non-computability

Tuesday, December 10, 2024 11:46 AM

Prove non-computability:
- ✓ I. halting problem as a counter example.
- ✓ II. show ∃ more languages than tms.
- ✓ III. show relationship between recursive - recursively enumerable and their complement

II finish proof.

proof:

1. show the set of all tms (under some Σ) is denumerable.
   how: * represent every tm as a binary string.
   - order them by value
   - count ie, put into 1-1 correspondence w/ Natural #s.

   ∴ denumerable.

2. show the set of all languages is not denumerable.
   how: • look @ Σ (eg. Σ = {a,b})
   ⇒ always finite.
   - look @ Σ* (eg. Σ* = {λ, a, aa, ab, ba, bb, ..}
     show denumerable by putting into shortest order and count.
   - power set of a denumerable set is not denumerable eg.

     $P_1 = \{a, aa, aaa, \dots\}$

     $P_2 = \{b, bb, bbb, \dots\}$

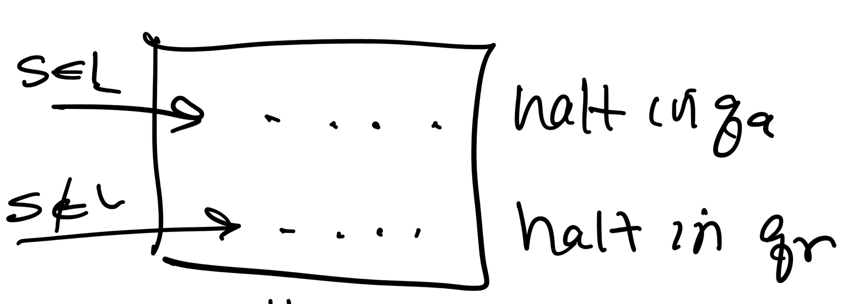
     $P_3 = \{abb, abbb, abbbb, \dots\}$

   look @ P_i = a language.
   Power set of Σ* is the set of all languages (under some Σ)
   ∴ set of all languages is not denumerable
3. ∴ since set of all languages is not denumerable and set of all tms is denumerable ⇒ more languages than tms.
   ⇒ ∃ languages w/out a corresponding tm.
4. turing's stro thesis says anything that is computable can be computed by a tm.
5. ∴ ∃ some languages that are not computable

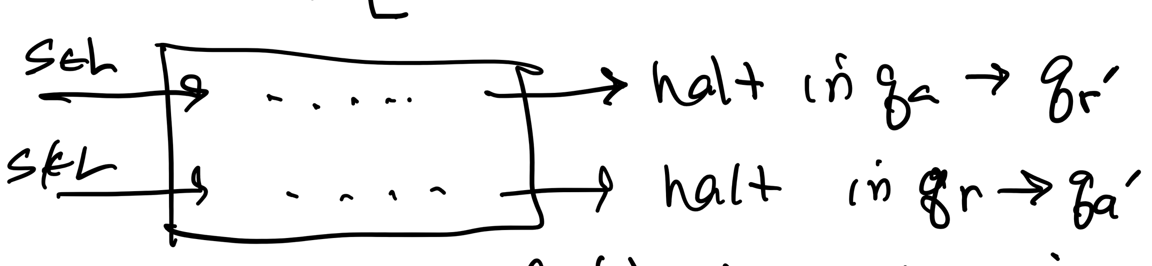
III show that ∃ languages that are not recursively enumerable.

1. recursive languages (tm is called decider)
   s∈L, M_{tm}(L) halt in q_a
   s∉L, M_{tm}(L) halt in q_r.
2. recursively enumerable languages (tm is called a recognizer)
   s∈L, M_{tm}(L) halt in q_a
   s∉L, M_{tm}(L) 1. halt in q_r or 2. not halt
3. recursive ≠ r.e.
   recursive ⊂ r.e.
   r.e. ⊄ recursive
4. complement of a recursive language is recursive.
   if L is recursive ⇒ L′ is also recursive

L M(L) is a decider



⇓ L′

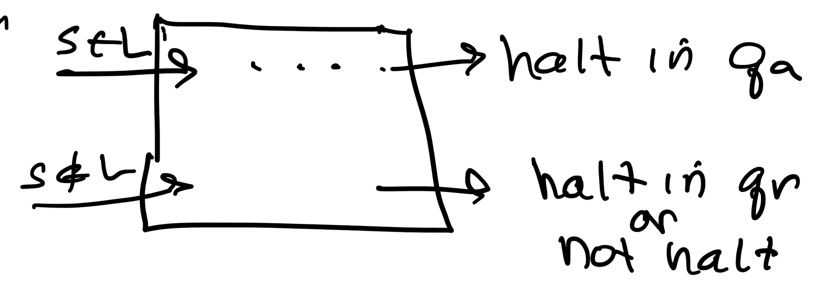


M(L′) is a decider

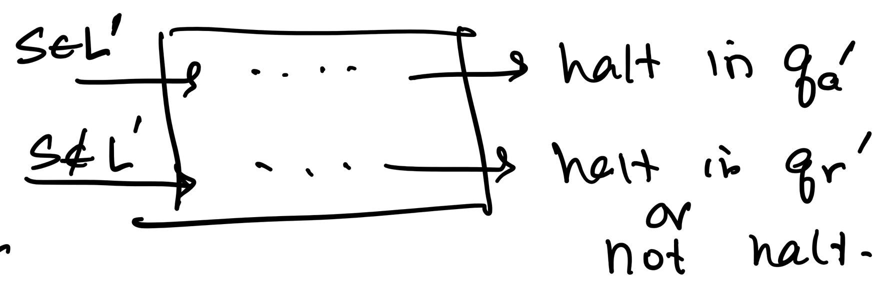
⇒ L′ is a recursive Language.

5. if L is r.e. and L′ is r.e.
   ⇒ both are recursive

L is r.e. ⇒ M(L) is a recognizer



L′ is r.e. ⇒ M(L′) is a recognizer



⇒ in all cases the "M" is going to halt. ∴ "M" is a decider for both L and L′
⇒ L, L′ are both recursive

6.

L	L′	both
Recursive	recursive	⇒ recursive
~~r.e.~~	~~r.e.~~	⇒ recursive
~~recursive~~ ⊂ r.e.	~~r.e.~~	⇒ recursive
~~r.e.~~	~~recursive~~ ⊂ r.e.	⇒ recursive
r.e.	neither recursive nor r.e.	

does not have an associated TM
∴ not computable