Recall that a *mapping reduction* is a computable map \( f : \Sigma^* \rightarrow \Sigma^* \) between two languages \( A \) and \( B \) such that for all strings \( \omega \in \Sigma^* \), \( \omega \in A \) if and only if \( f(\omega) \in B \).

**Problem 1.** Define the language

\[ R = \{ \langle M \rangle \mid L(M) \text{ is regular} \} . \]

Does Rice’s Theorem apply to \( R \)? If so, show it. If not, give a counter example.

**Solution 1.** Rice’s Theorem applies to \( R \).

First, any Turing machine which recognizes \( A_{TM} \notin R \). Moreover, we have previously shown that a Turing machine can emulate any DFA, so clearly there’s at least one Turing machine which belongs to \( R \).

Lastly, suppose \( A \) and \( B \) are two Turing machines with \( L(A) = L(B) \). Then if \( A \) belongs to \( R \), then the language of \( A \) is regular, hence so is \( B \)’s language. Thus \( B \) also belongs to \( R \). The same applies in the reverse direction, so \( R \) is a property of the Turing machine’s language.

**Problem 2.** To show that \( R \notin RE \), you will need a language not in \( RE \) to (mapping) reduce to it. Propose a language \( L \) and explain why you chose it.

**Solution 2.** We’ll go with \( \overline{A_{TM}} \). We can equivalently show \( A_{TM} \leq_m \overline{R} \), which is much easier. The reduction will follow almost identically to how it does in Rice’s Theorem.

**Problem 3.** To show \( L \leq_m R \), you will need to transform the input from \( L \) to input to \( R \). This is the (mapping) reduction \( f \). Give \( f \).

**Solution 3.** We want a reduction that transforms an input \( \langle M, \omega \rangle \) to \( A_{TM} \) into a single input \( I_{M,\omega} \) to \( \overline{R} \). Its language should be irregular if and only if \( M(\omega) \) accepts. This is easy to do.

Define the Turing machine \( I_{M,\omega} \) to be

\[ I_{M,\omega} = \text{On input } \nu \]
\[ 1. \text{If } \nu \text{ is not of the form } 0^n1^n, \text{ reject} \]
\[ 2. \text{Run } M(\omega) \]
\[ 3. \text{If } M \text{ accepts, accept; otherwise, reject} \]

Let \( f(\langle M, \omega \rangle) = \langle I_{M,\omega} \rangle \).

**Problem 4.** Prove that \( f \) is, in fact, a reduction.
Solution 4. If we look at $L(I_{M,\omega})$, we see $I_{M,\omega}$ can only accept strings from the irregular language $0^i1^i$ ($i \in \mathbb{N}$). If $M(\omega)$ accepts, it accepts all such strings, hence $L(I_{M,\omega})$ is irregular. If $M(\omega)$ does not accept, then it accepts no strings, hence $L(I_{M,\omega})$ is regular. So $L(I_{M,\omega})$ is irregular iff $M(\omega)$ accepts.

In other words, we have $\langle M, \omega \rangle \in \mathcal{ATM}$ iff $f(\langle M, \omega \rangle) \in \overline{R}$.

A mapping reduction need not apply only to abstract computational tasks. We can use it for to relate more immediately practical problems to each other as well. This will be the focus of the third section of the course on complexity theory.

A directed graph, or digraph, is pair $G = (V, E)$ of vertices $V$ connected by edges $E \subseteq V \times V$ (we disallow edges from a vertex to itself).

In the figure above, $V = \mathbb{Z}_6$ and $E = \{(2, 3), (0, 2), (0, 4), (2, 4), (0, 5), (4, 5), (1, 4), (1, 5)\}$. Note that in a digraph, the order we write the vertices in the edge matters. The first vertex is the source and the second vertex is the destination.

A cycle $c$ in a digraph $G = (V, E)$ is a sequence of vertices $v_1, \ldots, v_n \in V$ such that $v_1 = v_n$, all other vertices are distinct, and each $(v_i, v_{i+1}) \in E$. In the figure above, $0, 4, 2, 0$ is a cycle.

A Hamiltonian cycle is a cycle that visits each vertex. In the figure above, there is no Hamiltonian cycle. However, if the edge $(0, 3)$ were present, then there would be a Hamiltonian cycle by traversing the perimeter counterclockwise.

Problem 5. Define the language

$$HAMCYCLE = \{\langle G \rangle \mid G \text{ is a digraph with a Hamiltonian cycle}\}.$$  

Give an algorithm that decides $HAMCYCLE$ (it need not be the slightest bit efficient).

Solution 5. Our algorithm will simply try every permutation of the vertices of $V$ and check if that permutation (plus the first vertex duplicated at the end) is a Hamiltonian cycle. If any such permutation is a Hamiltonian cycle, we accept. If no such permutation is a Hamiltonian cycle, we reject.

A path $p$ in a digraph $G = (V, E)$ is a sequence of distinct vertices $v_1, \ldots, v_n \in V$ such that each $(v_i, v_{i+1}) \in E$. If $p$ begins at $s \in V$ and ends at $t \in V$, we call $p$ an $s$–$t$ path. In the figure above, $1, 4, 2, 3$ is a path from 1 to 3. Note that sometimes it is more convenient (or even necessary) to define a path (and cycle) as sequence of edges rather than vertices but to the same effect.

A Hamiltonian path is a path that visits each vertex. In the figure above $0, 5, 1, 4, 2, 3$ is a Hamiltonian path.
Problem 6. Define the language

\[ \text{HAMPATH} = \{ \langle G, s, t \rangle \mid G \text{ is a digraph with an } s \dashv t \text{ Hamiltonian path} \} . \]

Propose a reduction \( f \) such that \( \langle G, s, t \rangle \in \text{HAMPATH} \) if and only if \( \langle f(G, s, t) \rangle \in \text{HAMCYCLE} \).

Hint: A cycle starts nowhere, so you can think of it as starting (and ending) anywhere you choose.

Solution 6. We will create a new graph \( G' = (V', E') \) as follows. Add a vertex \( d \) to \( V \) to obtain \( V' \). Then add the directed edges \( (t, d) \) and \( (d, s) \) to \( E \) to obtain \( E' \).

The reduction is then \( f(G) = G' \).

Problem 7. Prove that \( \text{HAMPATH} \leq_m \text{HAMCYCLE} \) by showing that your reduction holds.

Solution 7. Suppose that \( \langle G, s, t \rangle \in \text{HAMPATH} \). Then there is a Hamiltonian path \( p = v_1, \ldots, v_n \) \((n = |V|)\) in \( G \) such that \( v_1 = s \) and \( v_n = t \). In \( G' \), \( p \) is still a valid \( s \dashv t \) path. Moreover, because \( G' \) contains the edges \( (t, d) \) and \( (d, s) \), the extended sequence of vertices \( p, d, s \) is a cycle. Since \( p \) visits each vertex of \( V \), and \( p, d \) visits each vertex of \( V' \), it follows that \( p, d, s \) is a Hamiltonian cycle in \( G' \).

In the other direction suppose \( \langle G' \rangle \in \text{HAMCYCLE} \). Then there is a Hamiltonian cycle \( c = v_1, \ldots, v_n \) \((n = |V'| + 1)\) in \( G' \) such that \( v_1 = v_n \). We may assume without loss of generality that \( v_1 = s \). Then since there is only one path through \( d \), namely \( t, d, s \), it follows that \( v_{n-2} = t \) and \( v_{n-1} = d \). This means that \( p = v_1, \ldots, v_{n-2} \) is an \( s \dashv t \) path in \( G' \). Further, \( p \) does not contain \( d \), so it is also an \( s \dashv t \) path of \( G \). In addition, it visits each vertex of \( V' \) except \( d \), so it visits each vertex of \( V \). This makes \( p \) a Hamiltonian path of \( G \).

Problem 8. What can you now conclude about \( \text{HAMPATH} \)?

Solution 8. \( \text{HAMPATH} \in \text{DEC} \) (or rather \( \text{HAMPATH} \) is decidable in \( O(n!) \) time).

As stated, the only requirement placed upon the reduction \( f \) is that it is computable. There is no reason we cannot place additional resource restrictions upon it (or even loosen that restriction for some exotic reduction in the arithmetic hierarchy). For instance, we can require that \( f \) be computable and requires only log space. Alternatively we might require that \( f \) be computable in linear time.

Problem 9. Suppose that \( \text{HAMCYCLE} \) can be computed in exponential time (e.g. \( O(2^n) \)). What can you conclude about the running time of \( \text{HAMPATH} \)? Why?

Solution 9. \( \text{HAMPATH} \) can be computed in exponential-time as well. The construction of our reduction took merely polynomial-time, so its polynomial running time plus \( \text{HAMCYCLE}'s \) exponential running time is still exponential-time. Since the composition of these two algorithms decides \( \text{HAMPATH} \), this becomes an upper bound on \( \text{HAMPATH}'s \) running time.

Problem 10. What minimum resource restriction can you place upon \( f \) to ensure your conclusion in the previous problem?

Solution 10. An exponential-time reduction (of whatever base \( \text{HAMCYCLE} \) is computed in). A better bound would be a polynomial-time reduction since we managed to create one that runs that fast.