CSE211-Formal Languages and Automata Theory

U4L7_Reducibility and Rice's Theorem

Dr. P. Saravanan
School of Computing
SASTRA Deemed University

What is Reducibility?

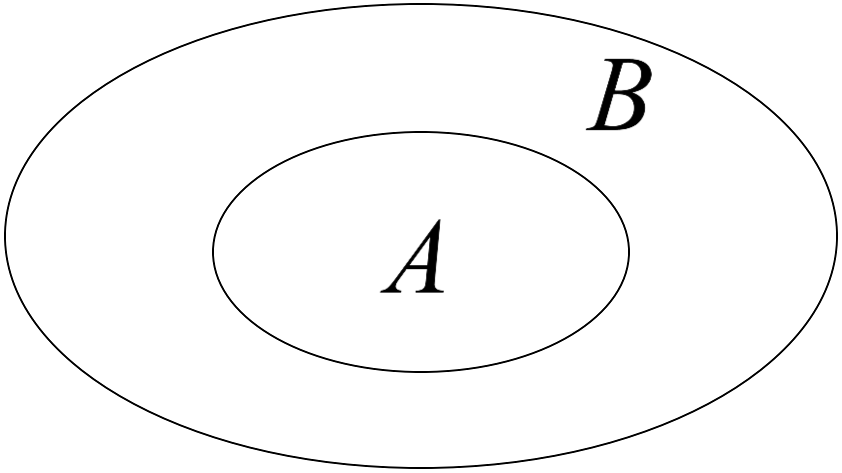
- A reduction is a way of converting one problem to another such that the solution to the second can be used to solve the first
  - We say that problem A is reducible to problem B
  - Example: finding your way around City is reducible to the problem of finding and reading a map
  - If A reduces to B, what can we say about the relative difficulty of problem A and B?
    - A can be no harder than B since the solution to B solves A
    - A could be easier
    - In example above, A is easier than B since B can solve any routing problem

Practice on Reducibility

- In our previous class work, did we reduce NFAs to DFAs or DFAs to NFAs?
  - We reduced NFAs to DFAs
    - We showed that an NFA can be reduced (i.e., converted) to a DFA via a set of simple steps
    - an DFA is a degenerate form of an NFA, we showed they have the same expressive power

Problem A is reduced to problem B

If we can solve problem B then we can solve problem A



Problem A is reduced to problem B

$\Downarrow$

If B is decidable then A is decidable

$\Downarrow$

If A is undecidable then B is undecidable

Example: the halting problem

is reduced to

the state-entry problem

The state-entry problem

Inputs:

- Turing Machine M
- State q
- String w

Question: Does M enter state q on input w ?

Theorem:

The state-entry problem is undecidable

Proof:

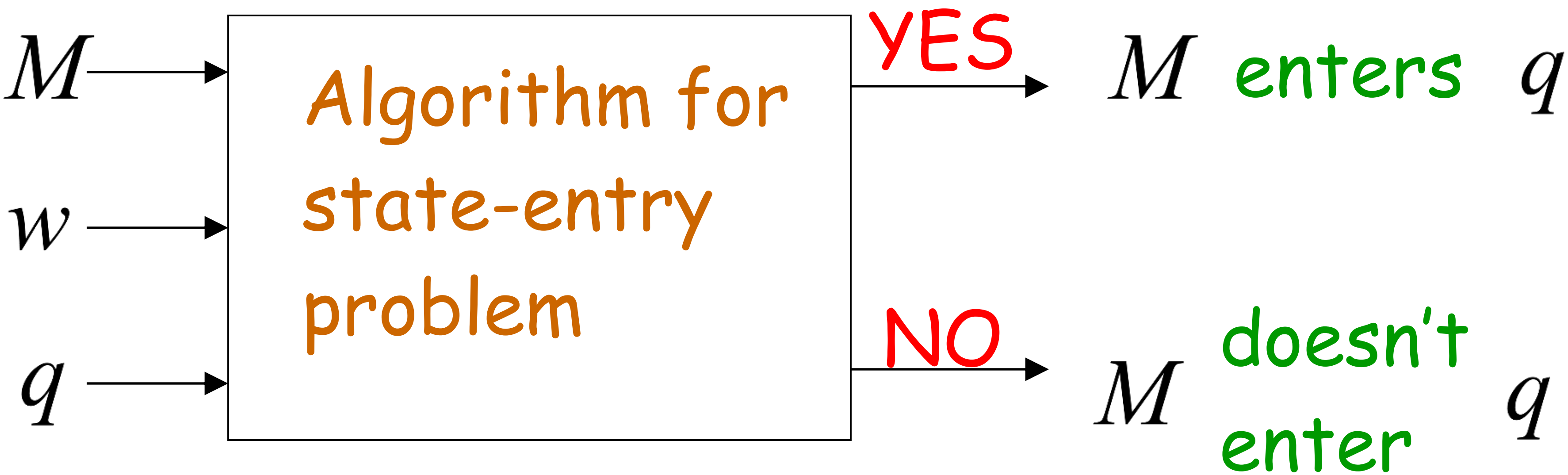
Reduce the halting problem to

the state-entry problem

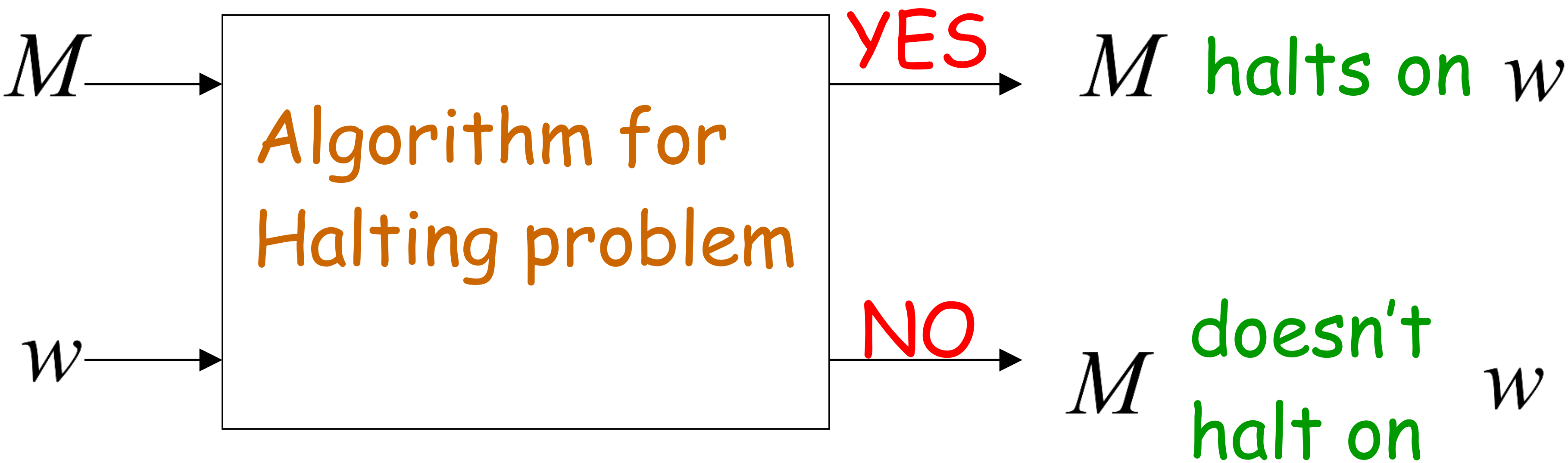
Suppose we have an algorithm (Turing Machine) that solves the state-entry problem

We will construct an algorithm that solves the halting problem

Assume we have the state-entry algorithm:

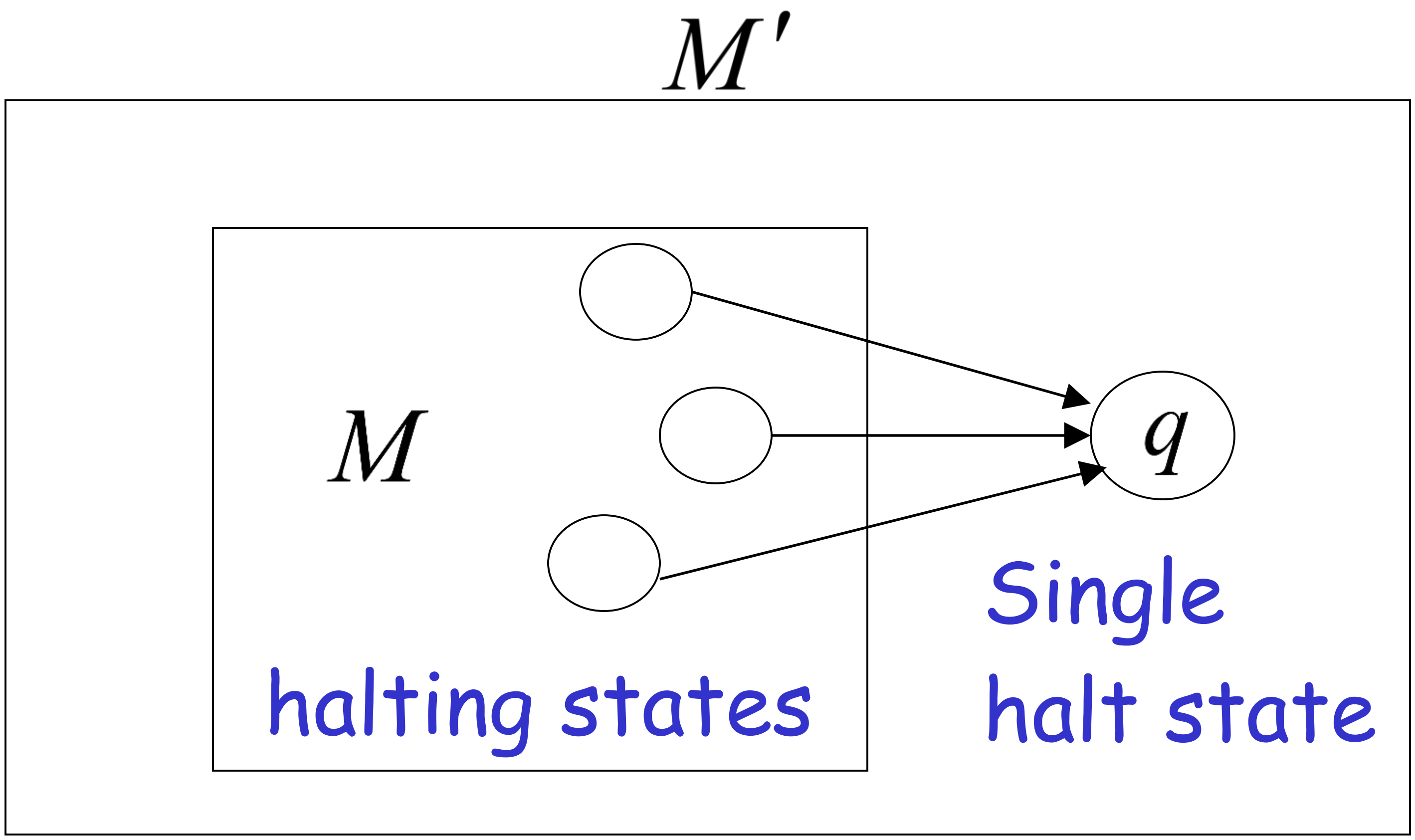


We want to design the halting algorithm:



Modify input machine M :

- Add new state q
- From any halting state add transitions to q



M halts

if and only if

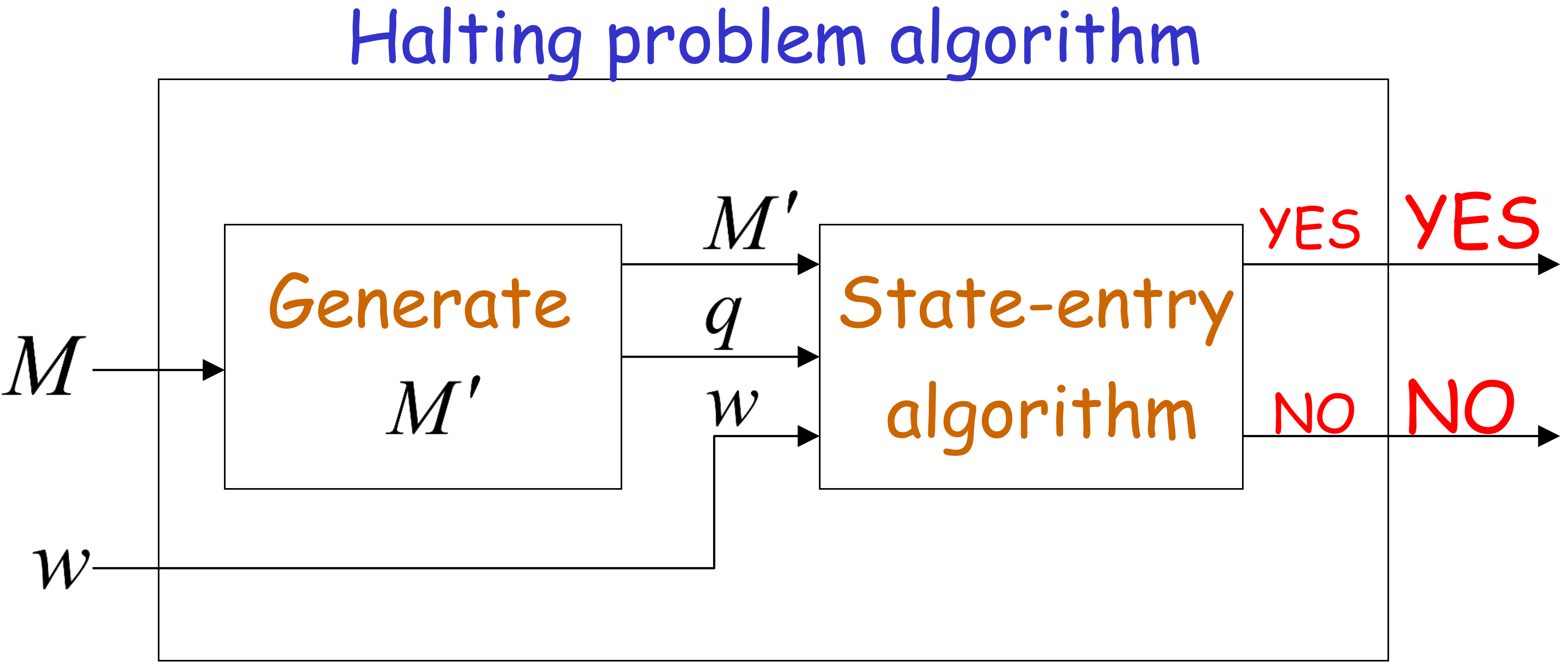
M' halts on state q

Algorithm for halting problem:

Inputs: machine M and string w

1. Construct machine M' with state q

2. Run algorithm for state-entry problem with inputs: M' , q , w


Halting problem algorithm
M
w
Generate
M'
M'
q
w
State-entry
algorithm
YES
NO
YES
NO

We reduced the halting problem
to the state-entry problem

Since the halting problem is undecidable,
it must be that the state-entry problem
is also undecidable

END OF PROOF

Another example:

the halting problem

is reduced to

the blank-tape halting problem

The blank-tape halting problem

Input: Turing Machine M

Question: Does M halt when started with a blank tape?

Theorem:

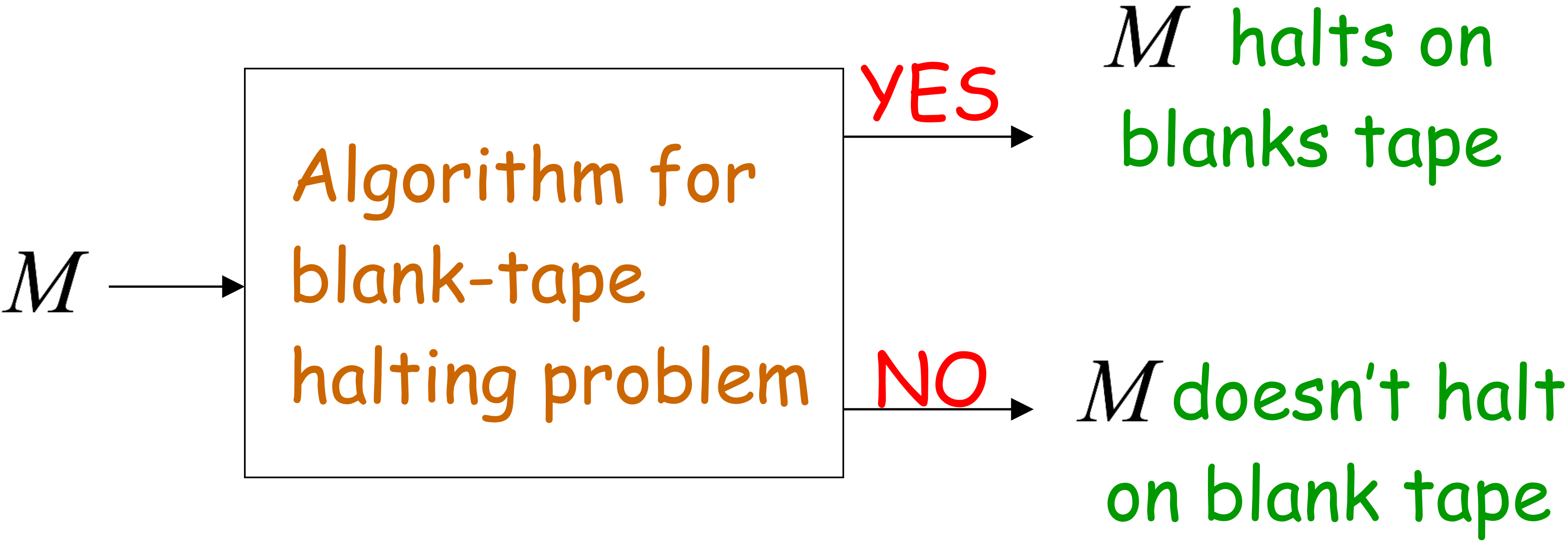
The blank-tape halting problem is undecidable

Proof: Reduce the halting problem to the blank-tape halting problem

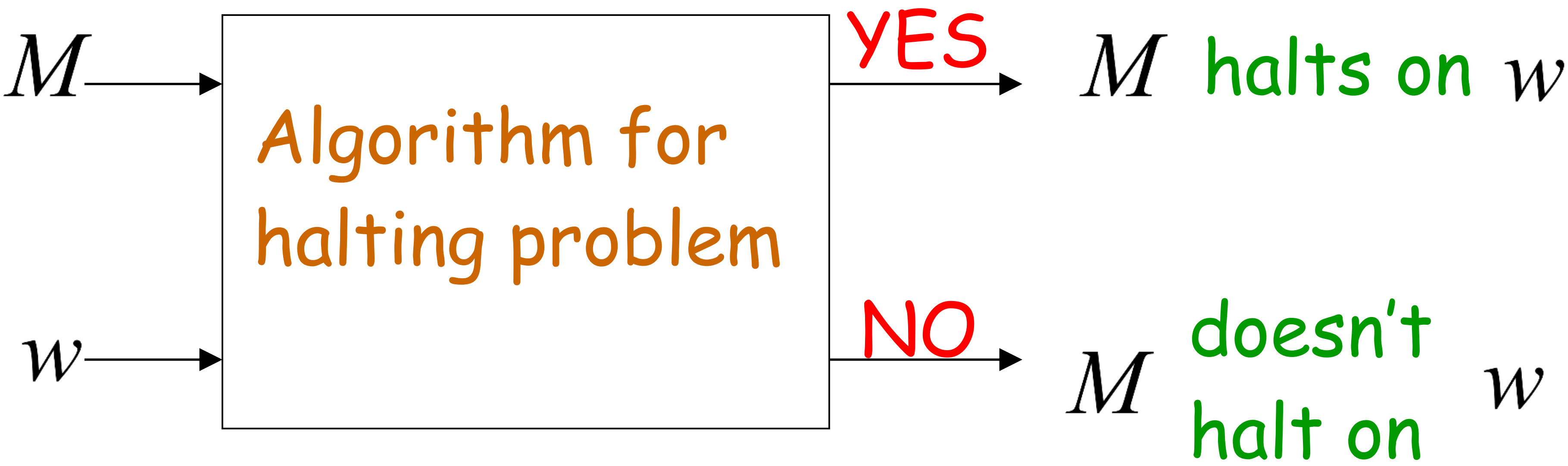
Suppose we have an algorithm
for the blank-tape halting problem

We will construct an algorithm
for the halting problem

Assume we have the
blank-tape halting algorithm:



We want to design the halting algorithm:



Construct a new machine M_w

- On blank tape writes w

- Then continues execution like M

M_w

step 1

if blank tape

then write w

step2

execute M

with input w

M halts on input string w

if and only if

M_w halts when started with blank tape

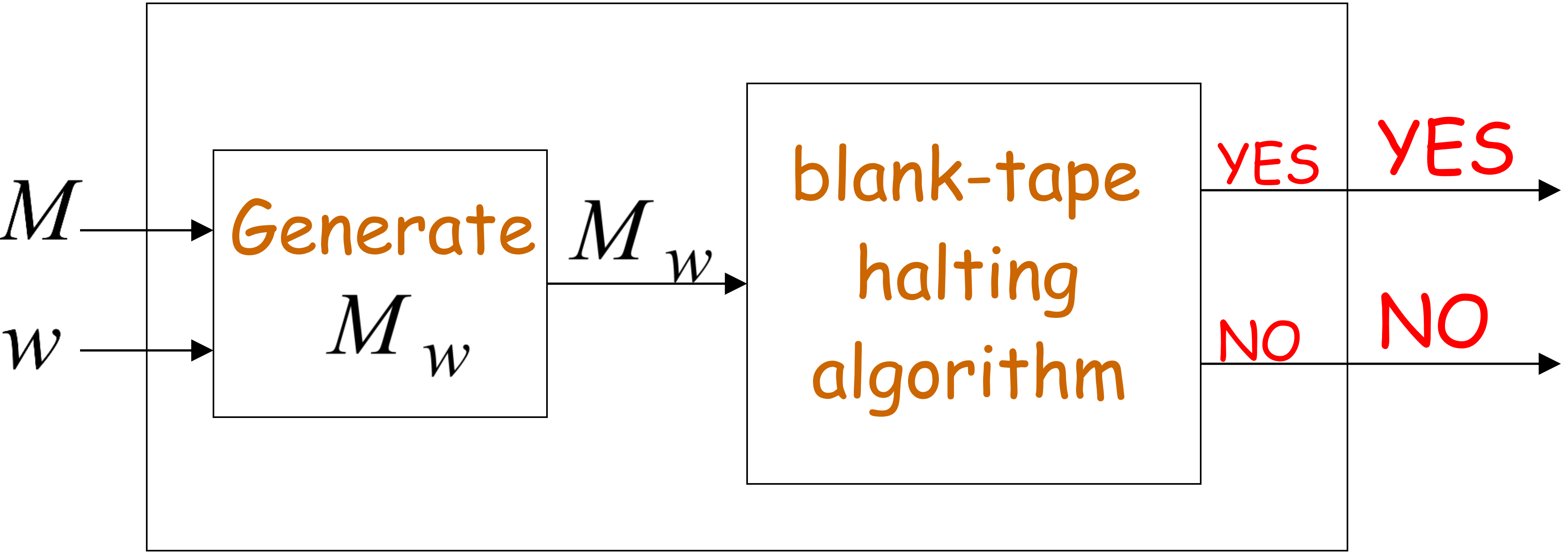
Algorithm for halting problem:

Inputs: machine M and string w

1. Construct M_w

2. Run algorithm for blank-tape halting problem with input M_w

Halting problem algorithm



We reduced the halting problem
to the blank-tape halting problem

Since the halting problem is undecidable,
the blank-tape halting problem is
also undecidable

END OF PROOF

Summary of Undecidable Problems

Halting Problem:

Does machine M halt on input w?

Membership problem:

Does machine M accept string w?

In other words: Is a string w member of a recursively enumerable language L?

Blank-tape halting problem:

Does machine M halt when starting on blank tape?

State-entry Problem:

Does machine M enter state q on input w ?

References

John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Pearson, 3rd Edition, 2011.

Peter Linz, *An Introduction to Formal Languages and Automata*, Jones and Bartle Learning International, United Kingdom, 6th Edition, 2016.

Next Class: Unit IV

Uncomputable Problems and Rice Theorem

Thank you.