Introduction to Computing Thoery

February 20, 2023
collected by whaleyy

Problem 1: Rice Theorem in Undecidability

Classify whether each of the following languages are recursive, recursively enumerable but not recursive, or non-recursively enumerable. Prove your answers, but you may not simply appeal to Rice's theorem.

1. $L_5 = \{"M"|M$ is a TM, and $L(M)$ is uncountable $\}$
2. $L_6 = \{"M"|$ TM M accepts at least two strings of different lengths$\}$

Solution for 1. L_5 is recursive. $L_5 = \phi$. Any of the languages is countable based on the finite alphabet.
Solution for 2. L_6 is recursively enumerable but not recursive.
L_6 is recursively enumerable. Constructing a Universal Turing Machine which takes "M" as input. In the following order, it generates strings of different lengths from 0 to infinity according to the alphabet marked as $\omega_0, \omega_1.\omega_2....$ And with the lexicographical input strings, it simulates M. When M halts on two strings of different lengths, it will halt.

- Do 1 step of M's computation on ω_0
- Do 2 steps of M's computation on ω_0 and ω_1
- Do 3 steps of M's computation on ω_0, ω_1 and ω_2

L_6 is not recursive. Use the same method used to prove the Rice Theorem. If L_6 is recursive, then there is a reduction to the halting problem $H = \{"M"|M$ halts on $e\}$.
Assume L_6 is decidable, then for any TM, we can decide whether it belongs to L_6. Given an TM M, construct a TM M_e with input x: if $x \neq 0, 10$, reject; else simulate M on e. $L(M_e) = \phi$ if M doesn't halt on e, else $L(M_e) = \{0, 10\}$. However, which is also another form of L_6 that is assumed recursive. Contradiction.

Recall : the proof of Rice Theorem.
Given a TM M, we construct a TM M_e with input ω. First take certain TM "M^*" which belongs to the subset of r.e. languages.
It executes as follows: first simulate M on input e, then simulate M^* on input ω.
$L(M_e) = \phi$ if M doesn't halt on e, else $L(M_e) = L(M^*)$.
However, which is also another form of the problem to be proved. Contradiction.

Problem 2: Encoding of Language

The encoding of an object O is represented as $"O"$. Similarly, the encoding of several objects $O_1, ..., O_k$ is represented as $"O_1""O_2"..."O_k"$. Convert the following problems into the corresponding languages.

1. Given a DFA A and a string ω, does A accept ω ?
2. Let D be a DFA, given a string ω, does D accept ω ?
3. Given two DFAs A and B, is $L(A) = L(B)$?

Solution for 1. $A_{DFA} = \{"A""\omega" : A \text{ is a DFA that accepts } \omega\}$
Solution for 2. $A_\omega = \{"\omega" : D \text{ accepts } \omega\}$
Solution for 3. $EQ_{DFA} = \{"A""B" : A \text{ and } B \text{ are two DFAs}, L(A) = L(B)\}$

> Pay attention to the description of **GIVEN**.

Problem 3: Reduction from SB_{DFA} to E_{DFA}

Let $SB_{DFA} = \{"D_1""D_2" : D_1 \text{ and } D_2 \text{ are two DFAs with } L(D_1) \subset L(D_2)$. Give a reduction from SB_{DFA} to E_{DFA}.
Solution Suppose TM M_E decides E_{DFA}.
Construct a TM M that decides SB_{DFA}: $M =$ on input $"D_1""D_2"$:

1. construct a DFA D' with $L(D') = L(D_1) - L(D_2) = L(D_1) \cap \neg L(D_2)$
2. run M_E on $"D'"$
3. output the result

the reduction $f("D_1""D_2") = "D'"$
SB_{DFA} is decidable iff D' is decidable.

> Some abbreviation of TM decision problems.
>
> 1. $A_{DFA} = \{"D""\omega" : D \text{ is DFA that accepts } \omega\}$
> 2. $A_{NFA} = \{"D""\omega" : D \text{ is NFA that accepts } \omega\}$
> 3. $E_{DFA} = \{"D" : D \text{ is DFA and } L(B) = \phi\}$
> 4. $EQ_{DFA} = \{"A""B" : A, B \text{ are DFAs and } L(A) = L(B)\}$
> 5. $A_{CFG} = \{"G""\omega" : G \text{ is CFG that accepts } \omega\}$
> 6. $E_{CFG} = \{"G" : G \text{ is CFG and } L(G) = \phi\}$

7. $ALL_{CFG} = \{"G" : G \text{ is CFG and } L(G) = \Sigma^*\}$

8. $EQ_{CFG} = \{"A""B" : A, B \text{ are CFGs and } L(A) = L(B)\}$

9. $A_{TM} = \{"M""\omega" : M \text{ is TM that accepts } \omega\}$

Note the definition of reduction.
$A \leq B$ implies that there is a reduction $f : \Sigma^\star \to \Sigma^\star$ from A to B such that for any $x \in \Sigma^\star$, $x \in A \leftrightarrow f(x) \in B$

some theorem:
If A is a recursively enumerable language and $A \not\leq A$, then A is recursive.

Problem 4: Encoding and Halting Problem

Prove that exists undecidable subset of $\{1\}^\star$.
Solution. We can establish a map from $\{0,1\}^\star$ to $\{1\}^\star$. $\forall x \in \{0,1\}^\star$, $f(x) = bin(x)\ of\ 1's \in \{1\}^\star$
$A_{TM} = \{"M""\omega" : M \text{ is TM that accepts } \omega\}$ can also be encoded by $\{0,1\}^\star$, however it's undecidable.
So there exists undecidable subset of $\{1\}^\star$

Problem 5: Reduction and Regular

If $A \leq B$ and B is a regular language, does it imply that A is a regular language?
Solution No. Consider two languages, $A = \{a^n b^n : n \geq 0\}$ and $B = \{a\}$ with $\Sigma = \{a, b\}$. Then A is not regular but recursive while B is regular. Assume TM M_A decides A
Construct mapping from A to B, $\forall \omega \in \Sigma^\star$

$$f(\omega) = \begin{cases} a & \omega \in A \\ b & \omega \notin A \end{cases} \tag{1}$$

To show f is recursive, construct TM M as follows:

- with input ω, simulate M_A
- if M_A accepts it, output a
- else M_A rejects it, output b

A is not necessarily regular even if $A \leq B$ and B is regualr.

note on proof of reduction:

- one way, construct a TM which satisfies the sufficient and necessary transition.
- another way, find a mapping function and prove it recursive

Problem 6: Negative and Recursive

Prove: If a language L is recursively enumerable but not recursive, then $\neg L$ is not recursively enumerable.
Solution. We have the theorem L is recursive iff L and $\neg L$ is r.e.
So if L is not recursive, either L or $\neg L$ is not r.e.

Problem 7: Reduction from General-like H to Certain Problem

Prove that the following language is not recursive.
$L = \{"M_1""M_2""k" : M_1$ and M_2 are two TM with $|L(M_1) \cap L(M_2)| \geq k\}$
Solution.
Consider the non-recursive problem $A = \{"M" : M$ halts on some strings$\}$, we reduce A to L.
Construct a TM M_{all} that accepts all input. So $M \in A \leftrightarrow "M""M_{all}""1" \in L$.

Problem 8: Reduction from H to Certain Problem

Prove that the following language is not recursive, but is recursively enumerable.
$L_1 = \{"M" : M$ is a TM that halts on at least 2023 strings$\}$
Solution. **L_1 is not recursive.** Construct a reduction from H to L_1.
$f("M""\omega") = M_{"M""\omega"}$
$M_{"M""\omega"}$ run M on ω with any input u.
So there is $\forall "M""\omega", "M""\omega" \in H \leftrightarrow M_{"M""\omega"} \in L_1$. For H is not recursive, L_1 is not recursive.
L_1 is recursively enumerable.
We can easily construct a TM semi-deciding L_1 just like the construction we use to prove "a language is r.e. iff it is turing enumarable".

Problem 9: Reduction from H to Certain Problem

Prove that the following language is not recursively enumerable.
$L_2 = \{"M" : M$ is a TM that halts on at most 2022 strings$\}$
Solution 1. Use the same construction as Problem 8 and reduce the $\neg H$ to L_2.
Solution 2. Use the proved conclusion in Problem 8, so $\neg L_2$ is r.e. but not recursive. Thus, L_2 is not r.e.

Problem 10: Reduction from $\neg H$ to Certain Problem

Prove that the following language is not recursively enumerable.
$L_3 = \{"M" : M$ is a TM such that there are at least 2023 strings on which M does not halt $\}$
Solution.
Consider the non r.e. problem $\neg H = \{"M""\omega" : M$ does not halt on $\omega\}$, we reduce $\neg H$ to L_3.
Construct recursive function $M_{"M""\omega"}$. With any input u. it runs M on ω.
$"M""\omega" \in \neg H \leftrightarrow M_{"M""\omega"} \in L_3$.
Since $\neg H$ is not r.e., L_3 is not r.e.

Problem 11: Reduction from $\neg H$ to Certain Problem

Prove that the following language is not recursively enumerable.
$L_4 = \{"M" : M$ is a TM such that there are at most 2022 strings on which M does not halt $\}$
Solution.
Still consider the non r.e. problem $\neg H = \{"M""\omega" : M$ does not halt on $\omega\}$. We reduce the $\neg H$ to L_4.
Construct TM $M_{"M""\omega"}$ with any input u.

$M_{"M""\omega"}$ $= on\ input\ u :$
run M $on\ \omega$
if M $halts\ on\ \omega\ in$ u $steps,\ loop\ forever$
$else\ M\ does\ not\ halt\ on\ \omega\ in\ u\ steps,\ halt$

So if M halts on ω in n steps, $\exists u < n, st\ M_{"M""\omega"}$ loops forever. Then $M_{"M""\omega"}$ accepts no strings.
Else if M does not halt on ω, $M_{"M""\omega"}$ accepts all strings.
$"M""\omega" \in \neg H \leftrightarrow M_{"M""\omega"} \in L_4$.
Since $\neg H$ is not r.e., L_4 is not r.e.

Problem 12: Primitive Recursive

Let $f : N \to N$ be a primitive recursive function. Define $F : N \to N$ to be
$F(n) = f(f(...f(n)...))$
where there are n compositions. For example, $F(0) = f(0)$ and $F(1) = f(f(1))$. Show that F is primitive recursive.
Solution.
$g(m, n) = f(f(...f(n)...))$ where there is m compositions of f.

$$\begin{cases} g(0, n) = f(n) \\ g(m+1, n) = f(g(m.n)) \end{cases}$$

For f is primitive recursive, then g is primitive recursive.
$F(n) = g(n, n) = g(id_{1,1}(n), id_{1,1}(n))$
For g is primitive recursive, then F is primitive recursive.

Important basic functions and primitive recursive functions: $N^k \to N$

1. k-zero function: $zero_k(n_1, ..., n_k) = 0$
2. jth-k identification function: $id_{k,j}(n_1, ..., n_k) = n_j$
3. successor function: $succ(n) = n + 1$
4. plus function: $plus(m, n) = m + n$
5. mult function: $mult(m, n) = m \times n$
6. predecessor function: $pred(n + 1) = n\, pred(0) = 0$
7. substraction function: $m \sim n = max\{m - n, 0\}$
   $m \sim 0 = m$
   $m \sim (n + 1) = pred(m \sim n)$
8. iszero function: $iszero(0) = 1\, iszero(m + 1) = 0$
9. greater than or equal function: $geq(m, n) = iszero(n \sim m)$
10. less than function: $lt(m, n) = 1 \sim geq(m, n)$
11. primitive recursive predicate: and/or/not
    (a) $\neg p(m) = 1 \sim iszero(p(m))$
    (b) $p(m, n) \wedge q(m, n) = 1 \sim zero(p(m, n) + q(m, n))$
    (c) $p(m, n) \vee q(m, n) = 1 \sim iszero(p(m, n)\dot{q}(m, n))$
12. condition function:
$$f(n_1, ..., n_k) = \begin{cases} g(n_1, ..., n_k) & p(n_1, ..., n_k) \\ h(n_1, ..., n_k) & otherwise \end{cases}$$
13. remainder function: $rem(m, n)$
$$rem(0.n) = 0 rem(m + 1, n) = \begin{cases} 0 & equal(rem(m, n), pred(n)) \\ rem(m, n) + 1 & otherwise \end{cases}$$
14. division function: $div(m, n)$
$$div(0, n) = 0$$
$$div(m + 1, n) = \begin{cases} div(m, n) + 1 & equal(rem(m, n), pred(n)) \\ div(m, n) & otherwise \end{cases}$$

Problem 13: Alphabet and Language

Judge: There is no non-empty language over ϕ.
Solution. May be false. $\{e\}$ is a language over ϕ that is non-empty.

Problem 14: Undecidability and Reduction

Judge the following languages are recursive, recursively enumerated, not recursively enumerated.
$L_1 = \{"M"|M$ is a TM that accepts every palindrome over its alphabet$\}$
$L_2 = \{"M"|M$ is a TM that accepts/halts on at most k strings $\}$
$L_3 = \{"M"|M$ is a TM that accepts/halts on at least k strings $\}$
$L_4 = \{"M"|M$ is a TM that accepts/halts on exactly k strings$\}$
$L_5 = \{"M"|M$ is TM that accepts all even numbers$\}$
$L_6 = \{"M"|M$ is TM that does not accept all even numbers$\}$
$L_7 = \{"M"|M$ is TM that rejects all even numbers$\}$
$L_8 = \{"M"|M$ is TM that contains at least one sting of even number of bs $\}$
$L_9 = \{"M"|M$ is a TM and $L(M)$ is infinite$\}$
$L_{10} = \{"M_1""M_2"|M_1$ and $"M_2"$ are two TMs, and $\epsilon \in L(M_1) \cup L(M_2)\}$
$L_{11} = \{"M_1""M_2"|M_1$ and $"M_2"$ are two TMs, and $\epsilon \in L(M_1) \cap L(M_2)\}$
$L_{12} = \{"M_1""M_2"|M_1$ and $"M_2"$ are two TMs, and $\epsilon \in L(M_1) - L(M_2)\}$
$L_{13} = \{"M"|\exists x, |x| \equiv 1 \pmod{k}$, and $x \in L(M)\}$
$L_{14} = \{"M\backslash|M$ is a TM such that both $L(M)$ and $\neg L(M)$ are infinite$\}$
$L_{15} = \{"M"|M$ is a TM, and $|L(M)|$ is prime$\}$
$L_{16} = \{"M"|\exists x \in \Sigma^\star, \forall y \in L(M), xy \notin L(M)\}$
$L_{17} = \{"M_1""M_2"|L(M_1) \leq_m L(M_2)\}$
$L_{18} = \{"M""\omega"|M$ is a TM that accepts ω using at most $2^{|\omega|}$ squares of its tape$\}$
$L_{19} = \{"M_1""M_2""M_3"|L(M_1) = L(M_2) \cup L(M_3)\}$
$L_{20} = \{"M_1""M_2""M_3"|L(M_1) \subset L(M_2) \cup L(M_3)\}$
$L_{21} = \{"M_1"|\exists TM's\ M_2, M_3, st\ L(M_1) \subset L(M_2) \cup L(M_3)\}$
Solution 1. Not Recursively Enumerated. By Rice Theorem, it's easy to judge it's not recursive. To prove L_1 not recursively enumerated, we reduce the $\neg H$ to L_1.

Notice the following construction is wrong ! Because the thing we need to prove is $"M""\omega" \in \neg H \leftrightarrow \tau(M) \in L_1$.
Construct TM M', with any input u. M' runs M on ω.

$$"M""\omega" \in \neg H \rightarrow L(M') = \phi \rightarrow "M'" \notin L_1$$

$$"M""\omega" \notin \neg H \rightarrow L(M') = \Sigma^\star \rightarrow "M'" \in L_1$$

Thus, $\neg H \leq L_1$. L_1 is not recursively enumerated.
Notice, if we want inverse the conclusion. We can't say if M does not halt on ω, accept. Because, M' simulate the execution of M, if M does not halt, M' will never halt.

When we require to reduce from $\neg H$, take care of the fact that chosen M may not $halt$ on ω, so we should limit the running steps to construct the reduction.
Construct TM M' with input ω. If ω is a palindrome, run M on ω for $|\omega|$ steps. If M has not accepted, accept, else, reject.

$$"M""\omega" \in \neg H \rightarrow L(M') = \Sigma^{\star} \rightarrow "M'" \in L_1$$

$$"M""\omega" \notin \neg H \rightarrow L(M') \subset L_{palindrome} \rightarrow "M'" \notin L_1$$

Solution 2. Not Recursively Enumerated.
Prove it by reducing from the $\neg H$. Construct a TM M', with any input u, runs M with ω.

$$"M""\omega" \in \neg H \rightarrow L(M') = \phi \rightarrow "M'" \in L_2$$

$$"M""\omega" \notin \neg H \rightarrow L(M') = \Sigma^{\star} \rightarrow "M'" \notin L_2$$

Thus, $"M""\omega" \in \neg H \leftrightarrow "M'" \in L_2$
Solution 3. Recursively enumerated but not recursive.
For the proof of recursively enumerated, we can design a TM $M^{\star}$, which enumerates all strings on the alphabet and gradually run M with these strings one by one. As long as M accepts more than k strings, halts.
For the proof of not recursive, we reduce from H. Construct a TM M' with any input u, run M with ω.

$$"M""\omega" \in H \rightarrow L(M') = \Sigma^{\star} \rightarrow "M'" \in L_3$$

$$"M""\omega" \notin H \rightarrow L(M') = \phi \rightarrow "M'" \notin L_3$$

Thus, $"M""\omega" \in H \leftrightarrow "M'" \in L_3$. L_3 in not recursive.
Solution 4. Not recursively enumerable.
To prove not r.e., reduce the $\neg H$ to L_4. Construct a TM M', with any input u, if $u \neq \omega_1, ..., \omega_k$, accept, else run M with ω. ($\omega_1, ...\omega_k$ are k different strings on the alphabet)

$$"M""\omega" \in \neg H \rightarrow L(M') = \omega_1, ..., \omega_k \rightarrow |L(M')| = k \rightarrow "M'" \in L_4$$

$$"M""\omega" \notin \neq H \rightarrow L(M') = \Sigma^{\star} \rightarrow "M'" \notin L_4$$

Thus, there is $"M""\omega" \in \neq H \leftrightarrow "M'" \in L_4$. So L_4 is not recursively enumerated.
Solution 5. Not recursively enumerable.
The same, we reduce from $\neq H$.
Construct a TM M', with any input u, if u is even, run M on ω in $|u|$ steps. If M does not accept ω, accept, else, loop forever.

$$"M""\omega" \in \neq H \rightarrow L(M') = L_{even} \rightarrow "M'" \in L_5$$

$$"M""\omega" \notin \neq H \rightarrow \exists |u| > |\omega| \rightarrow \exists x, x\ is\ even, x \notin L(M') \rightarrow "M'" \notin L_5$$

Thus, there is $"M""\omega" \in \neq H \leftrightarrow "M'" \in L_5$. L_5 is not recursively enumerable.

Solution 6. Not recursively enumerated.

The same, we reduce from $\neq H$.

Construct a TM M', with any input u, if $u = \omega_0$, run M on ω in k steps, if M does not accept, reject, else, accept. Else, accept. (ω_0 is one of the even number, $k > |\omega|$)

$$"M""\omega" \in \neq H \rightarrow \exists \omega_0 \in L_{even}, \omega_0 \notin L(M') \rightarrow "M'" \notin L_6$$

$$"M""\omega" \notin \neq H \rightarrow L(M') = \Sigma^\star \rightarrow "M'" \notin L_6$$

Thus, $"M""\omega" \in \neq H \leftrightarrow "M'" \in L_6$. L_6 is not recursively enumerated.

Solution 7. Not recursively enumerated.

The same, we reduce from $\neq H$.

Construct a TM M', with any input u, if u is even, run M on ω with $k(k > |\omega|)$ steps, if does not accept, reject, else accept, else reject.

$$"M""\omega" \in \neq H \rightarrow L(M') = \phi \rightarrow "M'" \in L_7$$

$$"M""\omega" \notin \neq H \rightarrow L(M') = L_{even} \rightarrow "M'" \notin L_7$$

Thus, $"M""\omega" \in \neq H \leftrightarrow "M'" \in L_7$. L_7 is not recursively enumerated.

Solution 8. Recursively enumerated but not recursive.

For its recursively enumerated, a UTM can be constructed to enumerated all strings with even number of bs and semi-decided the language.

For its not recursive, we reduce to it from H. Construct a TM M', with any input u, if u contains even number of bs, run M on ω, else accept.

$$"M""\omega" \in H \rightarrow L(M') = \Sigma^\star \rightarrow "M'" \in L_8$$

$$"M""\omega" \notin H \rightarrow L(M') = \phi \rightarrow "M'" \notin L_8$$

Thus,

$$"M""\omega" \in H \leftrightarrow "M'" \in L_8$$

. L_8 is not recursive.

Solution 9. Not recursively enumerated.

The same, reduce from $\neq H$. Construct a TM M' with any input u, run M on ω.

$$"M""\omega" \in \neq H \rightarrow L(M') = \phi \rightarrow "M'" \in L_9$$

$$"M""\omega" \notin \neq H \rightarrow L(M') = \Sigma^\star \rightarrow "M'" \notin L_9$$

Thus, there is $"M""\omega" \in \neq H \leftrightarrow "M'" \in L_9$. L_9 is not recursively enumerated.

Solution 10. Recursively enumerated but not recursive.

For L_{10} is recursively enumerated, a UTM can be designed to semi-decide this.

For L_{10} is not recursive, we reduce the H to it. Construct a TM M' with any input u, if $u = \epsilon$, run M with ω, else accept.

$$"M""\omega" \in H \rightarrow \epsilon \in L(M') \rightarrow "M'""M'" \in L_{11}$$

$$"M""\omega" \notin H \rightarrow \epsilon \notin L(M') \rightarrow "M'""M'" \notin L_{11}$$

Thus, $"M""\omega" \in H \leftrightarrow "M'""M'" \in L_{10}$. L_{10} is not recursive.

Solution 11. Recursively enumerated but not recursive.

The proof is the same as L_10.

Solution 12. Not recursively enumerated.

As the method to prove not r.e. before, we reduce from $\neq H$ to L_12. Assume M_{all} is the TM to accept all strings($L(M_{all} = \Sigma^\star$), construct a TM M' with input u. If $u = \epsilon$, run M with ω, else accept.

$$"M""\omega" \in\neq H \rightarrow \epsilon \notin L(M') \rightarrow \epsilon \in L(M_{all}) - L(M') \rightarrow "M_{all}""M'" \in L_{12}$$

$$"M""\omega" \notin\neq H \rightarrow \epsilon \in L(M') \rightarrow \epsilon \notin L(M_{all}) - L(M') \rightarrow "M_{all}""M'" \notin L_{12}$$

Thus, there is $"M""\omega" \in\neq H \leftrightarrow "M_{all}""M'" \in L_{12}$. L_{12} is not recursively enumerable.

Solution 13. Recursively enumerated but not recursive.

For L_{13} is recursively enumerated, it can be semi-decided by some TM.

For L_{13} is not recursive. Use Rice's theorem.

$$C = \{L \in r.e.|\exists x, |x| \equiv 1 \pmod 5 \; and \; x \in L\}$$

Notice that $C \subset r.e.$, $C \neq \phi$ and $C \neq r.e.$

Hence, $L_C = \{"M"|L(M) \in C\} = \{"M"|\exists x, |x| \equiv 1 \pmod 5, x \in L(M)\} \notin R$

Solution 14. Not recursively enumerated.

Reduce to L_{14} from $\neg H$. Construct a TM M', with any input u, if $|u|$ is even, run M on ω, else accept.

$$"M""\omega" \in \neg H \rightarrow L(M') = strings\, of\, even\, length, \neg L(M') = strings\, of\, odd\, length \rightarrow "M'" \in L_{14}$$

$$"M""\omega" \notin \neg H \rightarrow L(M') = \Sigma^\star, \neg L(M') = \phi \rightarrow "M'" \notin L_{14}$$

Thus, there is $"M""\omega" \in \neg H \leftrightarrow "M'" \in L_{14}$. L_{14} is not recursively enumerated.

Solution 15. Not recursively enumerated.

Reduce to L_{15} from $\neg H$. Construct a TM M', with any input u, if u is the 1st and 2nd strings of all the strings over the alphabet in lexicographic order, run M with ω, else if u is the 3rd and 4th strings of it, accept, else, reject.

$$"M""\omega" \in \neg H \rightarrow |L(M')| = 2 \rightarrow "M'" \in L_{15}$$

$$"M""\omega" \notin \neg H \rightarrow |L(M')| = 4 \rightarrow "M'" \notin L_{15}$$

Thus, there is $"M""\omega" \in \neg H \leftrightarrow "M'" \in L_{15}$. L_{15} is not recursively enumerated.
Solution 16. Not recursively enumerated.
Reduce to L_{16} from $\neg H$. Construct a TM M' with any input u, run M on ω.

$$"M""\omega" \in \neg H \to L(M') = \phi \to "M'" \in L_{16}$$

$$"M""\omega" \notin \neg H \to L(M') = \Sigma^\star \to "M'" \notin L_{16}$$

Thus, there is $"M""\omega" \in \neg H \leftrightarrow "M'" \in L_{16}$. L_{16} is not recursively enumerated.
Solution 17. Not recursively enumerated.
Reduce to L_{17} from $\neg H$. Assume M_0 is the TM that $L(M_0) = \phi$, construct a TM M_1 with any input u, run M on ω. If M halts on ω, run M' on u.

$$"M""\omega" \in \neg H \to L(M_1) = \phi \to "M_1""M_0" \in L_17(\phi \leq_m \phi)$$

$$"M""\omega" \notin \neg H \to L(M_1) = H \to "M_1""M_0" \notin L_{17}$$

Thus, there is $"M""\omega" \in \neg H \leftrightarrow "M_1""M_0" \in L_{17}$. L_{17} is not recursively enumerated.
Solution 18. Recursive.
Construct TM to decide this, for the maximum execution configurations is limited.
Solution 19. Not recursively enumerated.
Reduce to L_{19} from $EQ_{TM} = \{"M_1""M_2" | L(M_1) = L(M_2)\}$. Take M_3 as $L(M_3) = \phi$. Since EQ_{TM} is not recursively enumerated, L_{19} is also not recursively enumerated.
Solution 20. Not recursively enumerated.
Reduce to L_{20} from $\neg H$. Let M_2 and M_3 be TMs with $L(M_2) = L(M_3) = \phi$. Construct TM M_1, with any input u, run M with ω.

$$"M""\omega" \in \neg H \to L(M_1) = \phi \to L(M_1) \subset L(M_2) \cup L(M_3) \to "M_1""M_2""M_3" \in L_{20}$$

$$"M""\omega" \notin \neg H \to L(M_1) = \Sigma^\star \to L(M_1) \not\subset L(M_2) \cup L(M_3) \to "M_1""M_2""M_3" \notin L_{20}$$

Thus, there is $"M""\omega" \in \neg H \leftrightarrow "M_1""M_2""M_3" \in L_{20}$. L_{20} is not recursively enumerated.
Solution 21. Recursive.
All TMs satisfy this property.

Conclusion: Some Universal Problems of Undecidability

For DFA:

1. $A_{DFA} = \{"D""\omega" | D \text{ is a DFA and } D \text{ accepts } \omega\}$
   Recursive

2. $\neg A_{DFA} = \{"D""\omega"|D$ is a DFA and D deos not accepts $\omega\}$
   Recursive

3. $E_{DFA} = \{"D"|D$ is a DFA and $L(D) = \phi\}$
   Recursive

4. $\neg E_{DFA} = \{"D"|D$ is a DFA and $L(D) \neq \phi\}$
   Recursive

5. $EQ_{DFA} = \{"D_1""D_2"|D_1, D_2$ are DFAs and $L(D_1) = L(D_2)\}$
   Recursive

6. $\neg EQ_{DFA} = \{"D_1""D_2"|D_1, D_2$ are DFAs and $L(D_1) \neq L(D_2)\}$
   Recursive

7. $ALL_{DFA} = \{"D"|D$ is a DFA and $L(D) = \Sigma^\star\}$
   Recursive

8. $\neg ALL_{DFA} = \{"D"|D$ is a DFA and $L(D) \neq \Sigma^\star\}$
   Recursive

For NFA:
Since NFA can be converted to DFAs, all the problems are the same as DFAs.
For CFG:

1. $A_{CFG} = \{"G""\omega"|G$ is CFG and G accepts $\omega\}$
   Recursive

2. $\neq A_{CFG} = \{"G""\omega"|G$ is CFG and G does not accepts $\omega\}$
   Recursive

3. $E_{CFG} = \{"G"|G$ is CFG and $L(G) = \phi\}$
   Recursive

4. $\neg E_{CFG} = \{"G"|G$ is CFG and $L(G) \neq \phi\}$
   Recursive

5. $EQ_{CFG} = \{"G_1""G_2"|G_1, G_2$ are CFGs and $L(G_1) = L(G_2)\}$
   Not Recursively Enumerated

6. $\neg EQ_{CFG} = \{"G_1""G_2"|G_1, G_2$ are CFGs and $L(G_1) \neq L(G_2)\}$
   Recursively Enumerated but Not Recursive

7. $ALL_{CFG} = \{"G"|G$ is CFG and $L(G) = \Sigma^\star\}$
   Not Recursively Enumerated

8. $\neg ALL_{CFG} = \{"G"|G$ is CFG and $L(G) \neq \Sigma^\star\}$
   Recursively Enumerated but Not Recursive

For TM:

1. $A_{TM} = \{"M""\omega" | M$ is a TM and M accepts $\omega\}$
   Recursively Enumerated but Not Recursive

2. $\neg A_{TM} = \{"M""\omega" | M$ is a TM and M does not accept $\omega\}$
   Not Recursively Enumerated

3. $E_{TM} = \{"M" | M$ is a TM and $L(M) = \phi\}$
   Not Recursively Enumerated

4. $\neg E_{TM} = \{"M" | M$ is a TM and $L(M) \neq \phi\}$
   Recursively Enumerated but not Recursive

5. $EQ_{TM} = \{"M_1""M_2" | M_1, M_2$ are TMs and $L(M_1) = L(M_2)\}$
   Not Recursively Enumerated

6. $\neg EQ_{TM} = \{"M_1""M_2" | M_1, M_2$ are TMs and $L(M_1) \neq L(M_2)\}$
   Not Recursively Enumerated

7. $ALL_{TM} = \{"M" | M$ is a TM and $L(M) = \Sigma^\star\}$
   Not Recursively Enumerated

8. $\neg ALL_{TM} = \{"M" | M$ is a TM and $L(M) \neq \Sigma^\star\}$
   Not Recursively Enumerated

Problem 15: CFL, CFG and Pumping Theorem

Judge whether the following languages are CFLs.

	Recursive	R.E.	Not R.E.	Not Co-R.E.
A_DFA	x			
~A_DFA	x			
E_DFA	x			
~E_DFA	x			
EQ_DFA	x			
~EQ_DFA	x			
ALL_DFA	x			
~ALL_DFA	x			
A_CFG	x			
~A_CFG	x			
E_CFG	x			
~E_CFG	x			
EQ_CFG			x	
~EQ_CFG		x		
ALL_CFG			x	
~ALL_CFG		x		
A_TM		x		
~A_TM			x	
E_TM			x	
~E_TM		x		
EQ_TM				x
~EQ_TM				x
ALL_TM				x
~ALL_TM				x