

Exercise 1. Counting

We saw in lecture that it is possible to draw an uncountable number of circles on the plane. Given a point p in the plane, and two positive real numbers $x_1 < x_2$, we define a *band* to be the area with radius (x_1, x_2) (an open interval) about the point p . Prove that it is not possible to draw an uncountable number of non-intersecting bands on the plane.

See lecture 1 slides p. 26-27 for hints.

1. The set of rational numbers \mathbb{Q} is countable. This can be shown by diagonalization.
2. Consider some set S (of potentially infinite size) of non-intersecting bands in the plane. For band i , consider a horizontal line going through the point at its center. Because the rationals are dense and the band has finite thickness, we can find some point on this line that is also on the band and whose x coordinate x_i is rational. Now consider a vertical line going through x_i . This line also intersects the band, by construction of x_i , and thus there is a point (x_i, y_i) with $y_i \in \mathbb{Q}$ that is on the line and in the band. Furthermore, since the bands don't intersect, this point uniquely identifies band i .
3. Since \mathbb{Q} is countable, so is $\mathbb{Q} \times \mathbb{Q}$. Since we have found an injection from bands to $\mathbb{Q} \times \mathbb{Q}$, the number of bands is countable.

Exercise 2. Computability

- Prove that there are an uncountable number of functions in $\mathbb{N} \rightarrow \mathbb{N}$.

Suppose the number of functions were countable. Then we can enumerate them as f_0, f_1, f_2, \dots . Now define a function $g(x)$ as follows:

If $f_x(x) = c$ then $g(x) = a$, where $a \in \mathbb{N}$ s.t. $a \neq c$. For example, one valid definition would be $g(x) = f_x(x) + 1$.

Thus $g(x)$ is a function in $\mathbb{N} \rightarrow \mathbb{N}$ that differs from every f_i . But this contradicts the assumption that f_0, f_1, f_2, \dots enumerate all such functions. Therefore the set of functions $\mathbb{N} \rightarrow \mathbb{N}$ is uncountable.

- Consider your favorite programming language. A program is a string in Σ^* , where Σ is some finite alphabet like ASCII. Prove that there are only a countable number of programs. Assume ASCII encoding. For each valid program P , define a function $P \rightarrow \mathbb{N}$ by concatenating the three-digit ASCII codes (000 to 255) of the characters in P together to form a long but not infinite length number. Since this number is in \mathbb{N} , we have shown that all valid programs P can be mapped to a unique natural number. Thus there exists a injection from the set of valid programs P to \mathbb{N} . It follows that the set is countable.
- What does this suggest?

There are an uncountable number of functions $\mathbb{N} \rightarrow \mathbb{N}$. As any given programming language can only express a countable number of functions (programs), this suggests that some functions are not computable.
- Why is this not a proof that some functions are not computable?

We have shown that the set of functions encompassed by any programming language expressible as a string of some finite alphabet is a strict subset of all functions $\mathbb{N} \rightarrow \mathbb{N}$. However, we have not proved that all computable functions can be expressed as programs in Σ^* for some alphabet Σ .

Exercise 3. Induction The strings of balanced parentheses can be defined in at least two ways.

1. A string w over alphabet $\{(), ()\}$ is balanced iff:
 - w has an equal number of '('s and ')'s, and
 - any prefix of w has at least as many '('s as ')'s.
2.
 - ϵ is balanced,
 - If w is balanced, then (w) is balanced,
 - If x and y are balanced, so is xy ,
 - Nothing else is balanced.

Prove by induction on the length of the string that definitions (1) and (2) define the same class of strings.

It is very easy to prove that neither definition will accept odd-length strings, so we only worry about even length ones. We'll use the notation 1-balanced and 2-balanced to describe strings that are balanced by definition 1 and 2 respectively.

Base case *The string ϵ meets both definitions.*

Induction step *Assume any string of length k is 1-balanced iff it is 2-balanced. Prove that this is also true for any string w of length $k + 2$.*

1. *1-balanced implies 2-balanced: let $uv = w$ such that u is the shortest 1-balanced prefix of w . There are two cases:*

- $|u| < |w|$: *Since appending v to u must produce the 1-balanced string w , we can show that v must satisfy both rules of definition 1, and is therefore also 1-balanced. Since u and v both have length $\leq k$ and both are 1-balanced, both are also 2-balanced, by the induction hypothesis. Applying the second rule of definition 2, $uv = w$ must also be 2-balanced.*

- $|u| = |w|$: *Let $xsy = u$, where $|x| = |y| = 1$. Since u is 1-balanced, x must be (by the prefix rule. Furthermore, y must be), since otherwise xs would have more ')'s than '('s, violating the prefix rule for u .*

But now we can prove that s is 1-balanced. We know s must have equal numbers of '('s and ')'s, because we removed one of each from u , which was 1-balanced. We also know that it has no prefix with more ')'s than '('s.

We can prove this by contradiction. Find the shortest such prefix z . By construction, it must have exactly one more) than (. But the string xz would then have been a 1-balanced prefix of w , and we know that no such prefix exists, so this is a contradiction.

Thus s must be 1-balanced, and since $|s| = k$ this implies that it's also 2-balanced by the induction hypothesis. Therefore by the third rule of definition 2, w is 2-balanced.

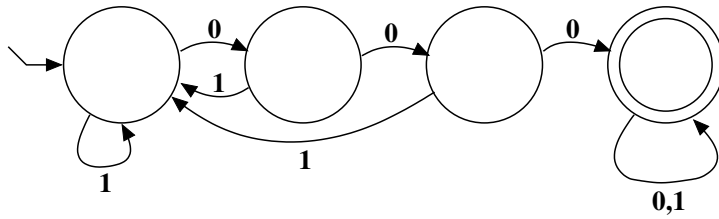
2. *2-balanced implies 1-balanced: By the definition of 2-balanced we know that either:*

- $w = (s)$ for 2-balanced s . *Since $|s| = k$, s is also 1-balanced by the induction hypothesis, and it is easy to see that enclosing a 1-balanced string in open and close parens produces a 1-balanced string.*
- $w = uv$ for 2-balanced u, v . $|u| \leq k$ and $|v| \leq k$, so they are also 1-balanced by the induction hypothesis. *Again, it's easy to see that appending a 1-balanced string to another 1-balanced string produces a 1-balanced string, since the number of parens remains equal and both substrings satisfy the prefix requirement individually.*

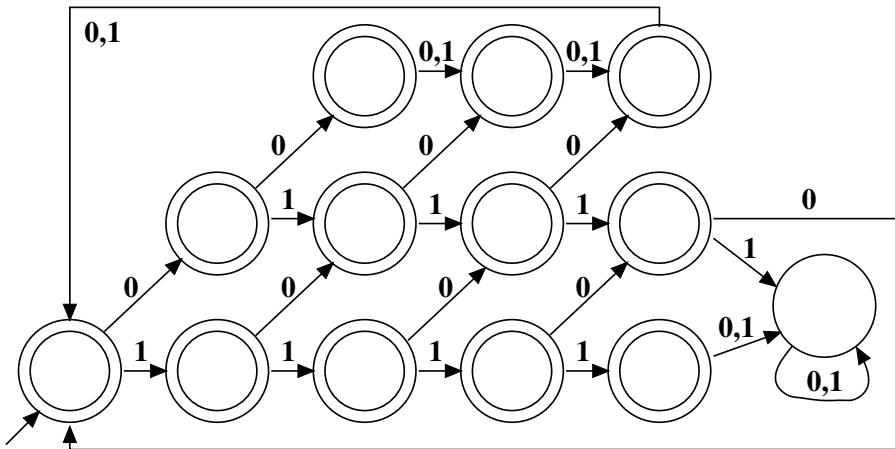
Exercise 4. Deterministic Finite Automata

Give deterministic finite automata accepting the following languages over $\Sigma = \{0, 1\}$.

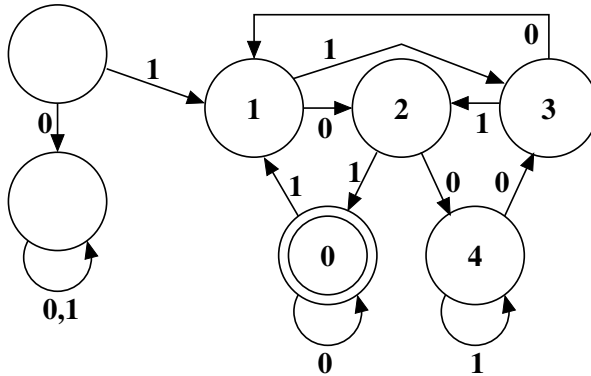
a) The set of all strings with three consecutive 0's.



b) The set of all strings such that every consecutive set of five symbols contains at least two 0's.



c) The set of all strings beginning with a 1 that, when interpreted as the binary representation of an integer, are congruent to zero modulo 5.



Exercise 5. Regular expressions

Let r and s denote regular expressions. Consider the equation $X = rX + s$ where rX is concatenation, and $+$ is union, Assuming that the set denoted by r does not contain ϵ , find the solution for X and prove that it is unique. What is the solution if $L(r)$ contains ϵ ?

*Solution is r^*s if r does not contain ϵ and $r^*(s + t)$, where t is some string in Σ^* , if r contains ϵ .*

$rX + s = rr^*s + s = rr^*s + \epsilon s = r^*s$ if r can't be ϵ .

Proof of uniqueness by contradiction:

*Assume that the solution is $X = r^*s + Y$, where $Y \cap r^*s = \emptyset$ and $Y \neq \emptyset$. Choose the shortest string $z \in Y$. Substituting our solution into the original equation we find*

$$X = rr^*s + rY + s = r^*s + rY$$

and thus $Y = rY$. So we can split z into two parts, $z = uv$, where $u \in r, v \in Y$. But since $\epsilon \notin r$, $|u| > 0$, which implies that $|v| < |z|$, contradicting our assertion that z was the shortest string in Y .

If r can be ϵ , then

$$\begin{aligned} rX + s &= rr^*(s + t) + s \\ &= rr^*s + rr^*t + s \\ &= rr^*s + s + rr^*t \\ &= r^*s + r^*t \\ &= r^*(s + t) \end{aligned}$$