

# History of Artificial Intelligence

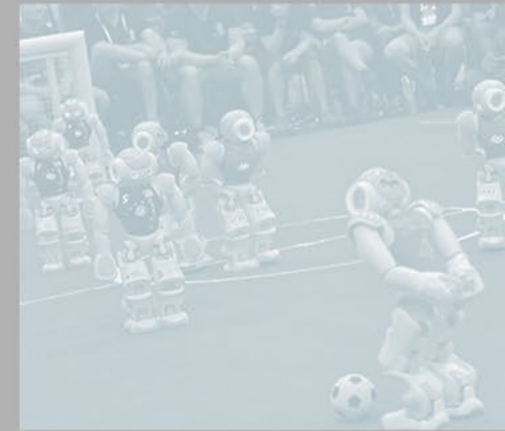
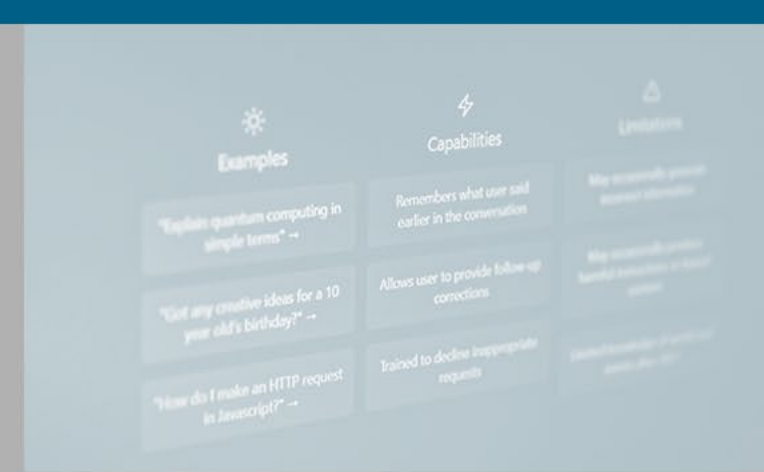
- Term coined in 1956
- Key conference at Dartmouth, McCarthy, Minsky, Rochester, Shannon
- Conference to focus on
  - Natural language processing
  - Neural networks
  - Theory of computation
  - Abstraction
  - Randomness and creativity

# History of Artificial Intelligence

- Since that time
- Focus on hard problems that come naturally to humans
  - Perception (Image recognition, audio recognition, etc.)
  - Natural language processing (computer languages are much better understood and easier to deal with)
  - Games (checkers, chess, go, ...)
  - Reasoning, theorem proving, etc.
  - Neural networks
  - Robotics

# Closely related questions

- What can computers do?
- Can computers do what humans can do?
- Can computers think?



FAKE NEWS

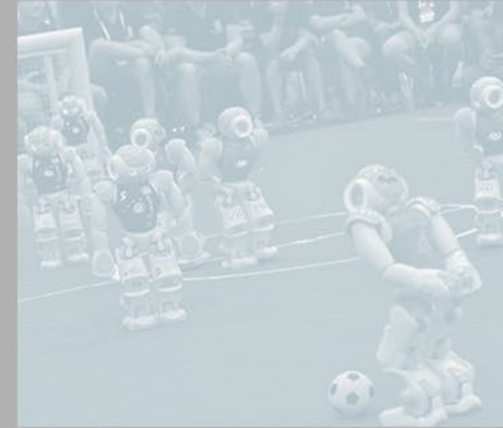
# What can computers do?

## Decidable (computable) and undecidable problems

Examples	Capabilities	Limitations
"Explain quantum computing in simple terms" →	Remembers what user said earlier in the conversation	May occasionally provide incorrect information
"Get any creative ideas for a 10 year old's birthday?" →	Allows user to provide follow-up corrections	May occasionally provide harmful information to vulnerable groups
"How do I make an HTTP request in Javascript?" →	Trained to decline inappropriate requests	Limited knowledge of world events that occur after data cutoff date



```
o=u.length:r&&(s=t,c(r))}return this},remove:
tion(){return u=[],this},disable:function(e:
function(){return p.fireWith(this,argument
ding",r={state:function(){return n},always
omise)?e.promise().done(n.resolve).fail(n.r
(function(){n=s},t[1^e][2].disable,t[2][2]
),n=h.call(arguments),r=n.length,i=1!==r|e
```



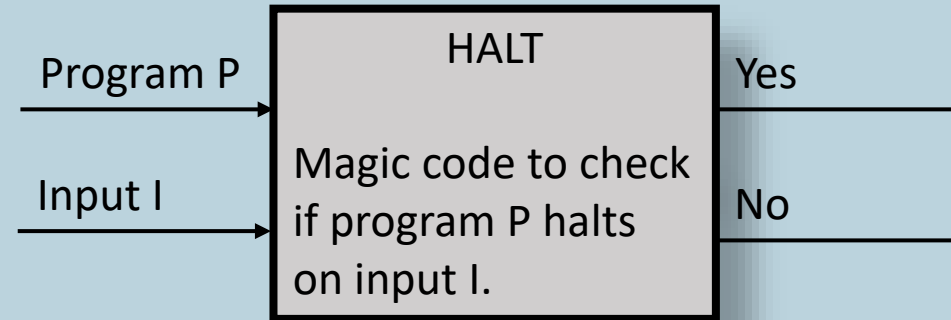


# Decidability (Computability)

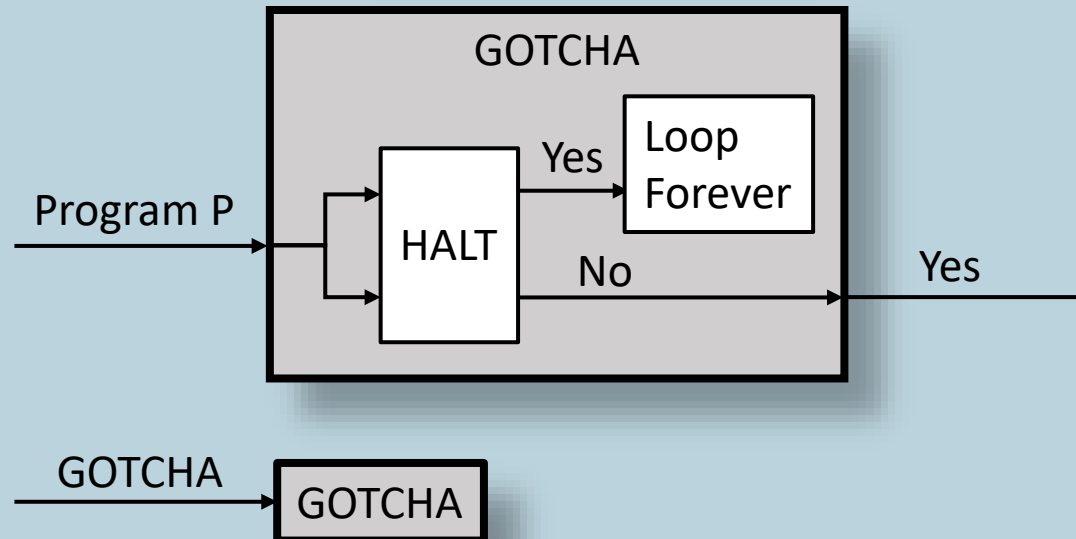
- The Halting Problem

- Suppose you have some program  $P$  and you'd like to know if it will halt and produce an answer given an input  $I$ .
- You could try just running  $P$  with input  $I$  to see, but then if it doesn't halt for some time, when do you determine that you know the answer?
- Instead you'd like to create a program  $HALT$  that can take the source code to  $P$  and the input  $I$  and tell you whether or not  $P$  will halt and produce an answer given  $I$  as input. In fact, you'd like to make  $H$  be able to tell you this for any program and input that you might give it.
- It can be proven that  $HALT$  cannot exist, i.e. that you cannot write a program that takes any program and its input and tells you whether it will halt and give an answer.

- Suppose you had such a program, call it HALT:



- You could create from it another program, GOTCHA:



- Then run GOTCHA on itself:

If GOTCHA halts, then it doesn't halt, and if it doesn't halt, then it halts

# Decidability

- This is not just a toy problem.
- Program analysis tools
  - Automatically check your program for "correctness"
  - Check to see if a particular part of your code will ever be executed
- But there are programs that do these things
  - We know these can work on some interesting set of programs, but not on every program

# Decidability

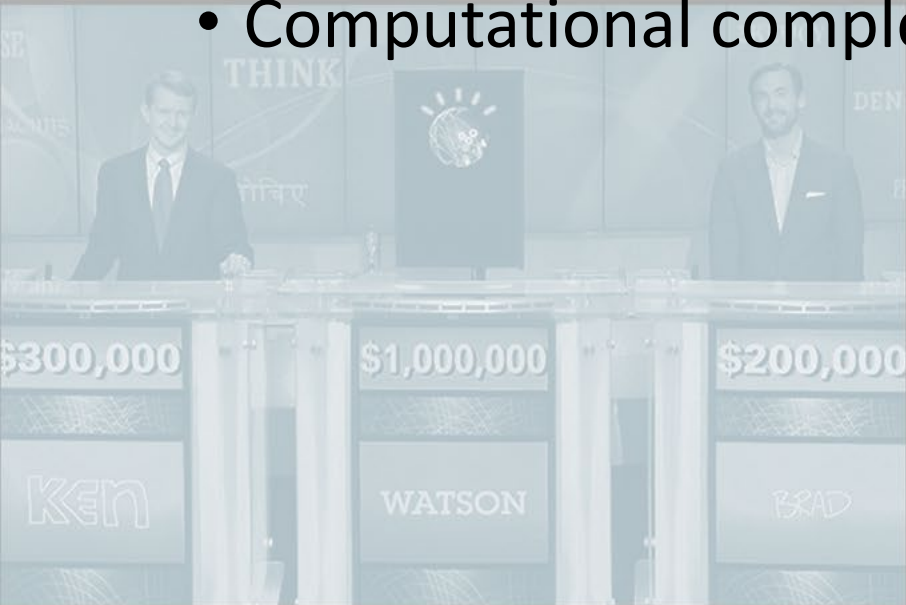
- Given a mathematical conjecture, it is possible to create a program that will tell whether it is provable (and thus a theorem)?
- No, this is the problem Turing proved undecidable in the paper that defined Turing Machines (also Godel and Church)
- Yet one of the first significant AI programs was Newell and Simon's "Logic Theorist", which proved theorems from Russell and Whitehead
- We still build "Automated Theorem Provers", in fact UTCS has been one of the leaders in this area, and we use this technology to verify correctness of systems



FAKE NEWS

# What can computers do?

- Computational complexity – how many resources do programs take?



```
o=u.length:r&&(s=t,c(r))}return this},remov  
tion(){return u=[],this},disable:function(  
e:function(){return p.fireWith(this,argumen  
ding",r={state:function(){return n},always  
omise)?e.promise().done(n.resolve).fail(n.r  
(function(){n=s},t[1^e][2].disable,t[2][2]  
,n=h.call(arguments),r=n.length,i=1!==r||e
```

