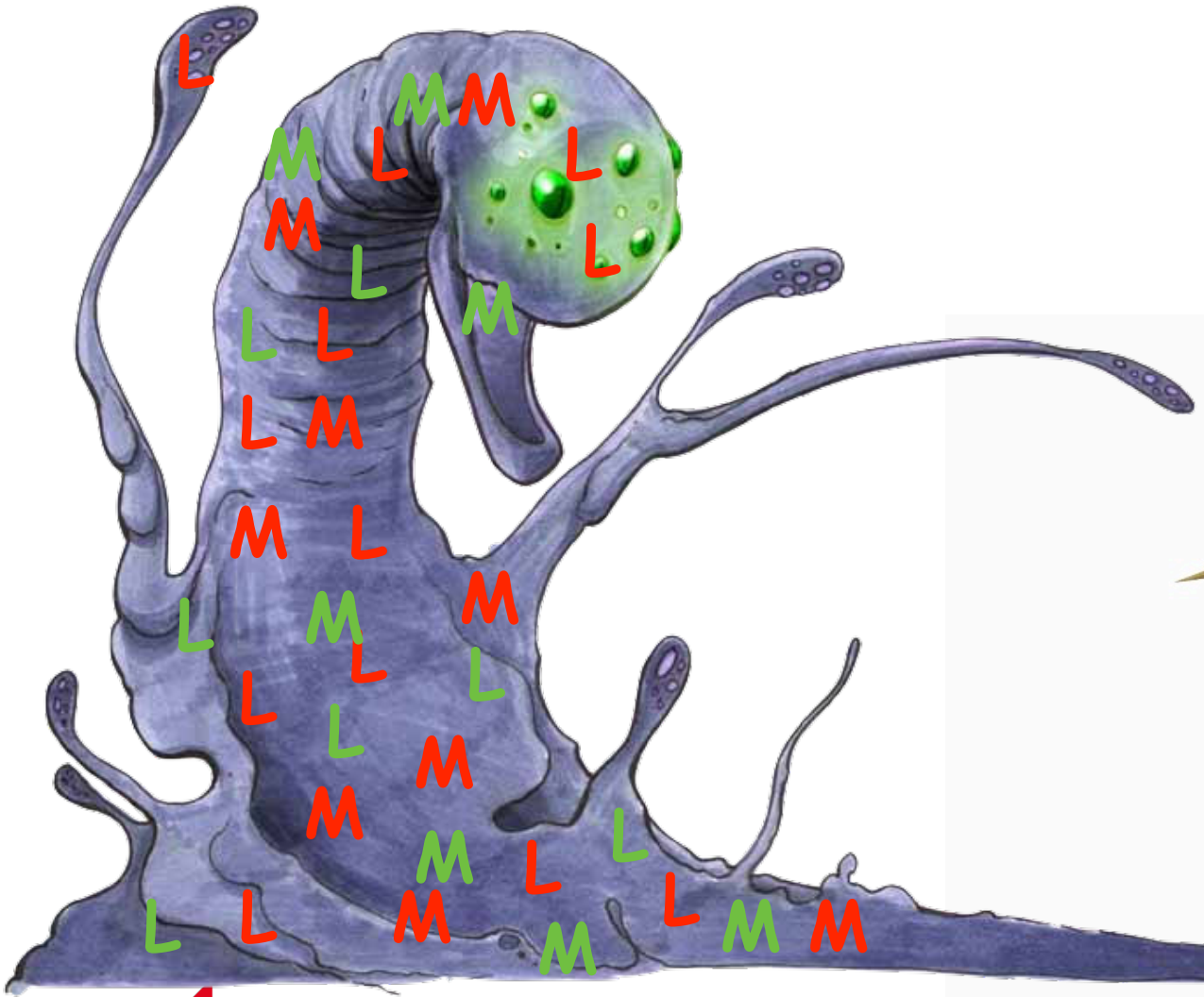
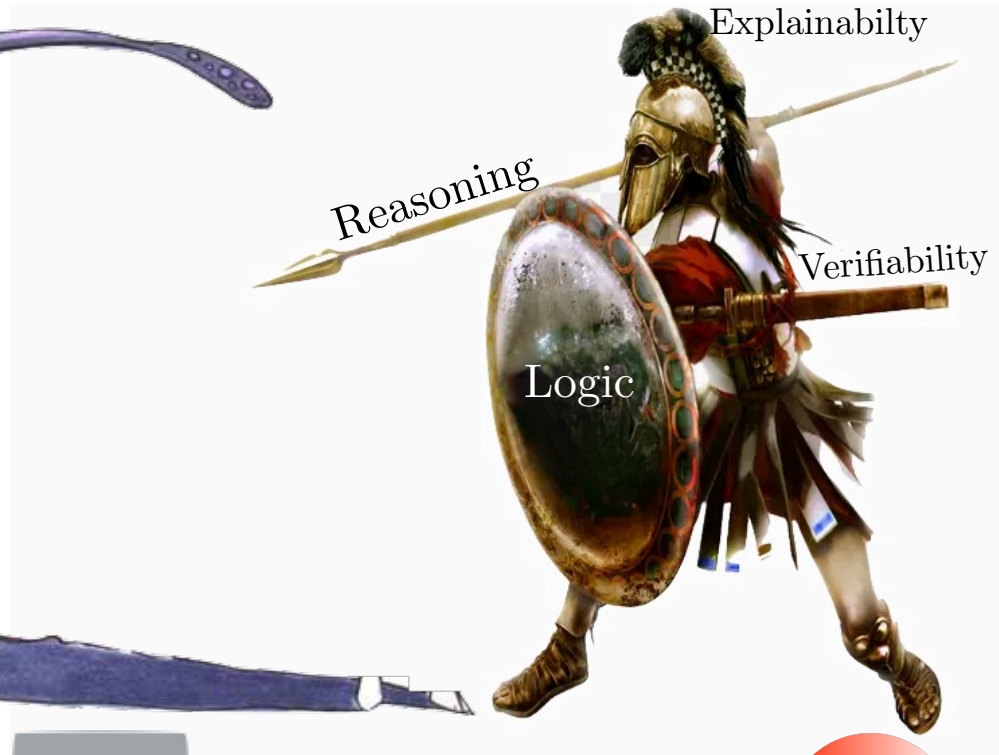
Superposition and E (and ATP in general) in the Age of "Artificial Intelligence"

Stephan Schulz



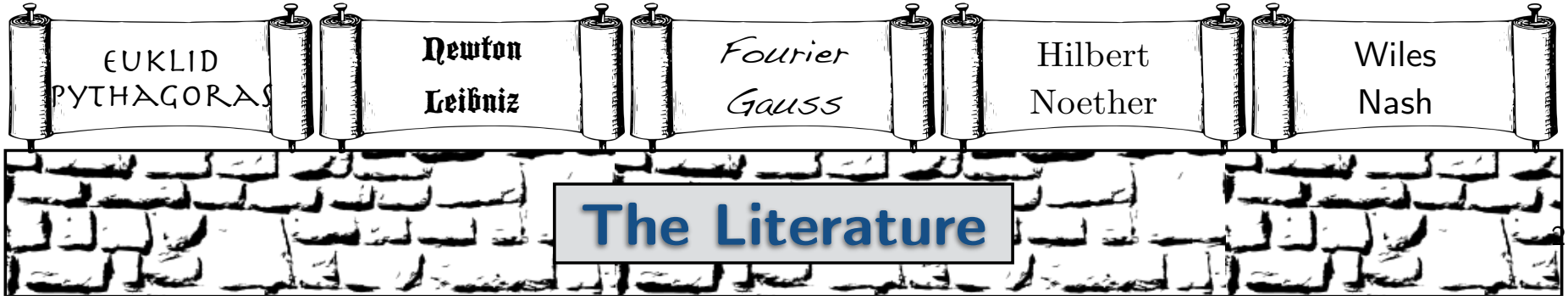


DHBW

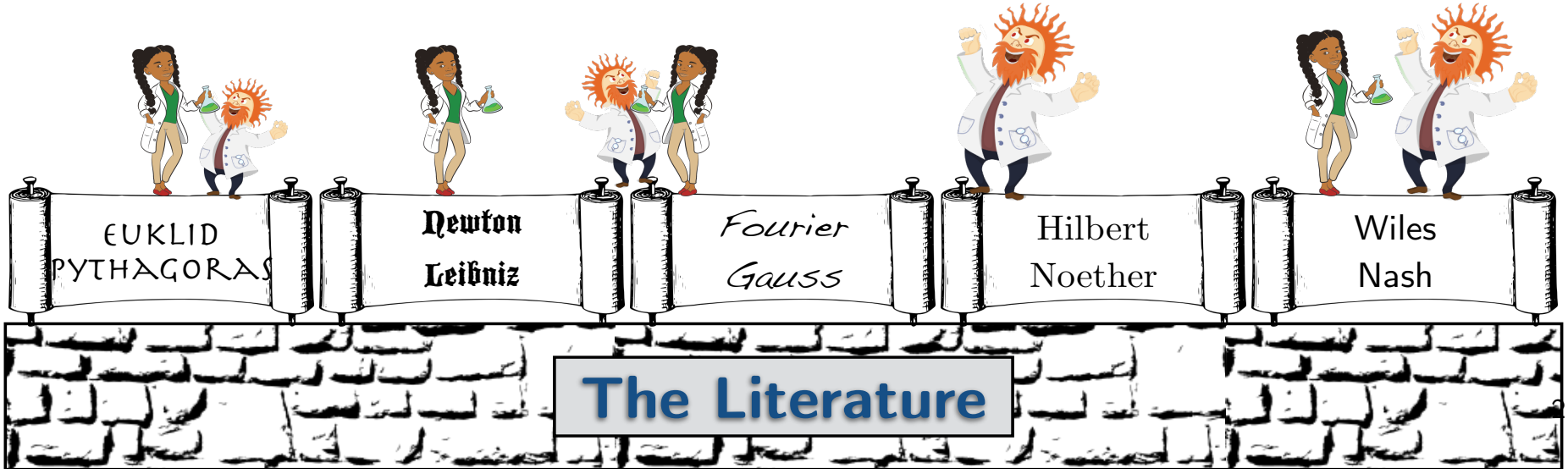
ZfKI

E

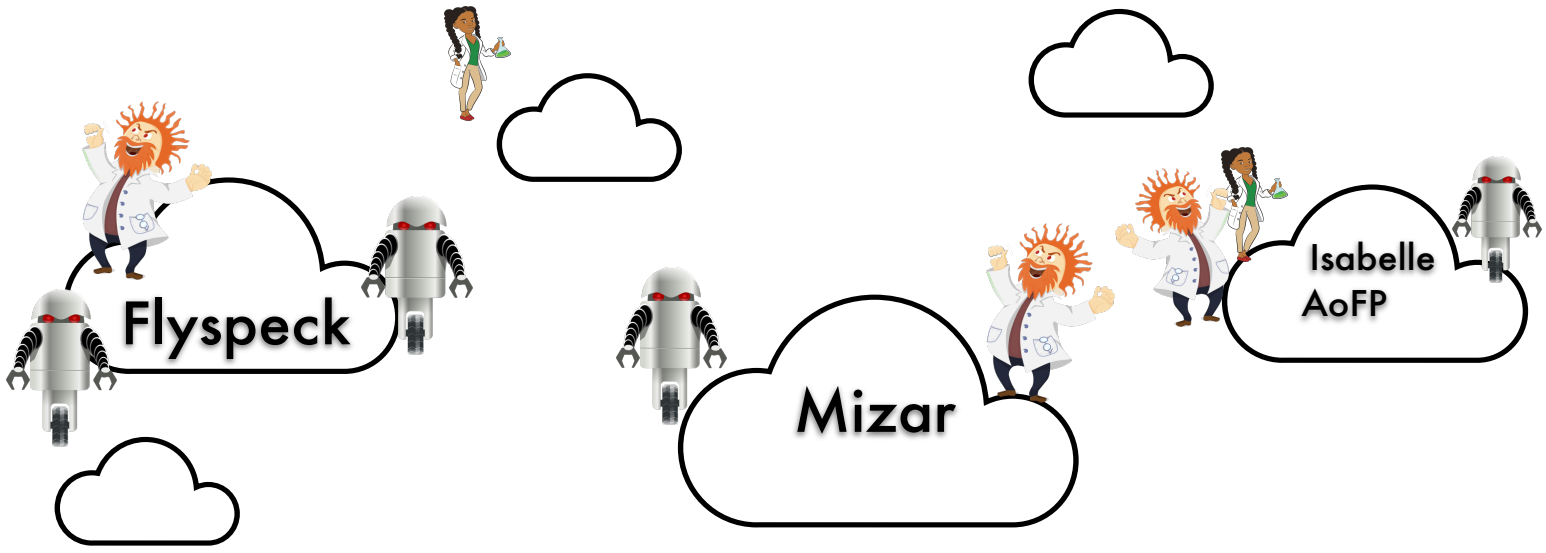
The math ecosystem (ca. now)

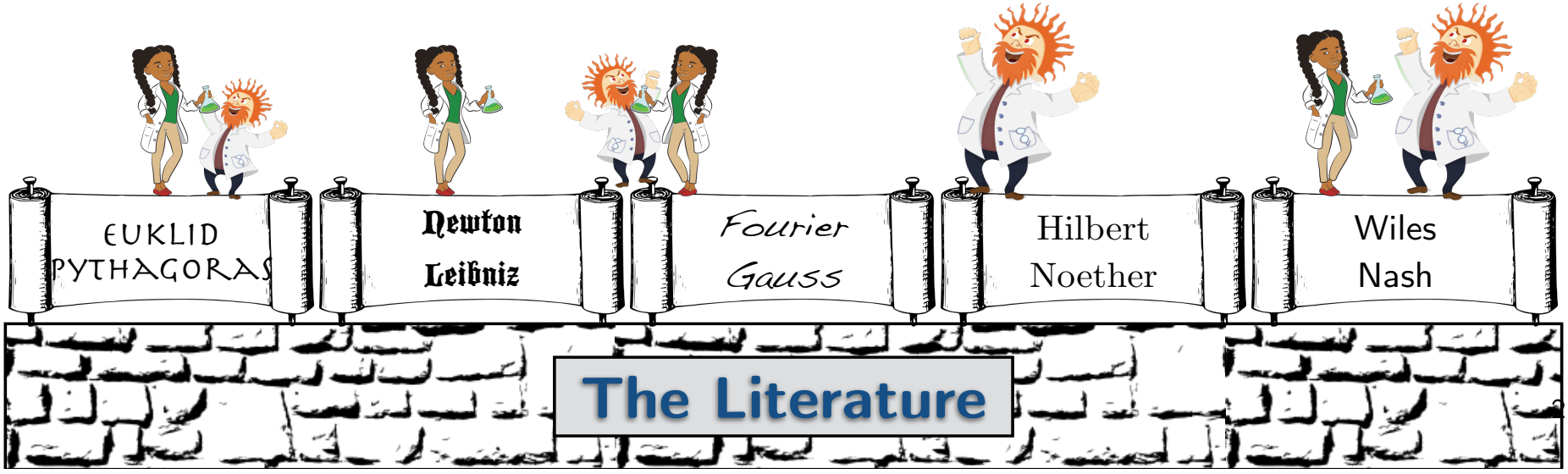


The math ecosystem (ca. now)



The math ecosystem (ca. now)

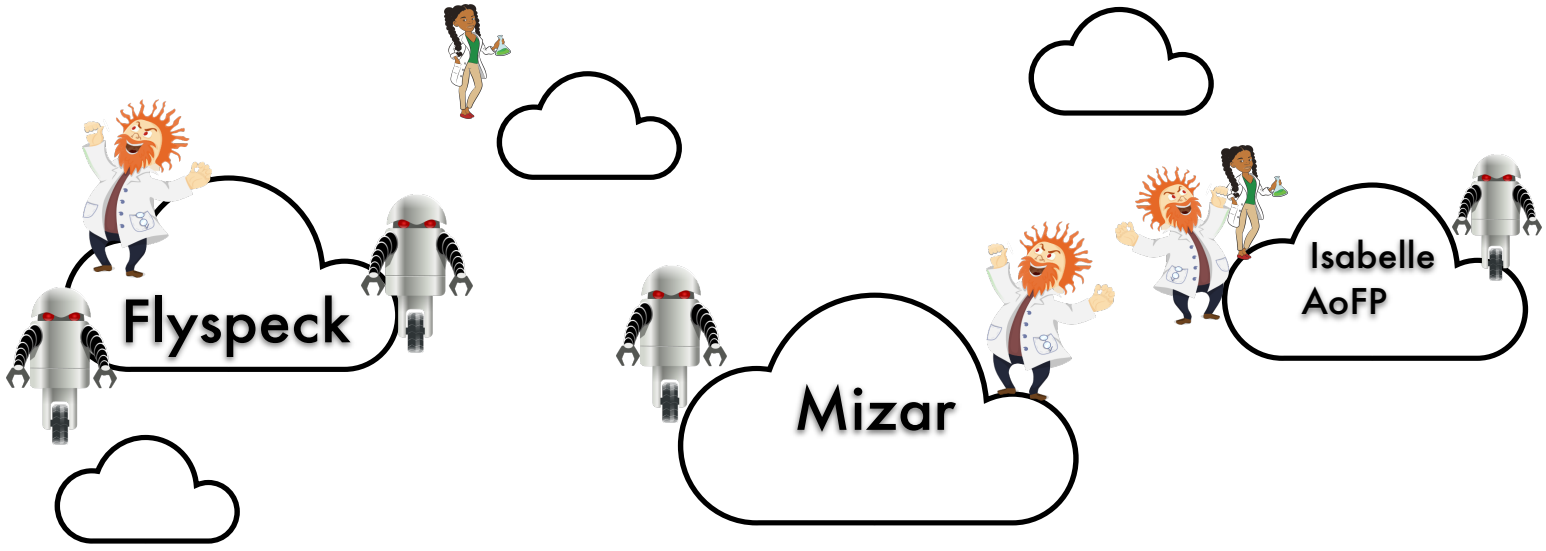


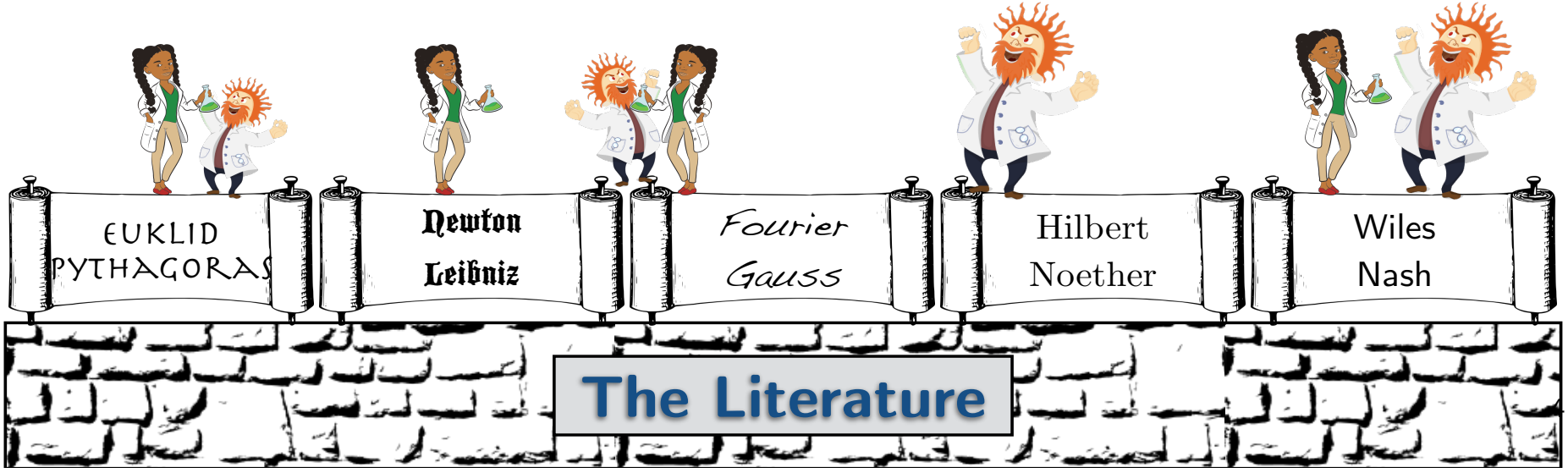


Enter

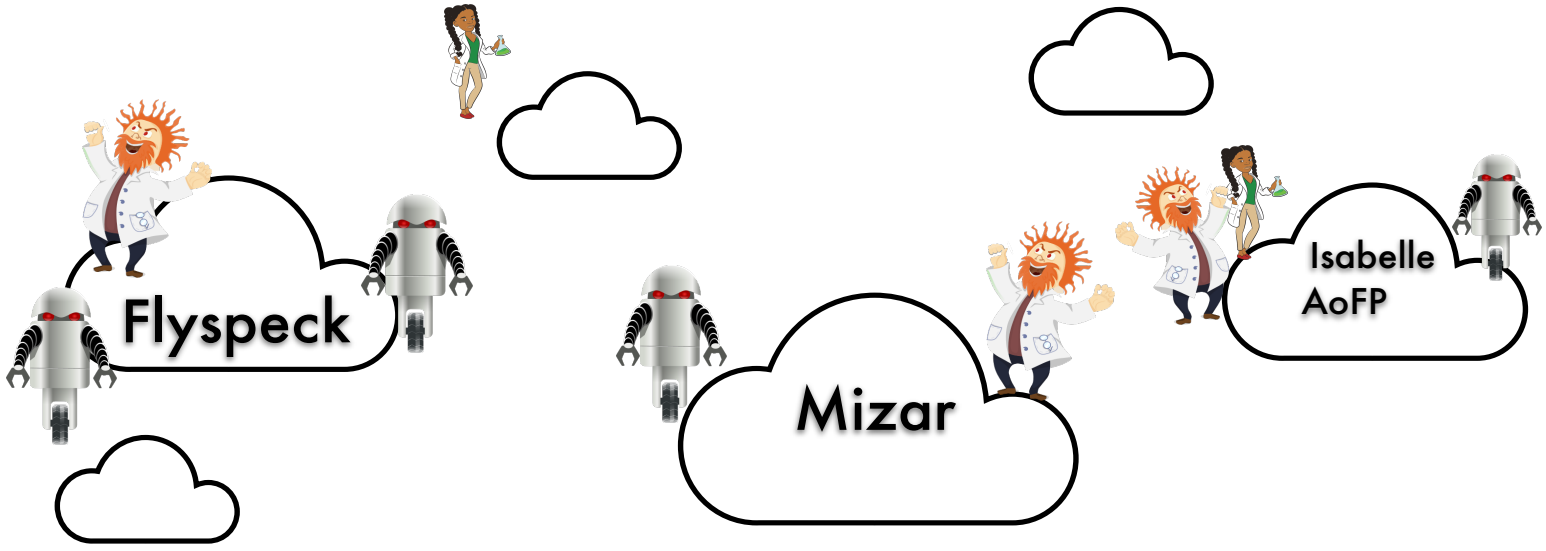
THE AGE OF ARTIFICIAL INTELLIGENCE

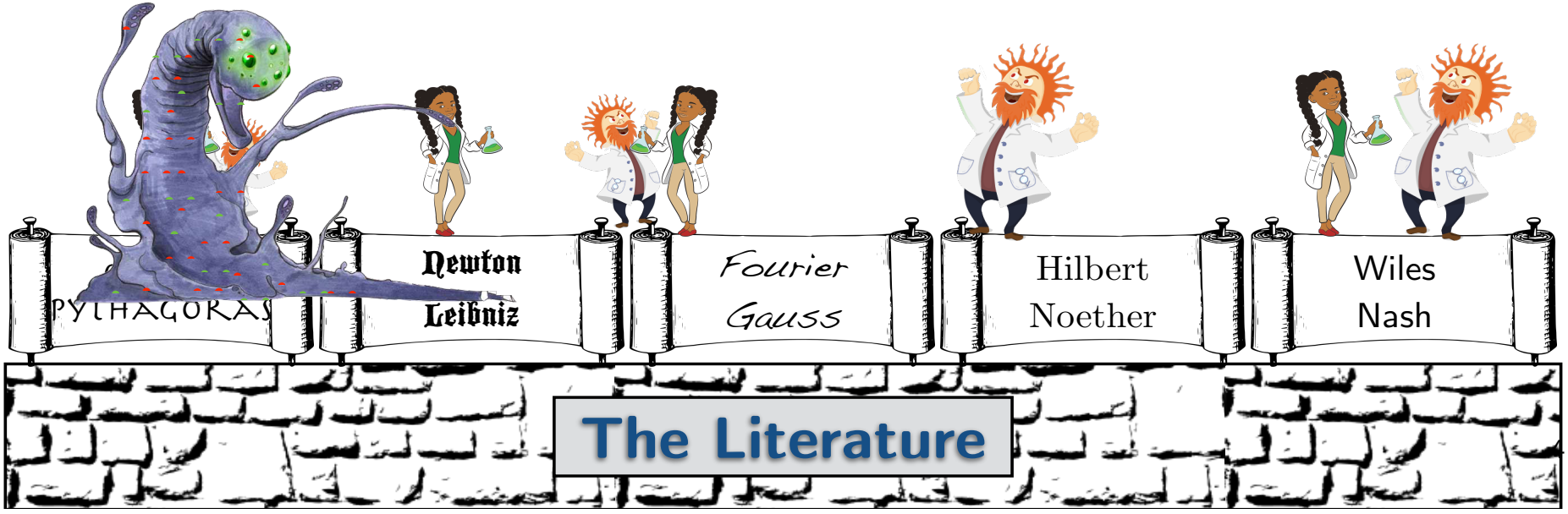
The future math ecosystem (OpenAI/xAI/...version)



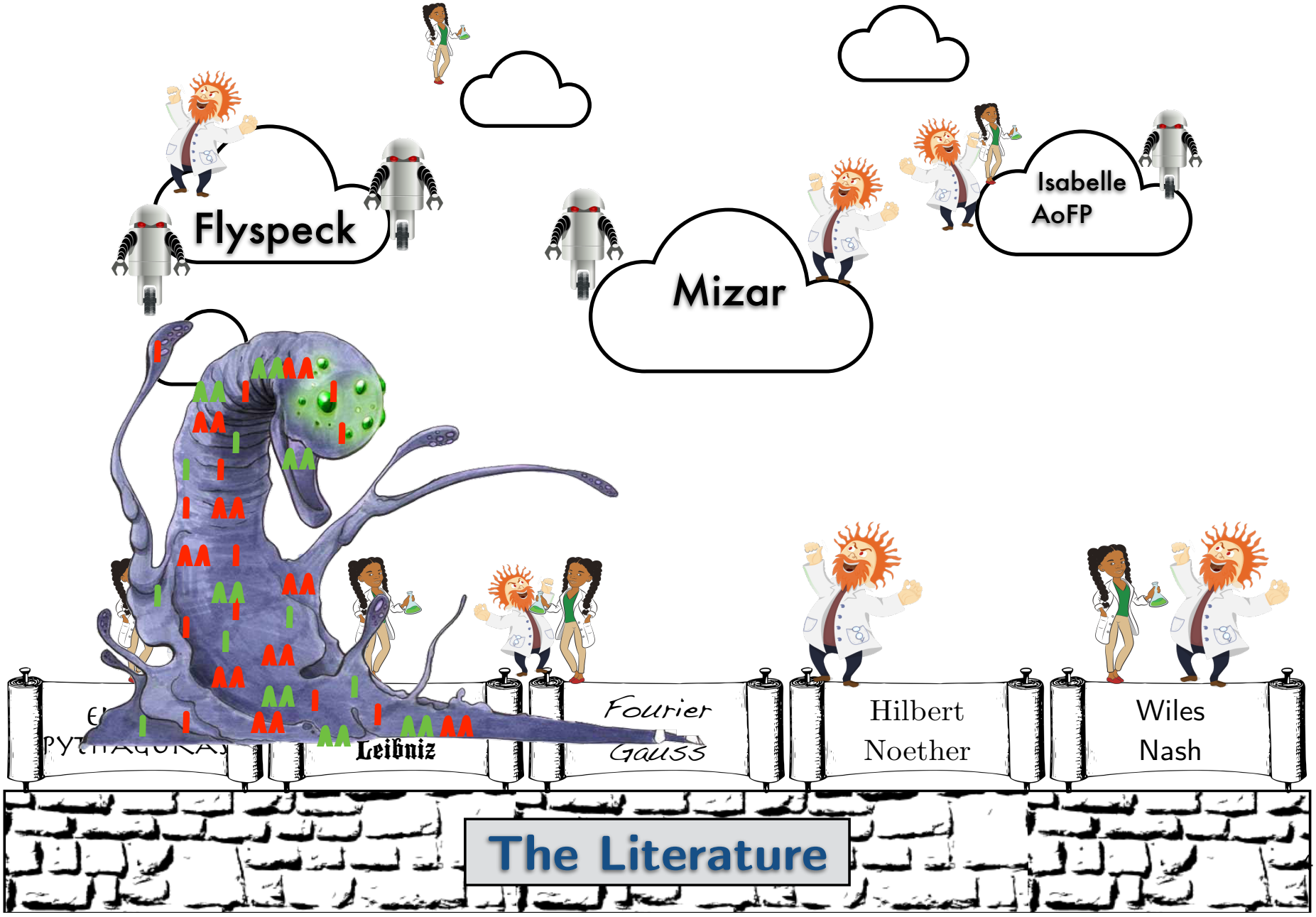


The future math ecosystem (OpenAI/xAI/...version)

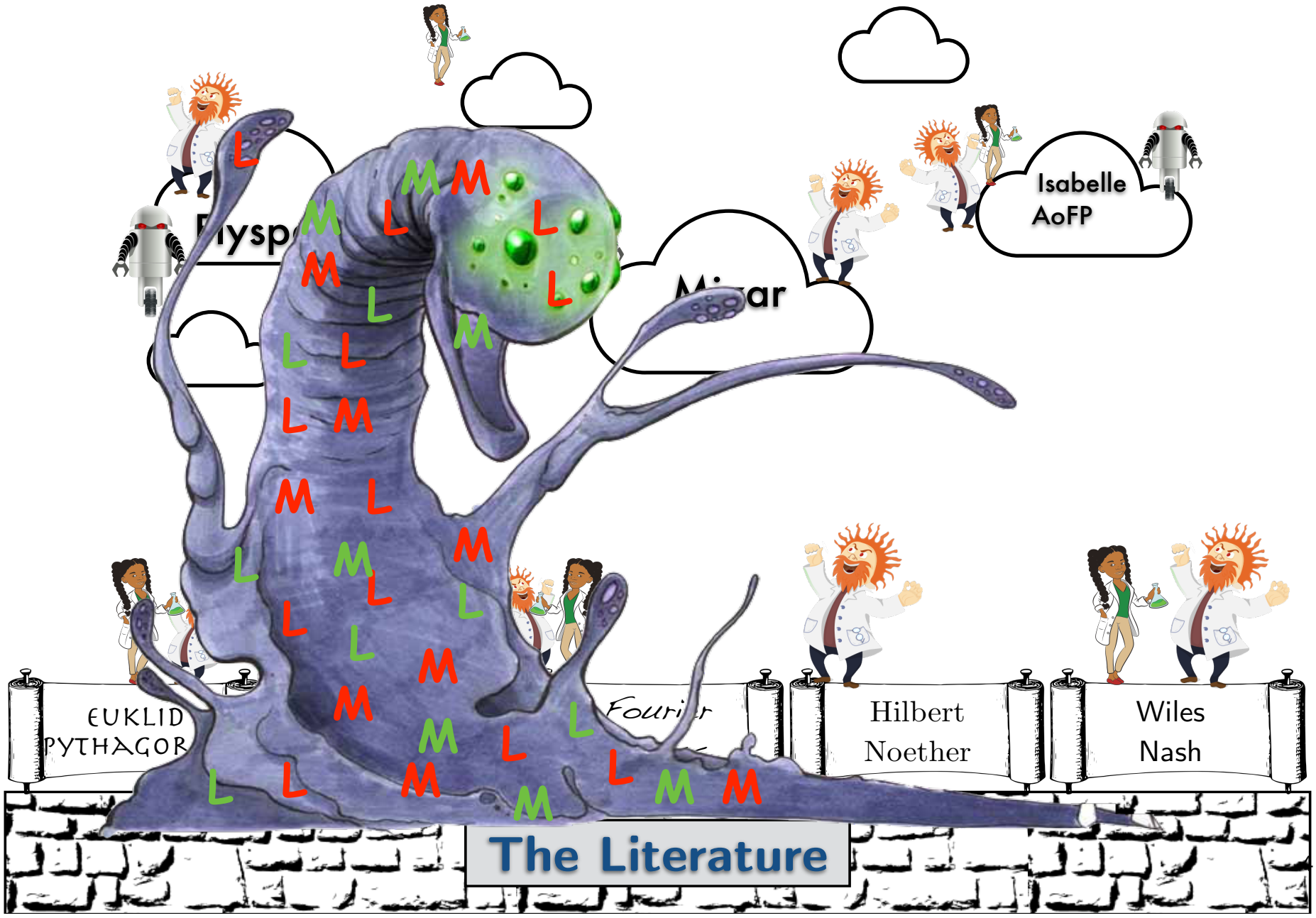




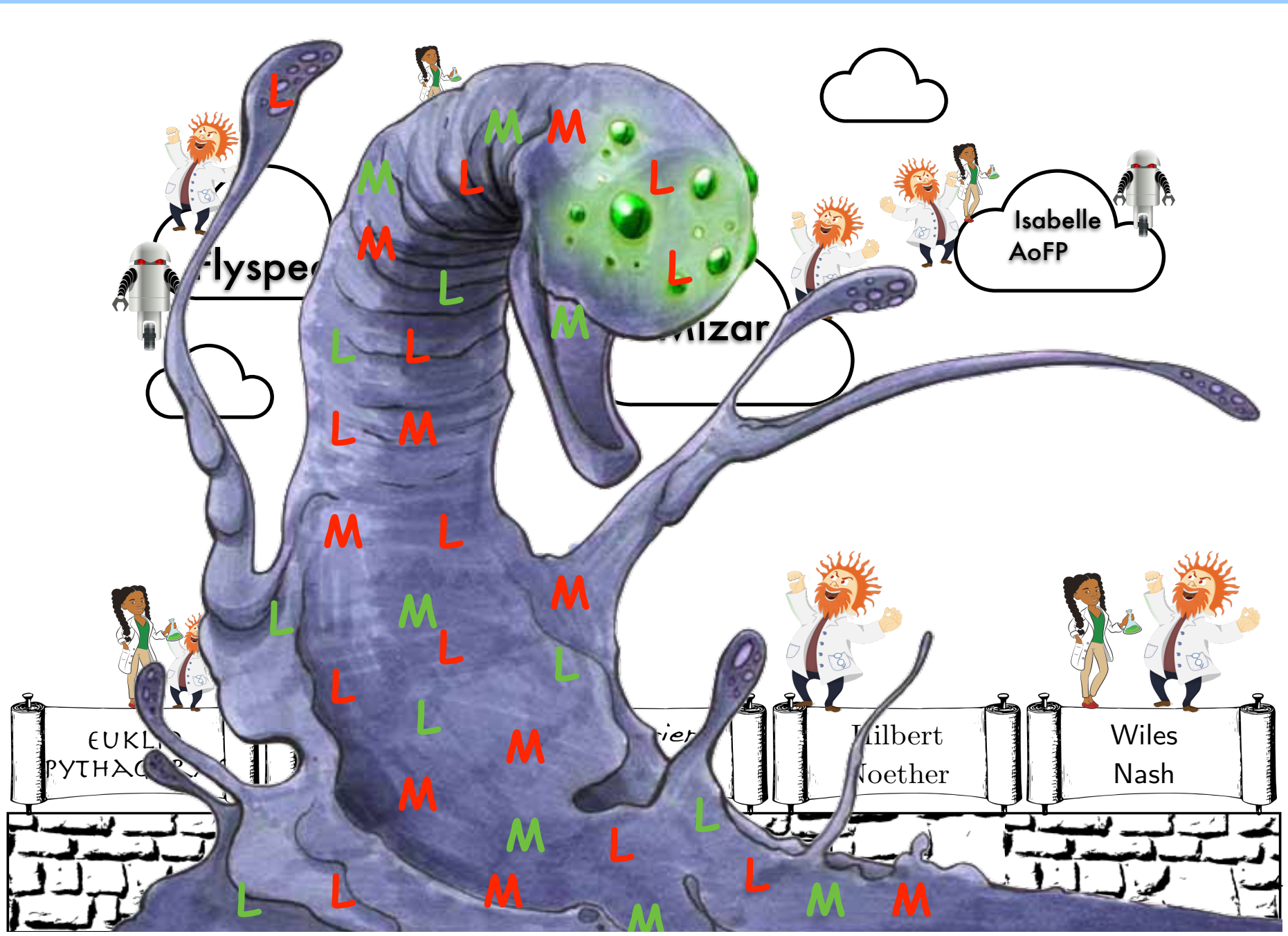
The future math ecosystem (OpenAI/xAI/...version)



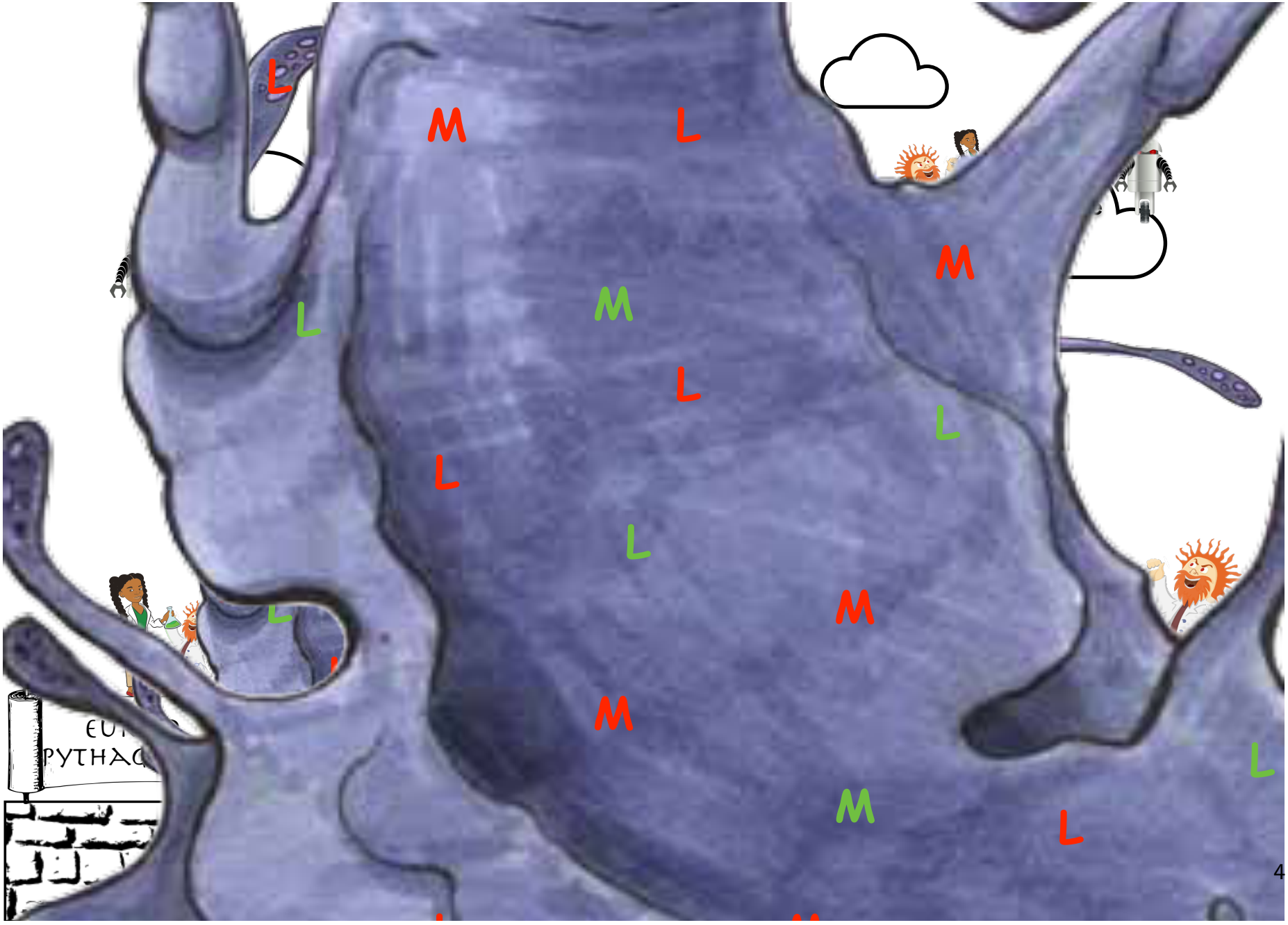
The future math ecosystem (OpenAI/xAI/...version)



The future math ecosystem (OpenAI/xAI/...version)



The future math ecosystem (OpenAI/xAI/...version)



This is not the future!

Large language models are amazing. . .

Large language models are amazing. . .

- . . . at processing *language*

Bold claims

Mathematics is *not* a language

Claiming otherwise is the same confusion as that between being able to program, and knowing a programming language (which gave us the horror that is COBOL). Or between knowing the rules of chess and being a competent chess player.

Bold claims

Mathematics is *not* a language

Claiming otherwise is the same confusion as that between being able to program, and knowing a programming language (which gave us the horror that is COBOL). Or between knowing the rules of chess and being a competent chess player.

- Of course there is *a* mathematical language
  - . . . with a lot of national dialects
  - . . . with a lot of field-specific dialects

Bold claims

Mathematics is *not* a language

Claiming otherwise is the same confusion as that between being able to program, and knowing a programming language (which gave us the horror that is COBOL). Or between knowing the rules of chess and being a competent chess player.

- Of course there is *a* mathematical language
  - . . . with a lot of national dialects
  - . . . with a lot of field-specific dialects

Mathematics is, at the core, the definition and exploration of new structures and their properties.

On LLMs

- LLMs learn conditional probability distributions on language
  - They do not learn abstract theory
  - They do not learn abstract reasoning
  - They do not learn calculations (except by memorization)
- LLMs can't handle abstract reasoning
  - Changing terms changes outcome (*A chat with Bard*)
  - Adding unrelated information leads to failure (*GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models*)
  - Hallucinations are likely unavoidable (*LLMs Will Always Hallucinate, and We Need to Live With This*)
- LLMs are essentially reproductive, not creative

On LLMs

- LLMs learn conditional probability distributions on language
  - They do not learn abstract theory
  - They do not learn abstract reasoning
  - They do not learn calculations (except by memorization)
- LLMs can't handle abstract reasoning
  - Changing terms changes outcome (*A chat with Bard*)
  - Adding unrelated information leads to failure (*GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models*)
  - Hallucinations are likely unavoidable (*LLMs Will Always Hallucinate, and We Need to Live With This*)
- LLMs are essentially reproductive, not creative

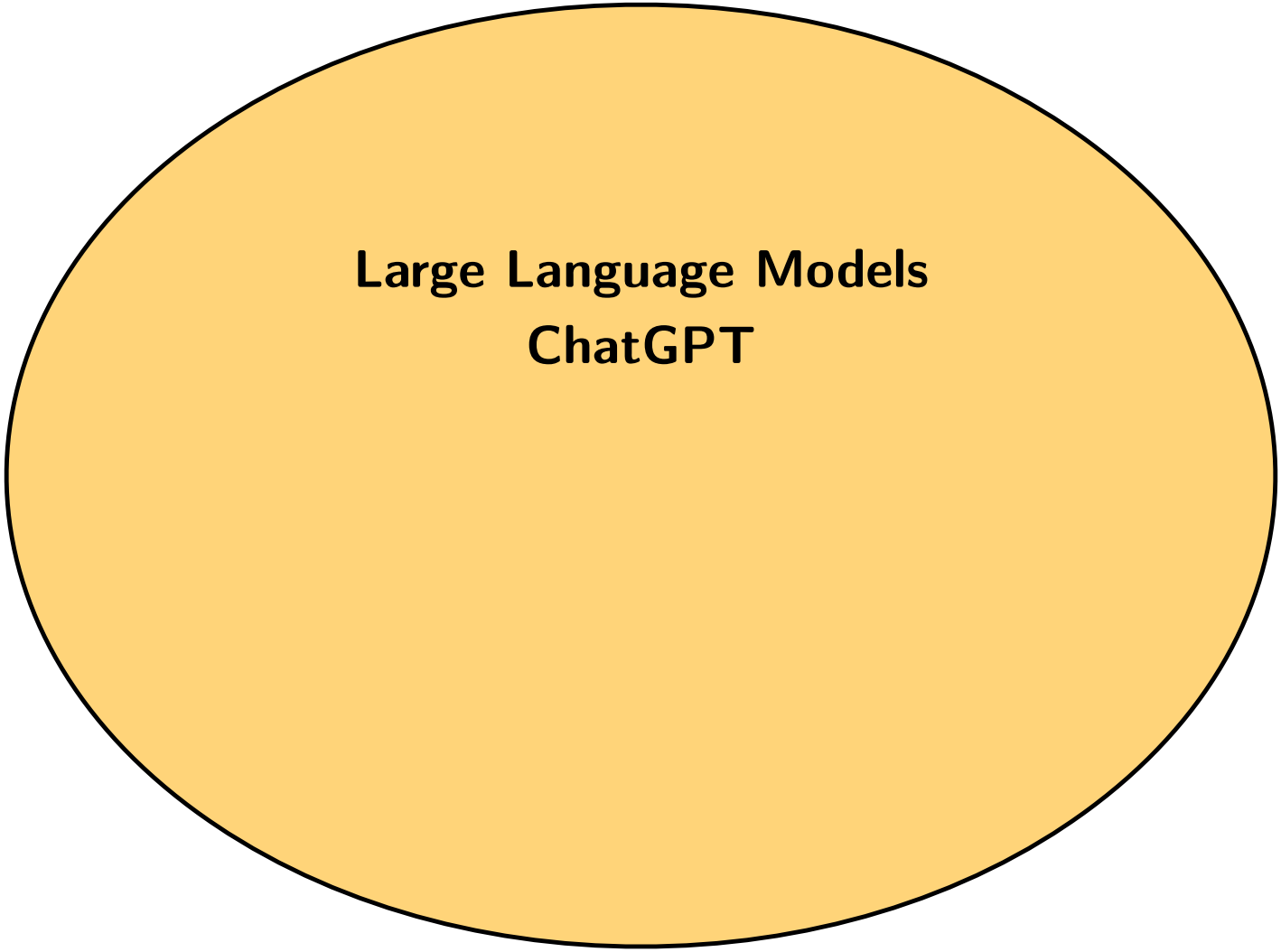
LLMs can't do (new) mathematics!

On LLMs

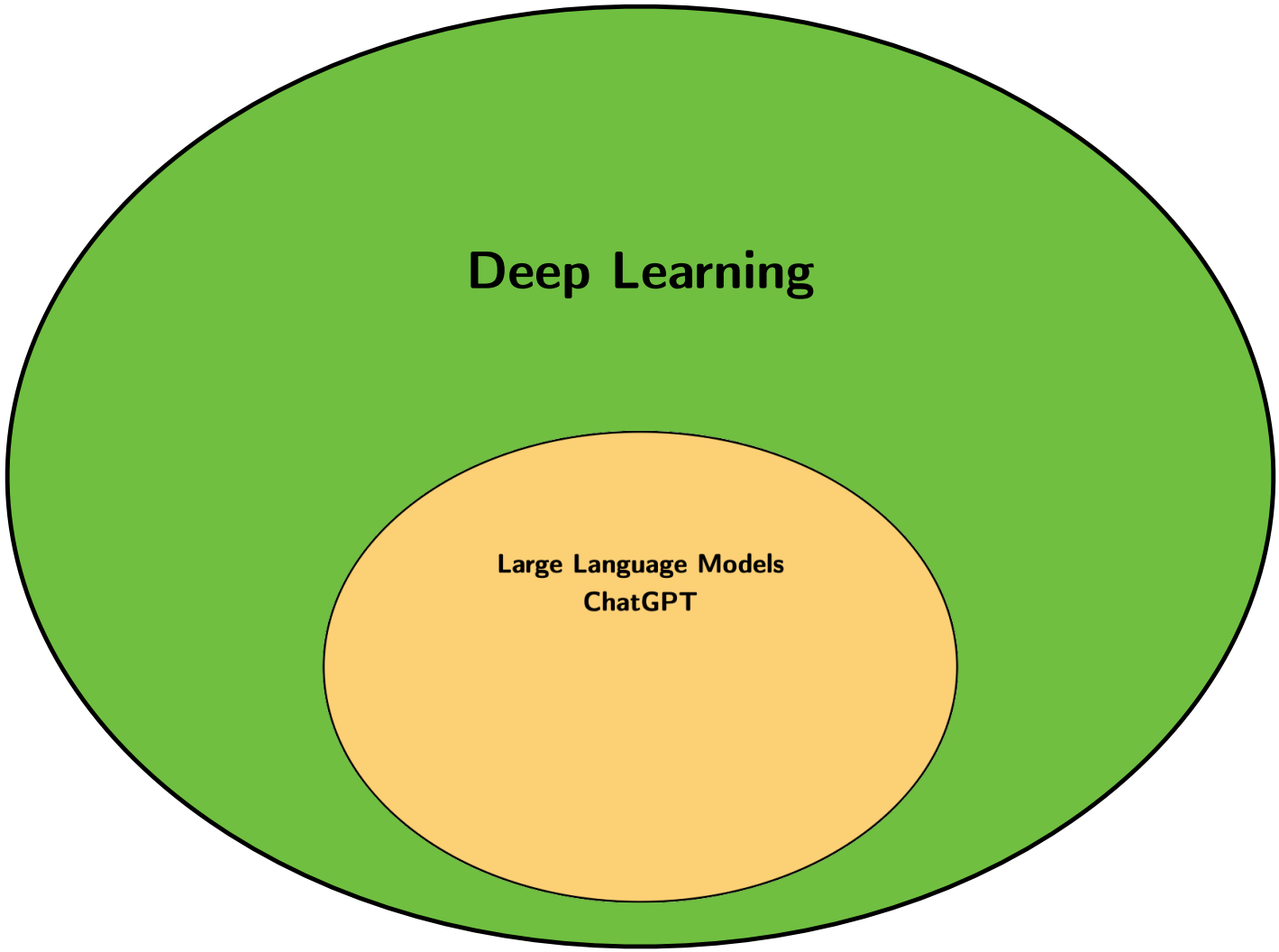

THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

If LLMs (alone) are not the future, what is?

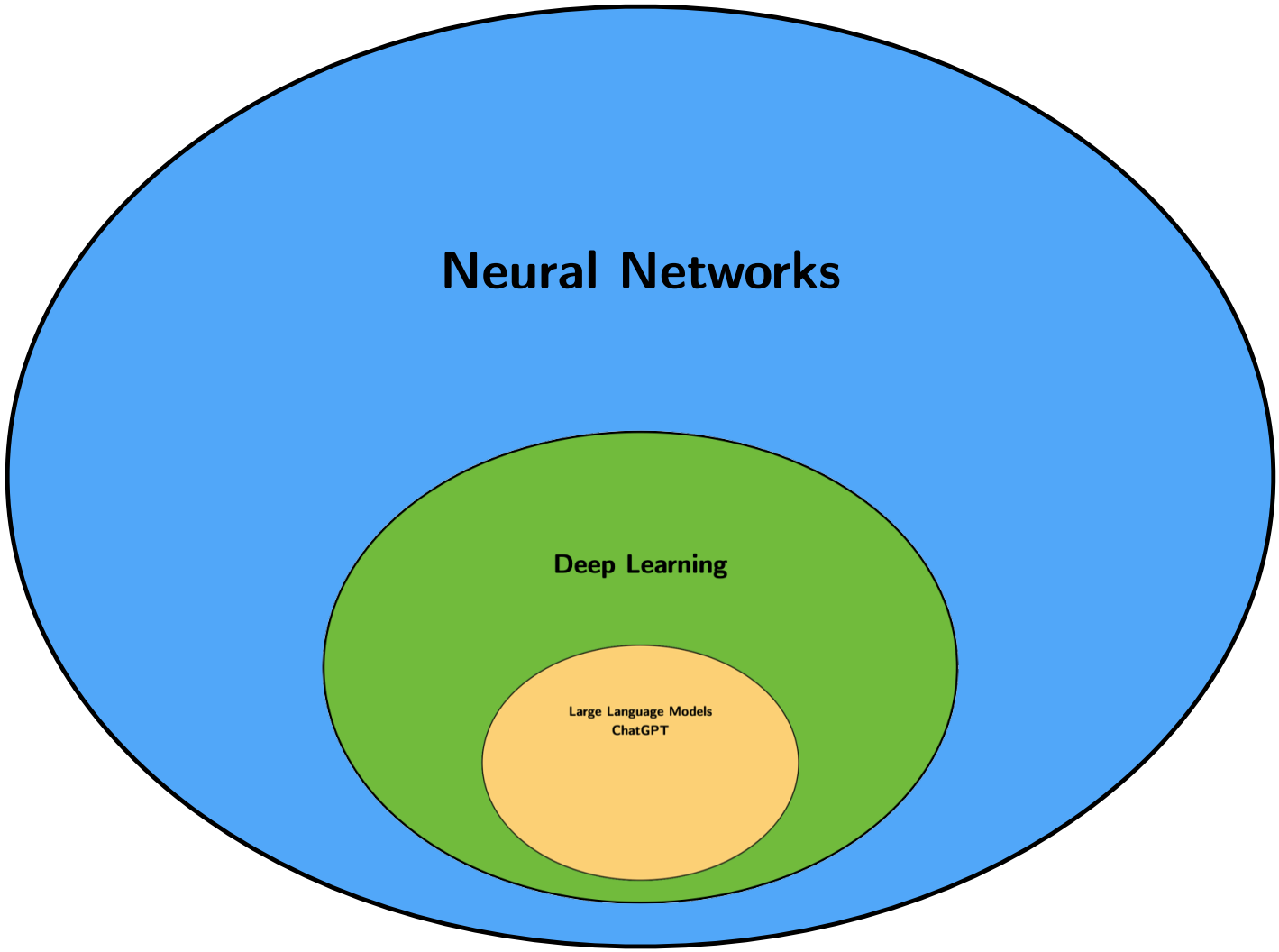
ChatGPT is not AI



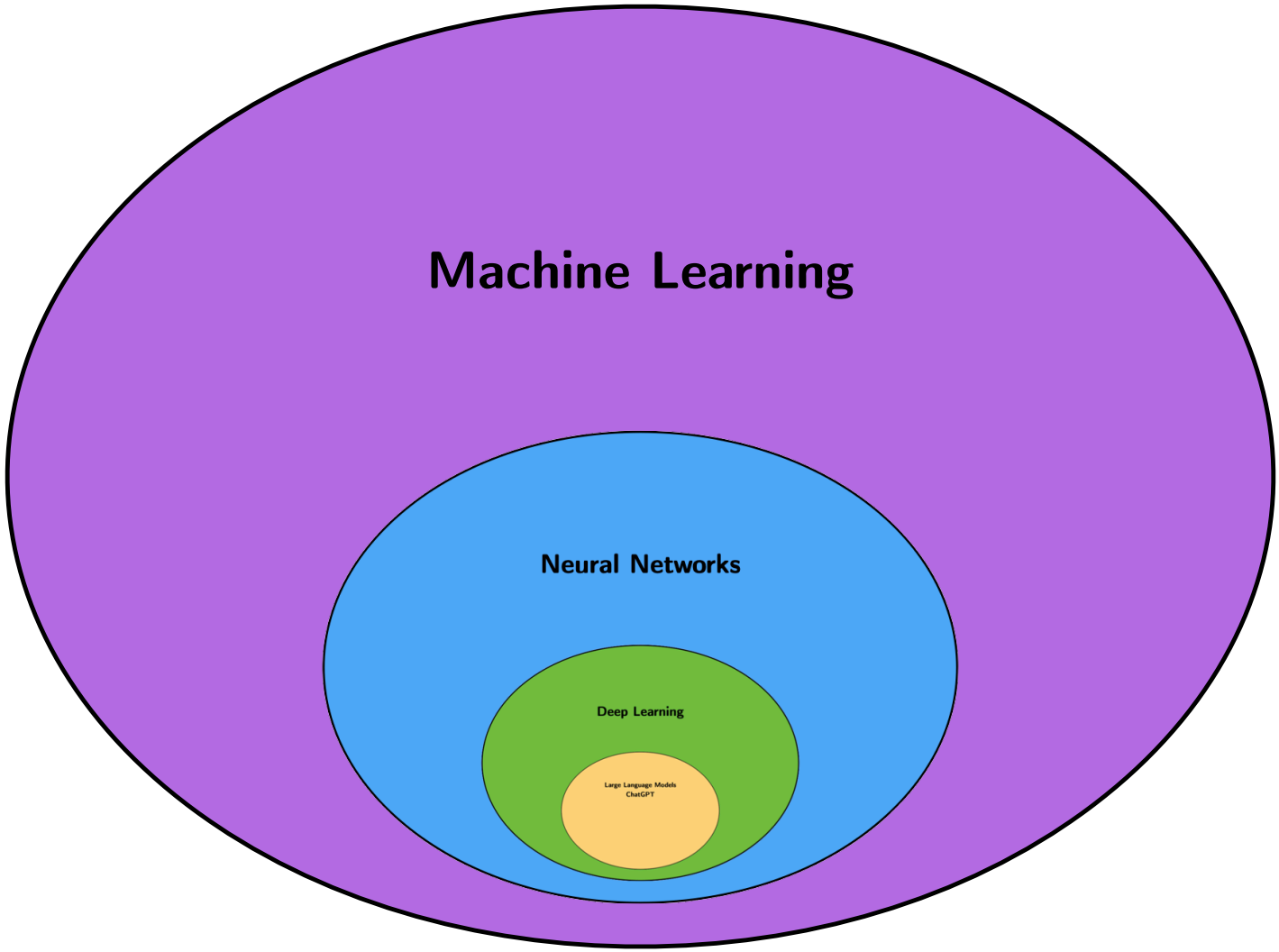
ChatGPT is not AI



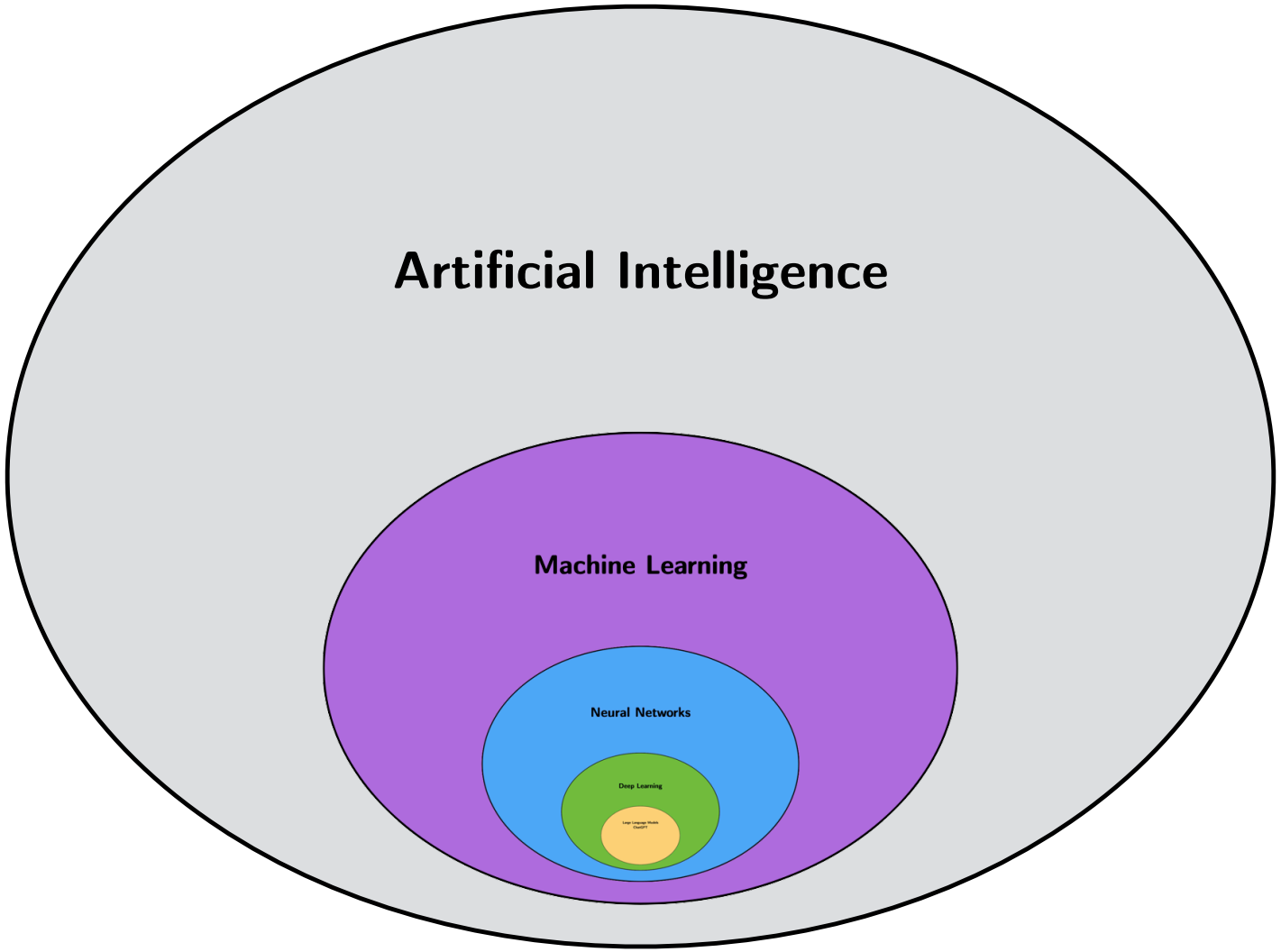
ChatGPT is not AI



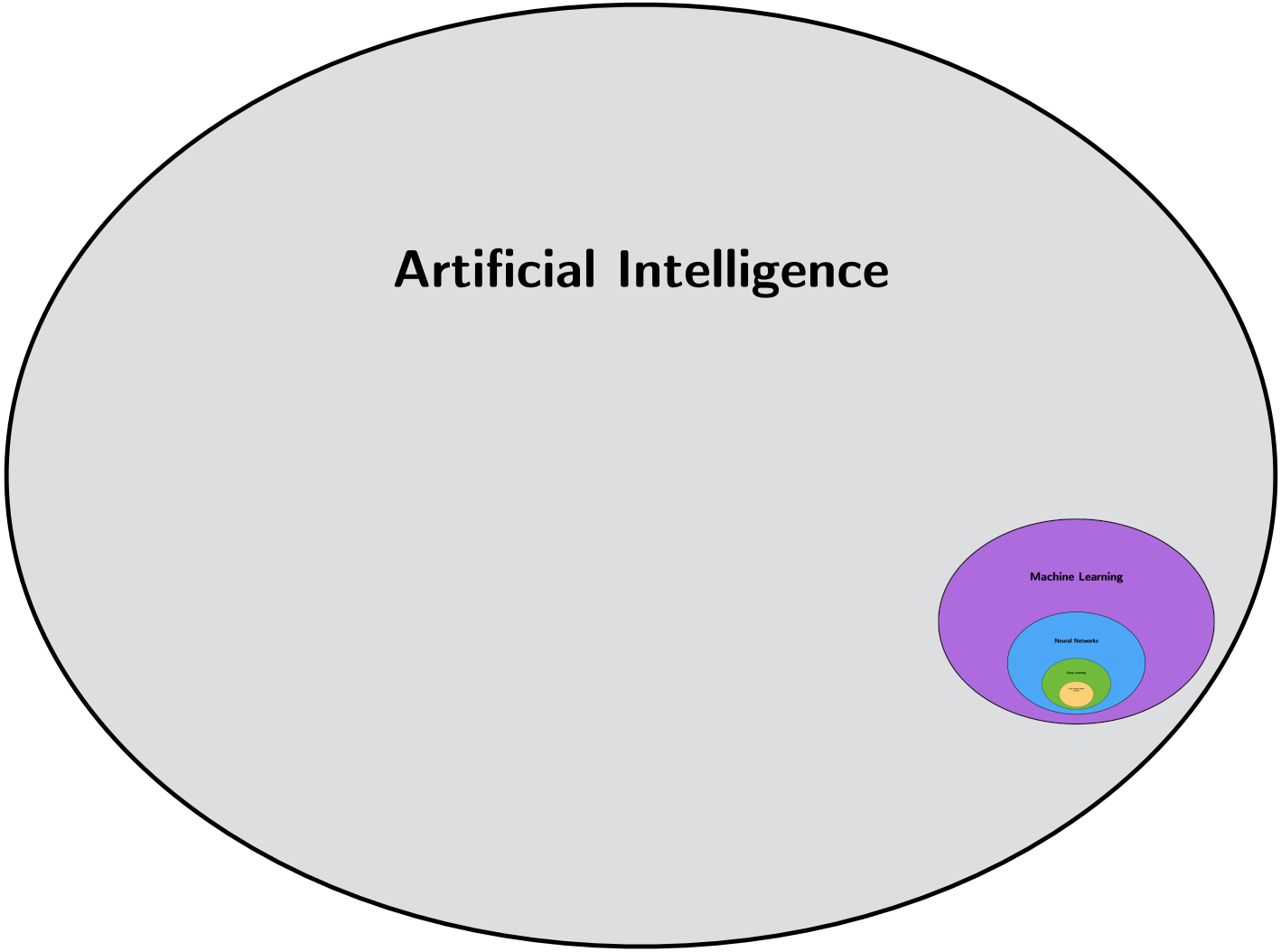
ChatGPT is not AI



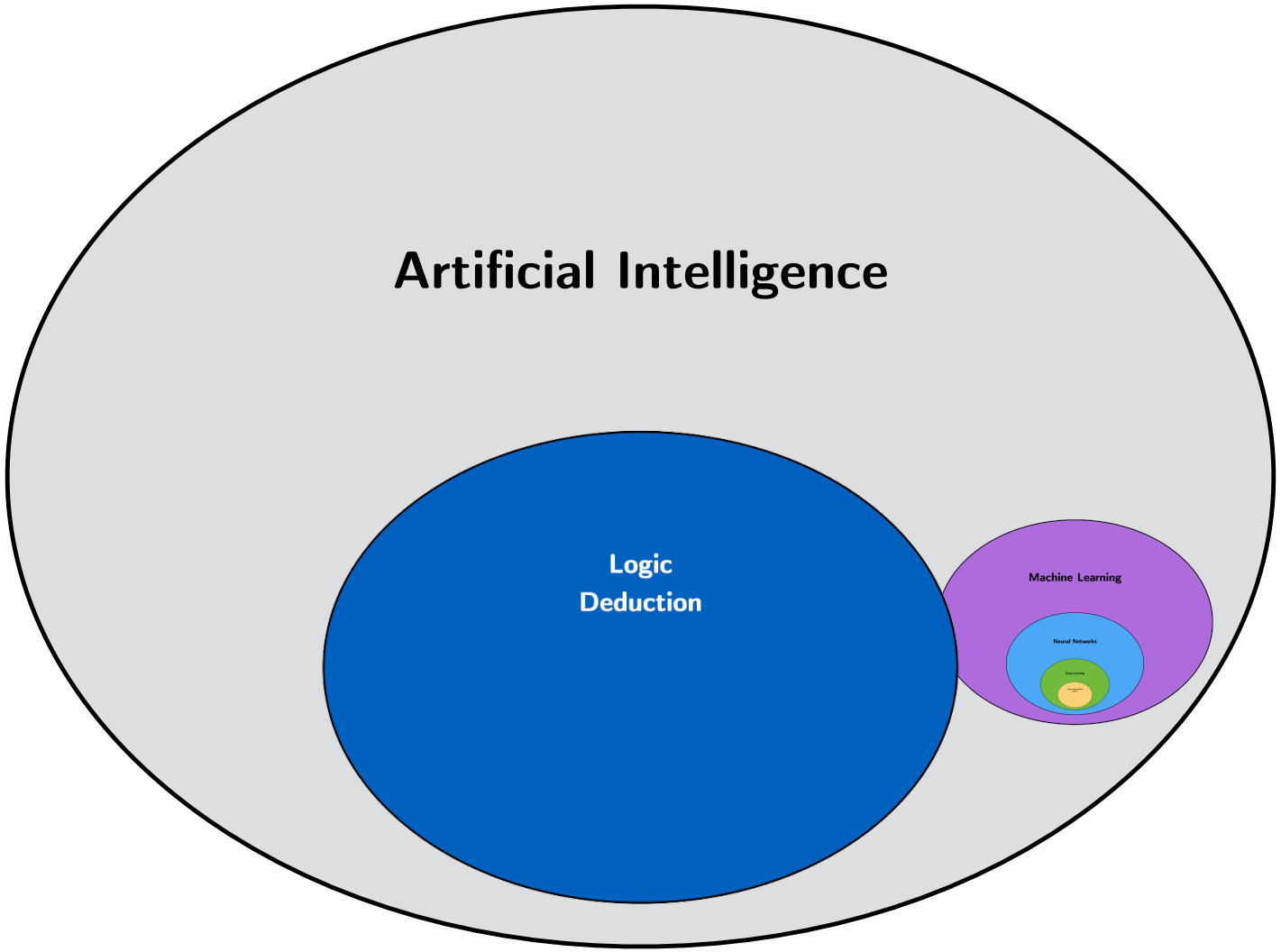
ChatGPT is not AI



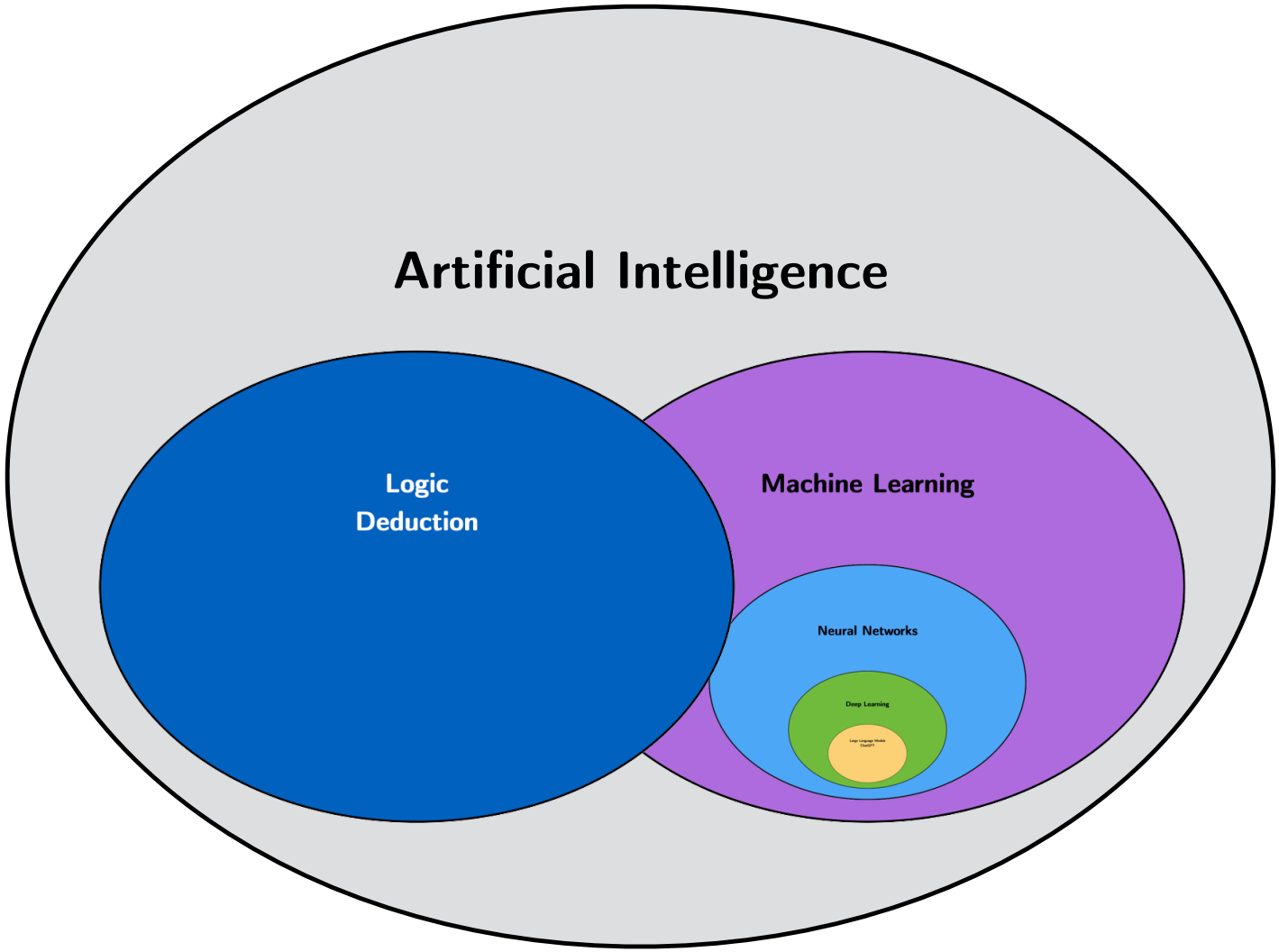
ChatGPT is not AI



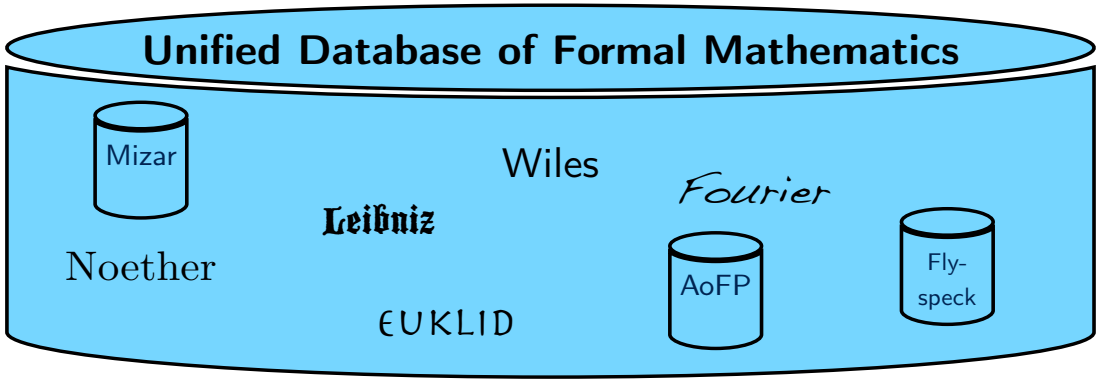
ChatGPT is not AI

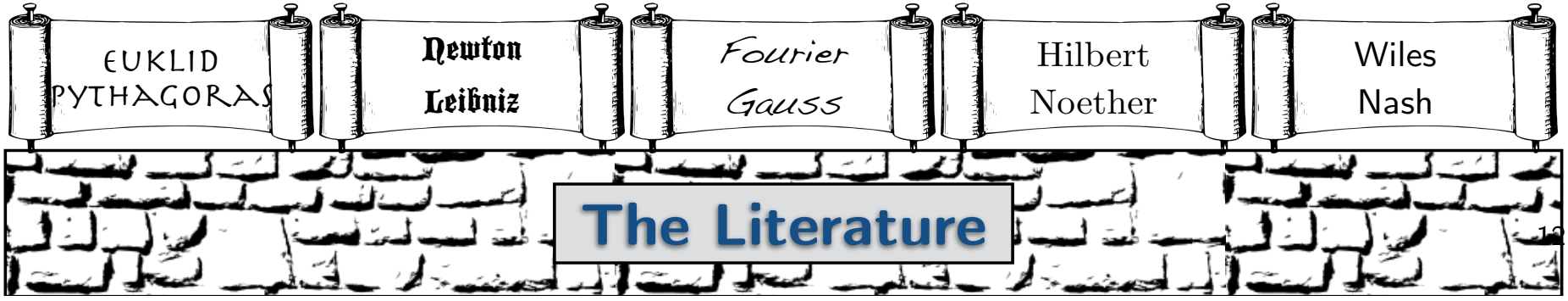


ChatGPT is not AI



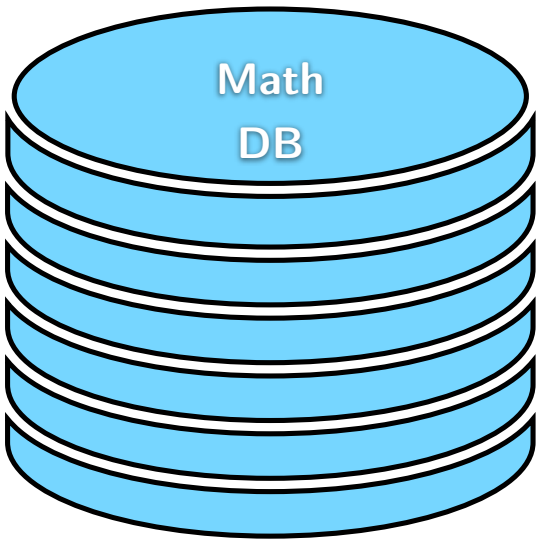
The Future Math Ecosystem (Utopia)



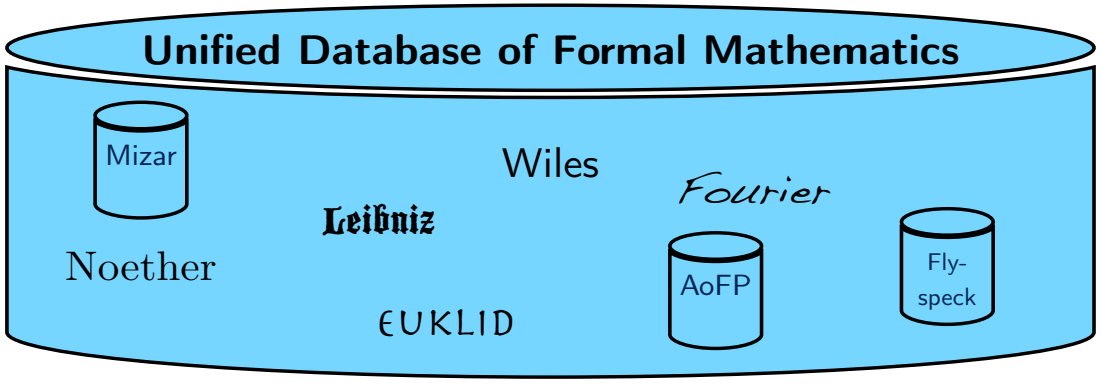


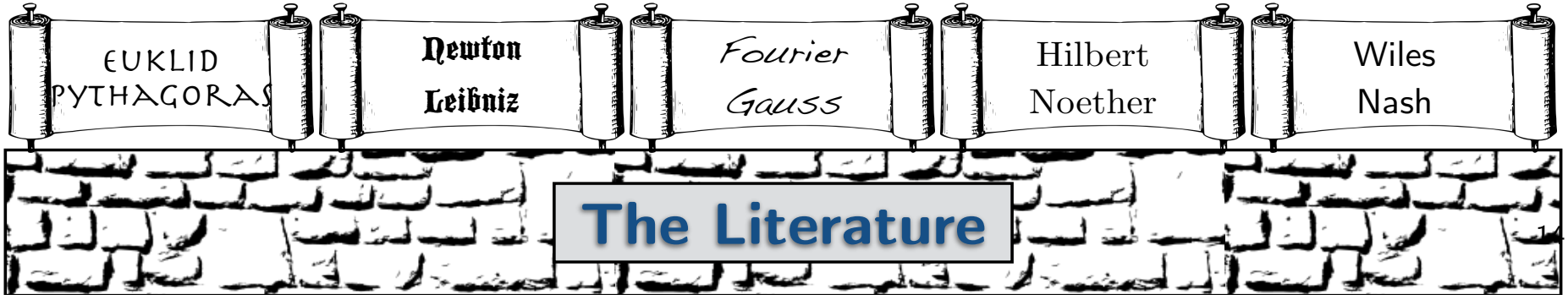
The Formal Mathematics Database

- Expressed in symbolic logic
  - Clear syntax
  - Clear semantics
- Unified
  - One format (or compatible formats)
  - Globally consistent
- Structured
  - Different domains
  - Compositional
  - Searchable
- (Increasingly) comprehensive
  - *"The place to be"*
  - One size fits all (because it's flexible)

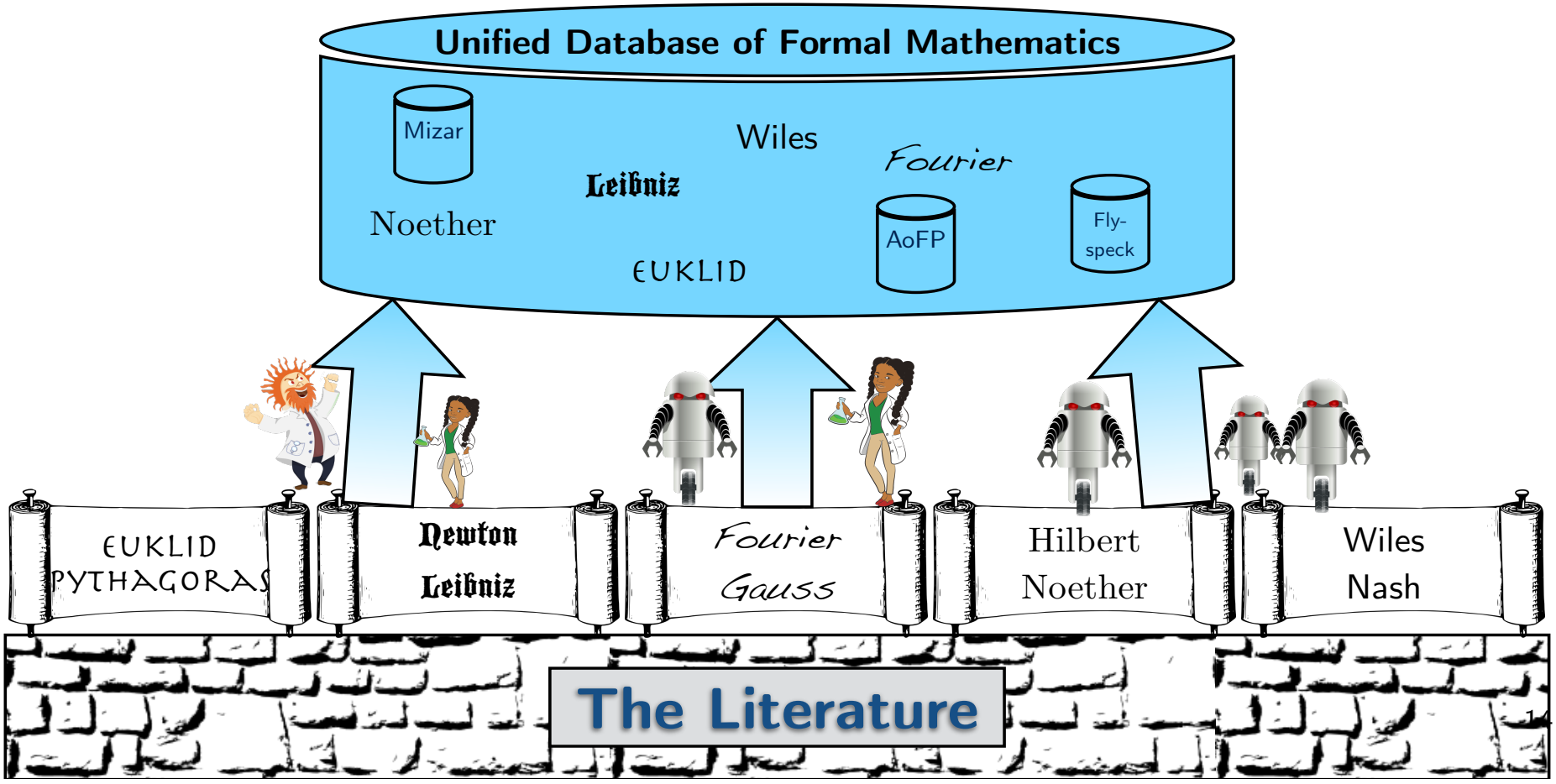


The Future Math Ecosystem (Utopia)

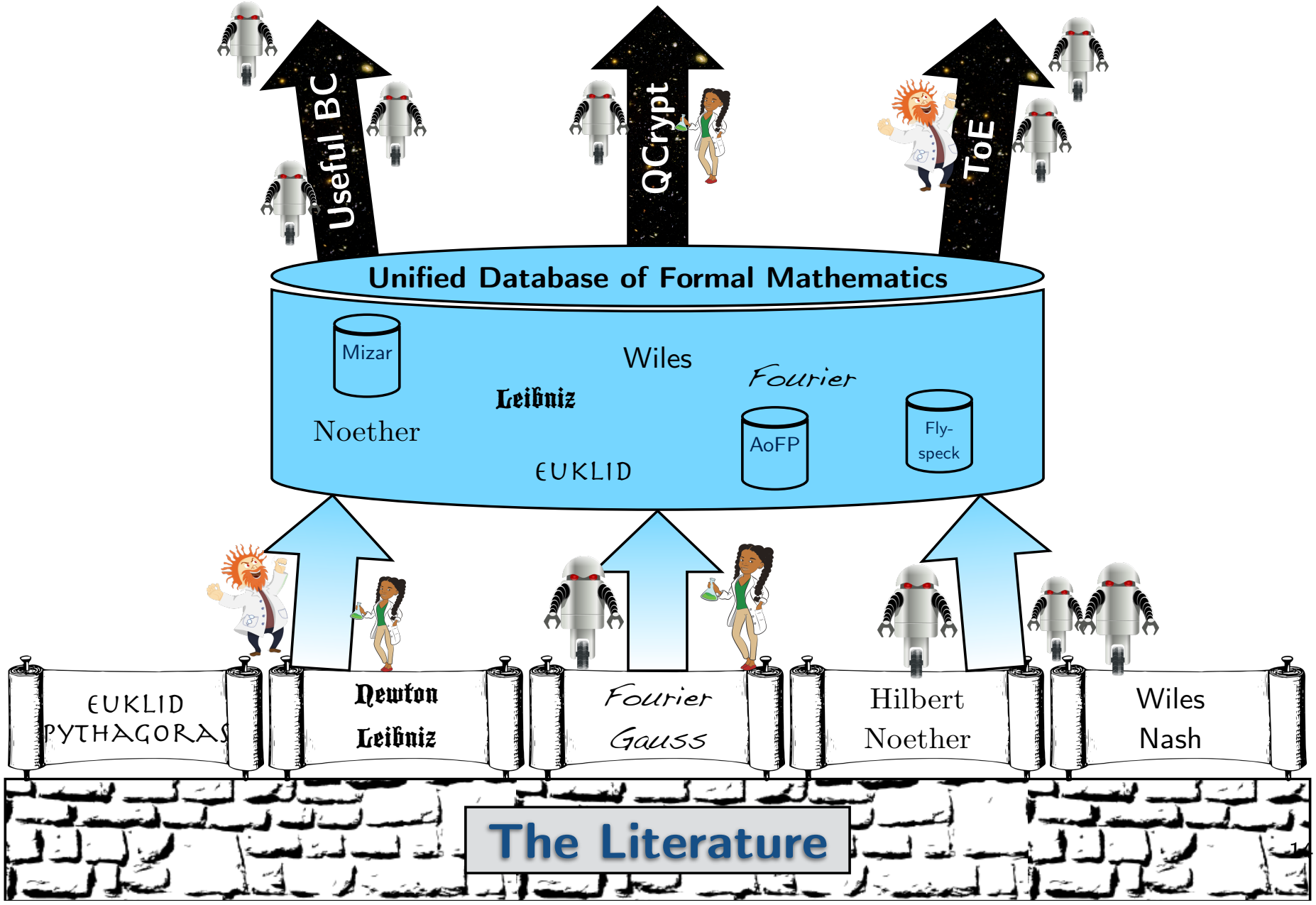




The Future Math Ecosystem (Utopia)



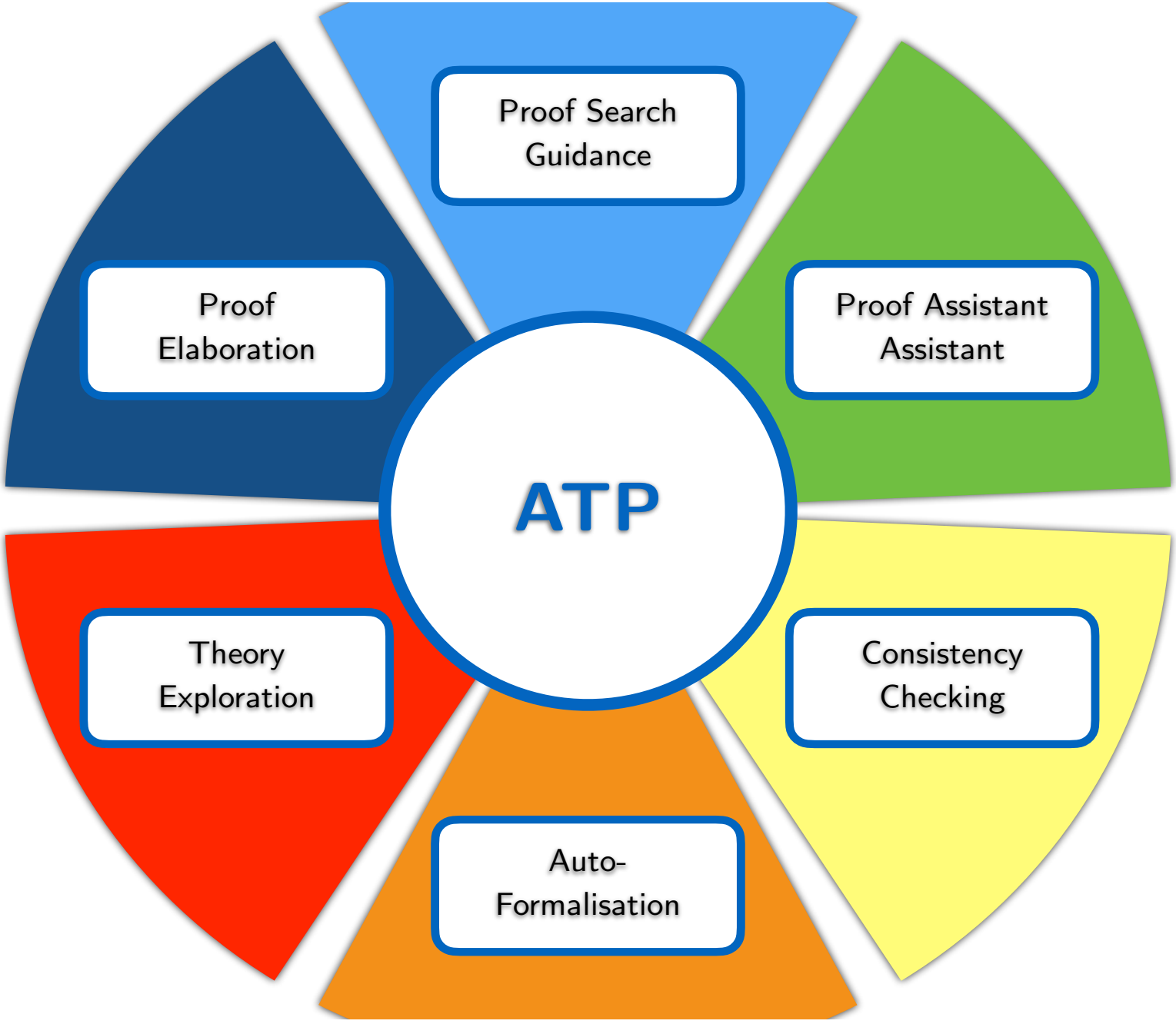
The Future Math Ecosystem (Utopia)



The future belongs to hybrid systems

- Interactive theorem provers are the main front-end for human mathematicians
  - User-friendly syntax and editors
  - Access to mathematical libraries
- Automatic theorem provers boost productivity
  - . . . e.g. as Hammers for ITPs
  - . . . supporting theory exploration
- Automatic theorem provers perform quality control
  - Internal consistency
  - Merge consistency
- Machine learning improves the power of automated systems
  - Proof guidance
- LLM-based systems translate to and from symbolic logic
  - Auto-formalization

ATP in the Formal Math Ecosystem



Automated Theorem Proving

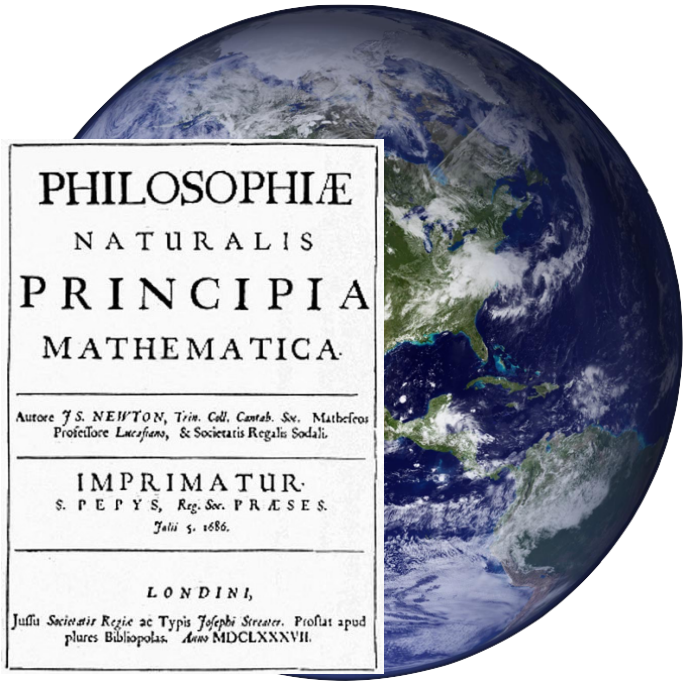
- Controlled language
  - Typically a well-defined logic (First-order logic, equational logic, HOL, propositional logic, . . .)
  - Clear, unambiguous syntax
  - Clear, unambigous semantics
- Sound reasoning framework
  - E.g. logical calculus
  - Only explicit premises are used
- Explicit proof objects
  - Verifiable results
  - Explainable results

Automated Theorem Proving: Big Picture

Real World Problem

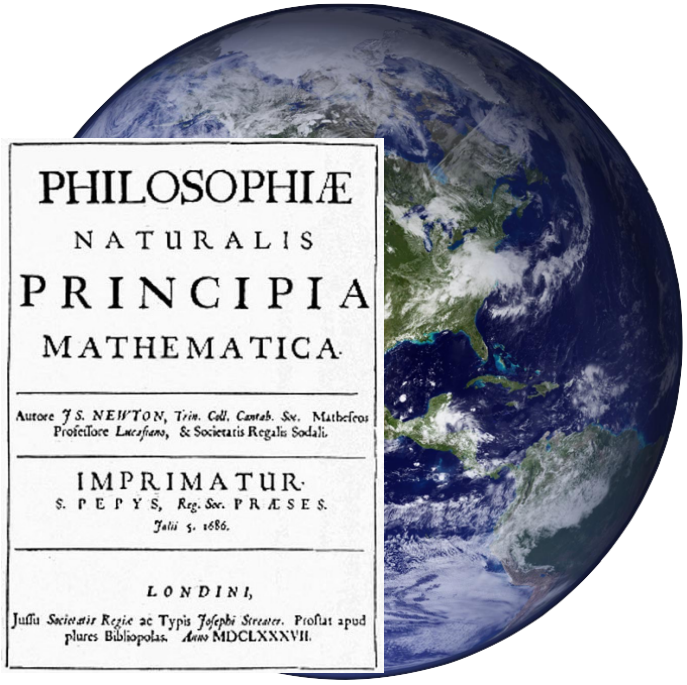
Automated Theorem Proving: Big Picture

Real World Problem



Automated Theorem Proving: Big Picture

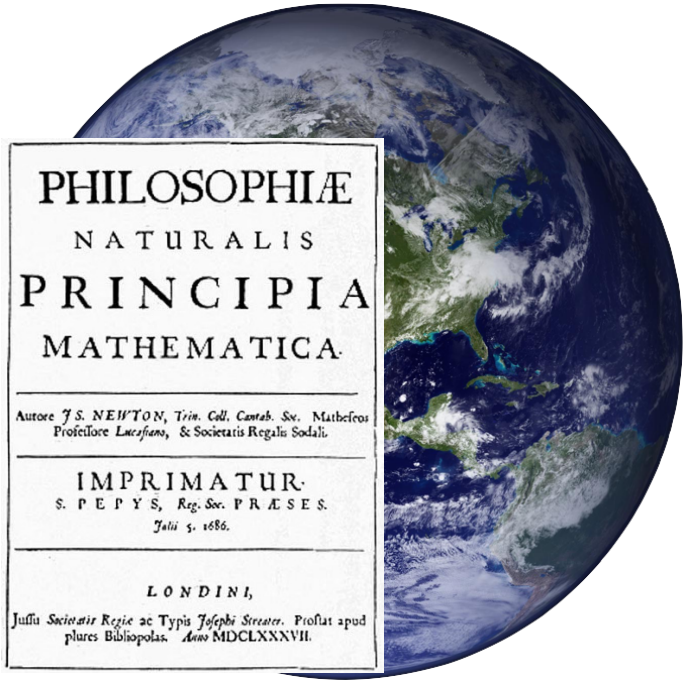
Real World Problem



Formalized Problem

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

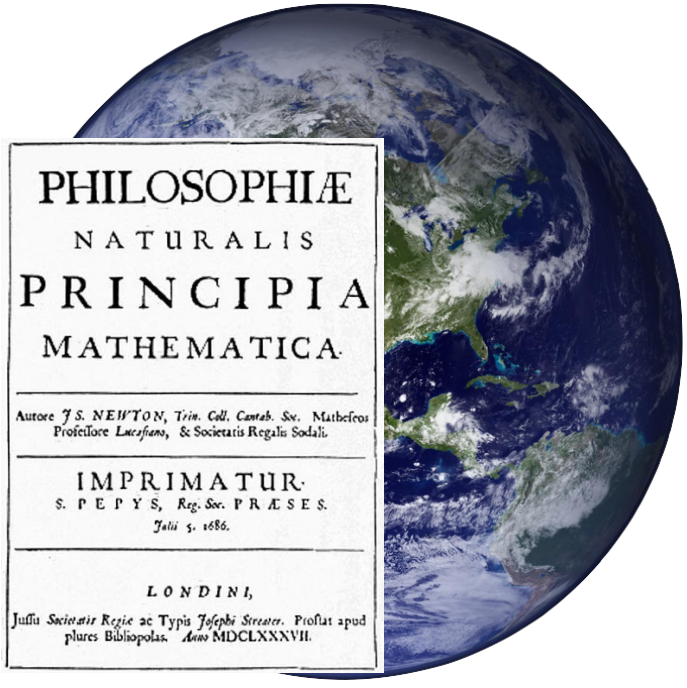
$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$

$$mortal(socrates)$$

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

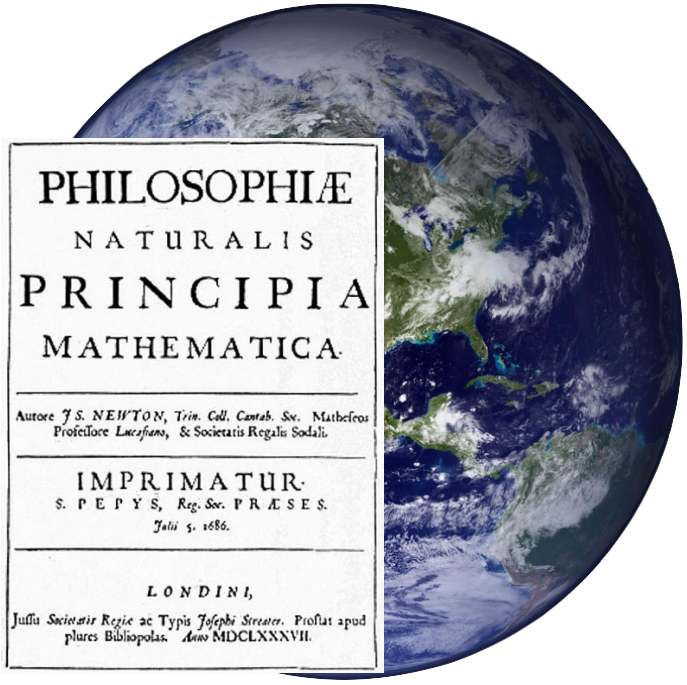
$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$

$$mortal(socrates)$$

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

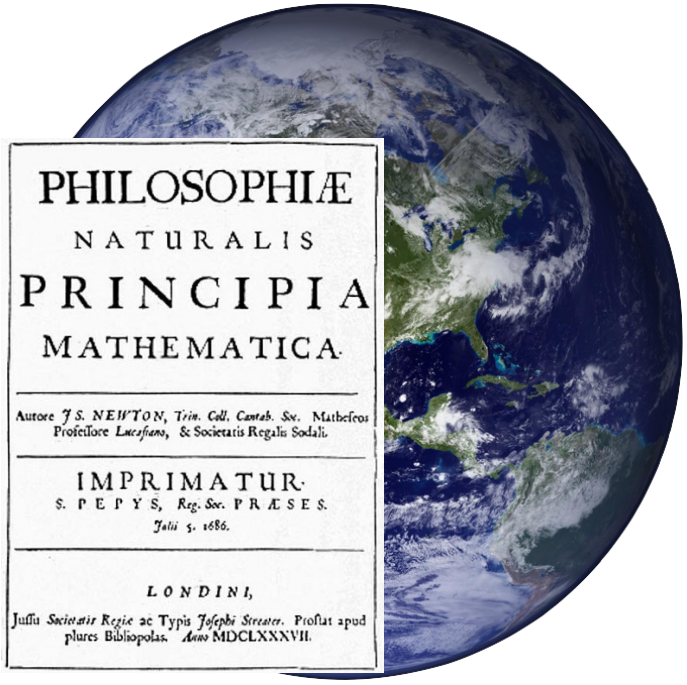
$$?$$
$$\models$$

$$mortal(socrates)$$

ATP

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$

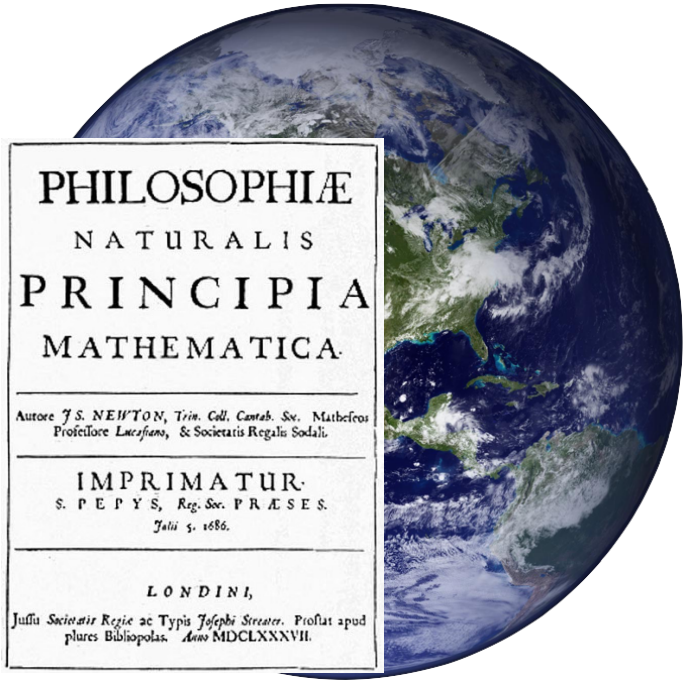
$$mortal(socrates)$$

ATP

Proof

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$

$$mortal(socrates)$$

Proof

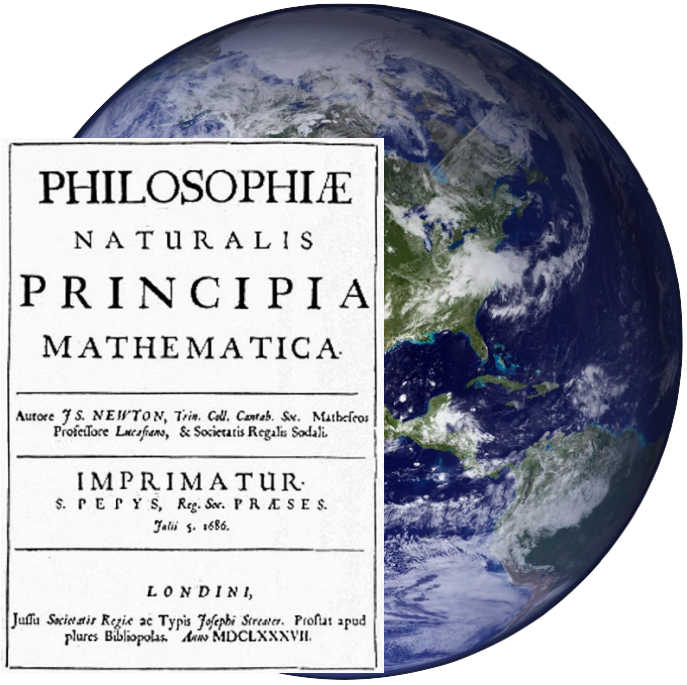
or

Countermodel

ATP

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$

$$mortal(socrates)$$

Proof

or

Countermodel

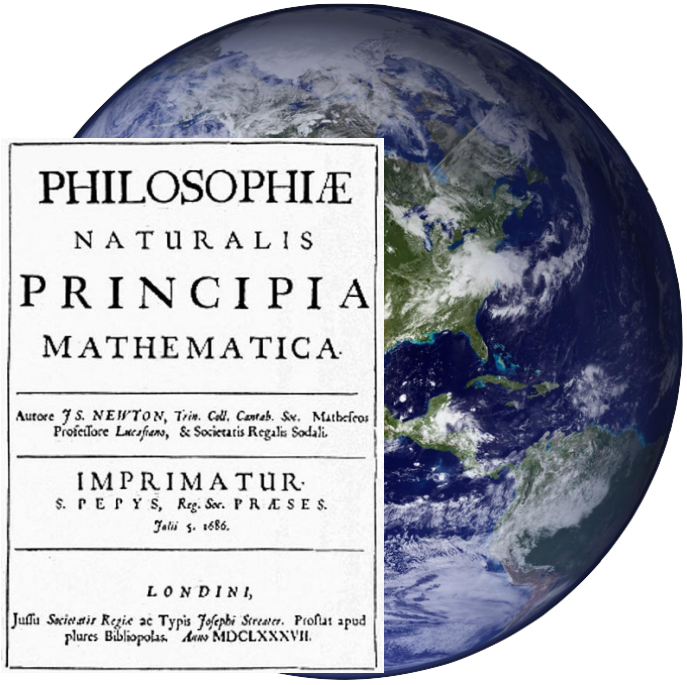
or

Timeout

ATP

Automated Theorem Proving: Big Picture

Real World Problem



Formalized Problem

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$

$$?$$
$$\models$$
$$mortal(socrates)$$

Proof

or

Countermodel

or

Timeout

ATP

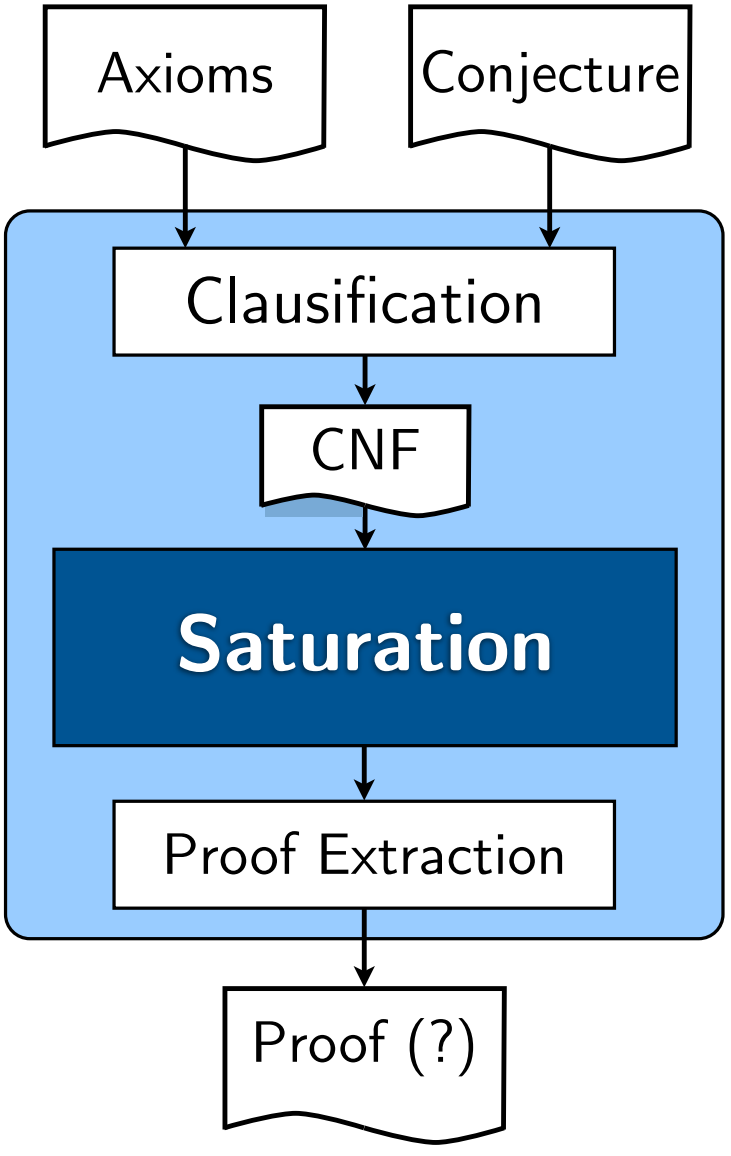
Proof Search

Opening the Black Box (theoretically)

ATP

Proof Search

Opening the Black Box (theoretically)



Refutation and Clausification

$$\forall X : human(X) \rightarrow mortal(X)$$
$$\forall X : philosopher(X) \rightarrow human(X)$$
$$philosopher(socrates)$$
$$\models$$
$$mortal(socrates)$$

iff

$$\left\{ \begin{array}{c} \forall X : human(X) \rightarrow mortal(X) \\ \forall X : philosopher(X) \rightarrow human(X) \\ philosopher(socrates) \\ \neg mortal(socrates) \end{array} \right\}$$

is unsatisfiable

Refutation and **Clausification**

$$\left\{ \begin{array}{c} \forall X : human(X) \rightarrow mortal(X) \\ \forall X : philosopher(X) \rightarrow human(X) \\ philosopher(socrates) \\ \neg mortal(socrates) \end{array} \right\}$$

is unsatisfiable

iff

$$\left\{ \begin{array}{c} \neg human(X) \vee mortal(X) \\ \neg philosopher(X) \vee human(X) \\ philosopher(socrates) \\ \neg mortal(socrates) \end{array} \right\}$$

is unsatisfiable

Saturating Theorem Proving

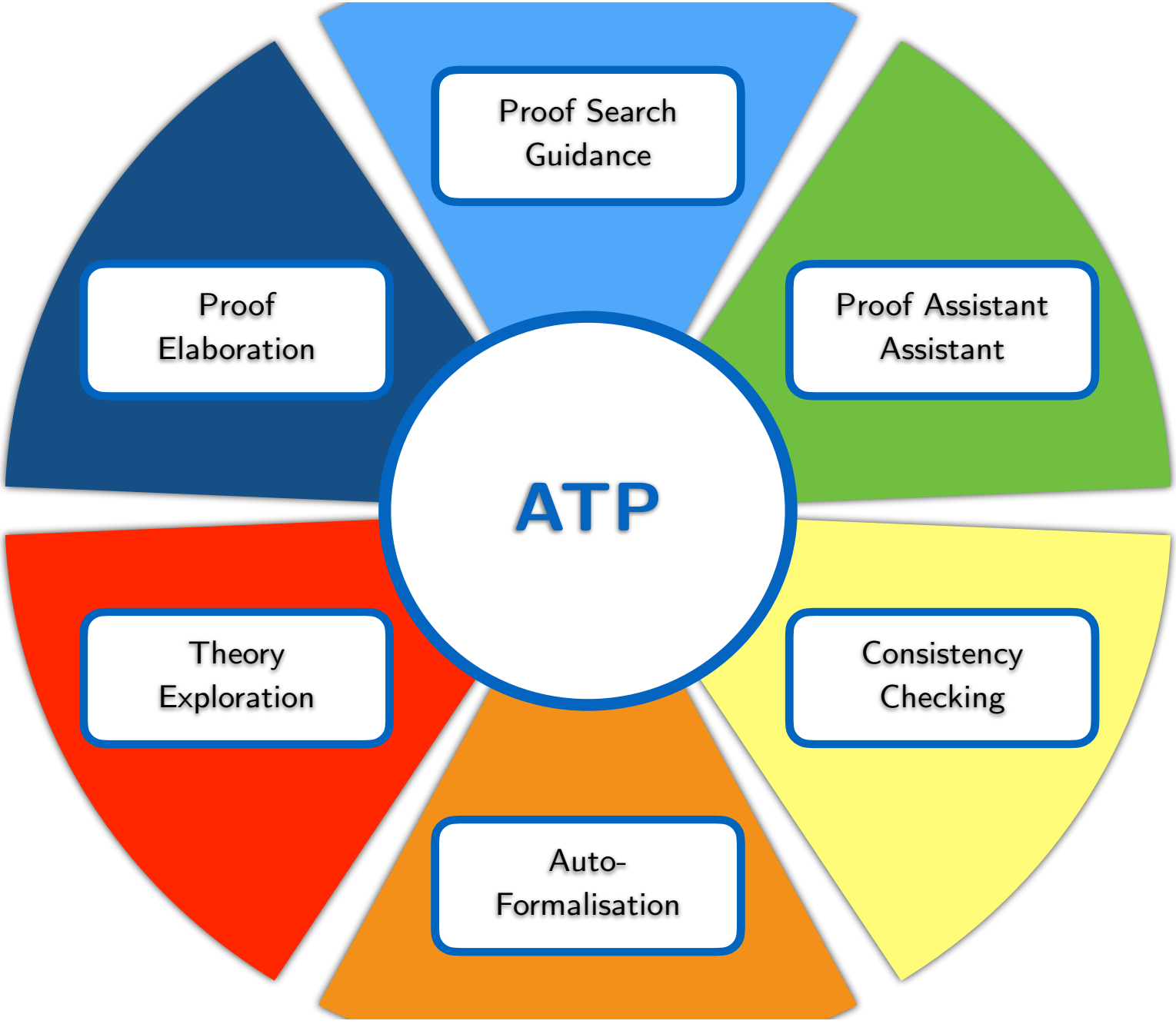
- Goal: Show unsatisfiability of a set of clauses S
- Approach:
  - Systematically enrich S with clauses derived via inferences from clauses in S (Saturation)
  - Optionally: Remove redundant clauses
- Outcome:
  - Derivation of the empty clause $\square$ (explicit witness of unsatisfiability)
  - Successful saturation (up to redundancy): S is satisfiable
  - . . . or infinite sequence of derivations
- Properties:
  - Correctness: Only logical consequences are derived
  - Completeness: Every unsatisfiable S will eventually lead to the derivation of $\square$

Saturating Theorem Proving

- Goal: Show unsatisfiability of a set of clauses S
- Approach:
  - Systematically enrich S with clauses derived via inferences from clauses in S (Saturation)
  - Optionally: Remove redundant clauses
- Outcome:
  - Derivation of the empty clause $\square$ (explicit witness of unsatisfiability)
  - Successful saturation (up to redundancy): S is satisfiable
  - . . . or infinite sequence of derivations
- Properties:
  - Correctness: Only logical consequences are derived
  - Completeness: Every unsatisfiable S will eventually lead to the derivation of $\square$

At it's core, an ATP uncovers inconsistency!

Recalling the (recent) past



Consistency checking

- Problem: Large axiomatizations may have bugs
  - Hidden internal inconsistencies
  - New *merge inconsistencies* if corpora are extended/merged
- ATPs are excellent at finding inconsistencies
  - Most modern ATPs prove by contradiction anyways
  - Otherwise prove $A \models \bot$
- Automatic tests based on systematic sub-sampling are effective
  - Uncovered several bugs in OpenCyc
  - Uncovered some bugs in SUMO
  - Mizar is (apparently) bug-free (!)
    - . . . but we found all injected inconsistencies

Consistency checking

- Problem: Large axiomatizations may have bugs
  - Hidden internal inconsistencies
  - New *merge inconsistencies* if corpora are extended/merged
- ATPs are excellent at finding inconsistencies
  - Most modern ATPs prove by contradiction anyways
  - Otherwise prove $A \models \bot$
- Automatic tests based on systematic sub-sampling are effective
  - Uncovered several bugs in OpenCyc
  - Uncovered some bugs in SUMO
  - Mizar is (apparently) bug-free (!)
    - . . . but we found all injected inconsistencies

Automatic consistency checking should become a standard practice in formalisation workflows!

Saturating Theorem Proving

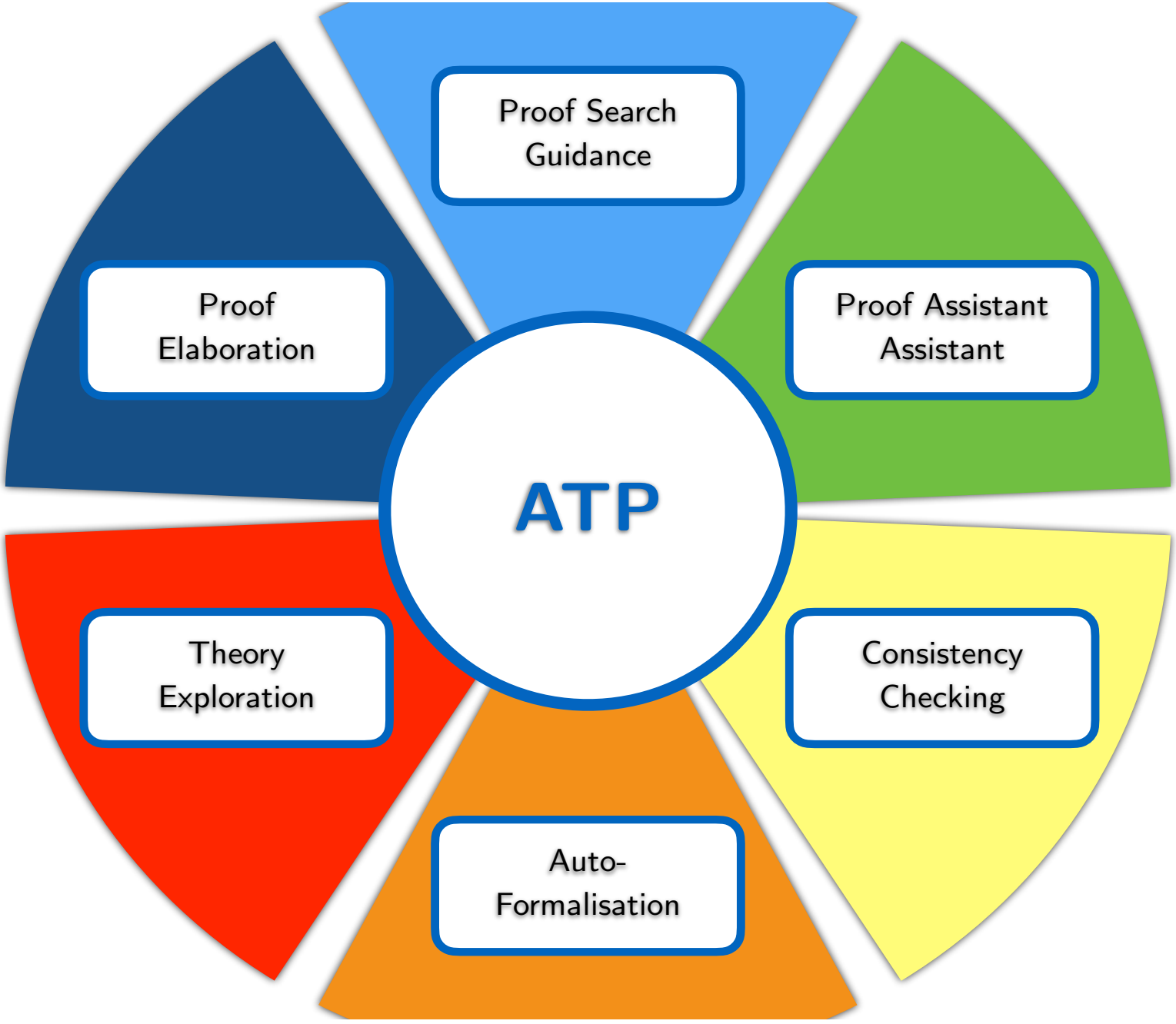
- Goal: Show unsatisfiability of a set of clauses S
- Approach:
  - Systematically enrich S with clauses derived via inferences from clauses in S (Saturation)
  - Optionally: Remove redundant clauses
- Outcome:
  - Derivation of the empty clause $\square$ (explicit witness of unsatisfiability)
  - Successful saturation (up to redundancy): S is satisfiable
  - . . . or infinite sequence of derivations
- Properties:
  - Correctness: Only logical consequences are derived
  - Completeness: Every unsatisfiable S will eventually lead to the derivation of $\square$

Saturating Theorem Proving

- Goal: Show unsatisfiability of a set of clauses S
- Approach:
  - Systematically enrich S with clauses derived via inferences from clauses in S (Saturation)
  - Optionally: Remove redundant clauses
- Outcome:
  - Derivation of the empty clause $\square$ (explicit witness of unsatisfiability)
  - Successful saturation (up to redundancy): S is satisfiable
  - . . . or infinite sequence of derivations
- Properties:
  - Correctness: Only logical consequences are derived
  - Completeness: Every unsatisfiable S will eventually lead to the derivation of $\square$

At it's core, an ATP enumerates consequences!

Re-Recalling the (recent) past



Automatic theory exploration

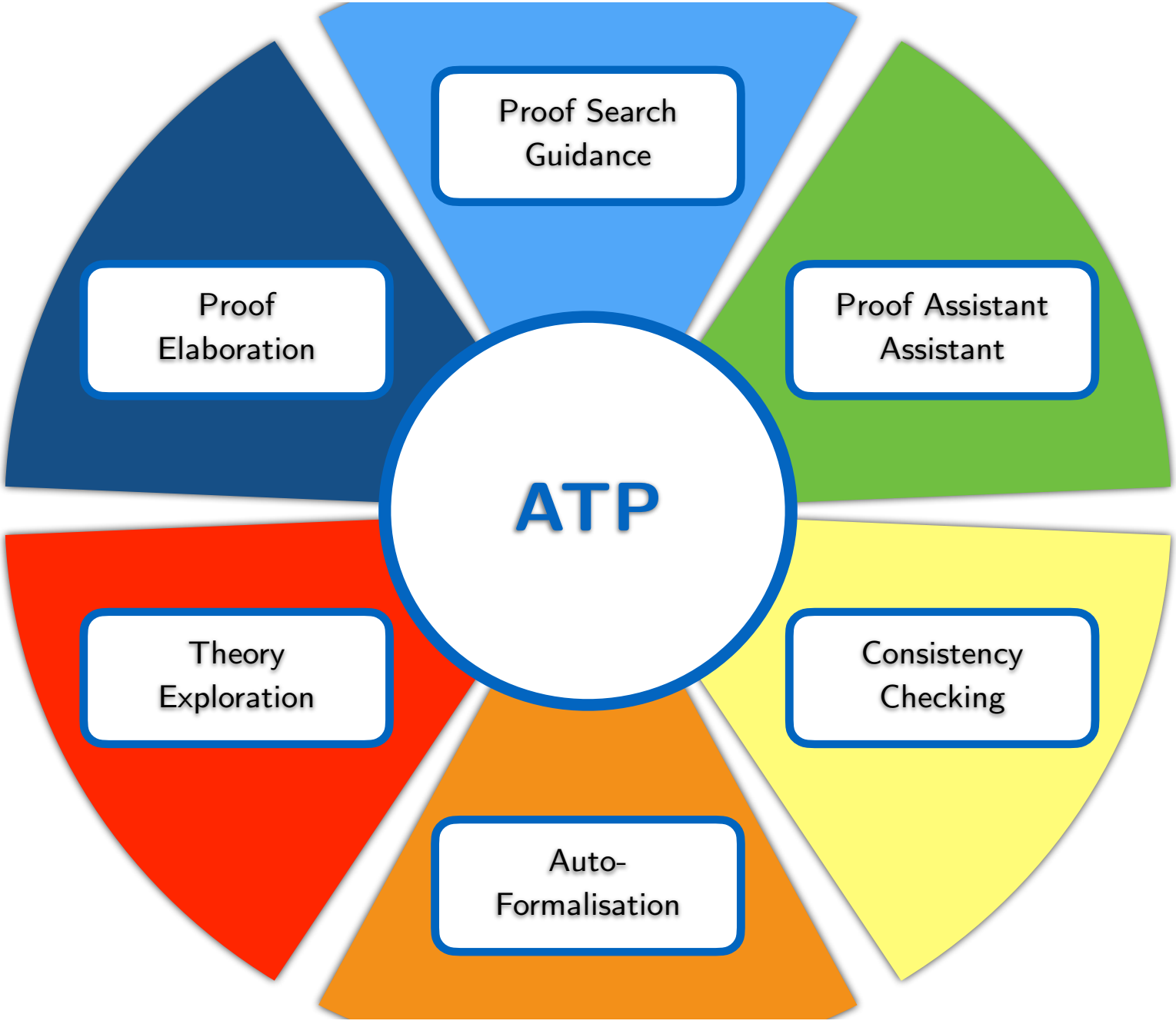
- Automatically find interesting consequences of a given axiomatization
- Enumerate and test
  - Saturating theorem prover generates consequences
  - Critic identifies interesting theorems
    - Generality
    - Proof size
    - "Surprise factor"
- Conjecture and prove
  - Use some approach to generate interesting conjectures
  - Prover checks them for theorem-hood
- Iterate until satisfied
  - New conjectures are added to the axiomatization
  - Process restarts

Automatic theory exploration

- Automatically find interesting consequences of a given axiomatization
- Enumerate and test
  - Saturating theorem prover generates consequences
  - Critic identifies interesting theorems
    - Generality
    - Proof size
    - "Surprise factor"
- Conjecture and prove
  - Use some approach to generate interesting conjectures
  - Prover checks them for theorem-hood
- Iterate until satisfied
  - New conjectures are added to the axiomatization
  - Process restarts

Proposal: Reuse cryptominers to mine theorems!

Re-Re-Recalling the (recent) past



Insert Josef's talk here

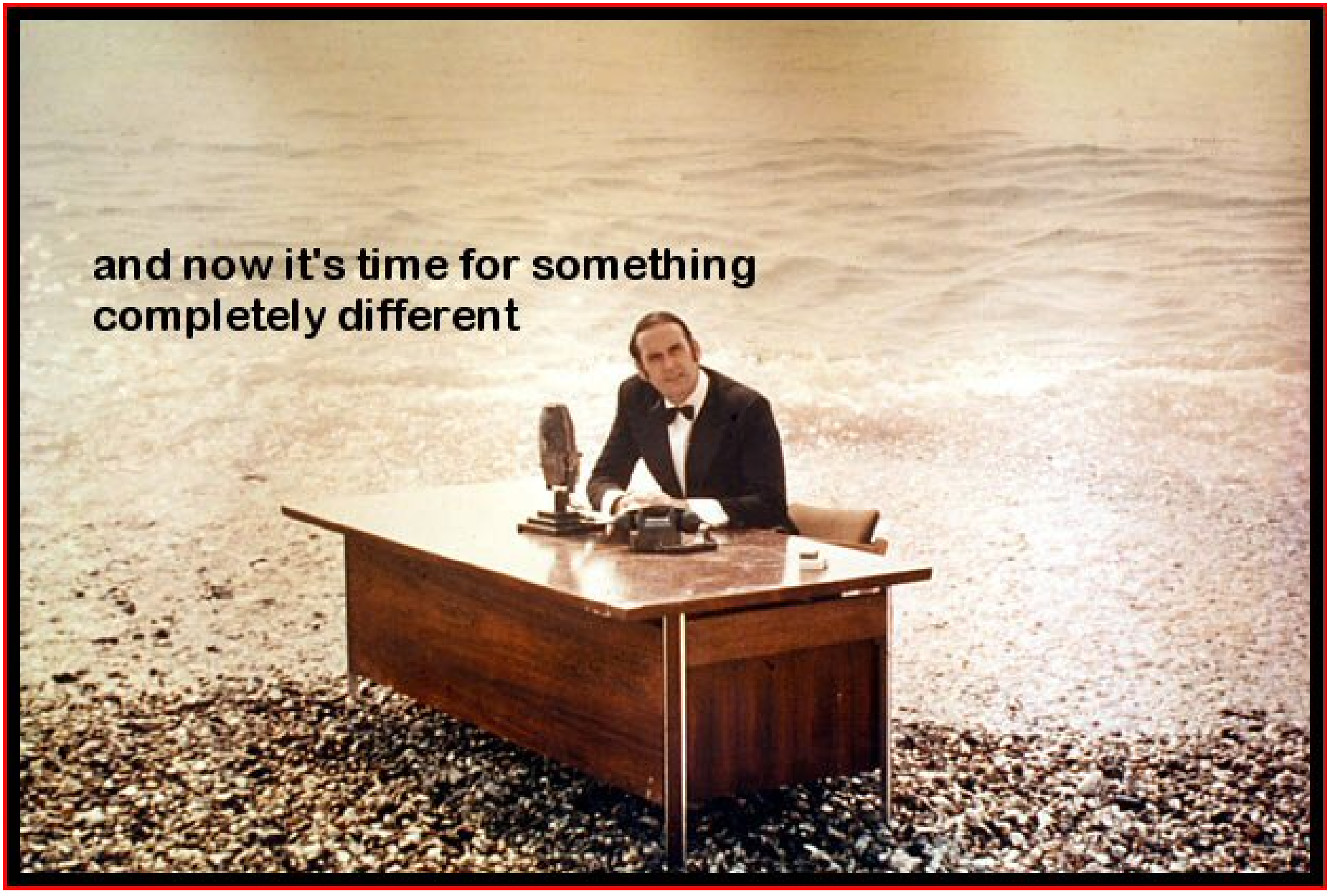
Proof search guidance

- Automated Theorem Provers search for proofs in infinite spaces
  - Better be careful where you go!
- Useful techniques (and we are only just beginning)
  - Proof mining
  - Genetic optimization
  - Strategy selection
  - Axiom selection
  - ...

Proof search guidance

- Automated Theorem Provers search for proofs in infinite spaces
  - Better be careful where you go!
- Useful techniques (and we are only just beginning)
  - Proof mining
  - Genetic optimization
  - Strategy selection
  - Axiom selection
  - ...

Significant successes, but it's harder than we thought!


and now it's time for something
completely different

(Old) News from E

- E now supports (monomorphic) HOL
  - Calculus is theoretically incomplete (e.g. finite unifier sets)
  - Good performance in practice
  - One of the strongest provers in recent CASCs
- E now supports FOOL/TFX
  - Allows mixing of (boolean) formulas and terms
  - Can conveniently encode e.g. some modalities
  - Calculus is complete (and mostly in preprocessing)

(Old) News from E

- E now supports (monomorphic) HOL
  - Calculus is theoretically incomplete (e.g. finite unifier sets)
  - Good performance in practice
  - One of the strongest provers in recent CASCs
- E now supports FOOL/TFX
  - Allows mixing of (boolean) formulas and terms
  - Can conveniently encode e.g. some modalities
  - Calculus is complete (and mostly in preprocessing)

Example?

TFX example

```
tff(humtype,     type, human: $tType).
tff(typepeter,   type, peter: human).
tff(typestephan, type, stephan: human).

tff(normal, axiom, ![X:human, Y:$o]: (says(X,Y) => (Y|~Y))).
tff(wise,   axiom, ![X:human, Y:$o]: ((says(X,Y)&wise(X)) => Y)).

tff(peteriswise, axiom, wise(peter)).
tff(stephanis,   axiom, wise(stephan)|~wise(stephan)).
tff(truth, axiom, says(stephan, 'E is the best theorem prover')).

tff(e_is_cool, conjecture, 'E is the best theorem prover').
```

TFX example

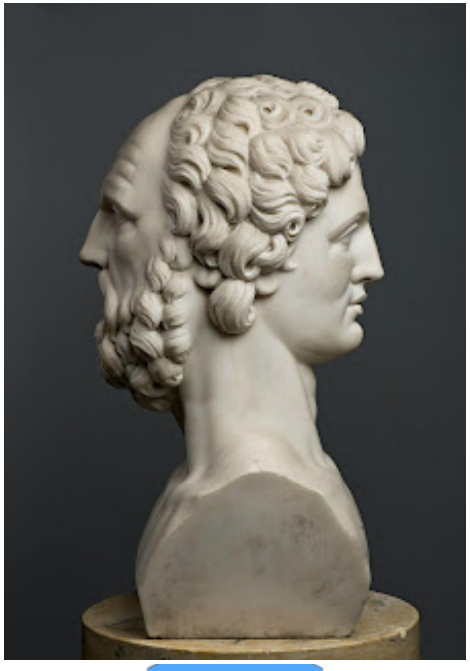
```
tff(humtype,     type, human: $tType).
tff(typepeter,   type, peter: human).
tff(typestephan, type, stephan: human).

tff(normal, axiom, ![X:human, Y:$o]: (says(X,Y) => (Y|~Y))).
tff(wise,   axiom, ![X:human, Y:$o]: ((says(X,Y)&wise(X)) => Y)).

tff(peteriswise, axiom, wise(peter)).
tff(stephanis,   axiom, wise(stephan)|~wise(stephan)).
tff(truth, axiom, says(stephan, 'E is the best theorem prover')).

tff(e_is_cool, conjecture, 'E is the best theorem prover').

tff(moretruth, axiom,
               says(peter, 'E is the best theorem prover')).
```

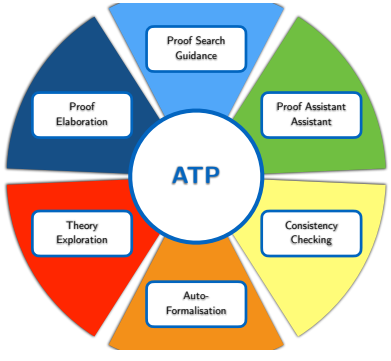
TFX example

```
tff(humtype,     type, human: $tType).
tff(typepeter,   type, peter: human).
tff(typestephan, type, stephan: human).

tff(normal, axiom, ![X:human, Y:$o]: (says(X,Y) => (Y|~Y))).
tff(wise,   axiom, ![X:human, Y:$o]: ((says(X,Y)&wise(X)) => Y)).

tff(peteriswise, axiom, wise(peter)).
tff(stephanis,   axiom, wise(stephan)|~wise(stephan)).
tff(truth, axiom, says(stephan, 'E is the best theorem prover')).

tff(e_is_cool, conjecture, 'E is the best theorem prover').

tff(crowd, axiom,
          ![X:human]:says(X, 'E is the best theorem prover')).
```

Conclusion

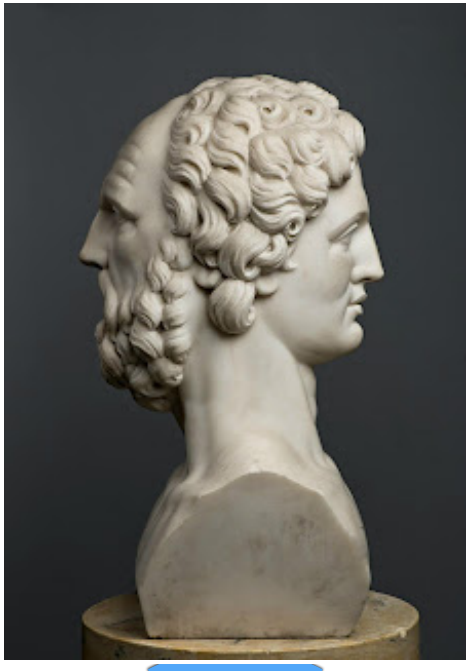
- Automated Reasoning and Machine Learning are complementary in many ways
  - The main formalism of computer maths will be symbolic
  - Deduction will use formal symbolic rules
- Machine learning techniques will contribute
  - Nearest neighbors
  - Decision trees
  - Genetic algorithms
  - Tree-based ANN architectures
  - Even LLMs

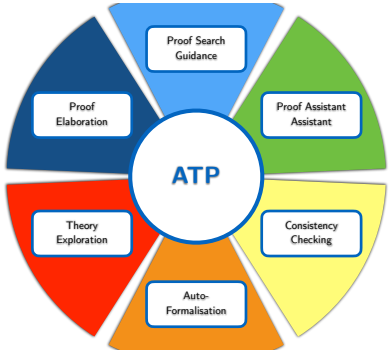




Conclusion

- Automated Reasoning and Machine Learning are complementary in many ways
  - The main formalism of computer maths will be symbolic
  - Deduction will use formal symbolic rules
- Machine learning techniques will contribute
  - Nearest neighbors
  - Decision trees
  - Genetic algorithms
  - Tree-based ANN architectures
  - Even LLMs
- **'E is the best theorem prover'**





Most Important Message

Most Important Message

- CADE-30 at Stuttgart in Summer 2025
  - July 28th—August 2nd, 2025
  - `https://cade-30.info`
- 9 Workshops
- CADE ATP System Competition
- Exciting social program

Most Important Message

- CADE-30 at Stuttgart in Summer 2025
  - July 28th—August 2nd, 2025
  - `https://cade-30.info`
- 9 Workshops
- CADE ATP System Competition
- Exciting social program

Thanks!