 1$ to ...). Each tiny step is right, so the overall conclusion must follow, but sometimes it's hard to see the intuition behind this global picture.

Once we move into situations that are not fully specified mathematically, this kind of reasoning can lead to situations that clash much more severely with intuition. In fact, the use of induction as a part of everyday, "common-sense" reasoning — as opposed to strictly as a proof technique — is a topic that has occupied philosophers and mathematicians for many years.

We now discuss three examples where inductive reasoning in informal situations (i.e. situations that are not entirely mathematically precise) leads to trouble. Each of these illustrates a genre of induction problem that has been the subject of considerable philosophical debate. And all three share the property that there is no "simple" explanation of how to get around the problem — they each point at something fundamental in the way inductive reasoning works.

6.1 Lifting a Cow

Farmers in Vermont, according to folklore, like to discuss the following conundrum. On the day a calf is born, you can pick it up and carry it quite easily. Now, calves don't grow that much in one day, so if you can lift it the day it's born, you can still lift it the next morning. And, continuing this reasoning, you can go out the next morning and still lift it, and the next morning after that, and so forth.

But after two years, the calf will have grown into a full-sized, thousand-pound cow — something you clearly can't lift. So where did things break down? When did you stop being able to lift it?

There's clearly an inductive argument going on here: there was a basis, that you can lift it on day 1, and an induction step — if you could lift it up to day n, then it doesn't grow much at all in one day, so you can still lift it on day $n+1$. Yet the conclusion, that you can lift it on every day of its life, clearly fails, since you can't lift the thousand-pound cow it becomes.

As promised above, there's no simple "mistake" in this argument; it's genuinely puzzling. But it's clearly related to things like (a) the fact that growth is a continuous process, and we're sampling it at discrete (day-by-day) intervals, and (b) the fact that "lifting" is not entirely well-defined — maybe you just lift it less and less each day, and gradually what you're doing doesn't really count as "lifting."

6.2 The Surprise Quiz

You're take an intensive one-month summer course that meets every day during the month of June. Just before the course starts, the professor makes an announcement. "On one of the days during the course, there will be a surprise quiz. The date will be selected so that, on the day when the quiz is given, you won't be expecting it."

Now, you think about this, and you realize that this means the quiz can't be on June 30: that's final day of the course, so if the quiz hasn't been given before then, it has to be given on the 30th, and so when you walk into class on the 30th you'll certainly be expecting it.

But that means the quiz can't be on June 29: we just argued that the quiz can't be on the 30th, so if you walk into class on the 29th, you'll know it has to be given that day.

But that means the quiz can't be on June 28: we just argued that the quiz can't be on the 29th or the 30th, so if you walk into class on the 28th, you'll know this is the last chance, and it has be given that day.

But that means the quiz can't be on June 27 — well, you get the idea.

You can argue in this way, by induction (going down from large numbers to small), that if the quiz can't be given from day n onwards, then when you walk into class on on day $n-1$, you'll be expecting it. So it can't be given on day $n-1$ either.

Yet something is wrong with all this. For example, suppose it turns out that the quiz is given on June 14. Why June 14? Exactly the point — you're surprised by the choice of June 14, so it was a surprise quiz after all.

This paradox (also called the "unexpected hanging", where the (more grisly) story is of a surprise execution) is truly vexing, and it's been claimed that over a hundred research papers have been written on the topic. Again, rather than try to identify the difficulties in the reasoning, let's simply note that any attempt to formalize what's going on will have to be very careful about what is means to be "surprised," and what is means to "know" that the quiz "must" be given on a certain date.

As part of this challenge in formalizing the reasoning, I should add the Joe Halpern, who teaches CS 280 in the spring semester, claims that the surprise quiz, in a fundamental sense, isn't really about induction in the end. I'll leave it to him to expand on this argument ...

6.3 The smallest number that cannot be described in fewer than 80 symbols

To introduce the third example, let's start with something that's completely correct — a way in which one can view induction as a style of proof by contradiction. Suppose we've established the basis and the induction step. Then $P(n)$ must hold for all n, since if it failed, there'd be a *smallest* n on which $P(n)$ failed. But this n can't be 1 (since we proved $P(1)$ directly in the basis), and this means that $P(1), \ldots, P(n-1)$ are all true. But then the induction step implies that $P(n)$ must be true — a contradiction.

In other words, one way to reason by induction is to suppose that $\forall n : P(n)$ is false, consider the *smallest counterexample* n, and then use the fact that $P(1), \ldots, P(n-1)$ are all true to get a contradiction.

This is a useful style of argument to keep in mind. (For example, one can go back over induction proofs we've already done and see how they'd look if written in this style.)

Here, though, we use this notion of the smallest counterexample to motivate a third and final induction paradox.

Many numbers can be described using a mixture of numbers and letters as symbols, without explicitly writing their decimal digits. For example, "the smallest prime number greater than 50" describes the number 53. Or, "Avogadro's number" (the number of molecules in a mole, from chemistry) succinctly describes an enormous number — one that would take more decimal digits to write out than the number of symbols it takes to write "Avogadro's number".

Now, some numbers can be described in fewer than 80 symbols and some cannot. So consider the following number: *The smallest number that cannot be described in fewer than 80 symbols.* The problem is that the phrase, "The smallest number that cannot be described in fewer than 80 symbols" contains fewer than 80 symbols, so I've just described it in fewer than 80 symbols, contradicting its description.

Again, there's no really simple way to explain what's wrong with this. But it gets at a subtle issue — self-reference, in which one refers to the means of description in the description itself. This notion will show up in more advanced CS courses like CS 381/481.

7 Induction and Strategic Reasoning

The Surprise Quiz Paradox describes a particular kind of reasoning in a very stylized setting. But in fact, this style of reasoning shows up concretely in certain human activities, and analogues have been studied by economists interested in the way induction informs strategic interaction.

Here's a specific instance that interests economists, where induction leads to a "paradoxical" outcome for strategic behavior. Consider the following two-player game, which we'll call the Trust Game. (People also refer to it as the "Centipede Game," for reasons we won't go into.)

Two players, A and B, sit across a table from one another. We start with two piles of money on the table — a larger one containing \$4, and a smaller one containing \$1. Player A can either *take* or *pass*: if he takes, he gets the larger pile and the game ends; if he passes, it becomes B's turn. After each *pass*, the size of each pile is doubled; so in the second turn, B has the option of taking \$8; in the third turn, A has the option of take \$16, and so forth.

The game can run for a maximum of $2n$ turns, for some number $2n$ known to both players.

To make this more concrete, suppose the game can go for at most 6 turns. If the games ends in the

- first turn, then A get 4 and B gets 1.
- second turn, then B get 8 and A gets 2.
- third turn, then A get 16 and B gets 4.
- fourth turn, then B get 32 and A gets 8.
- fifth turn, then A get 64 and B gets 16.
- sixth turn, then B get 128 and A gets 32.

Note that if both players pass for a while, they *each* benefit considerably. For example, if the game ran 20 turns, and the players passed all the way to the end, then they'd each get at least half a million dollars.

But if the players reason strategically, what's going to happen. Since the maximum number of turns is even, B will have the final move in turn $2n$, and will clearly take the larger pile. This means that if he makes it to turn $2n-1$, player A will take rather than passing. (By taking in turn $2n-1$, he stands to make twice what he would by passing.) But player B knows this, and so B will take if she makes it to turn $2n-2$. Player A, knowing this, will take if he makes it to turn $2n-3$, and Player B, knowing this, will take if she makes it to turn $2n-4$, This reasoning can be continued by induction: once we know that a player will take in turn j, we conclude that a player should take in turn $j-1$.

Continuing this induction all the way down to turn 1, we conclude that player A will take in turn 1 — in other words, he will talk away with \$4 even though both players stand to make much more if they could somehow hold on to a much later turn.

Think about it concretely. Suppose that in Uris Hall, the Psychology Department were running this experiment with a game that lasted up to 20 turns. This means that you and your opponent could walk in and come out fifteen minutes later with at least half a million dollars each. And yet, if you really followed the strategic principles above, and you were player A, you'd listen to the rules, think about it, and walk off with the \$4.

This sounds crazy, but the inductive logic is more inexorable than it first appears. Unlike the paradoxes in the previous section, everything here is mathematically precise — taking the \$4 really is the "right" thing to do under our specific model of strategic reasoning. Indeed, the appeal of this game to economists is the way in which it exposes some of the challenges in applying models of strategic reasoning to real-life situations.

The Trust Game has been the subject of experiments by economists (for relatively small stakes), and it's been found that real players tend to pass for one or two turns, and then take the money. In other words, they deviate from ideal strategic behavior, even when they seem to understand the rules and can execute the inductive reasoning above.

One explanation that people have proposed is the following. Suppose that some small fraction of the population consisted not of perfectly self-interested people, but of "altruists," who favored passing simply because they personally liked outcomes in which both players did well. Then when you sit down across from the other player, there's some chance that she's an altruist. If you knew she were an altruist, you should definitely pass — and so passing one or two times may be a good strategy even for purely self-interested players in this situation, as they "gamble" on what kind of player their opponent is.

There is a reasonably large literature on this problem, and a much larger one on the broader topic of *behavioral game theory*, which seeks to reconcile mathematical models of strategic behavior with the ways in which people behave in real life.