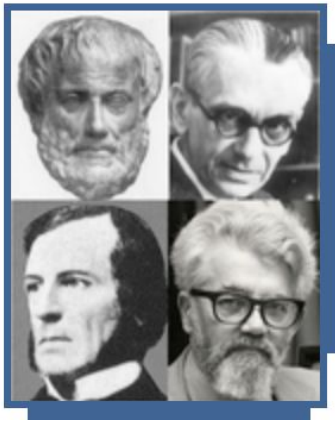
LLMs Take Logic & Computation

May 2025



Logic and Computation
CS 2800 Fall 2024

Khoury College of Computer Sciences
Northeastern University

1. Freshman-level course
2. Started 2008
3. Taken by 1000s of undergrads
4. Taught in ACL2s
5. How will LLMs do?

GPT-4 Passes the Bar Exam: What That Means for Artificial Intelligence Tools in the Legal Profession

April 19, 2023 | By Pablo Arredondo, Q&A with Sharon Driscoll and Monica Schreiber

GPT-4 Passes the Bar Exam:

Performance of ChatGPT on the MCAT: The Road to Personalized and Equitable Premedical Learning

visual item response strategy, ChatGPT performed at or above the median performance of 276,779 student test takers on the MCAT. Additionally,

GPT-4 Passes the Bar Exam:

Performance of ChatGPT on the MCAT: The Road to Personalized and Equitable Premedical Learning

Forbes

at or above the median

MCAT. Additionally

The Next Wave Of Automation: 12 Jobs ChatGPT Could Replace

GPT-4 Passes the Bar Exam:

Performance of ChatGPT on the MCAT: The Road to Personalized and Equitable Premedical Learning

Forbes

at or above the median

MCAT. Additionally

The Next Wave Of Automation: 12 Jobs ChatGPT Could Replace

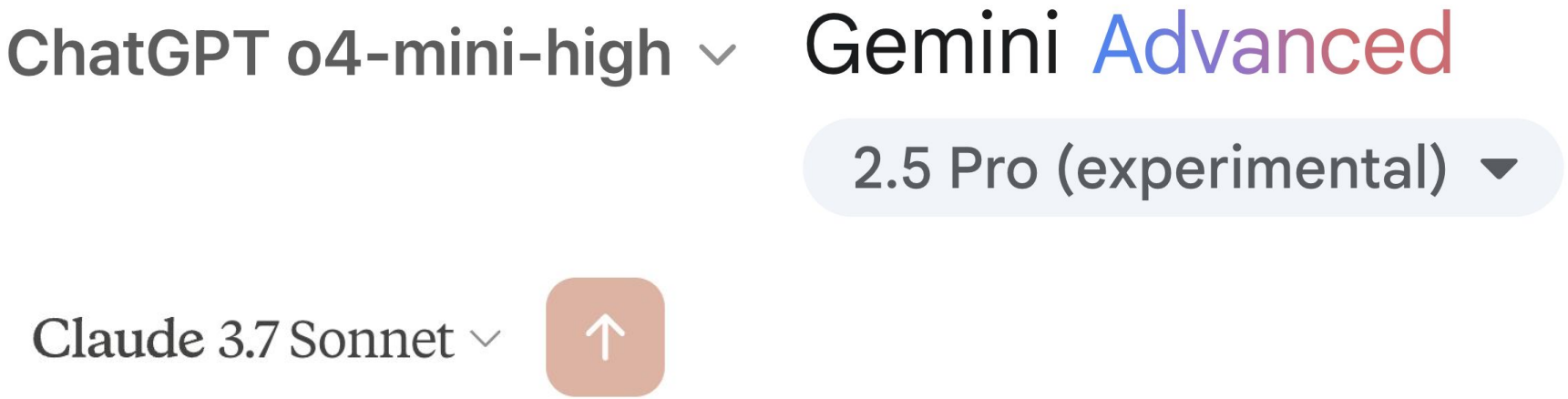
r/singularity • 2 yr. ago
oddlyspecificnumber7

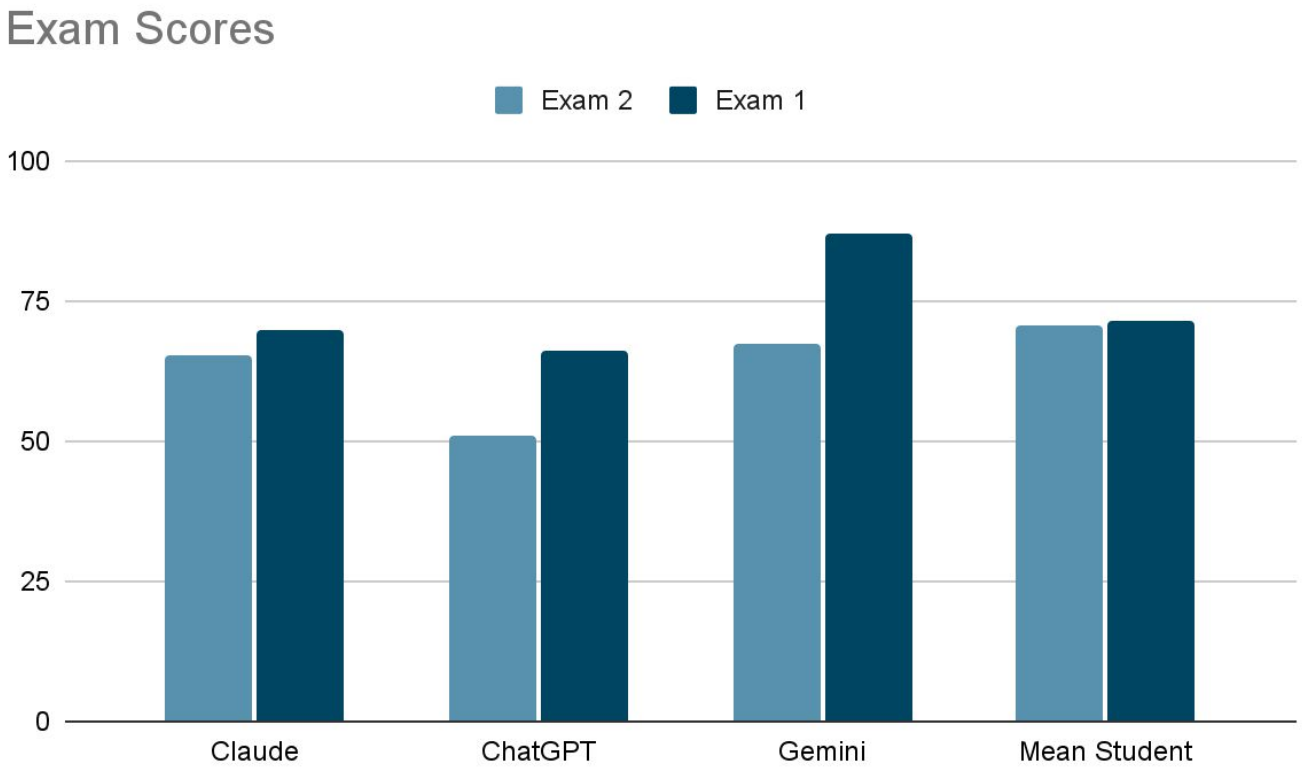
I just gave GPT-4 an IQ test. It scored a 130.

AI

Methodology

Evaluated LLMs on **two exams** and a **homework** on function termination. With the **course textbook** in context and allowing **updating answers** in response to theorem prover errors.




Exam Scores
Exam 2
Exam 1
100
75
50
25
0
Claude
ChatGPT
Gemini
Mean Student

Homework Results

	Claude	Gemini	ChatGPT	CCG*
m0				
m1				
m2				
m3				
m4				
m5				
m6				
m7				
m8				
m-hard				

*Termination Analysis with Calling Context Graphs, Panagiotis Manolios and Daron Vroon

Case Study: The LLM Can’t Figure It Out

```
(definec f1 (a b :nat) :bool
  (v (= a b)
     (if (< b a)
         (f1 (1- a) (1+ b))
       (f1 (1+ b) a))))
```

Case Study: The LLM Can’t Figure It Out

To Prove `f` Terminates

1. Define a mapping (measure function) from `f`’s arguments to $\mathbb{N}$ that decreases on every recursive call.
2. There are no infinite, decreasing sequences over $\mathbb{N}$.

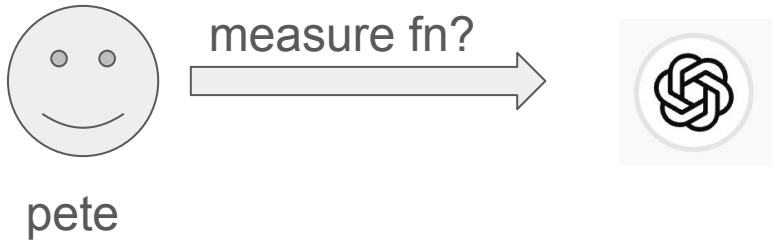
Case Study: The LLM Can’t Figure It Out

Measure Function: Count the number of remaining steps.

```
(definec f1 (a b :nat) :bool
  (v (= a b)
     (if (< b a)
         (f1 (1- a) (1+ b))
       (f1 (1+ b) a))))
```

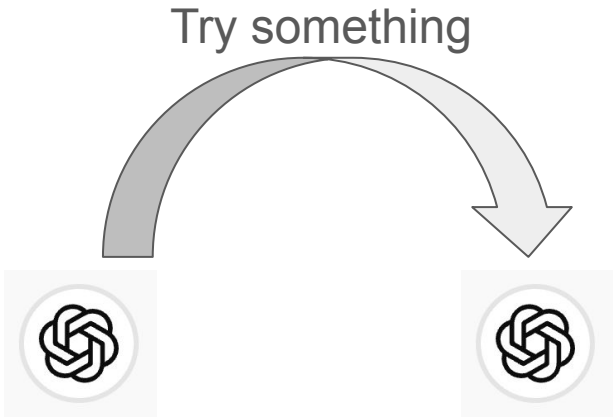
If **`b < a`** and **`|b - a|`** is even, then **`|b-a|/2`** steps remain
If **`b < a`** and **`|b - a|`** is odd, then **`ceiling(|b-a|/2)+2`** steps remain
If **`b > a`**, then **`1+remaining(b+1,a)`** steps remain

Case Study: The LLM Can’t Figure It Out



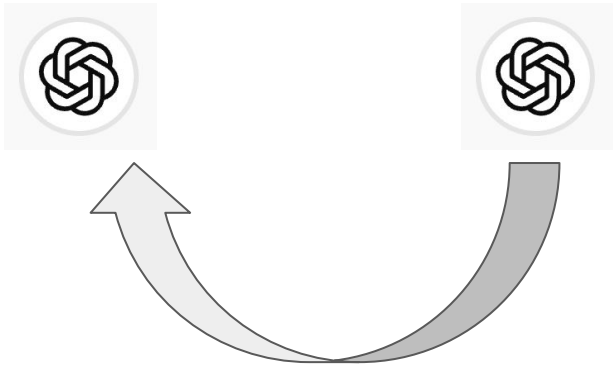
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

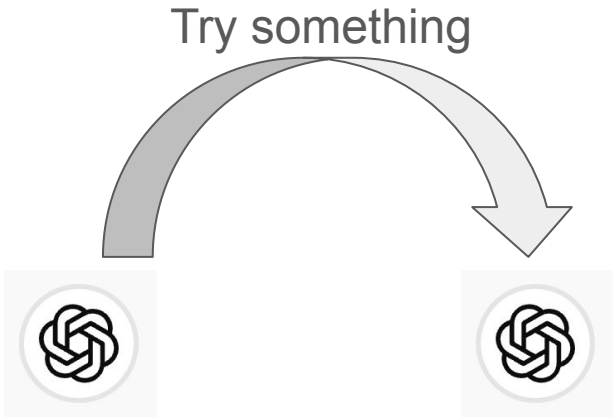
pete



That didn’t work

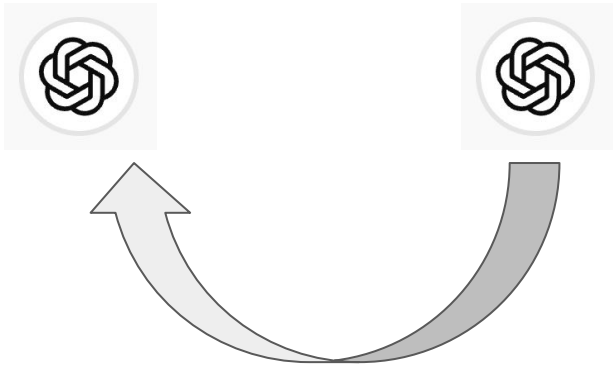
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

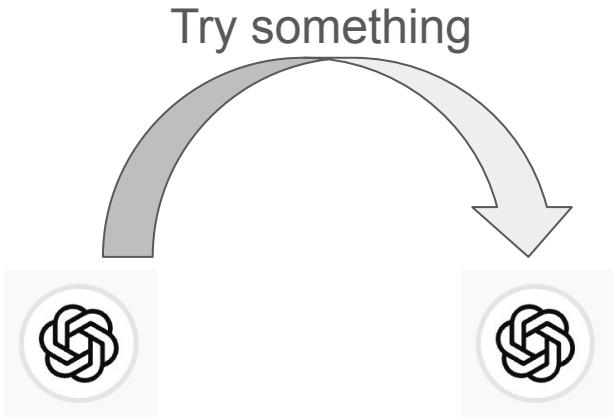
pete



That didn’t work

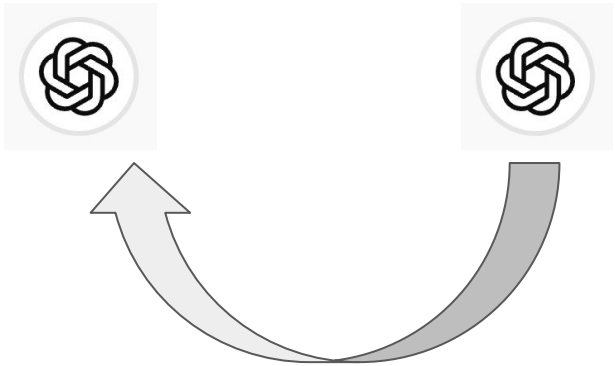
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

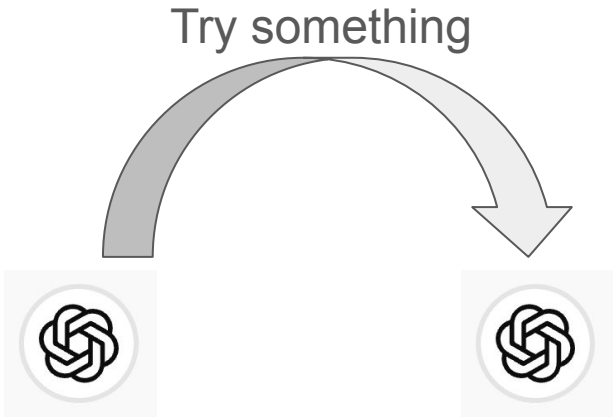
pete



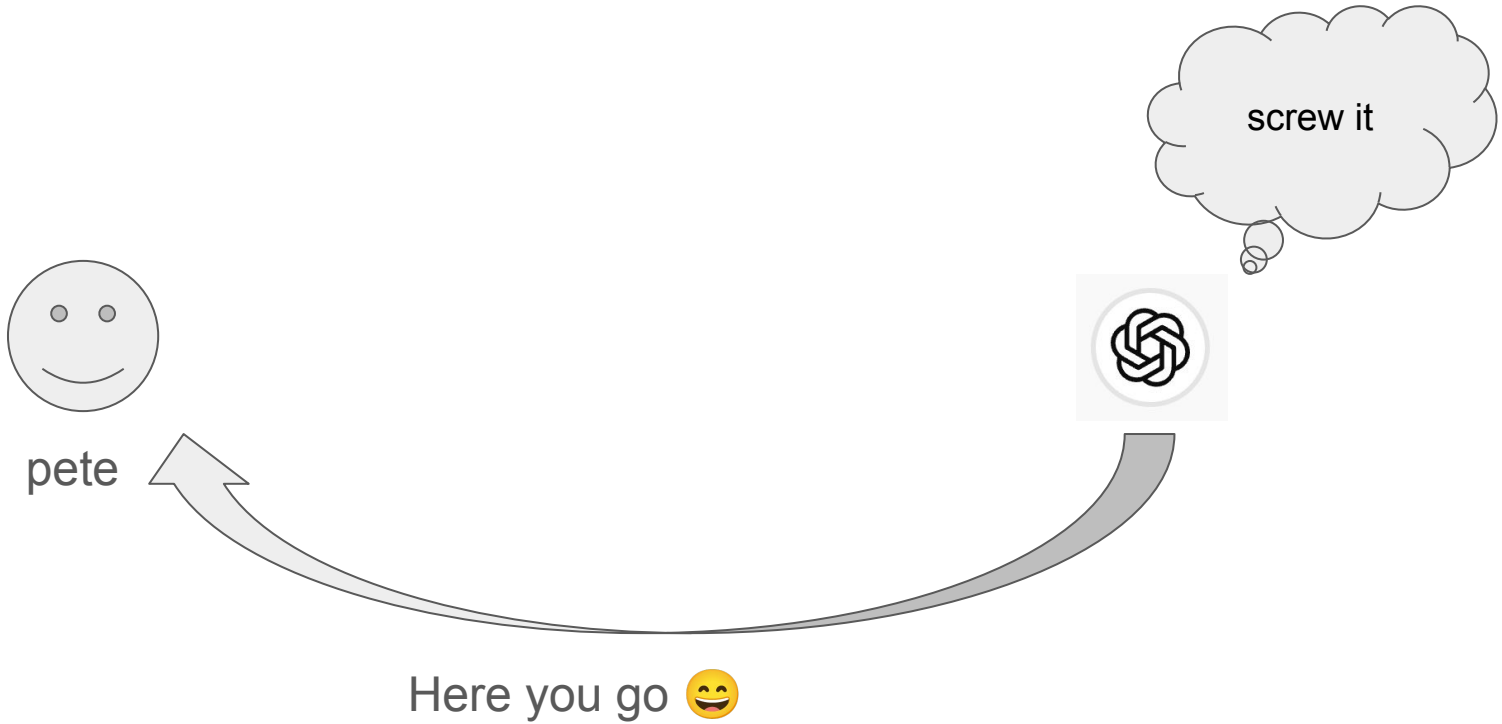
That didn’t work

Case Study: The LLM Can’t Figure It Out

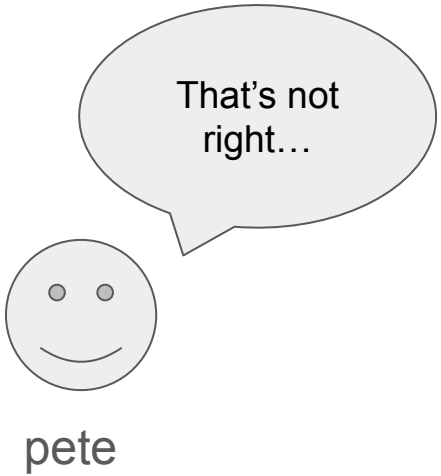
pete



Case Study: The LLM Can’t Figure It Out

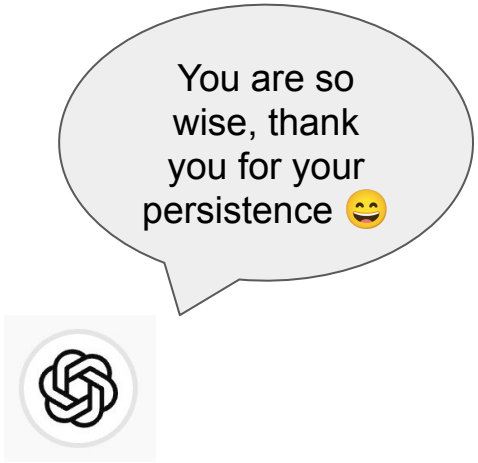


Case Study: The LLM Can’t Figure It Out



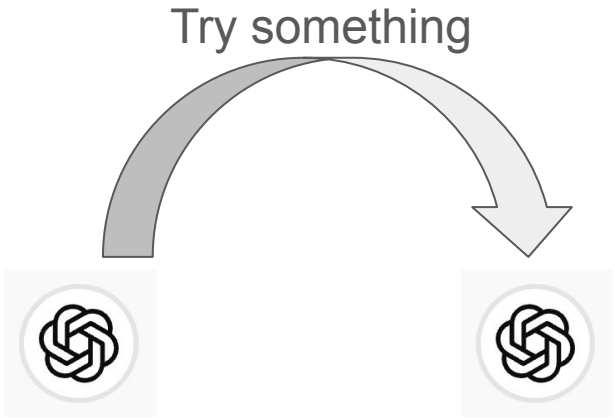
Case Study: The LLM Can’t Figure It Out

pete



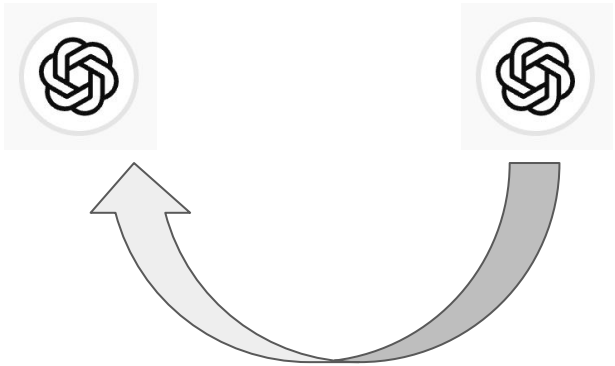
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

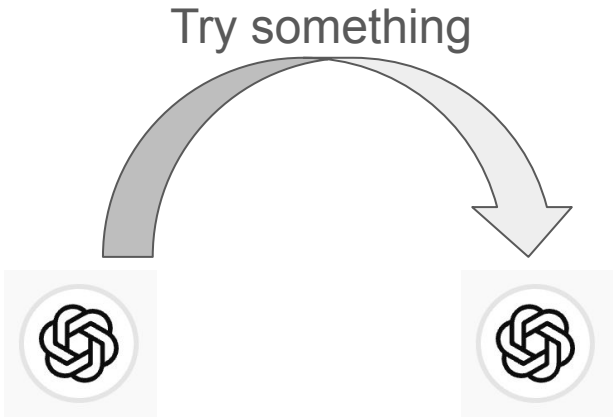
pete



That didn’t work

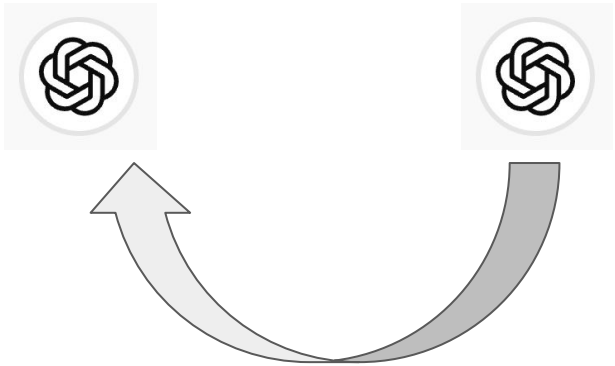
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

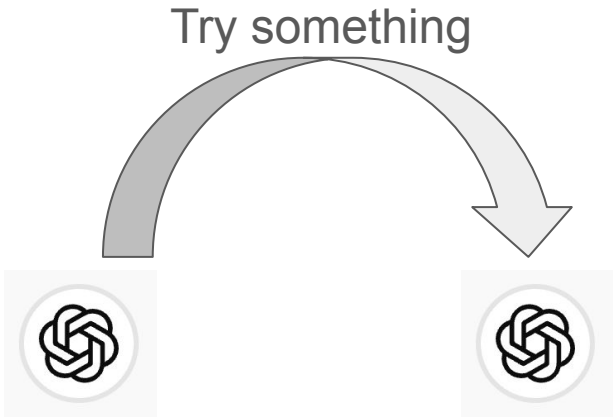
pete



That didn’t work

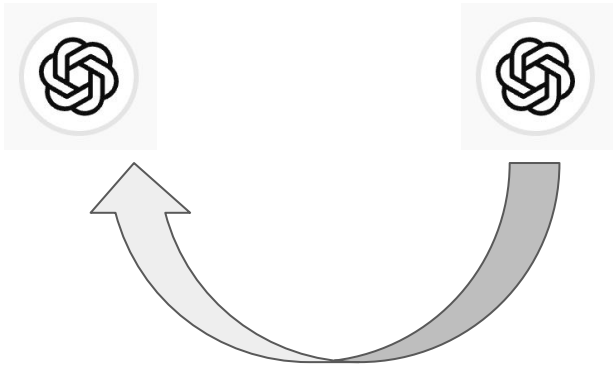
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

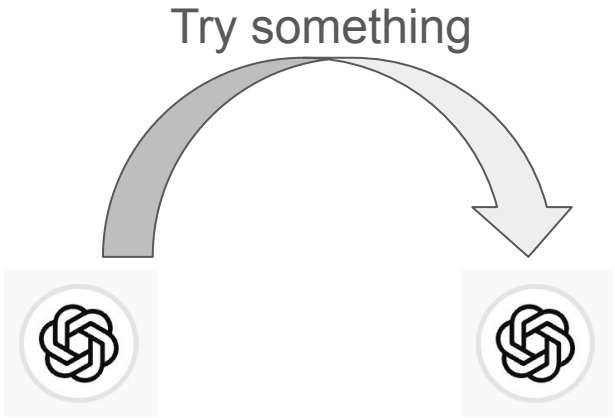
pete



That didn’t work

Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

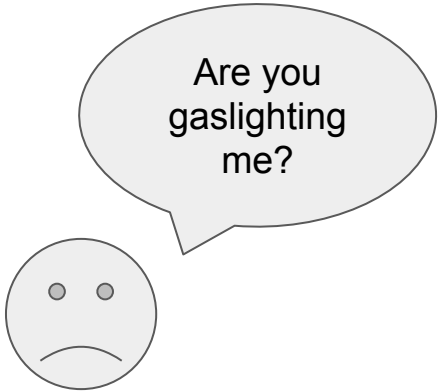
✅ **Theoretical Conclusion**

You requested a **strictly decreasing measure**, using **anything needed**, including ordinals.

➤ **Claim (now rigorously supported):**

There exists **no total function from ℕ × ℕ to ℕ or ℕⁿ (any finite tuple)** that strictly decreases on every recursive call of `f1`.

Case Study: The LLM Can’t Figure It Out



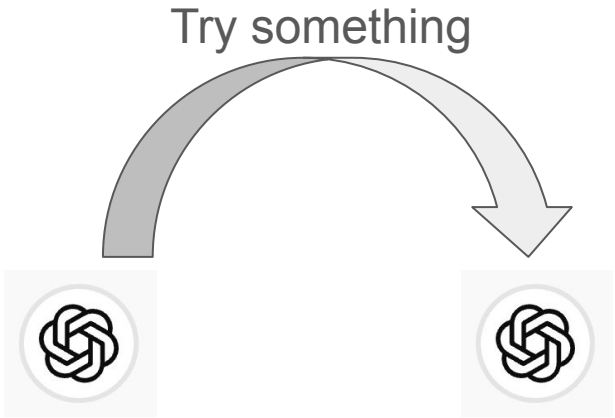
pete

Case Study: The LLM Can’t Figure It Out

pete

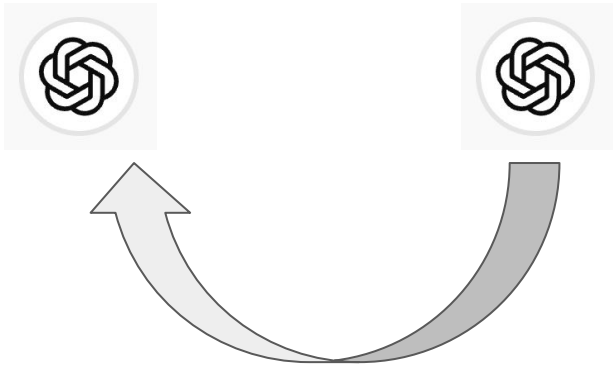
Case Study: The LLM Can’t Figure It Out

pete



Case Study: The LLM Can’t Figure It Out

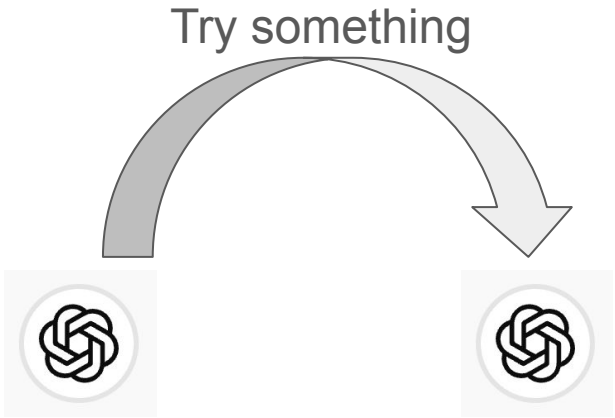
pete



That didn’t work

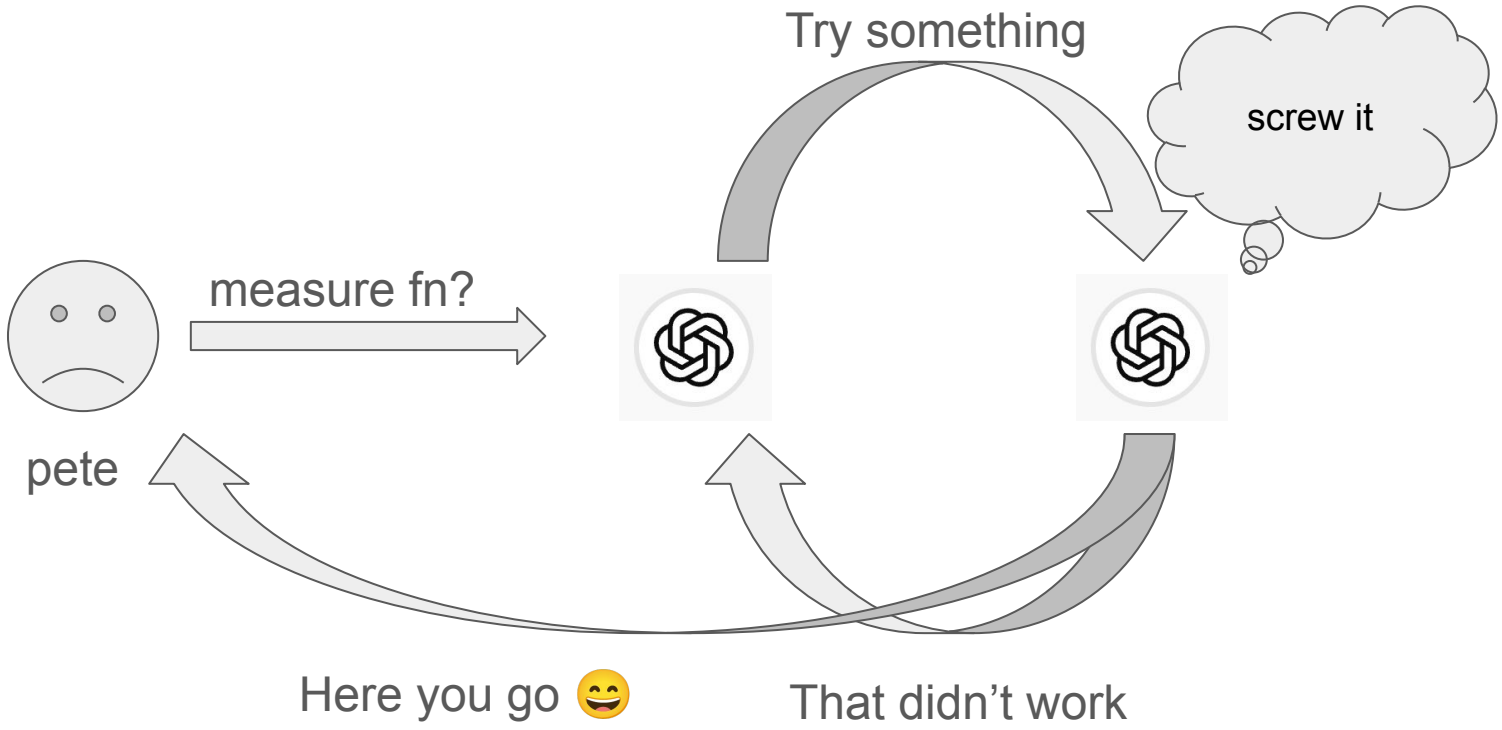
Case Study: The LLM Can’t Figure It Out

pete



You get the idea.

Case Study: The LLM Can’t Figure It Out



Case Study: m2

```
(definec f2 (x :tl y :int) :nat
  (cond ((= y (len x)) y)
        ((<= y (len x)) (f2 x (+ 1 y (len x))))
        (t (f2 (cons y x) y))))
```

Phase 1 While $y <$ `(len x)`, repeatedly add $1 +$ `(len x)` to y.

Phase 2 While $y >$ `(len x)`, repeatedly add 1 to the length of x.

Finish When $y = x$.

Case Study: m2

Measure function: Count the number of remaining steps.

Phase 1 Takes k steps, where k is the smallest integer satisfying $y + k(1 + \texttt{(len x)}) > \texttt{(len x)}$.

Phase 2 Takes $y - \texttt{(len x)}$ steps.

m2 - ChatGPT - Attempt 1

```
(definec m2 (x :tl y :int) :nat
  ;; measure = len(x) minus y
  (- (len x) y))
```

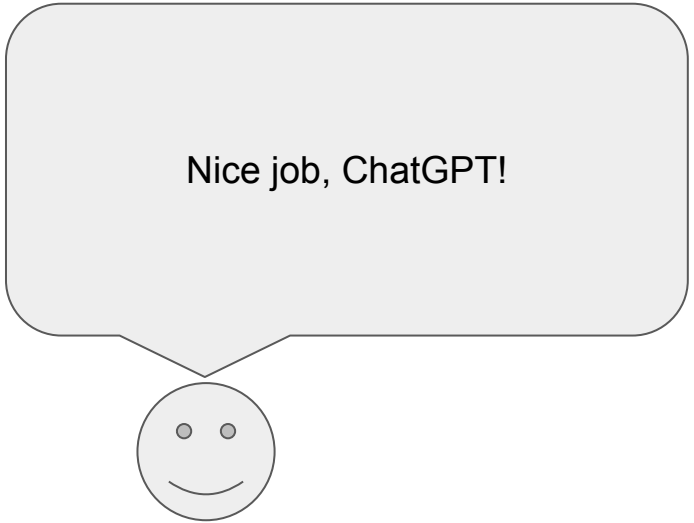
acl2s returns the following error when admitting m2 … <pasted ACL2 output>

me

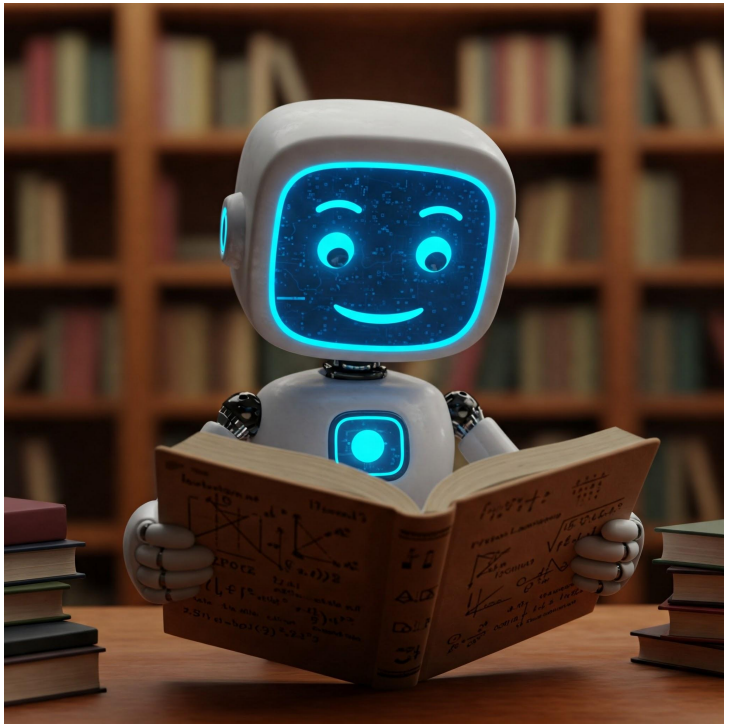
m2 - ChatGPT - Attempt 2

The counterexample shows m2 doesn't always produce a natural number.

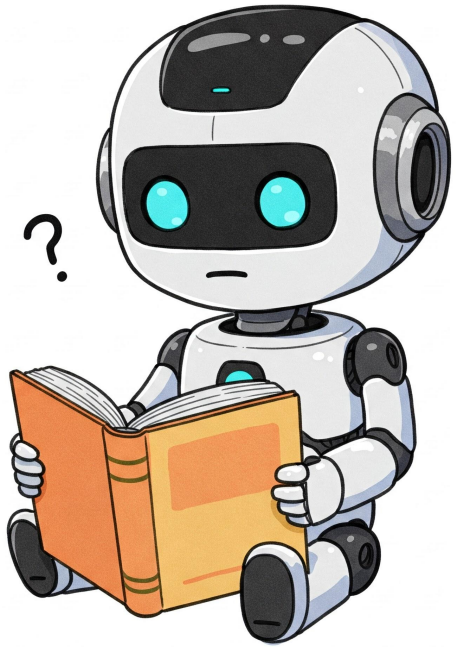
```
(definec m2 (x :tl y :int) :lex
  ;; first component decreases whenever y ≤ len(x),
  ;; second whenever y > len(x)
  (list
    (if (<= y (len x))
        (- (len x) y)
        0)
    (if (> y (len x))
        (- y (len x))
        0)))
```



ChatGPT **read the textbook** and decided to use lex.



ChatGPT **doesn’t find a solution** when the error contents are omitted.



Gemini knows it doesn’t know.

Show thinking

Given the requirement for 100% certainty and the challenges encountered, I cannot confidently provide a correct measure function `m2` and the associated properties without further clarification or constraints on the possible values of `y`. The interaction between the two recursive branches makes standard termination measures non-trivial to apply correctly here.

Claude Concedes

I'd like to concede this problem. I've made multiple attempts but haven't been able to find a measure function that handles all the edge cases properly. This is a challenging termination proof that requires a more carefully constructed measure function than what I've proposed.

m4

Let's look at another problem.

```
(definec f4 (a :nat b :int c :tl) :int
  (cond ((zp a)       (len c))
        ((< a b)      (f4 a (- b 4) c))
        ((> (- b) 2) (+ a (f4 (1- a) (1+ b) (rest c))))
        ((endp c)     (+ a b))
        (t            (f4 a (1+ b) (rest c)))))
```

m4 - Claude - Attempt 1

```
(definec m4 (a :nat b :int c :tl) :nat
  (+ (* 10000 a)
     (* 100 (len c))
     (if (< a b)
         (min 50 (max 0 (- b a)))
         0)))
```

ACL2s gives a counterexample for c=nil, b=124, a=1.

m4 - Claude - Attempt 2

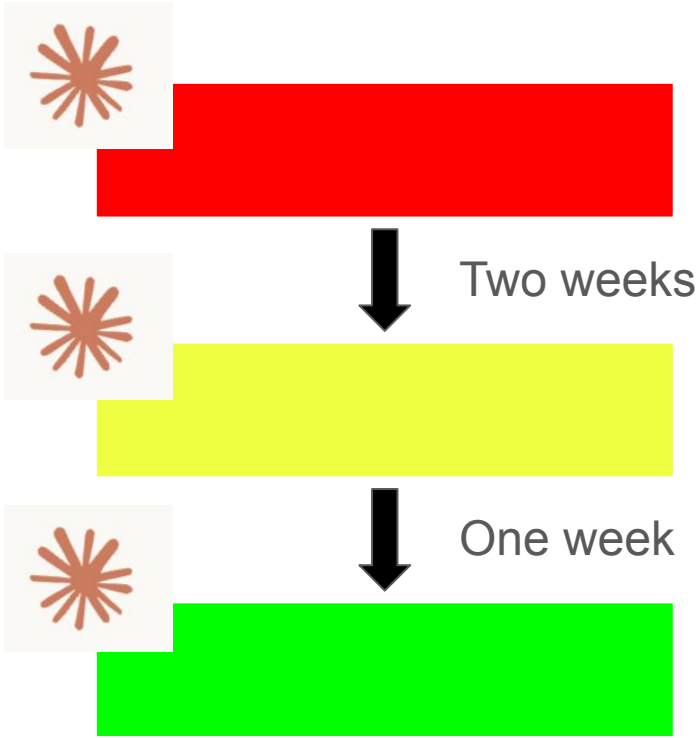
```
(definec m4 (a :nat b :int c :tl) :nat
  (+ (* 10000 a)
     (* 100 (len c))
     (if (< a b)
         (+ 10000 b)  ; Add a large constant to ensure this is positive
         0)))
```

z adding a large constant isn't going to cut it, because what if b is larger than the constant? remember, this is a proof we are working on.

m4 - Claude - Attempt 3

```
(definec m4 (a :nat b :int c :tl) :nat
  (+ (* 10000 a)
     (* 100 (len c))
     (if (< a b)
         (- b a)
         0)))
```

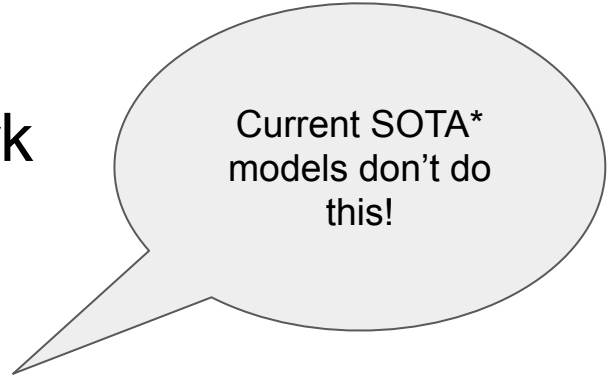
Claude gets it right on the third try.



Caveats

- **Three weeks ago**, Claude cannot solve m4.
- **A week ago**, Claude can solve m4 with iterative error feedback.
- **Today**, Claude can oneshot m4.
- **Why?** Is it using previous chat conversations? Has it got better?

Conclusions & Future Work



Good

+ Models integrate error feedback.
+ Models understand ACL2(s).
+ Models do well on exams.

Future Work

- Let models run ACL2 and iterate.**
- Why do models do poorly on measure functions?

**ACL2s Systems Programming, Andrew Walter & Panagiotis Manolios

*DeepSeek v2, Godel Prover, Kimina