1. In order to show that the language T is undecidable we will utilize a proof by contradiction which supposes that T is decidable. If T is decidable then there is a Turing Machine $TM_T$ which decides it. We will show that if this supposition is true, then we can construct a Turing Machine $TM_S$ which decides the language $A_{TM}$. That of course will be the contradiction showing that $TM_T$ cannot exist, and therefore T is an undecidable language.

   Proof by contradiction:

   Assume for some contradiction's sake that the language T is decidable.

   Then Turing Machine $TM_T$ is the Turing Machine which decides it

   We will show how to construct a Turing Machine $TM_S$ which decides the language $A_{TM}$ using $TM_T$.

   Let T' be the complement language of T with the addition of also accepting input if the input M also accepts the null language. We can presume that $TM_T$ would have rejected this input, so by our construction below of $TM_{T'}$, the Turing Machine which decides T' we will show that the Turing Machine which accepts the null language will be decided and accepted by $TM_{T'}$.

   $TM_S =$ "On input $< M >$

   1. Construct a different Turing Machine $TM_{T'}$ from $TM_T$ which takes in an input $< M >$ It is identical to $TM_T$ but with one exception: For every accept state in $TM_T$, $TM_{T'}$ has a reject state, and for every reject state in $TM_T$, $TM_{T'}$ has an accept state.

   2. Simulate $TM_T$ on input $< M >$

   2. Simulate $TM_{T'}$ on input $< M >$

   3. If either $TM_T$ or $TM_{T'}$ accepts then $TM_S$ accepts, otherwise $TM_S$ rejects."

   In this way the problem of deciding $A_{TM}$ reduces to the problem of deciding the language T. Since $TM_T$ exists then for all possible Turing Machines exactly one of either $TM_T$ or $TM_{T'}$ (which functions as the decider for the complement of T) will accept. We can be sure we properly constructed a Turing Machine which decides $A_{TM}$ because the languages T and T' are mutually exclusive and exhaustive. For every Turing Machine including the one that accepts the null language if $TM_T$ does not accept, then $TM_{T'}$ will. Therefore our Turing Machine $TM_S$ decides the language of all turing machines. There in lies our contradiction because we know this language is not decidable so we have proved that LTmust be undecidable.

2. In order to show that the language L is undecidable we will utilize a proof by contradiction which supposes that L is decidable. If L is decidable then there is a Turing Machine $TM_L$ which decides it. We will show that if this supposition is true, then we can construct a Turing Machine $TM_S$ which decides the language $E_{TM}$. That of course will be the contradiction showing that $TM_L$ cannot exist, and therefore L is an undecidable language.

   Proof by contradiction:

Assume for some contradiction's sake that the language L is decidable.

Then Turing Machine $TM_L$ is the Turing Machine which decides it

We will show how to construct a Turing Machine $TM_S$ which decides the language $E_{TM}$ using $TM_L$.

$TM_S$ = "On input $< M >$

1. Construct a different Turing Machine $TM_{L'}$ from $TM_L$ which takes in an input $< M' >$ It is identical to $TM_L$ but with two exceptions. One: that it writes some character not in the tape language of $TM_L$ instead of the blank space. Two: instead of accepting or rejecting right away when $TM_L$ does, it reads every character printed on its tape. If the tape is blank then accept, if the tape has any characters written on it, then reject.
2. Simulate $TM_{L'}$ on input $< M >$
3. If $TM_{L'}$ accepts then $TM_S$ accepts, otherwise $TM_S$ rejects."

In this way the problem of deciding $E_{TM}$ reduces to the problem of deciding the language L. if $TM_L$ exists then for all possible Turing Machines $TM_{L'}$ can decide if that Turing Machine accepts the empty language. There in lies our contradiction so we have proved that L must be undecidable.

---

3. Let language L = { $< M, w >$ | M is a Turing Machine which ever tries to move its head left when the head is already on the leftmost part of the tape during its computation on w.}

   In order to show that the language L is undecidable we will utilize a proof by contradiction which supposes that L is decidable. If L is decidable then there is a Turing Machine $TM_L$ which decides it. We will show that if this supposition is true, then we can construct a Turing Machine $TM_S$ which decides the language $HALT_{TM}$. That of course will be the contradiction showing that $TM_L$ cannot exist, and therefore L is an undecidable language.

   Proof by contradiction:

   Assume for some contradiction's sake that the language L is decidable.

   Then Turing Machine $TM_L$ is the Turing Machine which decides it

   We will show how to construct a Turing Machine $TM_S$ which decides the language $HALT_{TM}$ using $TM_L$.

   $TM_S$ = "On input $< M, w >$

   1. Simulate $TM_L$ on input $< M, w >$
   2. If $TM_L$ accepts or rejects then $TM_S$ accepts, otherwise $TM_S$ rejects."

   In this way the problem of deciding $HALT_{TM}$ reduces to the problem of deciding the language L. if $TM_L$ exists then for all possible Turing Machines $TM_L$ can decide if the

Turing Machine passed into it as input ever moves left when on the leftmost part of the tape. In order to decide this as it simulates M, M must be able to finish computing the entirety of its input w. We know that $TM_L$, being a decider, must be able to decide for any input $< M, w >$. If it can decide for any input, then it can decide for the same input that is passed into $TM_S$. As a result we can deduce there must not be any combination of Turing Machines and input words which will ever fail to halt. That is, for every possible Turing Machine, $TM_L$ will not halt when deciding if its head ever moves left when on the leftmost part of the tape. Then clearly whatever Turing Machine passed into and being used by $TM_L$ must also also halt on input w. So then every Turing Machine must Halt on every input. But we know there are Turing Machines which on certain inputs do not halt. So there in lies our contradiction and we have proved that L must be undecidable.

---

4. Let language L = $\{ < M, w > \mid$ M is a Turing Machine which ever tries to move its head left at any point during its computation on w.$\}$

   In order to show that the language L is decidable it is sufficient to show that we can construct a turing machine which decides it.

   WE will utilize a proof by construction to show this. Let $TM_L$ be the Turing Machine which decides L.

   $TM_L =$ "On input $< M, w >$

   1. Construct a different Turing Machine $TM_{M'}$ from $M$ which takes in an input $< M', w' >$ It is identical to $M$ but with two exceptions. One: Every time there is a clause in M's $\delta$ function which tells M's head to move to the left we replace that clause in $TM_{M'}$'s $\delta$ function to immediately accept. Two: for every accept state in $M$ switch it to a reject state in $TM_{M'}$, but all of the reject states in $M$ remain identical in $TM_{M'}$.

   2. Simulate $TM_{M'}$ on input $< M, w >$

   3. If $TM_{M'}$ accepts then $TM_L$ accepts, otherwise $TM_L$ rejects."

   We can be sure that $TM_L$ decides the language L because if $TM_{M'}$ accepts then we know M would have attempted to move its head left at least once when computing the same word w. This does not violate Rice's principle because the feature we are deciding is that of a Turing Machine itself and not of language it recognizes.