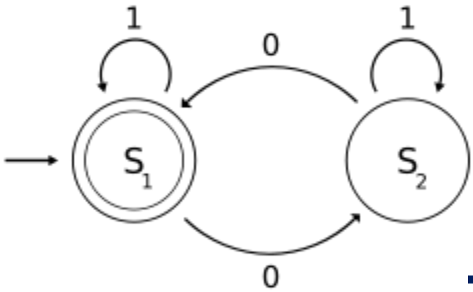
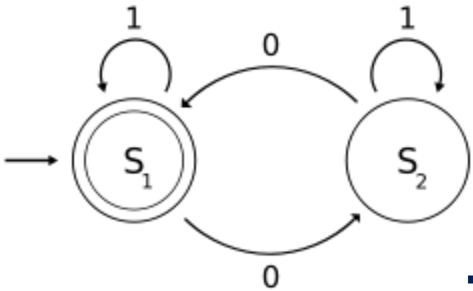
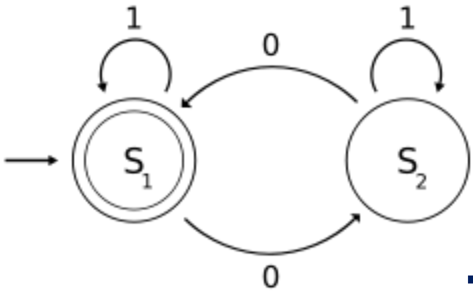


Undecidable Problems About Languages

Sipser: Section 5.1 pages 215 - 226

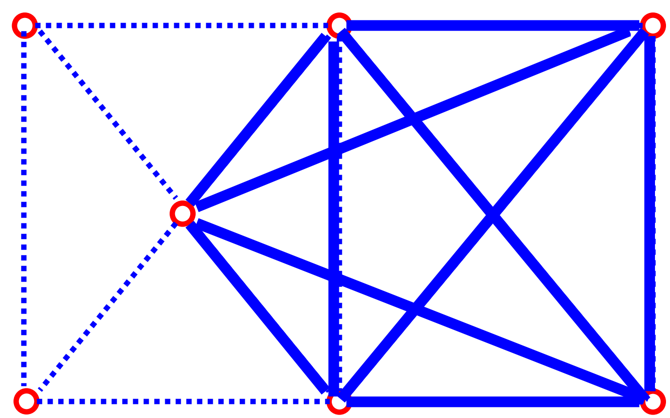


Reducibility

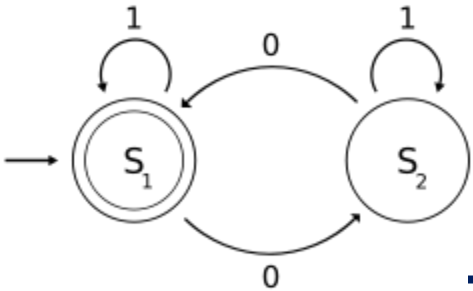


Clique and Independent Set

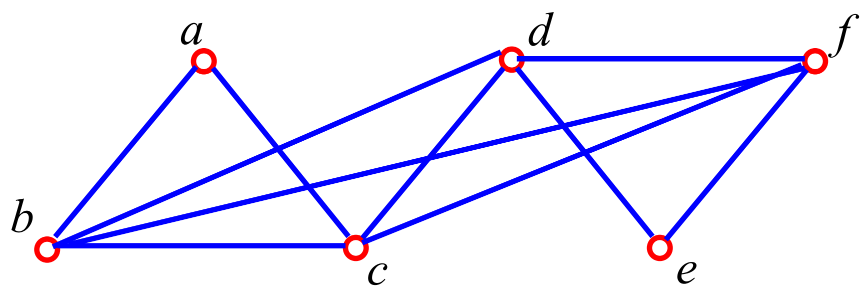
CLIQUE = {<*G*,*k*> | *G* is a graph with a *k*-clique}

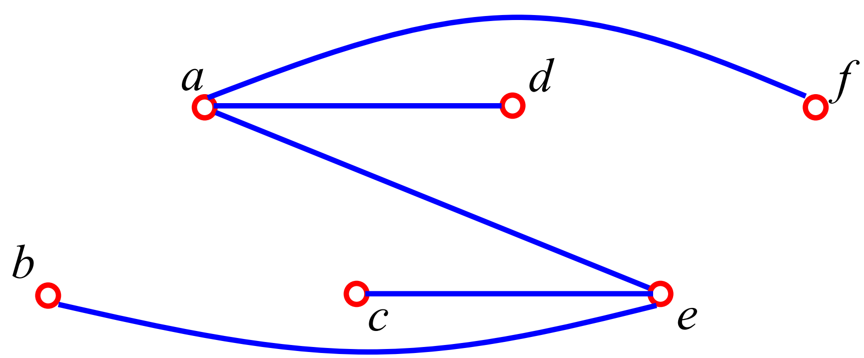


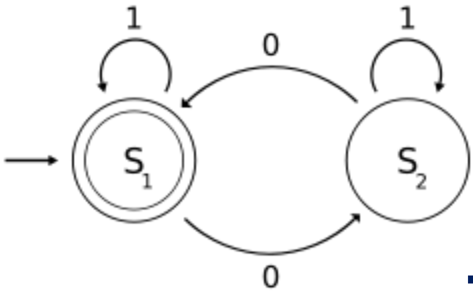
INDEPENDENT = {<*G*,*k*> | *G* is a graph containing an independent set of size *k*}



CLIQUE reduces to INDEPENDENT



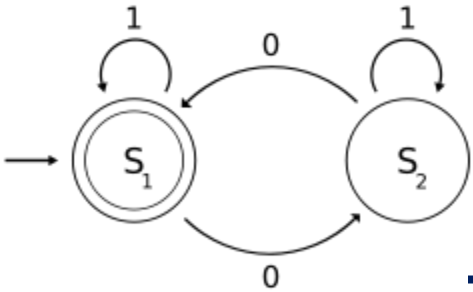




Certified Impossible

Theorem. A_{TM} = { <*M*, *w*> | *M* is a TM and *M* accepts *w* } is undecidable.

Definition. $HALT_{TM}$ = { <*M*, *w*> | *M* is a TM and *M* halts on input *w* }

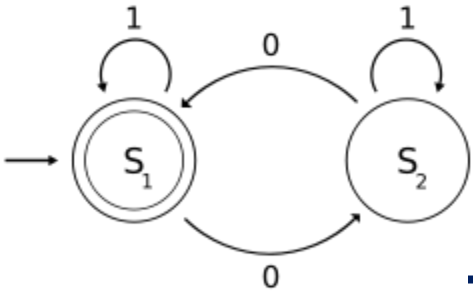


The Halting Problem (Again!)

Theorem. $HALT_{TM}$ is undecidable.

Proof Idea. We know A_{TM} is undecidable. We need to reduce one of $HALT_{TM}$ or A_{TM} to the other.

Which way to go?



$HALT_{TM}$ is undecidable

Proof. Suppose R decides $HALT_{TM}$. Define

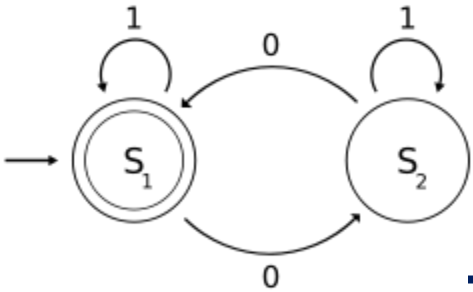
S = "On input <M, w>, where M is a TM and w a string:

1. Run TM R on input <M, w>.

2. If R rejects, then *reject*.

3. If R accepts, simulate M on input w until it halts.

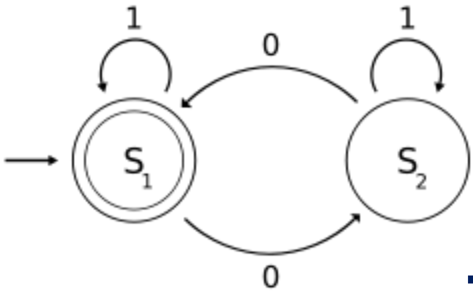
4. If M enters its accept state, *accept*. If M enters its reject state, *reject*."



Does *M* Accept Anything at All?

Definition. $E_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}$

Theorem. E_{TM} is undecidable.



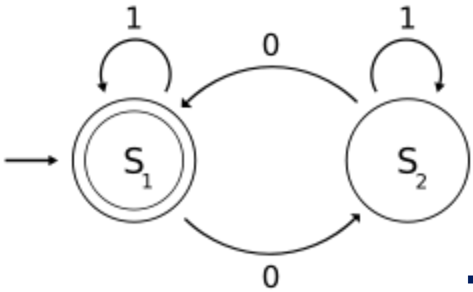
E_{TM} = {<M> | M is a TM & $L(M) = \emptyset$}

Proof. Given an input <M, w> we construct a machine M_w as follows:

M_w = "On input x:

1. If $x \neq w$, *reject*.
2. If $x = w$, run M on input w and *accept* if M does."

to be continued ...



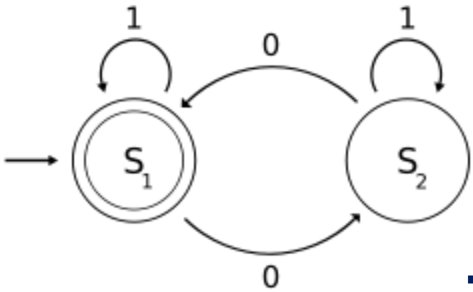
The Proof Continues

Proof continued.

Suppose TM *R* decides E_{TM}. Define

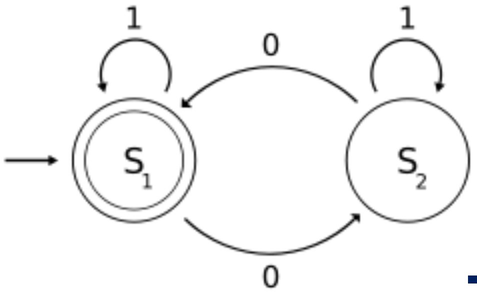
S = "On input <*M*, *w*>:

1. Use the description of *M* and *w* to construct M_w.
2. Run *R* on input <M_w>.
3. If *R* accepts, *reject*. If *R* rejects, *accept*."



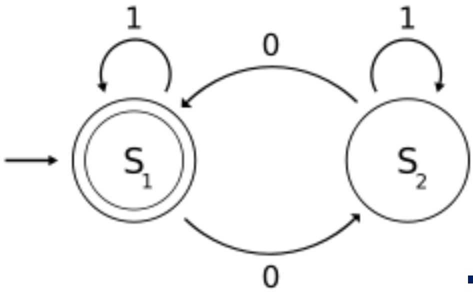
With Power Comes Uncertainty

	M accepts w	$L(M) = \emptyset$	$L(M_1) = L(M_2)$
Turing machines			
Pushdown machines			
Finite machines			



It's Even Worse Than You Thought

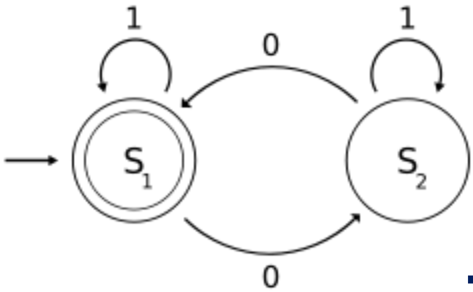
Rice's Theorem. *Any nontrivial property* of the languages recognized by Turing machines is undecidable.



For Example

Definition. $REGULAR_{TM}$ = { <*M*> | *M* is a TM and *L*(*M*) is regular}.

Theorem. $REGULAR_{TM}$ is undecidable.



$REGULAR_{TM}$ is undecidable

Proof. Let R be a TM that decides $REGULAR_{TM}$. Define

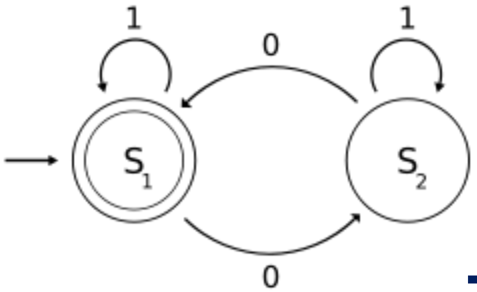
S = "On input $\langle M, w \rangle$:

1. Construct TM

   M_2 = "On input x:

   1. If x has the form 0^n1^n, *accept*.
   2. Otherwise, run M on input w and *accept* if M accepts w.

2. Run R on input $\langle M_2 \rangle$.
3. If R accepts, *accept*. Otherwise, if R rejects, *reject*."



Problems

- Let EQ_{TM} = { <M_1, M_2> | M_1 and M_2 are TMs and $L(M_1) = L(M_2)$ }. Show that EQ_{TM} is undecidable by reducing E_{TM} to EQ_{TM}.

- Consider the problem of determining whether a two-tape TM ever writes a nonblank symbol on its second tape when run on input *w*. Formulate this problem as a language and show that it is undecidable. (Hint: create an intermediary TM *T* that writes a nonblank symbol on its second tape iff *M* accepts *w*.)