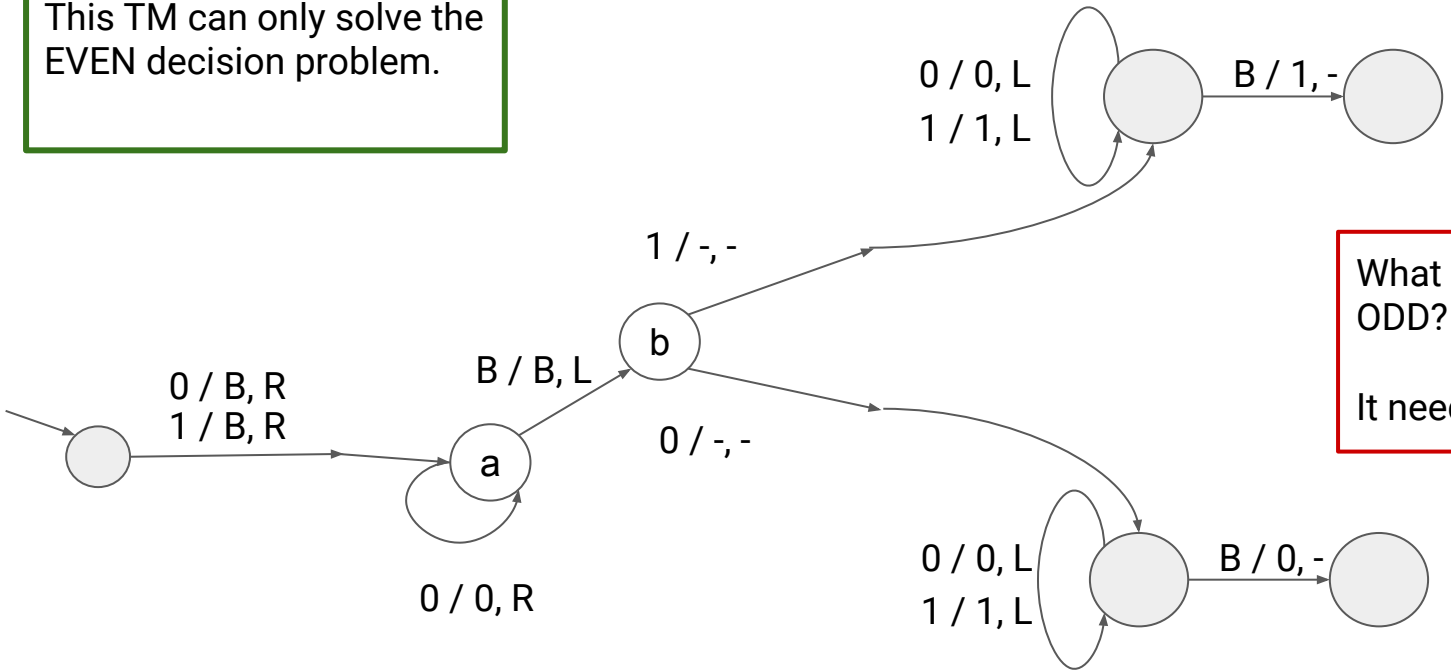
Universal Turing Machine

Ken Q Pu, Faculty of Science, Ontario Tech University

Turing machines are **hardwired** to the problem

This TM can only solve the EVEN decision problem.



What if we want to solve ODD?

It needs a different TM.

Ken Q Pu, Faculty of Science, Ontario Tech University

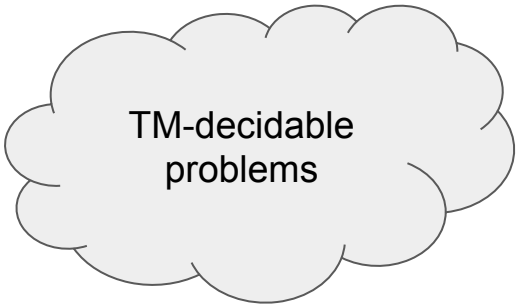
Decidability Refined

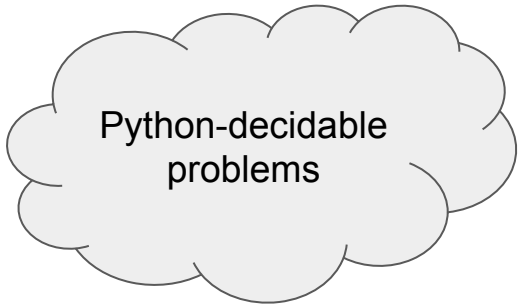
Definition:

A decision problem is decidable if there exists a TM that implements it.

The ENC function is *assumed*.

Relative powers of computing models

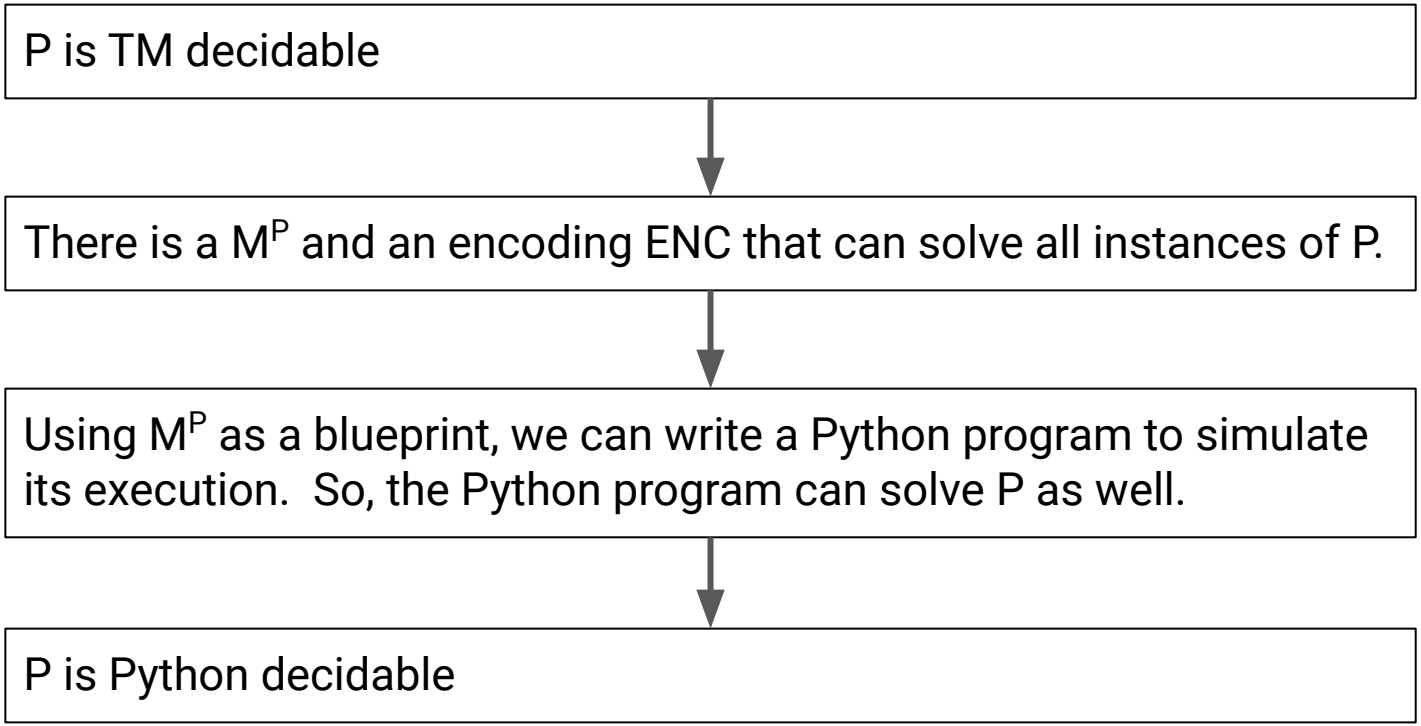




How do they compare?

The answer may surprise you, so read to the end.

TM vs Python



TM-Decidable

$\subseteq$

Python-Decidable

Some problems

Problem	TM-Decidable	Python-Decidable
EVEN?	Yes	Yes
ODD?	Yes	Yes
Divisibility of two integers	?	Yes
Shortest path in a graph	?	Yes
Deep learning	?	Yes
Solution of diophantine equation	No	No

Back in 1940s

- von Neumann needed computational power to design nuclear reactors, atomic bombs and the the hydrogen bomb.
- There were no computers, and hence no Python.
- TM-decidability was what he had at the time.
- How do we build **infinitely many** TM to solve all the problems we need to solve?

Alan Turing's brilliant vision - SIMULATION

Definition:

SIMULATION is a decision problem. The input is a TM, M, and the initial content of its tape, x.

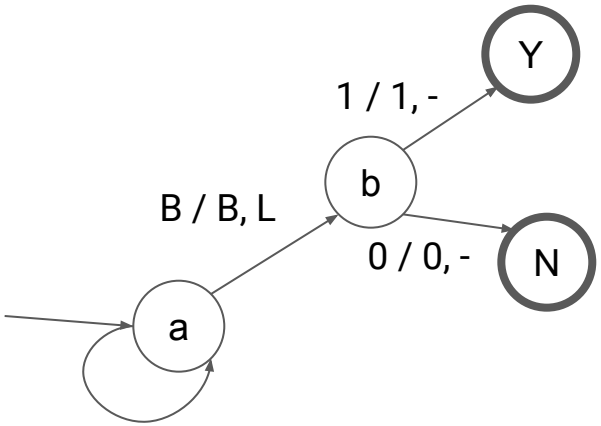
SIMULATION(M, x) = does M accept x?

SIMULATION is TM-decidable.
There exists M^{SIM}

We will prove it with the help of Python-decidability

Ken Q Pu, Faculty of Science, Ontario Tech University

Encoding of TM

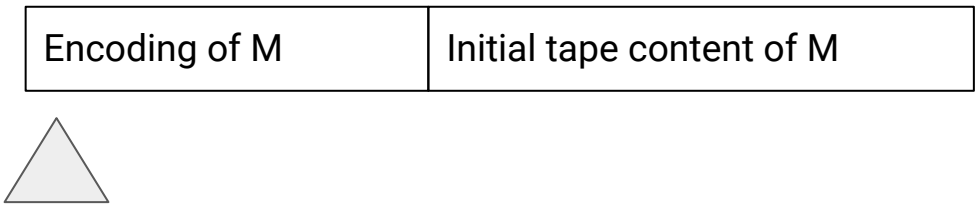


State	a
	b
	Y*
	N*
Transitions	a,a,0,0,R
	a,a,1,1,R
	a,b,B,B,L
	b,Y,1,1,-
	b,Y,0,0,-

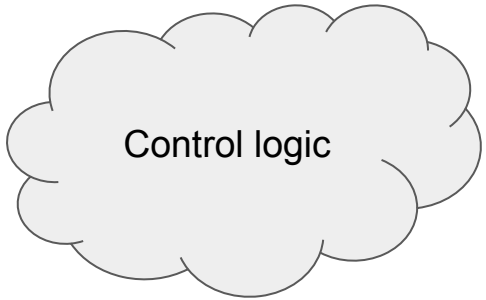
Encoding(M) = `a,b,Y*,N*;a,a,0,0,R;a,a,1,1,R;a,b,B,B,L;b,Y,1,1,-;b,Y,0,0,-;`

Ken Q Pu, Faculty of Science, Ontario Tech University

M^{SIM}



Tape of simulating TM



What's the control logic of the simulating TM?

Belief the millennium - Church-Turing Thesis

Church-Turing Thesis, 1936-1938

Any real-world computation is equivalent to some Turing machine.

Corollary#1:

Python-decidable $\subseteq$ TM-decidable

Corollary #2:

Python-decidable = TM-decidable

Another reason for Python-decidable $\subseteq$ TM-decidable

- von Neumann designed the computing architecture (of EDVAC) based on a TM.
- All modern computers are based the von Neumann architecture.
- So, all modern computers are running on top of a TM.
- Since Python runs on top of modern computers, all Python programs are being simulated by some TM.

SIM is TM-decidable

- Since Python is powerful enough to simulate an encoded TM and its input, there exists a TM that can do the same.
- This is called the Universal Turing Machine (UTM)

Programming **the** Universal Turing Machine

Programming involves tasks such as analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the **implementation of algorithms** (usually in a chosen programming language, commonly referred to as coding).

Wikipedia

Can you program a TM in general?

No

Can you program the UTM?

Yes

UTM Programming

Given a problem, P, you can go through the entire programming cycle to solve P using the UTM

Design	Same as modern programming, we need to design an algorithm in some pseudo code.
Create a source code in a programming language	The programming language is the control logic of a TM. The source code is the transition diagram of M^P
Compile the source code	We need to encode M^P into the string encoding for UTM to process.
Load the compiled binary	Write ENC(M^P) to the tape of UTM
Accept user input	Write input to MP to the tape of UTM after the program.
Execute program	Run UTM, hope it will halt.