Illinois Institute of Technology
Department of Computer Science

Lecture 3: January 23, 2019

CS 330 Discrete Structures
Spring Semester, 2019

Three Applications of Induction

Crossing the desert

The same idea as in the book-stacking example leads to the following clever observation: If you have to cross the desert by Jeep with no sources of fuel other than what you can carry, you can do it if you have enough Jeeps and drivers, no matter how wide the desert is.

We'll measure distance by ToGs, "tankfuls of gas." If we only have one Jeep, it can travel 1 ToG into the desert, at which point it is stranded. But if two Jeeps leave together, they can travel 1/3 ToG, at which point Jeep 2 can transfer half of its remaining fuel (1/3 tank) to Jeep 1; Jeep 2 can then use its remaining 1/3 tank to return to the starting point. Meanwhile, Jeep 1 will have traveled 1/3 ToG but now has a full tank, so it can continue 1 ToG, reaching a distance of $1 + 1/3$ ToG from the starting point.

With 3 Jeeps, they travel 1/5 ToG, at which point Jeep 3 transfers 1/5 to each of Jeeps 1 and 2, leaving its tank 2/5 full. Jeeps 1 and 2, whose tanks are now full, continue as in the previous paragraph: after Jeep 2 transfers 1/3 of a tank to Jeep 1, Jeep 2 returns to Jeep 3; Jeep 2's tank is empty, but Jeep 3 has 2/5 of a tank, allowing both of them to return to the starting point. Jeep 1 can now travel $1 + 1/3 + 1/5$ ToG from the starting point.

In general, with n Jeeps, the Jeeps travel $1/(2n-1)$ ToG at which point Jeep n transfers $1/(2n-1)$ of a tank to each of the other $n-1$ Jeeps which then have full tanks, leaving Jeep n with $1 - \frac{n}{2n-1} = \frac{n-1}{2n-1}$ tanks of gas. Jeeps $1, \ldots, n-1$ proceed recursively, with $n-2$ Jeeps returning empty to Jeep n, who then transfers $\frac{1}{2n-1}$ tanks of gas to each of them, leaving it also with $\frac{1}{2n-1}$ tanks of gas; Jeeps $2, \ldots, n$ can then return to the starting point. Jeep 1 can then travel $\frac{1}{2n-1}$ ToG further than it could with $n-1$ Jeeps. If Jeep 1 can travel D_k ToG with k Jeeps, then $D_1 = 1$ ToG and $D_k = D_{k-1} + 1/(2k-1)$.

That is, Jeep 1 can travel the sum of reciprocals of the odd numbers, which we have seen grows like $(\ln n)/2$.

Good guys versus bad guys

There is a room with a large number of people, say n, all knowledgeable about logic and mathematical induction, each wearing either a white hat or a black hat; nobody can see the color of his/her own hat, but everybody can see all the other hats in the room. An announcement is made that every hour a chime will sound and when it does, anybody who could logically deduce the color of his/her hat must leave the room; furthermore, the announcement says that there is at least one person wearing a black hat in the room.

Theorem. *If there are exactly k people in the room wearing black hats, those k leave the room at the kth chime.*

Proof. To get the idea of the induction, consider what happens with $k = 1$ black hat in the room. The person wearing that hat sees no other black hats, yet he knows that there is at least one, so it must be his/her own; therefore he leaves the room at the first chime. Suppose there are $k = 2$ black hats in the room worn by A and B. At the first chime, A sees another black hat and so can deduce nothing about his/her

own hat; similarly, B can deduce nothing about his/her own hat. But at the second chime, A knows that B did not leave the room on the first chime (and vice versa), so A can deduce that his/her own hat is black (if it were white, B would see only white hats and would have left at the first chime). B similarly deduces that he/she is wearing a black hat (again, if it were white, A would see only white hats and would have left at the first chime). Thus A and B leave at the second chime.

To make the inductive argument, let S_i be the statement "Everybody knows that at least i people are wearing black hats." We claim that S_i is true at the ith chime, that is, everybody knows that at least i people are wearing black hats. We have seen that this is true for $i = 1$ and $i = 2$. Suppose it is true for some $i \geq 2$. So, at the $(i+1)$st chime, each person reasons, "We all knew there were at least i black hats in the room, but nobody left; therefore *everybody* must be seeing at least i black hats because anybody seeing only $i-1$ black hats would have deduced that he was wearing a black hat and would have left the room. But "everyone" includes at least one person wearing a black hat because of the initial announcement: that person, too, must be seeing i black hats, so there must be at least $i+1$ black hats in the room; everybody in the room makes that same deduction. Therefore at the $(i+1)$st chime, everybody knows that at least $i+1$ people are wearing black hats, proving S_{i+1}.

At the kth chime, therefore, everybody in the room knows that there are at least k black hats, but each person wearing a black hat can see only $k-1$; black hats, so those people deduce that they are wearing black hats and simultaneously leave the room. □

This example appears useless, but in suitable generalizations it tells us how quickly information can spread among a network of computers ("Hat guessing games," *SIAM J. Discrete Math.*, vol. 22, no. 2, 2008, pp. 592–605).

Euclid's algorithm

Euclid's algorithm gives a method for computing the greatest common divisor of two integers u and v. WLOG we assume that $u \geq v$. The algorithm can be described by a sequence of remainders r_i, defined as:

$$r_0 = u$$

$$r_1 = v$$

$$r_{i+1} = r_{i-1} \bmod r_i$$

For any u and v, eventually one arrives at a remainder of 0; call it r_{k+1}. Then the greatest common divisor of u and v is r_k, so the algorithm gives a sequence

$$r_0 = u \geq r_1 = v > r_2 > r_3 > \cdots > r_k > r_{k+1} = 0.$$

This is a *decreasing* sequence of non-negative numbers, so it must be finite and hence the algorithm must terminate.

The algorithm can (should!) be expressed recursively:

$$\gcd(u, v) = \begin{cases} u & \text{if } v = 0, \\ \gcd(v, u \bmod v) & \text{otherwise,} \end{cases}$$

(Exercise: Prove this form is equivalent to the iterative form.)

We can ask two questions about this algorithm. First, how do we know that when it ends, r_k is actually the greatest common divisor of u and v? Second, for an input of a given size, how large is k? That is, how many iterations can proceed before a result is found?

The answer to the first question is given in Rosen on pages 282–284.

The second question is more interesting (see Rosen, pages 368–370). Let us assume that u (the larger number) has n decimal digits, or, equivalently, that

$$10^n \leq u < 10^{n+1}.$$

The remainders decrease at each iteration; we make this observation more precise by noting that in general

$$r_{i-1} \geq r_i + r_{i+1}$$

Why is this so? Assume that $u \geq v$ so that $r_0 \geq r_1$, since $r_0 = u$ and $r_1 = v$. We know that the definition of r_{i+1} is the remainder when r_{i-1} is divided by r_i:

$$r_{i+1} = r_{i-1} \bmod r_i$$

or, in other words,

$$r_{i-1} = \text{ (some multiple of } r_i) + r_{i+1}$$

Since $r_{i-1} \geq r_i$, the "multiple of r_i" cannot be zero, so

$$r_{i-1} \geq r_i + r_{i+1}$$

for $i = 1, 2, \ldots$.

This means we have a chain of inequalities:

$$\begin{aligned} r_0 &\geq r_1 + r_2 \\ r_1 &\geq r_2 + r_3 \\ r_2 &\geq r_3 + r_4 \\ &\vdots \end{aligned}$$

Suppose that the algorithm ends after k iterations with $r_{k+1} = 0$, so r_k is the greatest common divisor sought. Then successive substitution in the chain of inequalities leads to

$$\begin{aligned} r_0 &\geq r_1 + r_2 \\ &\geq (r_2 + r_3) + r_2 = 2r_2 + r_3 \\ &\geq 2(r_3 + r_4) + r_3 = 3r_3 + 2r_4 \\ &\geq 3(r_4 + r_5) + 2r_4 = 5r_4 + 3r_5 \\ &\geq 5(r_5 + r_6) + 3r_5 = 8r_5 + 5r_6 \\ &\vdots \\ &\geq F_k r_k + F_{k-1} r_{k+1} = F_k r_k \end{aligned}$$

where F_0, F_1, F_2, $\ldots$ is the Fibonacci sequence $F_0 = 0$, $F_1 = 1$, $F_{i+1} = F_i + F_{i-1}$. Since r_k is the greatest common divisor, $r_k \geq 1$ gives

$$u = r_0 \geq F_k.$$

If u has n digits, then because $u = r_0$, we know

$$10^{n+1} > F_k.$$

We will show later this semester that

$$F_k \approx \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^k.$$

Thus

$$10^{n+1} > \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^k$$

and taking logarithms and solving for k gives

$$\frac{n+1}{\log_{10}(\frac{1+\sqrt{5}}{2})} + \frac{\log_{10}\sqrt{5}}{\log_{10}(\frac{1+\sqrt{5}}{2})} > k$$

$$k \approx 4.785n,$$

so that k, the number of iterations, is at most approximately $4.785n$ for n-digit numbers.

Thus, the number of iterations executed in Euclid's algorithm grows at a rate *linear* in the number of digits of u and v, or, equivalently, *logarithmic* in their actual values.