1 Countability

Recall that a set S is **countable** (either finite or countably infinite) if and only if there exists a surjective mapping $\mathbb{N} \to S$, i.e. if and only if we can enumerate the elements of S. This means that the following are countable:

- Any subset of a countable set.
- A countable union of countable sets.
- The direct product of two countable sets.
- The image of a countable set under a set function.

Exercise. *Are the following sets countable or not?*

1. *The set of all infinite binary sequences* $\{0,1\}^{\mathbb{N}}$.
2. *The set of real numbers* $\mathbb{R}$.
3. *The set of rational numbers* $\mathbb{Q}$.
4. *The set of all English words (that is, all finite strings that can be written using the English alphabet).*
5. *The set of all English sentences.*

Solution. 1. Uncountable, by a diagonalization argument.

2. Uncountable; this is (almost) the same set as the previous problem.
3. Countable; this is the image of $\mathbb{N} \times \mathbb{N}$ given by $(n, m) \to n/m$.
4. Countable; this is the countable union of the sets S_i, where S_i is the set of words of length i and hence is finite for each i.
5. Countable; this is similar to the previous problem, but we can take the union over sentence length, rather than word length.

Exercise. *You are in a spaceship A standing still at the origin of $\mathbb{R}^3$. There is an enemy spaceship B somewhere in space that you wish to attach a spy probe to. This works by launching the probe in such a way that it ends up hitting B; there is also some constant amount of time T you have to wait between launching two probes. Suppose your intelligence is intelligent enough to know that B is moving with constant velocity v whose components are rational numbers, and, moreover, B is in some point p with rational coordinates at some point of time, but not intelligent enough to know v or p.*

Fortunately, you have a countably infinite supply of spy probes. Show that you have an algorithm that guarantees success in finite time.

Solution.
For each possible combination of v and p and for each time t, there is a velocity v_s such that a spy probe launched at time t with velocity v_s hits B. Now it remains to observe that $(v, p) \in \mathbb{Q}^6$, which is countable.

This is useful because the class of decidable languages is countable, as is the class of recognizable languages; consequently, because the number of languages is uncountable, we know that there exist undecidable and unrecognizable languages. In fact, "most" languages are both undecidable and unrecognizable.

2 Decidability and Undecidability

Recall that:

- A Turing Machine M **decides** a language L if it halts on every input and L is the set of words for which M has output 1. A language is decidable if there is a TM for which this is the case.
- A Turing Machine M **recognizes** a language L, where L is the set of words for which M has output 1. A language is recognizable if there is a TM for which this is the case.

2.1 Diagonalization

Theorem 1.
The languages $A_{TM}, A_{WR} = \{\langle M, w\rangle : M \text{ accepts the input } w\}$, where M is either a TM or a word-RAM, respectively, are not decidable.

Proof. (Diagonalization Argument) Assume $\{\langle M, w\rangle : M \text{ accepts the input } w\}$ is decidable.

Then the language $D = \{\langle M\rangle : M \text{ accepts } \langle M\rangle\}$ is decidable, hence $\overline{D} = \{\langle M\rangle : M \text{ does not accept } \langle M\rangle\}$ is decidable. Suppose $\overline{D}$ is decidable by M_1, then $\langle M_1\rangle \in \overline{D}$ iff M_1 accepts $\langle M_1\rangle$ iff $\langle M_1\rangle \in D$, which is a contradiction. □

We similarly showed via reduction that $HALT_{TM}$, the problem of deciding whether a Turing Machine M halts on a string w, is undecidable even if M or w (but not both!) is fixed.

Exercise. *Show that a language $L \subseteq \Sigma^*$ is r.e. iff there is a computable function $f : \mathbb{N} \to \Sigma^*$ whose image is exactly L. That is, we can* enumerate *the elements of L as $\{f(0), f(1), f(2), \ldots\}$ for some recursive (=computable) function f.*

Solution.
($\Longrightarrow$) If L is r.e., we know there is a Turing machine M that accepts exactly the strings in L. If L is a finite language, it is straightforward to make an enumerator M' for L.

Otherwise, we can enumerate all elements of L using the following enumerator M': choose some computable ordering on all strings in Σ^*, and for i going from 1 to infinity, let M' run M on the first i strings for i steps. Then the n-th string that M accepts in the course of this algorithm is the value of $f(n)$.

($\Longleftarrow$) Suppose we have an enumerator M for L. Then a recognizer M' for L on input x is obtained by simply computing $f(0), f(1), \ldots$ in order, and accepting if $f(n) = x$ for some n.

2.2 Reductions

Theorem 2.
If $L_1 \leq_m L_2$ and L_1 is undecidable, then so it L_2.

Exercise. *Reductions can be tricky to get the hang of, and as we have seen with NP-completeness, you want to avoid "going the wrong way" with them. In which of these scenarios does $L_1 \leq_m L_2$ provide useful information (and in those cases, what may we conclude)?*

(a) L_1's decidability is unknown and L_2 is undecidable

(b) L_1's decidability is unknown and L_2 is decidable

(c) L_1 is undecidable and L_2's decidability is unknown

(d) L_1 is decidable and L_2's decidability is unknown

Solution.(a) Nothing.

(b) L_1 is decidable. This is in Sipser.

(c) Undecidable. Corollary to the above question.

(d) Nothing.

Exercise. *Let $S = \{\langle M \rangle : M$ is a TM such that $L(M) = \{\varepsilon\}\}$. Using mapping reductions, prove that S is neither r.e. nor co-r.e.*

Solution.
To show S is not r.e., we reduce from a non-r.e. language. One such is $\overline{\mathrm{A_{TM}}}$. The reduction works by sending a given Turing machine M and string x to a Turing machine M_x which accepts ε, and for any other input runs M on x. Then observe that if M does not accept x, $L(M_x) = \{\varepsilon\}$, and converesely, if $L(M_x) = \{\varepsilon\}$, then M does not accept x.

To show that S is not co-r.e., it suffices to reduce from a non-r.e. language to the complement of S. We reduce from $\overline{\mathrm{A_{TM}}}$ again: given M, x, we get a Turing machine M_x such that M_x rejects everything different from ε, and on ε runs M on x. Then if M does not accept x, $L(M_x) = \emptyset$, and conversely, if $L(M_x) = \emptyset$, M does not accept x.

2.3 Rice's Theorem

Theorem 3 (Rice's theorem)**.**
Let $\mathcal{P}$ be any subset of the class of r.e. languages such that $\mathcal{P}$ and its complement are both nonempty. Then the language $L_{\mathcal{P}} = \{\langle M \rangle : L(M) \in \mathcal{P}\}$ is undecidable.

Intuitively, Rice's theorem states that Turing machines cannot test whether another Turing machine satisfies a (nontrivial) property. For example, let $\mathcal{P}$ be the subset of the recursively enumerable languages which contains the string a. Then Rice's theorem claims that there is no Turing machine which can decide whether a Turing machine accepts a.

Cautionary note: Rice's theorem only tells you about the language accepted by the Turing machine, not about how the Turing machine performs its computation. For example, Rice's theorem is not applicable to the first exercise in the next section.

2.4 Exercises: are these languages decidable?

Exercise. $L = \{\langle M, x\rangle : \textit{ At some point it its computation on } x, \textit{ } M \textit{ re-enters its start state}\}$

Solution.
Undecidable. Assume for the sake of contradiction that this language is decidable. We will show that a decider for this language can decide the halting problem. Given an input $\langle M, w\rangle$, modify M so that the start state is split into two states, and the new start state has no incoming transitions except for those we will specifically add in this construction. Also, change the accept and reject states so that they are no longer accept / reject states, but just normal states that (effectively) epsilon transition to the new start state. Make new accept / reject states that are inaccessible. Now the machine re-enters its start state iff the original machine would have halted. This construction was a finite transformation, and something a TM could do, so by combining this machine with a decider for the language in question, we could decide the halting problem. Contradiction.

Exercise. $L = \{\langle M\rangle : M \textit{ on input } i, \textit{ outputs the } i\textit{th digit of } \sqrt{2} \textit{ in binary}\}$

Solution.
First, let L' be the language which consists of the integers i for which the ith digit of $\sqrt{2}$ is a 1. Then, we have that $L = \{\langle M\rangle : L(M) = L'\}$.

Note that L' is r.e., since we can compute $\sqrt{2}$ to an arbitrary number of digits of precision using standard arithmetic. Consequently, by Rice's theorem, L is not decidable.

Exercise. $L = \{\langle x, y\rangle : f(x) = y\}$ *where* f *is a fixed computable function.*

Solution.
Decidable. Given $\langle x, y\rangle$, compute $f(x)$ and compare it to y.

3 Probability Review

A **discrete random variable** X which could take the values in some set S can be described by the probabilities that it is equal to any particular $s \in S$; we write this as $\mathbb{P}(X = s)$. Two random variables X, Y are independent if their outcomes are unrelated; i.e. if for all possible pairs of outcomes x, y, $\mathbb{P}((X, Y) = (x, y)) = \mathbb{P}(X = x)\mathbb{P}(Y = y)$.

Then, its **expected value** $\mathbb{E}(X)$ is the "average" value it takes on, with

$$\mathbb{E}(X) = \sum_{s \in S} s \cdot \mathbb{P}(X = s).$$

For example, the expected value of a standard die is $\frac{1}{6}(1+2+3+4+5+6) = 3.5$. As it turns out, it is true that

$$\mathbb{E}(aX + bY) = a\mathbb{E}(X) + b\mathbb{E}(Y),$$

even if X and Y are not independent (for example, you can easily verify that this is true when $X = Y$). This is known as *linearity of expectation.*

Two types of common random variables are:

- Geometric random variable: the number of tries until some event which happens with probability p on each try occurs. When computing with geometric random variables, the following facts often come in handy:

$$\sum_{i=0}^{\infty} p^i = \frac{1}{1-p} \text{ (for } 0 < |p| < 1) \text{ and } \sum_{i=0}^{n} p^i = \frac{1-p^{n+1}}{1-p} \text{ (for all } p).$$

- Binomial random variable: the number of heads given by n random coin flips

We can prove a few useful facts using the definitions above.

Exercise. *Prove that for random numbers X which only take on nonnegative integer values,*

$$\mathbb{E}(X) = \sum_{j=0}^{\infty} \mathbb{P}(X > j).$$

Solution.
We manipulate the definition of the expectation:

$$\mathbb{E}(X) = \sum_{k=0}^{\infty} k\mathbb{P}(X = k) = \sum_{j=1}^{\infty}\sum_{k=j}^{\infty} \mathbb{P}(X = k) = \sum_{j=1}^{\infty} \mathbb{P}(X \geq j) = \sum_{j=0}^{\infty} \mathbb{P}(X > j)$$

as wanted.

Exercise. *Show that if some event happens with probability p, the expected number of tries we need in order to get that even to occur is $1/p$.*

Solution.
By the previous exercise, and noting that the probability that an event still has not happened after i tries is $(1-p)^i$, the expectation is $\sum_{i=1}^{\infty}(1-p)^i = \frac{1}{p}$.

Alternatively, the probability that the event happens after exactly the ith try is $(1-p)^{i-1}p$. Thus, we can compute

$$\mathbb{E}(\#\text{ of tries}) = \sum_{i=1}^{\infty} i(1-p)^{i-1}p.$$

Additionally, one useful inequality about random variables is **Markov's inequality**: for any nonnegative random variable X and $\lambda > 0$,

$$\mathbb{P}(X > \lambda \cdot \mathbb{E}X) < \frac{1}{\lambda}.$$

In other words, this indicates that the probability of X being significantly larger than its expectation is small. For example, if an event happens with probability p, then Markov's inequality tells us that

$$\mathbb{P}(\text{more than } x \text{ tries are needed for the event to happen}) < \frac{p}{x}.$$

(Chebyshev's inequality, which can be proved using Markov's inequality, gives a somewhat similar bound which also deals with the case that X is significantly smaller than its expected value.)