Decidability

continued…

Theorem:

For a recursively enumerable language L it is undecidable to determine whether L is finite

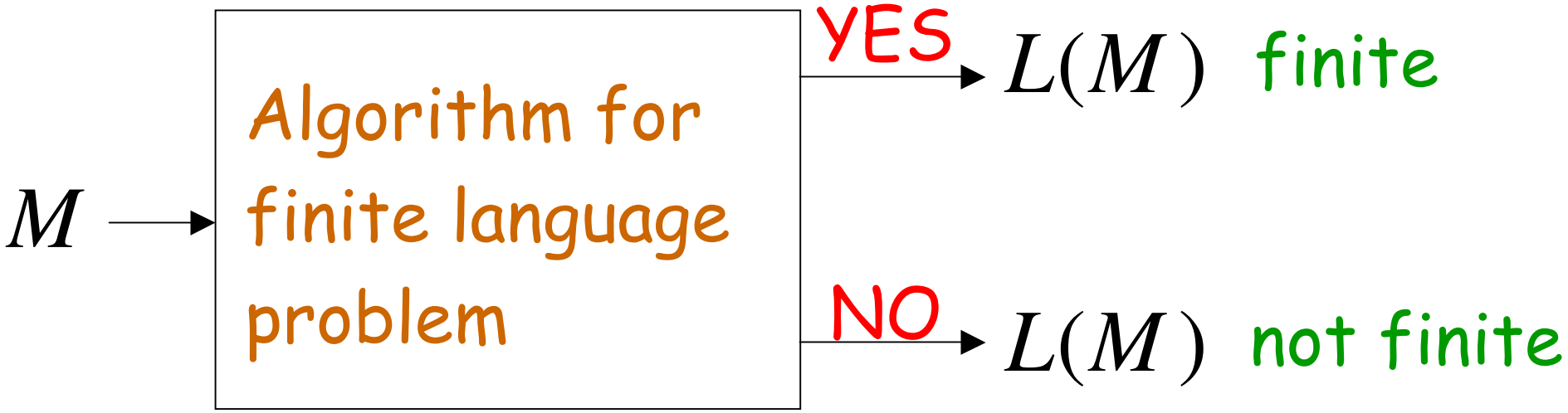
Proof:

We will reduce the halting problem to this problem

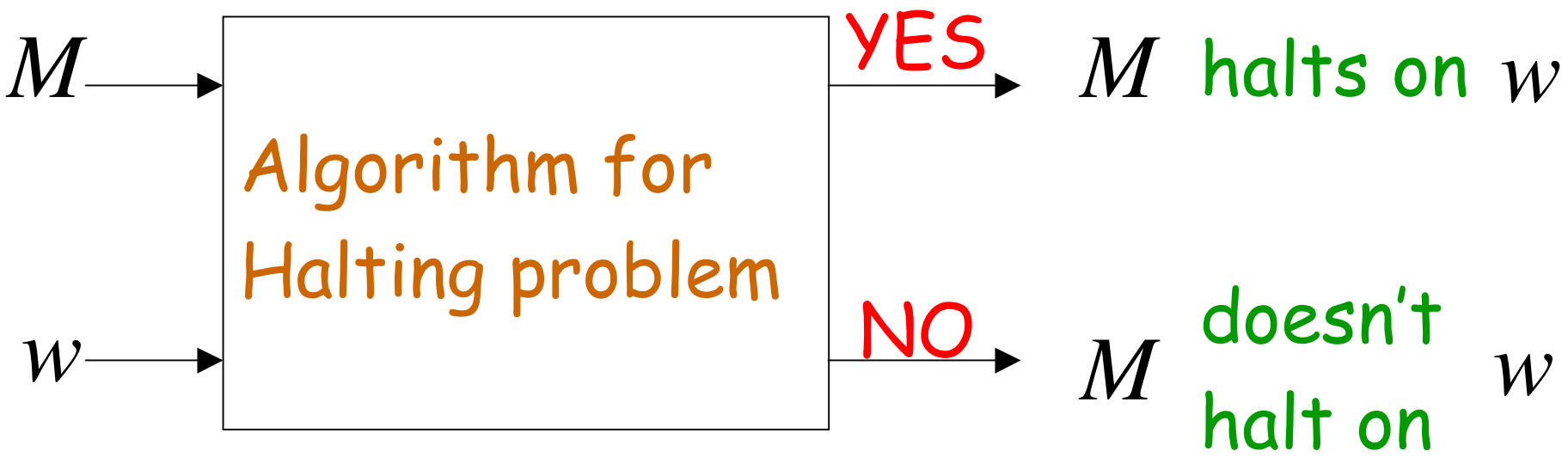
Let M be the machine that accepts L

$$L(M) = L$$

Assume we have the finite language algorithm:



We will design the halting problem algorithm:



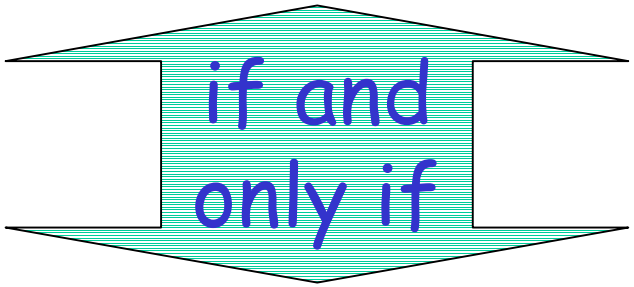
First construct machine M_w:

Initially, simulates M on input w

When M enters a halt state,
accept any input (inifinite language)

Otherwise accept nothing (finite language)

M halts on w



$L(M_w)$ is not finite

Algorithm for halting problem:

Inputs: machine M and string w

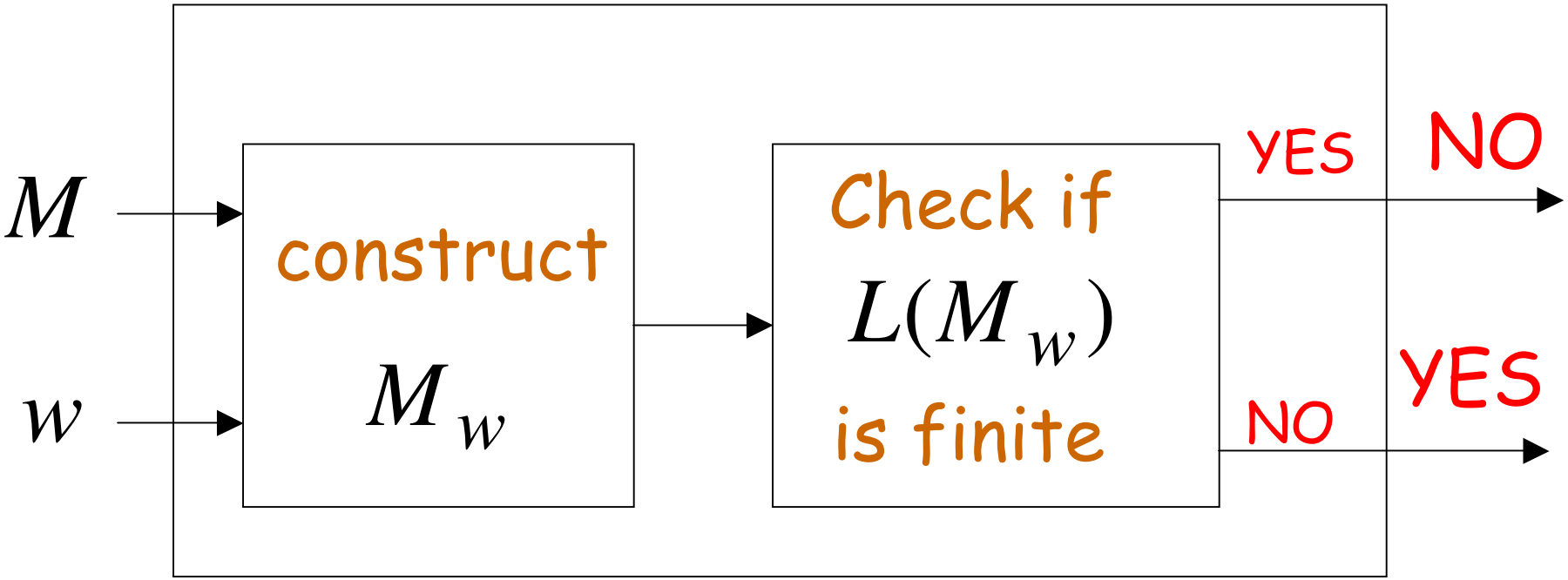
1. Construct M_w

2. Determine if $L(M_w)$ is finite

   YES: then M doesn't halt on w

   NO: then M halts on w

Machine for halting problem



Theorem:

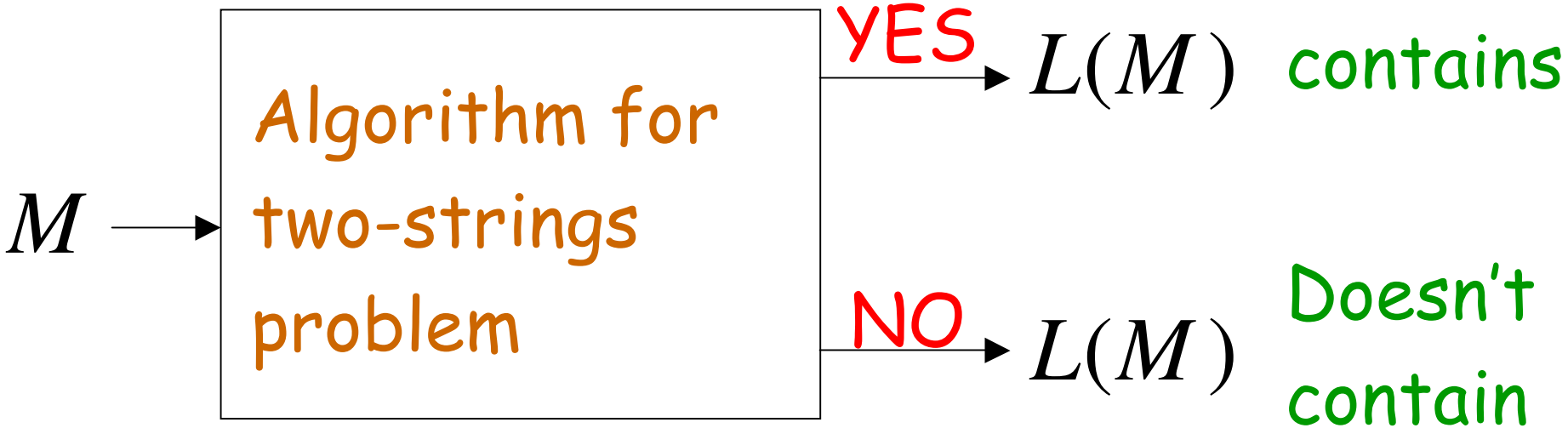
For a recursively enumerable language L
it is undecidable to determine whether
L contains two different strings of same length

Proof: We will reduce the halting problem
to this problem

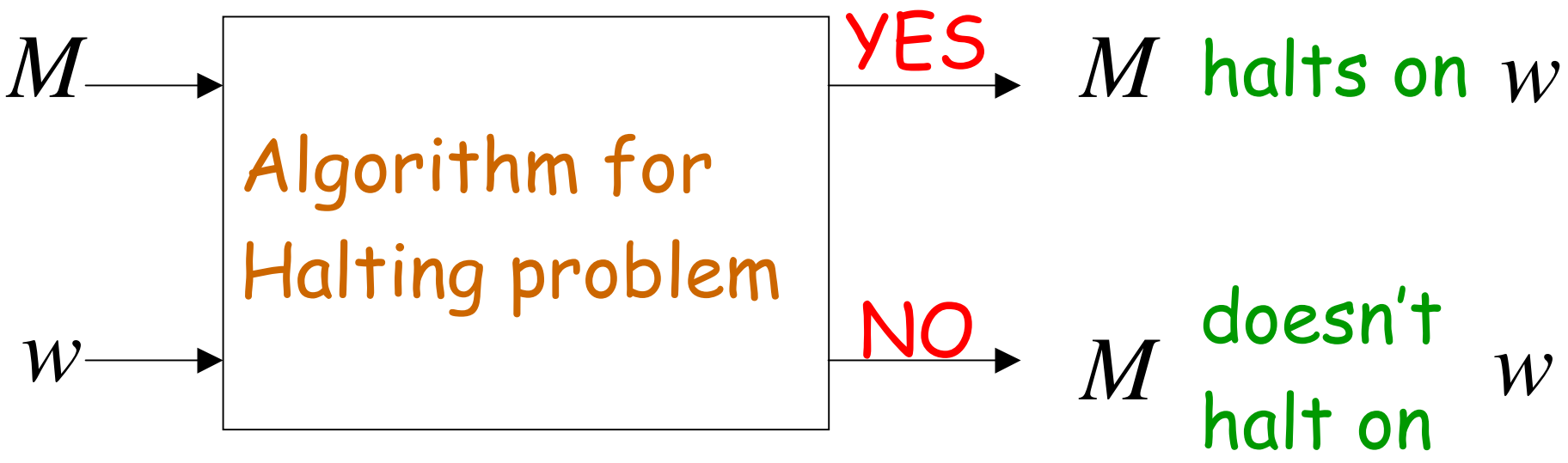
Let M be the machine that accepts L

$$L(M) = L$$

Assume we have the two-strings algorithm:



We will design the halting problem algorithm:



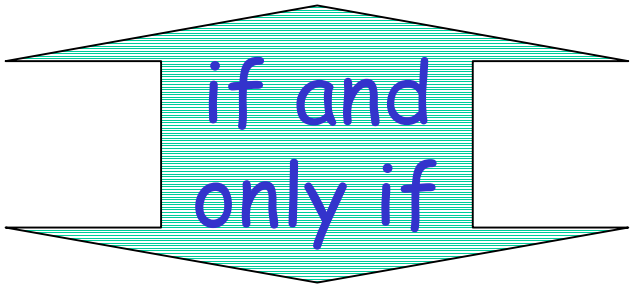
First construct machine M_w :

Initially, simulates M on input w

When M enters a halt state,
accept symbols a or b

(two equal length strings)

M halts on w



M_w accepts a and b

(two equal length strings)

Algorithm for halting problem:

Inputs: machine M and string w

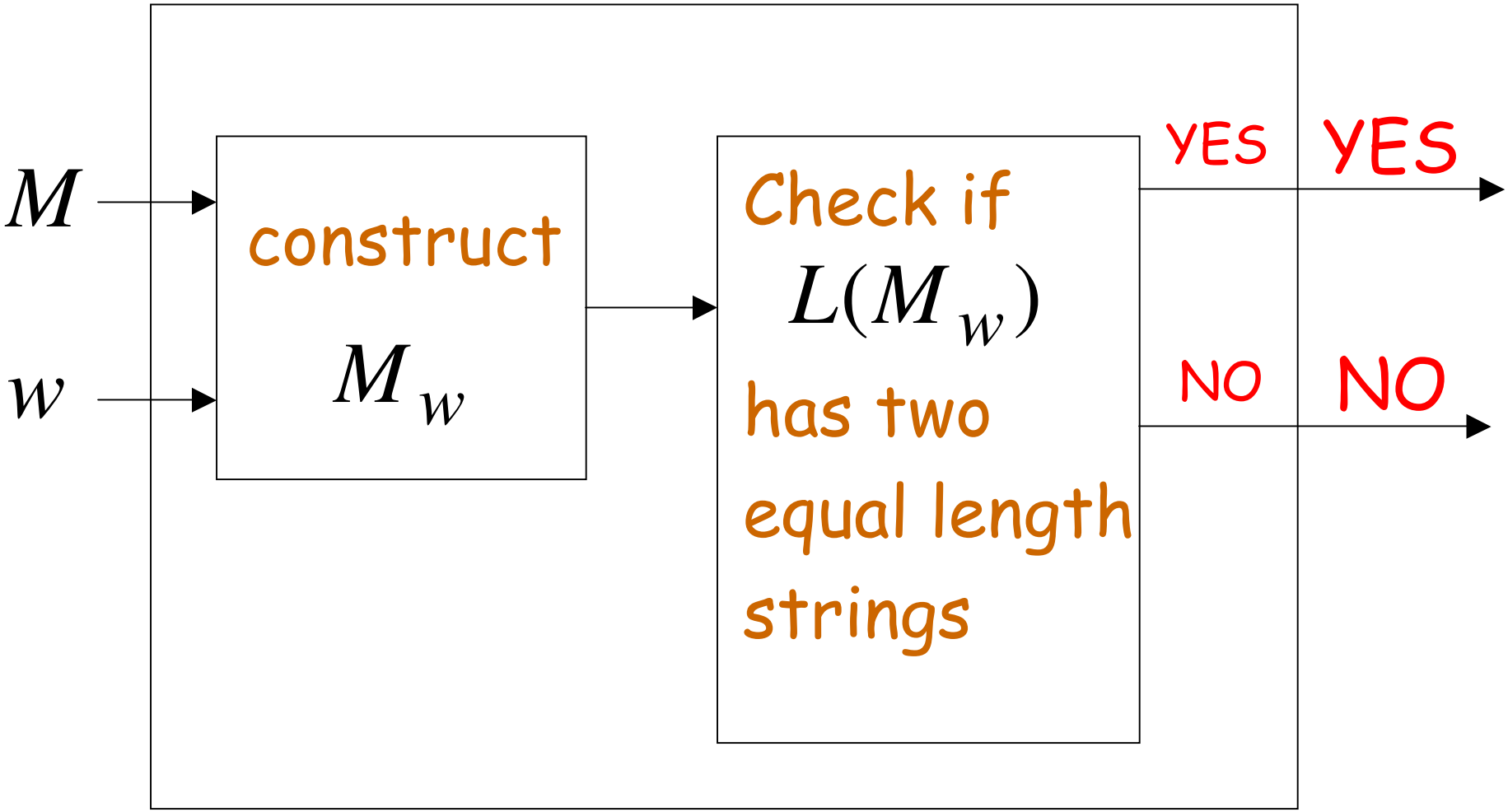
1. Construct M_w

2. Determine if M_w accepts two strings of equal length

YES: then M halts on w

NO: then M doesn't halt on w

Machine for halting problem



The Post Correspondence Problem

Some undecidable problems for context-free languages:

- Is context-free grammar G ambiguous?

- Is $L(G_1) \cap L(G_2) = \emptyset$?

We need a tool to prove that the previous problems for context-free languages are undecidable:

The Post Correspondence Problem

The Post Correspondence Problem

Input: Two sequences of strings

$$A = w_1, w_2, \ldots, w_n$$

$$B = v_1, v_2, \ldots, v_n$$

There is a Post Correspondence Solution if there is a sequence $i, j, \ldots, k$ such that:

$$w_i w_j \cdots w_k = v_i v_j \cdots v_k$$

PC-solution

Example:

	w_1	w_2	w_3
A:	100	11	111

	v_1	v_2	v_3
B:	001	111	11

PC-solution: 2,1,3

$$w_2 w_1 w_3 = v_2 v_1 v_3$$

$$11100111$$

Example:

	w_1	w_2	w_3
A:	00	001	1000

	v_1	v_2	v_3
B:	0	11	011

There is no solution

Because total length of strings from B
is smaller than total length of strings from A

The Modified Post Correspondence Problem

Inputs: $A = w_1, w_2, \ldots, w_n$

$$B = v_1, v_2, \ldots, v_n$$

MPC-solution: $1, i, j, \ldots, k$

$$w_1 w_i w_j \cdots w_k = v_1 v_i v_j \cdots v_k$$

Example:

	w_1	w_2	w_3
A:	11	111	100

	v_1	v_2	v_3
B:	111	11	001

MPC-solution: 1,3,2 $w_1 w_3 w_2 = v_1 v_2 v_3$

11100111

1. We will prove that the MPC problem is undecidable

2. We will prove that the PC problem is undecidable

1. We will prove that the
   MPC problem is undecidable

We will reduce the membership problem
to the MPC problem

Membership problem

Input: recursive language L

string w

Question: $w \in L$?

Undecidable

Membership problem

Input: unrestricted grammar G

string w

Question: $w \in L(G)$?

Undecidable

The reduction of the membership problem to the MPC problem:

For unrestricted grammar G and string w

we construct a pair A, B such that

A, B has an MPC-solution
if and only if $w \in L(G)$

A	B	Grammar G
$FS \Rightarrow$	F	S : start variable F : special symbol
a	a	For every symbol a
V	V	For every variable V

A	B	Grammar G
E	$\Rightarrow wE$	string w E : special symbol
y	x	For every production $x \rightarrow y$
$\Rightarrow$	$\Rightarrow$	

Example:

Grammar G :

$$S \to aABb \mid Bbb$$

$$Bb \to C$$

$$AC \to aac$$

String $w = aaac$

A		B	
$w_1:$	$FS \Rightarrow$	$v_1:$	F
$w_2:$	a	$v_2:$	a
	b		b
	c		c
$\vdots$	A	$\vdots$	A
	B		B
	C		C
$w_8:$	S	$v_8:$	S

A		B	
$w_9:$	$E \Rightarrow$	$v_9:$	$aaacE$
	$aABb$		S
	Bbb		S
$\vdots$		$\vdots$	
	C		Bb
	aac		AC
$w_{14}:$	$\Rightarrow$	$v_{14}:$	$\Rightarrow$

Grammar G :

$$S \to aABb \mid Bbb$$

$$Bb \to C$$

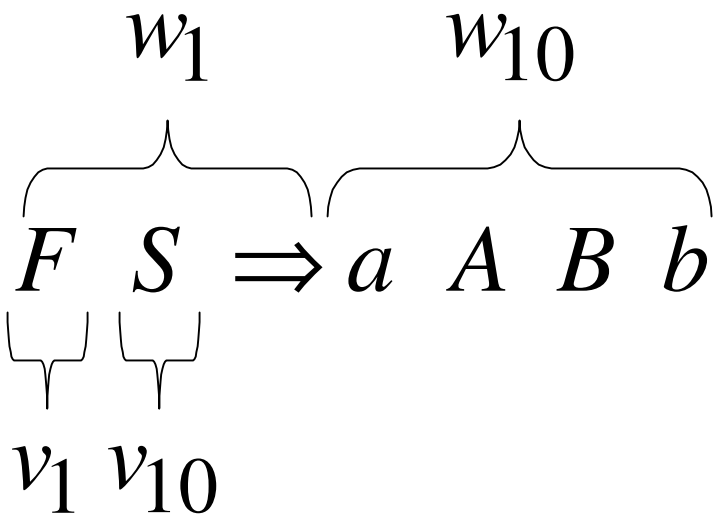
$$AC \to aac$$

$$aaac \in L(G)$$

$$S \Rightarrow aABb \Rightarrow aAC \Rightarrow aaac$$

$$S \Rightarrow aABb$$

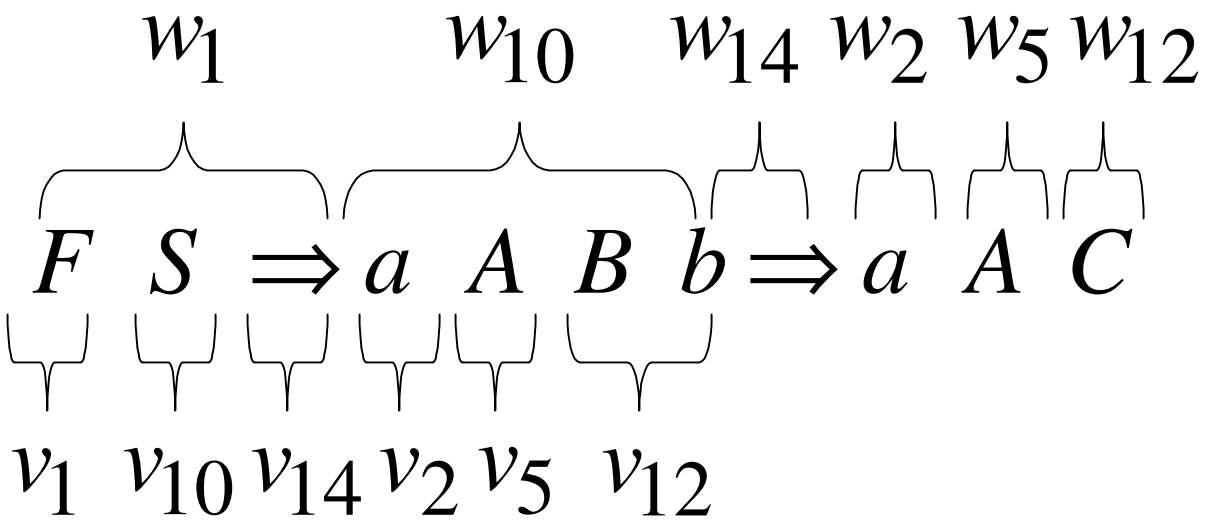
$$A$$



$$B$$

$$S \Rightarrow aABb \Rightarrow aAC$$

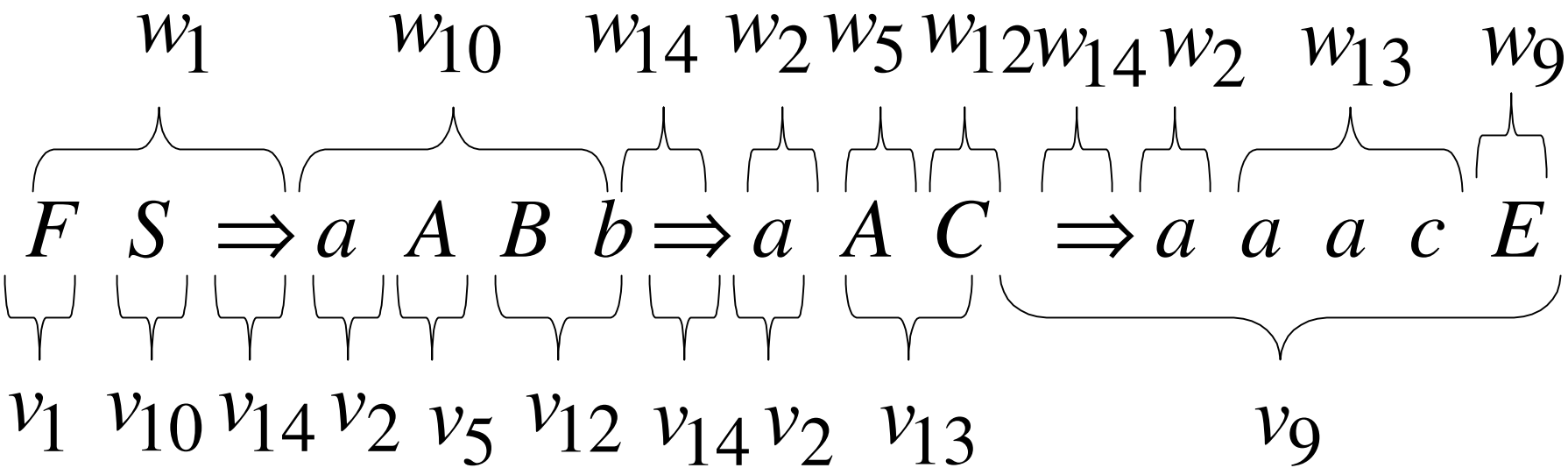
$$A$$



$$B$$

$$S \Rightarrow aABb \Rightarrow aAC \Rightarrow aaac$$

A



B

Theorem:

(A, B) has an MPC-solution

if and only if $w \in L(G)$

Algorithm for membership problem:

Input: unrestricted grammar G

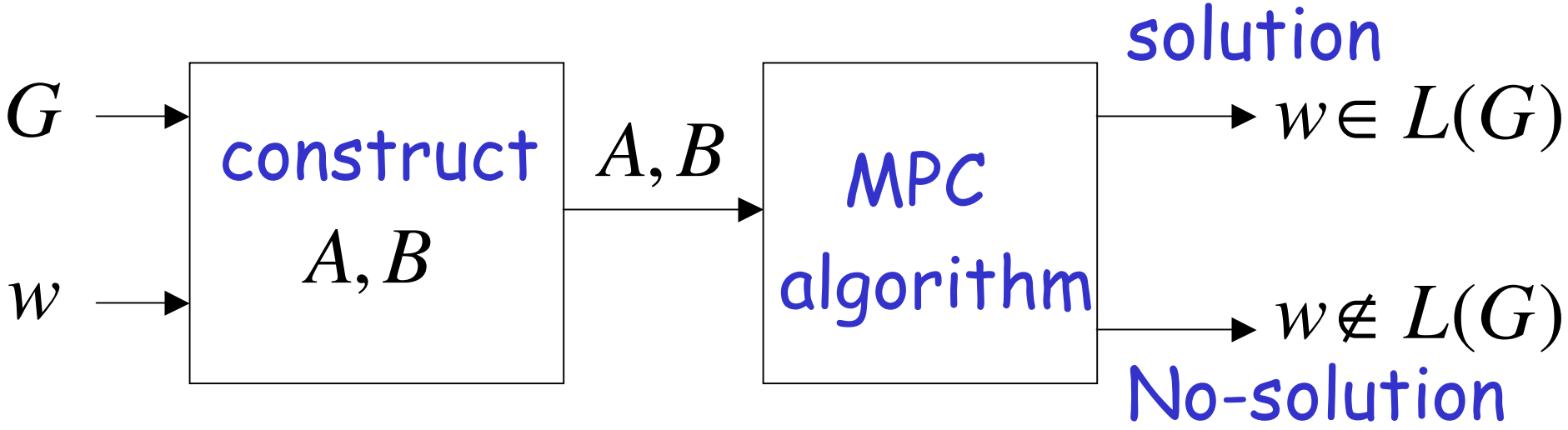
string w

Construct the pair A, B

If A, B has an MPC-solution then $w \in L(G)$

else $w \notin L(G)$

Membership machine



2. We will prove that the
   PC problem is undecidable

We will reduce the MPC problem
to the PC problem

A, B : input to the MPC problem

$$A = w_1, w_2, \ldots, w_n$$

$$B = v_1, v_2, \ldots, v_n$$

We construct new sequences C, D

$$C = w'_0, w'_1, \ldots, w'_n, w'_{n+1}$$

$$D = v'_0, v'_1, \ldots, v'_n, v'_{n+1}$$

$$A = w_1, w_2, \ldots, w_n$$

$$B = v_1, v_2, \ldots, v_n$$

A $\qquad\qquad$ $w_i = abcad$

C $\qquad\qquad$ $w_i' = a * b * c * a * d *$

We insert a special symbol between any two symbols

B $\quad v_i = abcad$

D $\quad v_i' = *a*b*c*a*d$

Special Cases

C

$$w'_0 = * \, w_1$$

$$w'_{n+1} = \Diamond$$

D

$$v'_0 = v'_1$$

$$v'_{n+1} = *\Diamond$$

Observation:

There is a PC-solution for C, D
if and only if
there is a MPC-solution for A, B

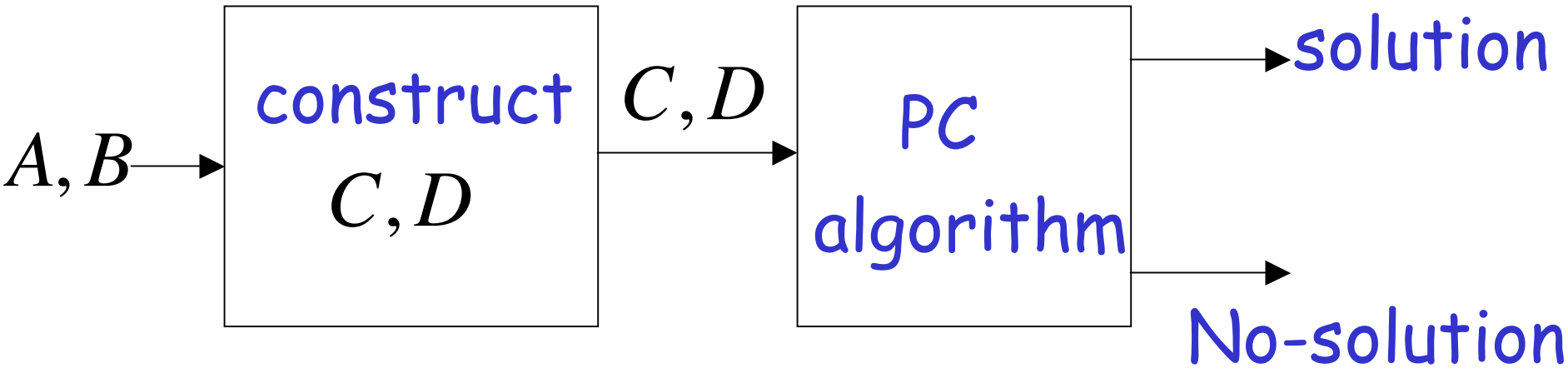
PC-solution $w'_0 \cdots w'_k w'_{n+1} = v'_0 \cdots w'_k v'_{n+1}$

MPC-solution $w_1 \cdots w_k = v_1 \cdots v_k$

MPC-algorithm

Input: sequences A, B

Construct sequences C, D

Solve the PC problem for C, D

MPC algorithm