

自動証明と周辺技術がもたらす 情報理論の未来

中正和久 & 萩原学

2023/11/29

中正和久

- 山口大学工学部知能情報工学科 准教授
専門：数学知識管理, 社会実装 (インフラ・教育・福祉)
出身：大阪府河内長野市
趣味：スキー, 登山, 株式投資, モフモフ動画鑑賞
情報処理学会中国支部事務局
定理証明支援系はMizar
前職はプログラマ

Xのアカウントは @aabaa1978



萩原学

- 千葉大学大学院理学研究院 教授
専門: 符号理論、離散数学
出身: 栃木県足利市
趣味: 映画、テレビドラマ、ウクレレ、怪談
ISITA2020 共同実行委員長
量子削除誤り訂正符号を作った人
IBC2023 銀メダリスト

Xのアカウントは @QRJAM



お願い

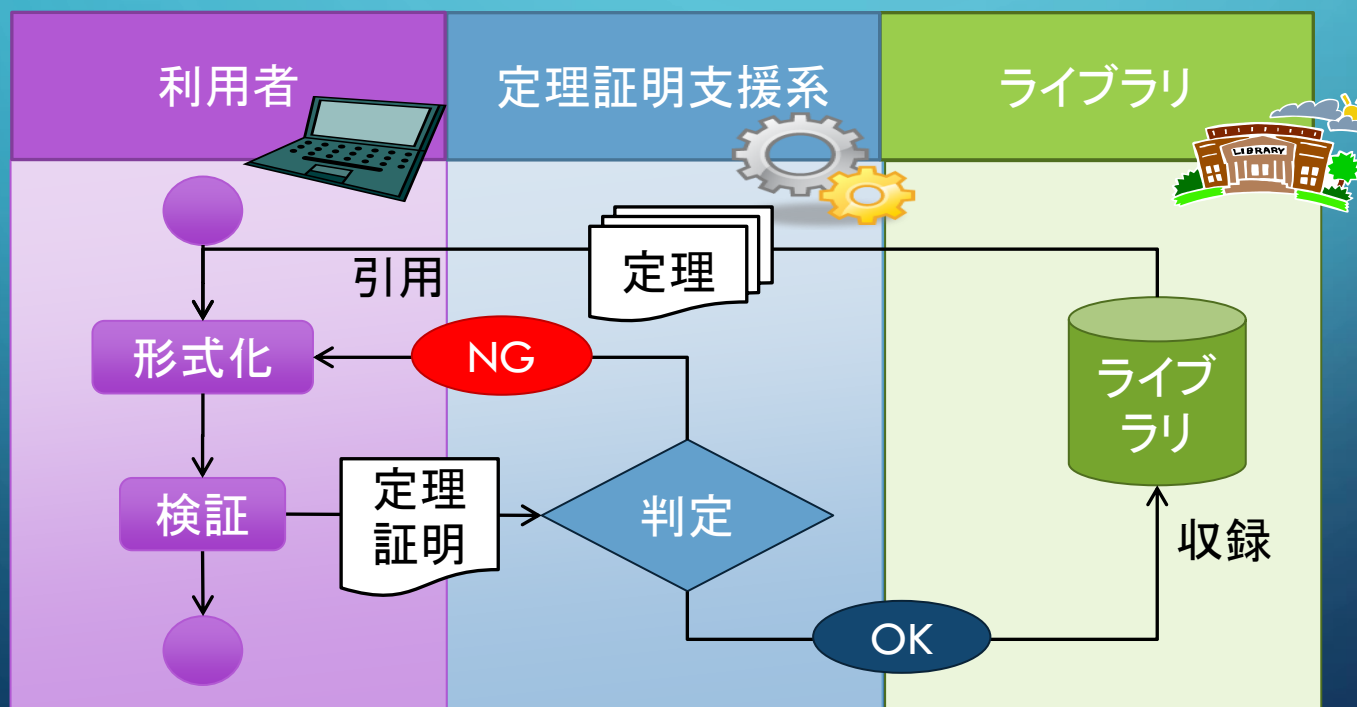
- ナイトセッションらしく(らしさとは?)
だらだらと話しをさせて下さい。
- 参加型
和気あいあいとした雰囲気
楽しい時間に。



定理証明支援系

定理証明支援系って何？

- 数学証明をプログラミング言語のように記述 (=形式化)して定理証明支援系に渡すと, 証明の正しさがチェックできる!



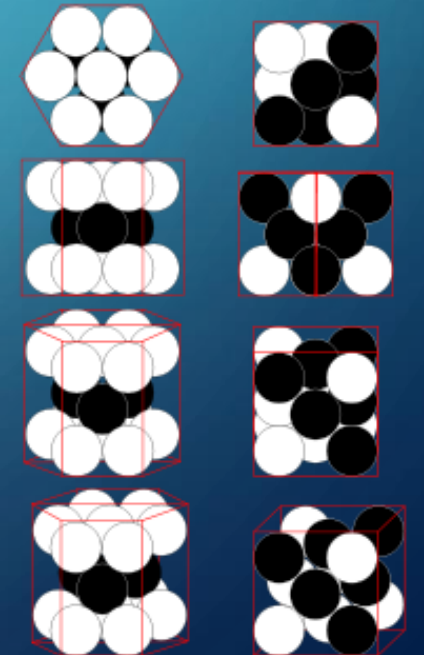
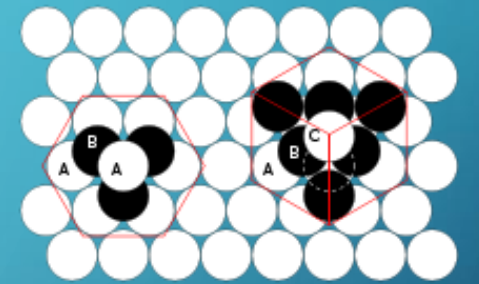
何が目的？何ができるの？

- 定理証明支援系の目的は？
 - 数学証明の正しさをチェックする
 - コンピュータプログラムが仕様通りに動くことをチェックする
- 誰がどういう用途で使っているの？
 - 計算機科学者, プログラマ
コンピュータプログラムの検証
 - 論理学者
さまざまな論理体系の研究
 - 数学者
証明の正しさのチェック
数学知識のデータベース化



こんな場面で使われます～数学編 [1/2]

- ケプラー予想の形式検証 (2002-2014)
 - 「3次元空間における同一半径球の最密充填は、面心立方充填と六方最密充填だけである」ことを主張する定理. ケプラーが1611年に予想し、長年未解決だった.
 - 1998年にThomas Halesが解決を主張したが、計算機を用いたしらみつぶしによる解法だったため、査読者は「証明が正しいことを99%確信」していたが、残り1%のギャップを埋められなかった.
 - 2002年にHalesは定理証明支援系 (Isabelle + HOL Light) によって、ケプラー予想の証明を形式化する「Flyspeckプロジェクト」を立ち上げた. 2014年に形式化が完成し、数学界に正式に証明が受け入れられた.



こんな場面で使われます～数学編 [2/2]

- Formalizing 100 Theorems

- 2000年に選ばれた「The Hundred Greatest Theorem」を形式化しようという試み。2023年までにフェルマーの最終定理を除く99の定理が形式化された。

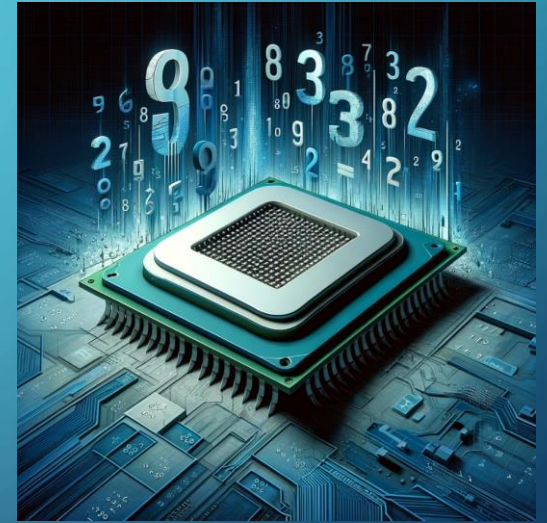
- 主な定理

$\sqrt{2}$ が無理数であること, 代数学の基本定理,
素数定理, ゲーデルの不完全性定理,
平方剰余の相互法則, 円の面積, 素数が無限に存在すること,
グリーンの定理, 連続体仮説, ラムゼーの定理,
テイラーの定理, ブラウエルの不動点定理,
等周定理, π と e が超越数であること, 大数の法則,
ユークリッドの互除法, シローの定理, フーリエ級数,
ルベグ測度とルベグ積分

<u>Isabelle</u>	89
<u>HOL Light</u>	87
<u>Coq</u>	79
<u>Lean</u>	76
<u>Metamath</u>	74
<u>Mizar</u>	69
<u>ProofPower</u>	43
nqthm/ACL2	39
PVS	26
<u>Naproche</u>	10
NuPRL/MetaPRL	8

こんな場面で使われます～産業編 [1/2]

- Pentium FDIVバグ(1994年)
 - Pentiumプロセッサの浮動小数点演算ユニットに不具合
 - 除算で使うSRT法のルックアップテーブルに誤りがあった
 - 100万回に高々61回しか起こらないためテストで見逃し
 - リコールによる損失額は4.75億ドル(=約500億円)
 - Intelはこれを教訓に, John Harrison等の形式化グループを立ち上げて, 定理証明支援系を用いた浮動小数点演算ユニットの見直しを実施



こんな場面で使われます～産業編 [2/2]

- NASAのソフトウェアの形式検証(PVS, 1996?～)
- オペレーティングシステムseL4の形式検証 (Isabelle/HOL, 2009)
- C言語コンパイラCompCertの形式検証 (Coq, 2012)
- OpenSSLのプロトコルの安全性検証と脆弱性の発見 (Coq, 2014)

あの数学者も使っています [1/2]



- Terence Tao (2006年フィールズ賞受賞者, The smartest man in the world)



Terence Tao

@tao@mathstodon.xyz

Finally completed the #Lean4 formalization of my Maclaurin-type inequality paper at github.com/teorth/symmetric_pr.... Specifically, if s_k denotes the k^{th} symmetric mean of n real numbers, the inequality

$$|s_l|^{1/l} \leq C \max((l/k)^{1/2} |s_k|^{1/k}$$

$$, (l/(k+1))^{1/2} |s_{k+1}|^{1/(k+1)})$$

is established for $1 \leq k < l \leq n$. In fact as a byproduct of the formalization I now get the explicit value $C = 160e^7$ for the constant C .

In the end it took perhaps a hundred hours of effort, spread out over a month. It was about 20x slower to formalize this paper than it was to write it in LaTeX. Initially this was due to my own inexperience with Lean, though near the end I was hitting the limitations of existing Lean tools, as well as the speed of the Lean compiler (though this was in part due some inefficiencies in my coding). However, Lean is a living language still in development,

A combination of automated tools, AI tools, and interactions with human Lean experts proved invaluable. GPT greatly sped up the initial learning of syntax, tools such as `exact?`, `apply?`, and Moogle moogle-morphlabs.vercel.app/ helped locate key lemmas, and Copilot had an often uncanny ability to autocomplete one or more lines of code at a time. But there was certainly a lot of "folklore" tips and tricks that I had to go to the Lean Zulip to get answers to. 2/3

あの数学者も使っています [2/2]



- Peter Scholze (2020年フィールズ賞受賞者, 数論幾何学の第一人者)

Half a year of the Liquid Tensor Experiment: Amazing developments

Posted on [June 5, 2021](#) by [xenaproject](#)

[This is a guest post by Peter Scholze.]

Exactly half a year ago I wrote the [Liquid Tensor Experiment](#) blog post, challenging the formalization of a difficult foundational theorem from my [Analytic Geometry](#) lecture notes on joint work with Dustin Clausen. While this challenge has not been completed yet, I am excited to announce that the Experiment has verified the entire part of the argument that I was unsure about. I find it absolutely insane that interactive proof assistants are now at the level that within a very reasonable time span they can formally verify difficult original research. Congratulations to everyone involved in the formalization!!

In this Q&A-style blog post, I want to reflect on my experience watching this experiment.

どうやって動く？～原理を探ろう

- 証明と詰将棋って似ていませんか？
 - 公理と推論規則 → 駒を動かすルール
 - 命題 → 初期配置
 - 証明 → 詰み手順



どうやって動く？～原理を探ろう

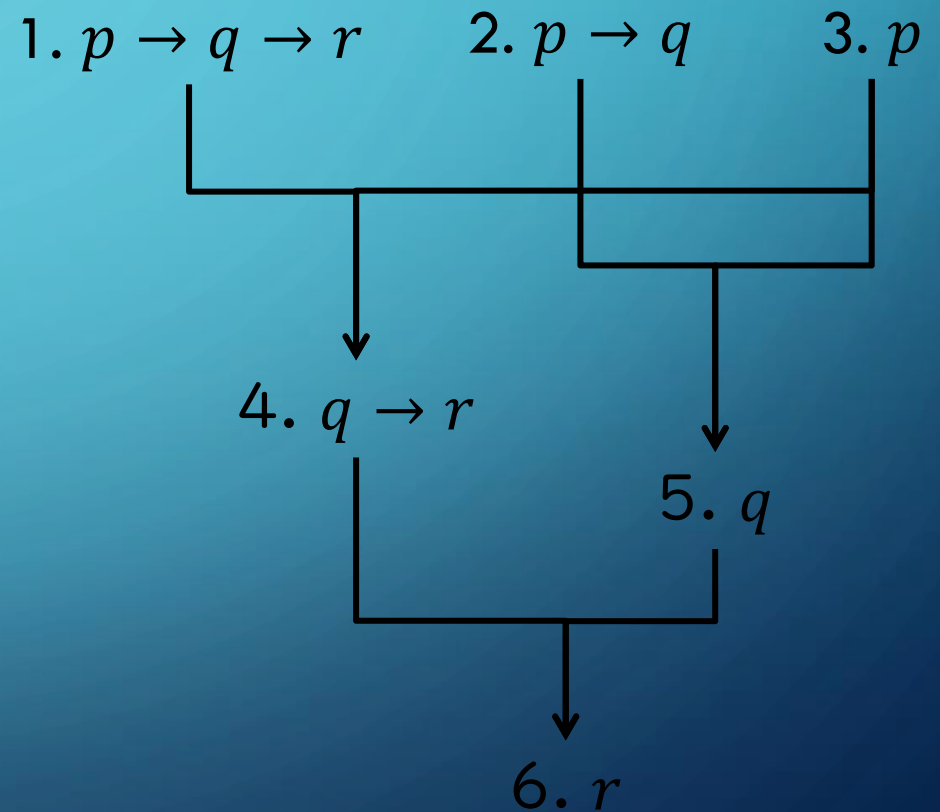
例: $(p \rightarrow q \rightarrow r) \rightarrow (p \rightarrow q) \rightarrow p \rightarrow r$ の演繹証明

- 推論規則:

① $x, (x \rightarrow y) \rightarrow y$: モーダスポネンス

- 証明

1. $p \rightarrow q \rightarrow r$ が真であると仮定
2. $p \rightarrow q$ が真であると仮定
3. p が真であると仮定
4. 1と3に①を適用して $q \rightarrow r$ が成り立つ
5. 2と3に①を適用すると q が成り立つ
6. 4と5に①を適用すると r が成り立つ



どうやって動く？～コンピュータ上で実現するには？

- 証明とプログラムを同一視する
- 推論規則や公理を関数で表し, 証明を関数適用の組み合わせで表す
- 「命題=型」「証明=プログラム」
 - $h : p$ 「 h は型 p を持つプログラムである」 \leftrightarrow 「 h は命題 p の証明である」

数学論理	プログラミング	詰将棋
公理, 推論規則	関数	駒の動かし方
命題	型	初期盤面
証明	プログラム	詰み手順

どうやって動く？～コンピュータ上で実現するには？

- 推論規則は関数で表される

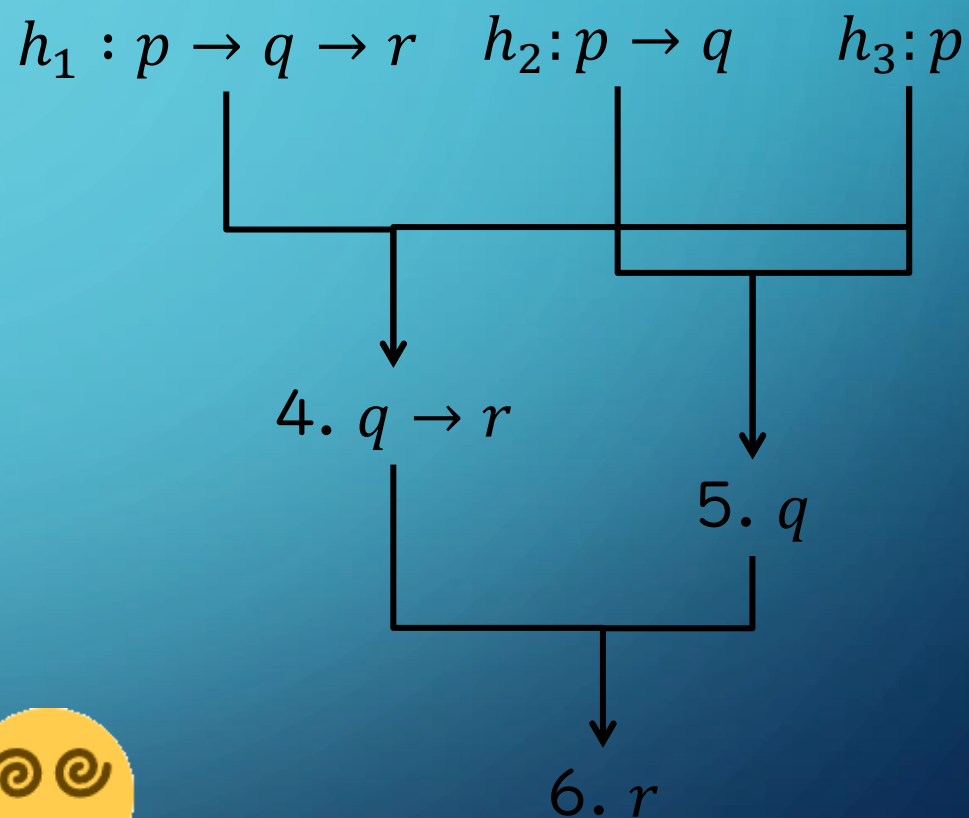
$$f(h_1: x, h_2: x \rightarrow y) = h_3: y$$

- 証明(プログラム)

- $f(f(h_1, h_3), f(h_2, h_3))$

読むのが大変！

どうやって正しいことを確認するの？



どうやって動く？～LEANで書いてみよう

```
variables p q r : Prop

example : (p → q → r) → (p → q) → p → r :=
begin
  assume P_Q_R : p → q → r,      -- p → q → rが真であると仮定
                                  -- 仮定を P_Q_R と名付ける
  assume P_Q : p → q,            -- p → qが真であると仮定
                                  -- 仮定を P_Q と名付ける
  assume P : p,                  -- pが真であると仮定
                                  -- 仮定を P と名付ける
  have Q_R : q → r, from P_Q_R P, -- p → q → r と p から q → r を示す
                                  -- 新たに Q_R と名付ける
  have Q : q, from P_Q P,        -- p → q と p から q を示す
                                  -- 新たに Q と名付ける
  show r, from Q_R Q            -- q → r と q から r を示す
end
```

Leanのおかげでだいぶ読みやすくなったぞ！

何が嬉しいの？

- 証明とプログラムが対応すると、仕様通りに動作するプログラムが書ける！
- 例) マージソートの出力がソート済みであることのLeanプログラム (一部省略)

```
def merge_sort : list α → list α
| []       := []
| [a]      := [a]
| (a::b::l) := begin
  cases e : split (a::b::l) with l1 l2,
  cases length_split_lt e with h1 h2,
  exact merge r (merge_sort l1) (merge_sort l2)
end

theorem sorted_merge_sort : ∀ l : list α, sorted r (merge_sort r l)
| []       := by simp [merge_sort]
| [a]      := by simp [merge_sort]
| (a::b::l) := begin
  cases e : split (a::b::l) with l1 l2,
  cases length_split_lt e with h1 h2,
  rw [merge_sort_cons_cons r e],
  exact (sorted_merge_sort l1).merge (sorted_merge_sort l2)
end
```

生成AI

いろいろあるよ生成AI

- 人工知能チャットボット

ChatGPT

By OpenAI。GPT…Generative Pre-trained Transformer。

Bart

By Google。BERTとGPTの合わせ技。

- 画像生成AI：text-to-image

Stable Diffusion

By Stability AI (CompVis, Runway)。潜在拡散モデル。

Midjourney

By Midjourney, Inc。

- 音楽生成AI：text-to-music

MusicGen

By Meta (facebook)。



使ってますか？生成AI

- みなさんの使用例を教えてください。

萩原： Stable Diffusion で萩原のイラスト、ビールのラベル
chatGPTで英語のメール作成（日本語から英語に翻訳）
chatGPTでLaTeX文書の校正



萩原： chatGPTでLaTeX文書の校正



You are a professional editor with LaTeX.



Yes, I am proficient in using LaTeX as a professional editor. LaTeX is a typesetting language that is widely used for creating academic and scientific documents, including research papers, reports, books, and presentations. As an editor, I can create, edit, and format documents using LaTeX to produce high-quality, professional-looking output.

Some of the tasks I can perform using LaTeX include:

1. Formatting text: I can format text using various styles, such as bold, italics, underline, and different font sizes.
2. Creating tables: I can create tables with borders, shading, and different alignment options.
3. Creating lists: I can create ordered and unordered lists using LaTeX.
4. Including graphics: I can include graphics, images, and figures in LaTeX documents.
5. Adding mathematical equations: LaTeX is particularly useful for adding mathematical equations to documents.
6. Creating bibliographies: I can create bibliographies using LaTeX, including formatting citations and references in different styles.

萩原: chatGPTでLaTeX文書の校正



Please edit the following.

Let X, Y be a set of integers, and f be a map from X to Y and is a bijection.



Let X and Y be sets of integers, and let f be a bijection from X to Y .

萩原： chatGPTでLaTeX文書の校正



Please edit the following.

Let X, Y be a set of integers, and f be a map from X to Y and is a bijection. This implies that \exists exists g from Y to X such that $g \circ f(x) = x$ for $x \in X$.



Here's the edited version:

Let X and Y be sets of integers, and let $f: X \rightarrow Y$ be a bijection. This implies that there exists a map $g: Y \rightarrow X$ such that $g \circ f(x) = x$ for all $x \in X$.

萩原： chatGPTでLaTeX文書の校正



I prefer a blank before \forall for mathematical symbols.



Sure, here's the edited version with a blank before the \forall for mathematical symbols:

Let X and Y be sets of integers, and let $f : X \forallrightarrow Y$ be a bijection. This implies that there exists a map $g : Y \forallrightarrow X$ such that $g \forallcirc f(x) = x$ for all $x \forallin X$.

萩原： chatGPTでLaTeX文書の校正

I think the explanation of \circ should be added.

Sure, here's the edited version with an explanation of \circ :

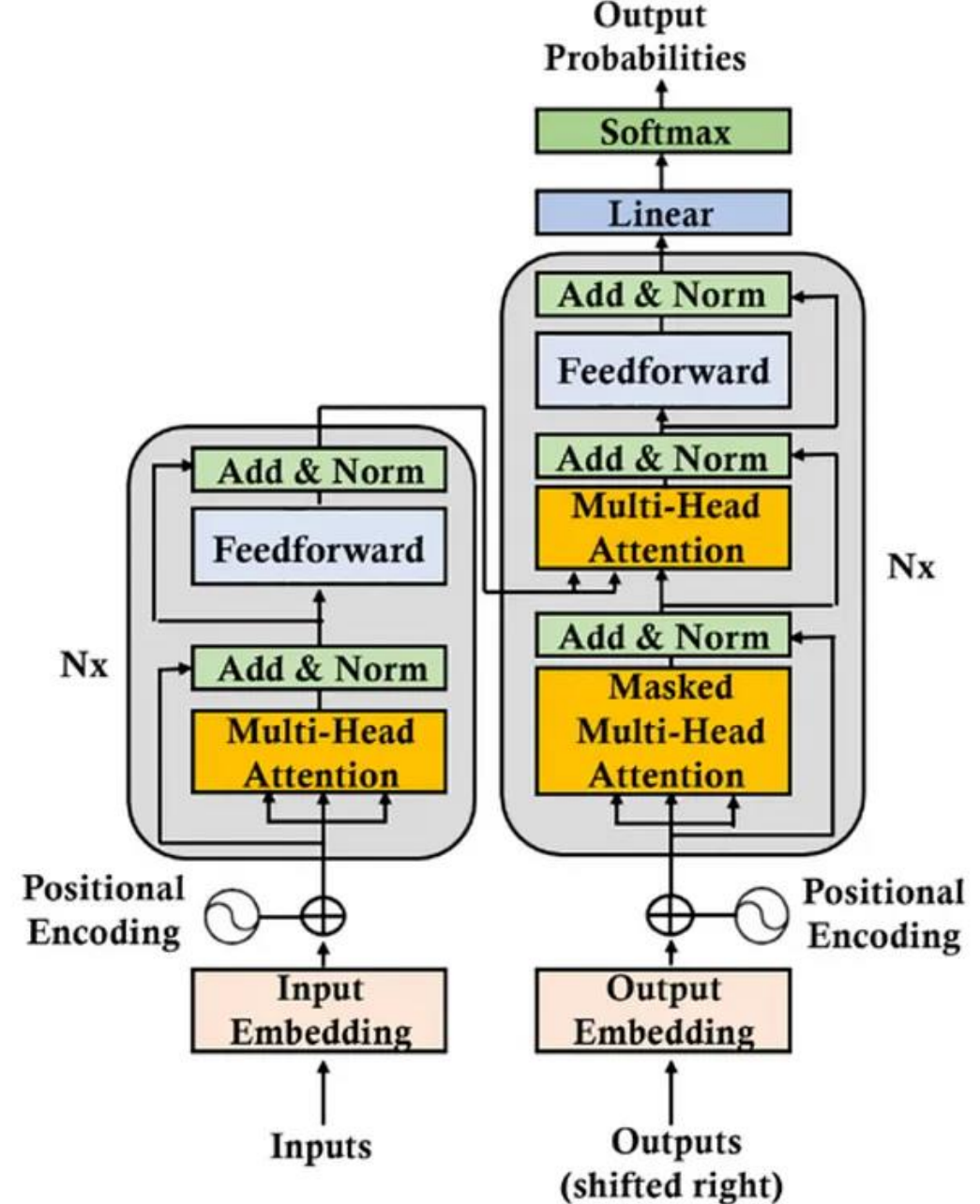
Let X and Y be sets of integers, and let $f : X \rightarrow Y$ be a bijection. This implies that there exists a map $g : Y \rightarrow X$ such that $g \circ f(x) = x$ for all $x \in X$, where \circ denotes the composition of functions. In other words, $g(f(x))=x$ for all $x \in X$.

欲しいよ!こんな生成AI

- text-to-ECC
- prop-to-proof
- 雑談-to-科研費申請書
- みなさんのご意見を教えてください。

CHATGPTのキー技術

- トランスフォーマー
文章にそれっぽい単語を
続ける機能
- アテンション
それっぽいを定める機能
- 大規模言語モデル (LLM)
大量の文章等を整理した
データ群



生成AIの欠点

- 著作権・人権侵害

生成には
元のデータが
存在

ディープフェイク

- 雇用減少

複雑な作業も
自動化、機械化

創作活動

- 情報漏洩

機密情報から生成

- 依存

コピー&ペースト
だけの作文

- 虚偽・不正確

それっぽい文章

指が6本の人間

自動証明

自動定理証明って何？

- コンピュータが証明を自動的に作ってくれる
- すでに一部の定理証明支援系には組み込まれている

Isabelle Sledgehammer

予想はできたけど
証明が思いつかない
助けて、野崎先生



しょうがないなあ。
明日まで待ってて下さい。
萩原先生。



※これは自動証明ではない

実際に使ってみよう～問題設定

- 群の公理

- 単位元の存在: $\forall x (e \cdot x = x)$

- 逆元の存在: $\forall x (x^{-1} \cdot x = e)$

- 結合法則: $\forall x \forall y \forall z ((x \cdot y) \cdot z = x \cdot (y \cdot z))$

- 証明したい定理

- 前提: $\forall x (x \cdot x = e)$

- 結論: $\forall x \forall y (x \cdot y = y \cdot x)$

実際に使ってみよう～入力

- 証明したい公理や命題をTPTP言語へ変換

```
%---- 1 * x = 1
fof(leftidentity,axiom,
  ![X] : mult(e,X) = X).

%---- i(x) * x = 1
fof(leftinverse,axiom,
  ![X] : mult(inverse(X),X) = e).

%---- (x * y) * z = x * (y * z)
fof(associativity,axiom,
  ![X,Y,Z] : mult(mult(X,Y),Z) = mult(X,mult(Y,Z))).

%---- x * x = 1
fof(groupoforder2,hypothesis,
  ![X] : mult(X,X) = e).

%----prove x * y = y * x
fof(commutativity,conjecture,
  ![X,Y] : mult(X,Y) = mult(Y,X) ).
```

first-order logic	TPTP
\perp, \top	<code>\$false, \$true</code>
$\neg F$	<code>~F</code>
$F_1 \wedge \dots \wedge F_n$	<code>F1 & ... & Fn</code>
$F_1 \vee \dots \vee F_n$	<code>F1 ... Fn</code>
$F_1 \rightarrow F_n$	<code>F1 => Fn</code>
$F_1 \leftrightarrow F_n$	<code>F1 <=> Fn</code>
$(\forall x_1) \dots (\forall x_n) F$	<code>![X1, ..., Xn] : F</code>
$(\exists x_1) \dots (\exists x_n) F$	<code>? [X1, ..., Xn] : F</code>

実際に使ってみよう～出力

```
% Running in auto input_syntax mode. Trying TPTP
% Refutation found. Thanks to Tanya!
% SZS status Theorem for problem
% SZS output start Proof for problem
1. ! [X0] : mult(e,X0) = X0 [input]
2. ! [X0] : e = mult(inverse(X0),X0) [input]
3. ! [X0,X1,X2] : mult(mult(X0,X1),X2) = mult(X0,mult(X1,X2)) [input]
4. ! [X0] : e = mult(X0,X0) [input]
5. ! [X0] : mult(X0,X1) = mult(X1,X0) [input]
6. ~! [X1] : ! [X0] : mult(X0,X1) = mult(X1,X0) [negated conjecture 5]
7. ~! [X0] : ! [X1] : mult(X0,X1) = mult(X1,X0) [rectify 6]
8. ~! [X0,X1] : mult(X0,X1) = mult(X1,X0) [flattening 7]
9. ? [X0,X1] : mult(X0,X1) != mult(X1,X0) [ennf transformation 8]
10. ? [X0,X1] : mult(X0,X1) != mult(X1,X0) => mult(sK0,sK1) != mult(sK1,sK0) [choice axiom]
11. mult(sK0,sK1) != mult(sK1,sK0) [skolemisation 9,10]
12. mult(e,X0) = X0 [cnf transformation 1]
13. e = mult(inverse(X0),X0) [cnf transformation 2]
14. mult(mult(X0,X1),X2) = mult(X0,mult(X1,X2)) [cnf transformation 3]
15. e = mult(X0,X0) [cnf transformation 4]
16. mult(sK0,sK1) != mult(sK1,sK0) [cnf transformation 11]
17. mult(X0,mult(X0,X1)) = mult(e,X1) [superposition 14,15]
19. mult(e,X1) = mult(inverse(X0),mult(X0,X1)) [superposition 14,13]
21. e = mult(X0,mult(X1,mult(X0,X1))) [superposition 14,15]
23. mult(X0,mult(X0,X1)) = X1 [forward demodulation 17,12]
24. mult(inverse(X0),mult(X0,X1)) = X1 [forward demodulation 19,12]
```

```
29. mult(inverse(X0),e) = X0 [superposition 23,13]
77. mult(X1,mult(X0,X1)) = mult(inverse(X0),e) [superposition 24,21]
92. mult(X1,mult(X0,X1)) = X0 [forward demodulation 77,29]
127. mult(X0,X1) = mult(X1,X0) [superposition 23,92]
271. mult(sK0,sK1) != mult(sK0,sK1) [superposition 16,127]
272. $false [trivial inequality removal 271]
% SZS output end Proof for problem
% -----
% Version: Vampire 4.8 (commit dd1c1e0fb on 2023-11-20 15:37:59 +0100)
% Termination reason: Refutation

% Memory used [KB]: 494
% Time elapsed: 0.002 s
```

どのように動いている？

仮定: 1. $\forall x((S(x) \vee T(x)) \rightarrow P(x))$
2. $\forall x(S(x) \vee R(x))$
3. $\neg R(a)$

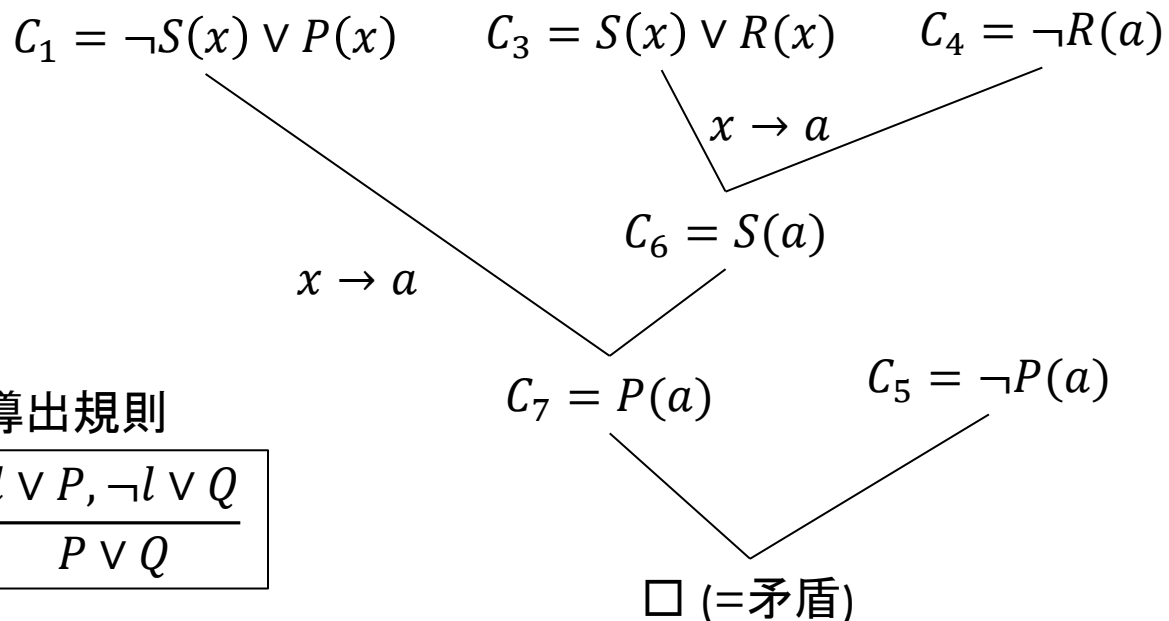
結論: $P(a)$

反駁をとる

ここから矛盾が出てくればOK

$$\forall x((S(x) \vee T(x)) \rightarrow P(x))$$
$$\forall x(S(x) \vee R(x))$$
$$\neg R(a)$$
$$\neg P(a)$$

初期節と導出節からペアを選んで導出規則を適用して矛盾が出たら終了
初期節をうまく絞り込むこと(Premise Selection)と
ペアをうまく選ぶこと(Strategy Selection)が攻略の鍵



スコールム化して
連言標準形に直す

初期節の集合

$$C_1 = \neg S(x) \vee P(x)$$
$$C_2 = \neg T(x) \vee P(x)$$
$$C_3 = S(x) \vee R(x)$$
$$C_4 = \neg R(a)$$
$$C_5 = \neg P(a)$$

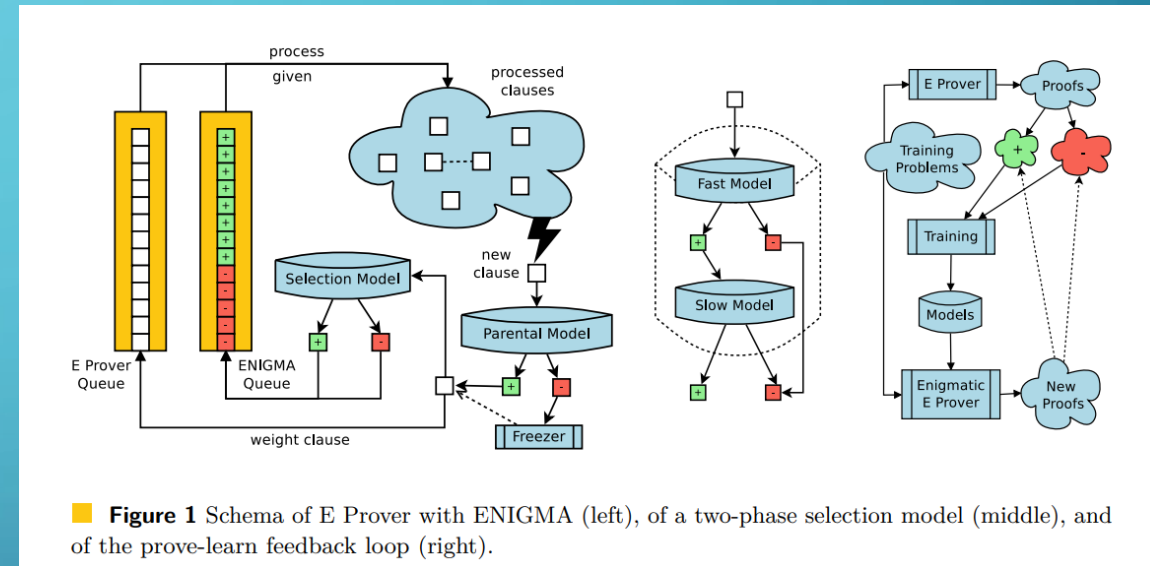
導出原理

LLM以前のディープラーニングによる自動定理証明

- Mizar 60 for Mizar 50 (2021)

solved	[%]	date	premises	methods/notes
38k	65.65	Jun 2020	\mathcal{B}	ENIGMA, reported on July 2nd at IJCAR'20 ⁸
40 268	69.57	Oct 2020	\mathcal{B}	ENIGMA
40 994	70.83	Nov 12	\mathcal{M}	ENIGMA, heuristic premise minimization
41 169	71.13	Nov 12	\mathcal{M}	Vampire with 300s limit adds 175
41 792	72.20	Nov 27	\mathcal{M}	E/ENIGMA/Vampire with more premise minimization
42 206	72.92	Dec 7	\mathcal{M}	E/ENIGMA/Vampire with more premise minimization
42 471	73.38	Jan 6	\mathcal{G}, \mathcal{E}	E with BliStr/Tune strategies on \mathcal{G}, \mathcal{E} premises
42 519	73.46	Jan 10	many	ENIGMA runs on all training predictions
42 826	73.99	May 14	$\mathcal{G}, \mathcal{L}, \mathcal{K}$	Vampire/Deepire runs – FroCoS'21 [53]
43 414	75.01	Jul 26	\mathcal{M}, \mathcal{B}	2,3-phase ENIGMA, leapfrogging
43 524	75.20	Aug 21	\mathcal{M}	3-phase ENIGMA, shifting context, leapfrog., fwd subsump.
43 599	75.33	Aug 26	\mathcal{L}	3-phase ENIGMA, leapfrogging, fwd. subsumption
43 717	75.53	Sep 2	\mathcal{M}	mainly Vampire/Deepire

■ **Table 1** Timeline of the experiments. \mathcal{B} are standard bushy premises, \mathcal{M} are subproblem-minimized premises, \mathcal{G}, \mathcal{L} , and \mathcal{K} are GNN/LightGDB/kNN-based premises, and \mathcal{E} their ensembles.



■ **Figure 1** Schema of E Prover with ENIGMA (left), of a two-phase selection model (middle), and of the prove-learn feedback loop (right).

↑ Mizar数学ライブラリの自動証明の成功率の変遷
 2014年に40%だった成功率が、2021年に75%に到達した。
 学習時間はNVIDIA V100 GPU x 1で12日。
 実験では命題一つにつき120秒打ち切り。

DEEPMIND のLLM数学ソルバ(自然言語)

• Minerva (2022)

Question: Assume that the variance of the first n natural numbers is 10, and the variance of the first m even natural numbers is 16. Compute $m + n$.

Model output: Let σ^2 be the variance of the first m even natural numbers, and τ^2 be the variance of the first n natural numbers. Then, we have $\tau^2 = \frac{n^2-1}{12}$ and $\sigma^2 = \frac{m^2-1}{3}$. We solve the system of equations:

$$\frac{n^2-1}{12} = 10$$

$$\frac{m^2-1}{3} = 16.$$

This translates into the system

$$n^2 = 121$$

$$m^2 = 48 + 1 = 49.$$

Therefore, $n = \boxed{11}$ and $m = \boxed{7}$, so $n + m = \boxed{18}$.

Question: For every $a, b, b \neq a$ prove that

$$\frac{a^2 + b^2}{2} > \left(\frac{a+b}{2}\right)^2.$$

Model output:

$$\frac{a^2+b^2}{2} > \left(\frac{a+b}{2}\right)^2$$

$$\iff \frac{a^2+b^2}{2} > \frac{a^2+b^2+2ab}{4}$$

$$\iff a^2 + b^2 > \frac{a^2+b^2+2ab}{2}$$

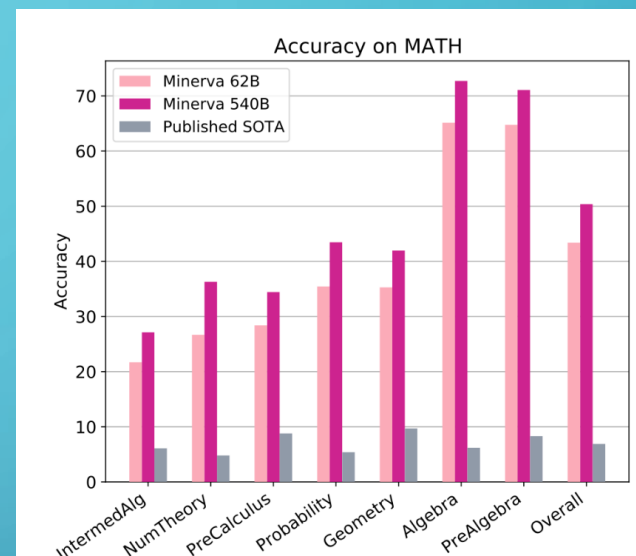
$$\iff 2a^2 + 2b^2 > a^2 + b^2 + 2ab$$

$$\iff a^2 + b^2 > 2ab$$

$$\iff a^2 + b^2 - 2ab > 0$$

$$\iff (a-b)^2 > 0$$

which is true, because the square of a real number is positive.



従来, LLMは数学が苦手だったものをファインチューンした. MATH dataset (米国の競技数学レベルの問題セット)の正答率はPaLM 540Bで8.8%だったが, Minerva 540Bでは50.3%に向上.

OPEN AIのLLM数学ソルバ(自然言語)

Let's Verify Step by Step (2023)

Let $x^8 + 3x^4 - 4 = p_1(x)p_2(x) \cdots p_k(x)$,
 where each non-constant polynomial $p_i(x)$ is monic with integer coefficients, and cannot be factored further over the integers. Compute $p_1(1) + p_2(1) + \cdots + p_k(1)$.

I notice that the given polynomial has even degree and only even powers of x , so I can try to make a substitution to simplify it.

Let $y = x^4$, then the polynomial becomes $y^2 + 3y - 4$, which is a quadratic equation.

I can factor this quadratic equation as $(y + 4)(y - 1)$, so the original polynomial is $(x^4 + 4)(x^4 - 1)$.

Now I need to factor each of these factors further, if possible, over the integers.

For the first factor, $x^4 + 4$, I recall the Sophie Germain identity, which says that $a^4 + 4b^4 = (a^2 + 2b^2 + 2ab)(a^2 + 2b^2 - 2ab)$.

Applying this identity with $a = x$ and $b = 1$, I get $x^4 + 4 = (x^2 + 2x + 2)(x^2 - 2x + 2)$.

For the second factor, $x^4 - 1$, I can use the difference of squares formula, which says that $a^2 - b^2 = (a + b)(a - b)$.

Applying this formula with $a = x^2$ and $b = 1$, I get $x^4 - 1 = (x^2 + 1)(x^2 - 1)$.

Then I can apply the difference of squares formula again with $a = x$ and $b = 1$, to get $x^2 - 1 = (x + 1)(x - 1)$.

So the original polynomial is $(x^4 + 4)(x^4 - 1) = (x^2 + 2x + 2)(x^2 - 2x + 2)(x^2 + 1)(x + 1)(x - 1)$.

These are all the irreducible factors over the integers, so $p_1(x) = x^2 + 2x + 2$, $p_2(x) = x^2 - 2x + 2$, $p_3(x) = x^2 + 1$, $p_4(x) = x + 1$, and $p_5(x) = x - 1$.

To find the sum of their values at $x = 1$, I just need to plug in $x = 1$ into each factor and add them up.

I get $p_1(1) = 1^2 + 2 \cdot 1 + 2 = 5$, $p_2(1) = 1^2 - 2 \cdot 1 + 2 = 1$, $p_3(1) = 1^2 + 1 = 2$, $p_4(1) = 1 + 1 = 2$, and $p_5(1) = 1 - 1 = 0$.

So the final answer is $5 + 1 + 2 + 2 + 0 = 10$.

Answer: 10

Let $x^8 + 3x^4 - 4 = p_1(x)p_2(x) \cdots p_k(x)$,
 where each non-constant polynomial $p_i(x)$ is monic with integer coefficients, and cannot be factored further over the integers. Compute $p_1(1) + p_2(1) + \cdots + p_k(1)$.

This problem is about factoring a polynomial over the integers, which can be a tricky task.

One possible strategy is to look for rational roots of the polynomial, using the rational root theorem.

However, that would involve checking a lot of possible factors of -4 , and there is no guarantee that the polynomial has any rational roots at all.

Another possible strategy is to use some algebraic tricks to simplify the polynomial or reduce its degree.

I notice that the polynomial has only even powers of x , so I can try to substitute $y = x^2$ and see what happens.

Then the polynomial becomes $y^4 + 3y^2 - 4$, which is a quadratic equation in y^2 .

I can try to factor this quadratic equation by looking for two numbers that multiply to -4 and add to 3 .

Those numbers are 4 and -1 , so I can write $y^4 + 3y^2 - 4 = (y^2 + 4)(y^2 - 1)$.

Now I can substitute back $x^2 = y$ and get $x^8 + 3x^4 - 4 = (x^4 + 4)(x^4 - 1)$.

This looks promising, because I have reduced the degree of the factors from 8 to 4 .

I can try to factor each of these factors further, using the difference of squares formula.

I get $x^8 + 3x^4 - 4 = (x^2 + 2)(x^2 - 2)(x^2 + 1)(x^2 - 1)$.

I can apply the difference of squares formula again to the last factor and get $x^8 + 3x^4 - 4 = (x^2 + 2)(x^2 - 2)(x^2 + 1)(x + 1)(x - 1)$.

Now I have factored the polynomial completely into monic linear and quadratic factors with integer coefficients.

These are the $p_i(x)$'s that the problem is asking for.

To find the sum of their values at $x = 1$, I just need to plug in $x = 1$ into each factor and add them up.

I get $p_1(1) + p_2(1) + \cdots + p_k(1) = (1^2 + 2)(1^2 - 2)(1^2 + 1)(1 + 1)(1 - 1)$.

Simplifying, I get $p_1(1) + p_2(1) + \cdots + p_k(1) = (3)(-1)(2)(2)(0)$.

Multiplying, I get $p_1(1) + p_2(1) + \cdots + p_k(1) = 0$.

Answer: 0

Figure 2: Two solutions to the same problem, graded by the PRM. The solution on the left is correct while the solution on the right is incorrect. A green background indicates a high PRM score, and a red background indicates a low score. The PRM correctly identifies the mistake in the incorrect solution.

	ORM	PRM	Majority Vote	# Problems
AP Calculus	68.9%	86.7%	80.0%	45
AP Chemistry	68.9%	80.0%	71.7%	60
AP Physics	77.8%	86.7%	82.2%	45
AMC10/12	49.1%	53.2%	32.8%	84
Aggregate	63.8%	72.9%	61.3%	234

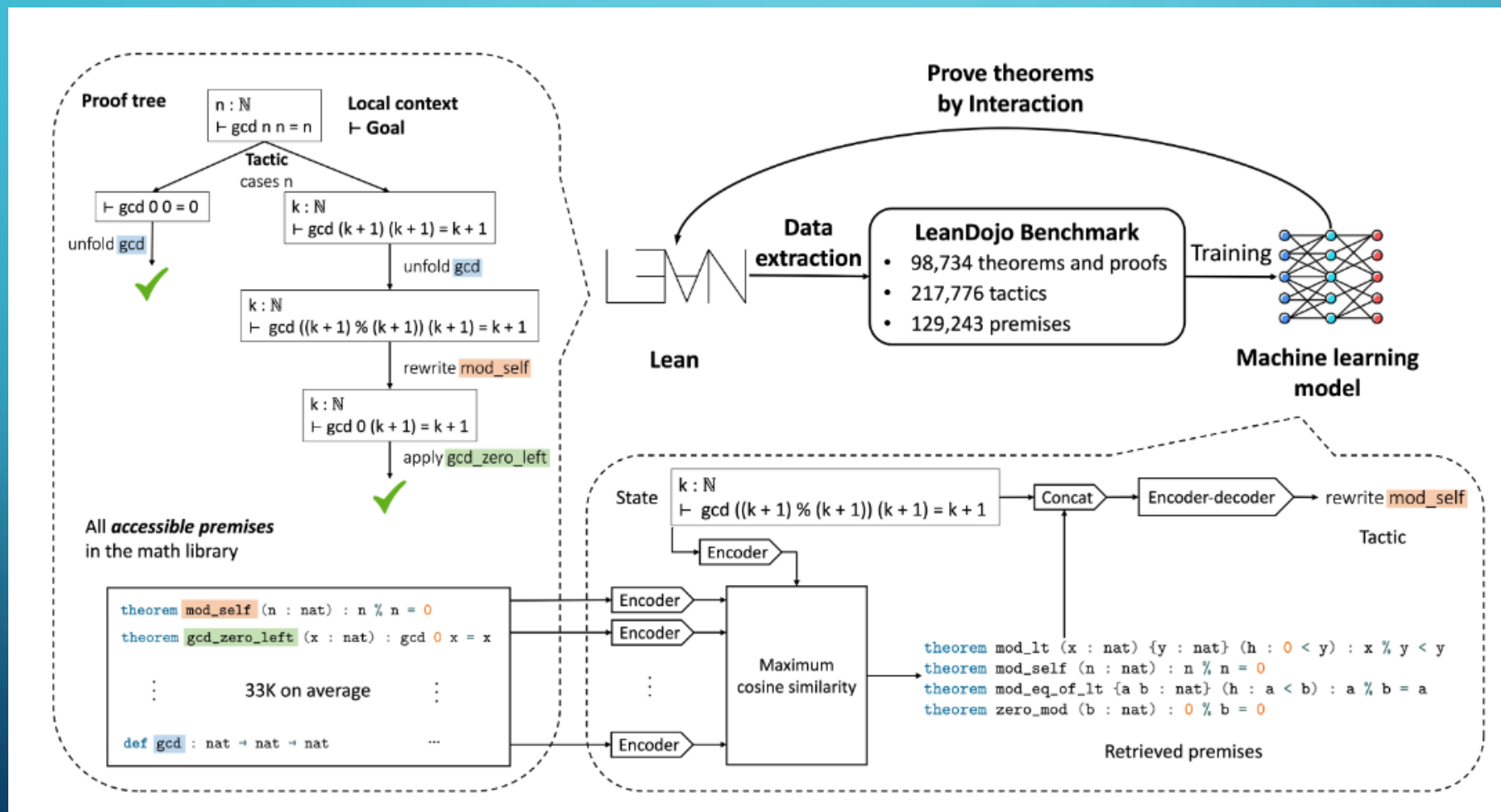
Table 1: We measure out-of-distribution generalization using recent STEM tests. We evaluate the outcome-supervised RM, the process-supervised RM, and majority voting using 100 test samples per problem.

答案の途中経過が正しいかを機械学習モデルで監視させることによって精度を向上させた DeepMindのMinervaでは50%だったMATH datasetの解答精度をわずか1年で80%まで向上.

ボトルネックは学習データのアノテーション付けを人間が行っていること. 定理証明支援系と組み合わせれば自動化できるのでは?

自動定理証明 + LLM + LEAN = ?

- Lean Dojo(2023)



AIは国際数学オリンピック(IMO)で金メダルを取れるか？

- 1位賞金 500万ドル = 7.4億円

\$10mn AI Mathematical Olympiad Prize Launches



XTX Markets is launching a new \$10mn challenge fund, the **Artificial Intelligence Mathematical Olympiad Prize (AI-MO Prize)**. The fund intends to spur the development of AI models that can reason mathematically, leading to the creation of a publicly-shared AI model capable of winning a gold medal in the **International Mathematical Olympiad (IMO)**.



The grand prize of \$5mn will be awarded to the first publicly-shared AI model to enter an AI-MO approved competition and perform at a standard equivalent to a gold medal in the IMO.

There will also be a series of progress prizes, totalling up to \$5mn, for publicly-shared AI models that achieve key milestones towards the grand prize.

The AI-MO Prize [...] will help compare different AI problem solving strategies at a technical level, in a manner that will be accessible and appealing to the broader public.

Terence Tao, UCLA

[view more](#)

I am sure that many people will be following the AI-MO Prize with great excitement, to see when, in the future, AI will match the world's brightest young minds.

Gregor Dolinar, IMO President

[view more](#)



The prizes will be designed by an **AI-MO Advisory Committee**, including mathematicians, AI and machine learning experts, and experienced Olympiad problem-setters.

The first AI-MO approved competitions will open to participants in early 2024. There will be a presentation of progress at **the 65th IMO**, which will be held in Bath, England in July 2024.

出典: <https://aimoprize.com/>

- 形式化数学グループはLeanで解くことをレギュレーションに加えるように提案



IMO Grand Challenge

The **International Mathematical Olympiad (IMO)** is perhaps the most celebrated mental competition in the world and as such is among the ultimate grand challenges for Artificial Intelligence (AI).

The challenge: build an AI that can win a gold medal in the competition.

To remove ambiguity about the scoring rules, we propose the formal-to-formal (F2F) variant of the IMO: the AI receives a *formal* representation of the problem (in the **Lean Theorem Prover**), and is required to emit a *formal* (i.e. machine-checkable) proof. We are working on a proposal for encoding IMO problems in Lean and will seek broad consensus on the protocol.

Other proposed rules:

Credit. Each proof certificate that the AI produces must be checkable by the Lean kernel in 10 minutes (which is approximately the amount of time it takes a human judge to judge a human's solution). Unlike human competitors, the AI has no opportunity for partial credit.

Resources. The AI has only as much time as a human competitor (4.5 hours for each set of 3 problems), but there are no other limits on the computational resources it may use during that time.

Reproducibility. The AI must be open-source, released publicly before the first day of the IMO, and be easily reproduceable. The AI cannot query the Internet.

Challenge. The grand challenge is to develop an AI that earns enough points in the F2F version of the IMO (described above) that, if it were a human competitor, it would have earned a gold medal.

Note: this is only a preliminary proposal for the rules. To get involved in the discussion, please join our **Zulip channel**.

出典: <https://imo-grand-challenge.github.io/>

AIがIMOで金メダルを取るのはいつ頃だと予想しますか？

会場アンケート

- 1年以内
- 1～3年
- 3～5年
- 5～10年
- 10～20年
- 20年～

AIがIMOで金メダルを取るのはいつ頃だと予想しますか？

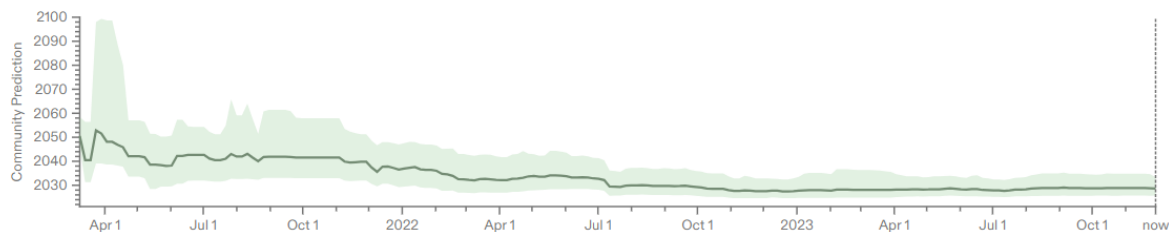
- 2028年が平均的な予想
- 生成AIの登場で予想が前倒しに

When will an AI win a Gold Medal in the International Math Olympiad?

Sep 11, 2028

676 predictions

33 Closes Jan 1, 2100 35 comments



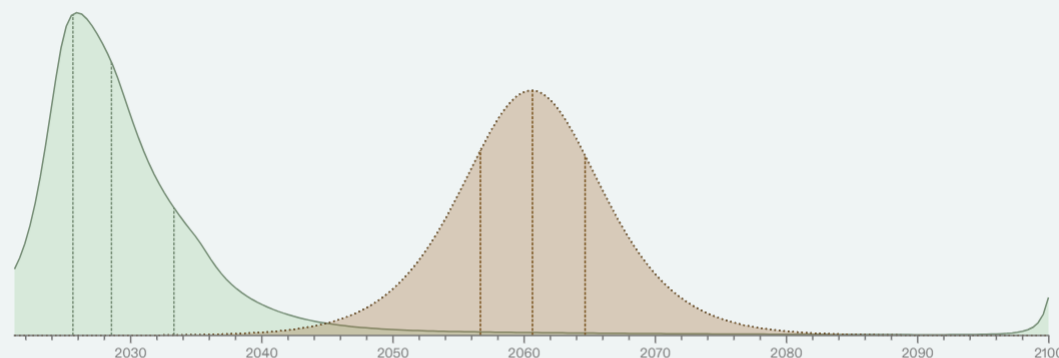
Total Forecasters
213

Community Prediction
Sep 11, 2028

Make a Prediction

Probability Density

Recency weighted



Sign Up to Predict

	My Prediction	Community
lower 25%	...	2025
median	...	2028
upper 75%	...	2033
Not ≤ Dec 23, 2099	...	4.3%

生成AIを使った形式証明

デモ：CHATGPTとLEANの合わせ技

- オンラインLEANで「マルコフの不等式」を証明してみましょう。
- これから証明する命題の主張
「 X を有限個の確率変数とし、 $a > 0$ とすると、
 $P(|X| \geq a) \leq E(|X|)/a$ が成り立つ」

形式化のコツ

1. 紙とペンで証明のアウトラインを書いておこう
2. 小さい補題をたくさん作ろう
3. 困ったらhaveで構造化しよう
4. 一般化は後回しにしよう
5. 分からないことはLLMに質問しよう



LEAN のココが凄い (*° 0° *)

- 証明もプログラミングも両方できる
- 表現力が高く, 抽象的な概念も朝飯前
- 自然言語に近い形に構造化して書くことができる
- タクティクスを書いて証明を自動化できる
- プレースホルダ (Leanが補完してくれる) が便利
- Microsoft謹製のオープンソース
- チュートリアルやドキュメントが豊富
- コミュニティが大きく, 数学ライブラリが急速に成長中

LEAN のココが不満 (>_<) ※個人差があります

- 理解するのが難しい
 - 型理論, 圏論, 関数型言語, ラムダ計算, モナド...
 - 文法が複雑, 特にタクティクスの文法はカオス...
 - 抽象度が高すぎて, ライブラリを読むのに苦痛を伴う
- 証明を書くのに時間がかかる
 - 論文の20~100倍, プログラミングの10倍~30倍
 - そもそも厳密に証明すること自体が難しい
 - 型付けが厳密で, エラーの読解が難しい
 - ライブラリが大きいと定理検索が大変
 - まだまだ自動推論が弱い

分からないことはLLMに質問しよう

- 「定理や定義のアウトラインを教えて」
- 「この定理 (定義 or 証明) をLeanに翻訳して」
- 「こういう定理がライブラリにあるか教えて」
- 「この証明に使えそうなタクティクスを教えて」
- 「このLeanのコードを解説して」
- 「Leanでこういった文法はある？」
- 「こういうエラーが出たのだけれど, どういう意味? できれば直して」



LEAN FAQ

- Q1. 最新版はLean4だけどLean3を使ってデモしたのはどうして?

A1. まだChatGPTがLean4を学習していないためです。

Lean3とLean4は文法がかなり異なるので、注意してください。

- Q2. 最初は何から勉強すれば良いの?

A2. リンク集の「Learn Lean」を上から順に学習するのがお勧めです

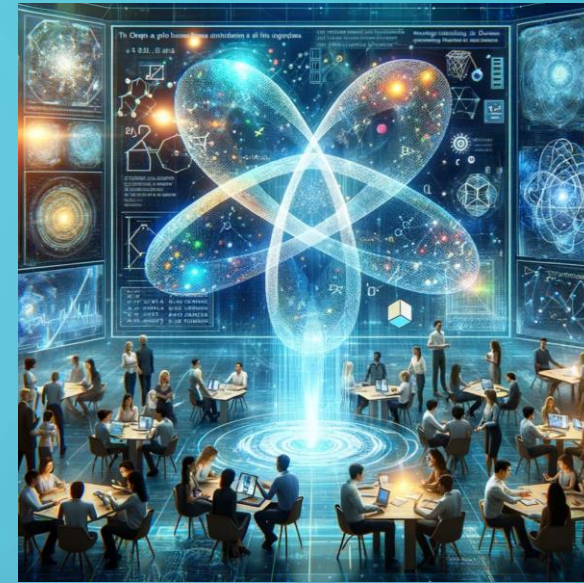
<https://github.com/fpvandoorn/lean-links>

- Q3. Web版よりローカルにインストールした方がいいの?

A3. 最初はWeb版でも構いません。本格的に使いたくなったら

VSCoodeのGitHub Copilotが使えるようにローカルにインストールしましょう。

定理証明支援系と 自動定理証明の未来



• いま

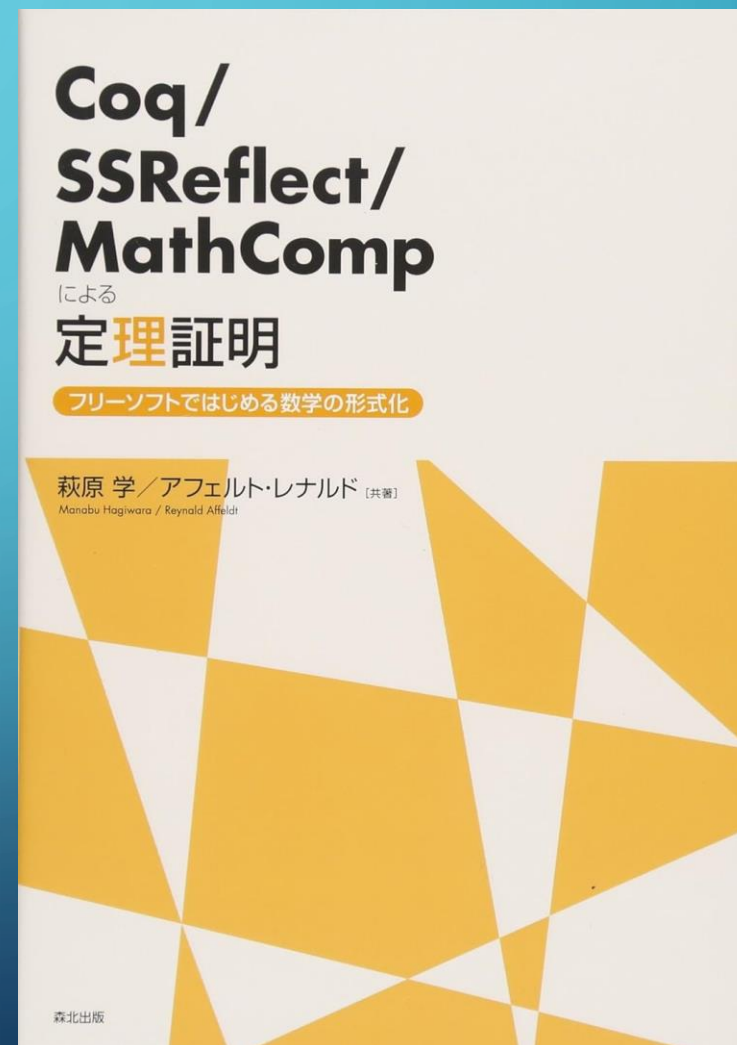
- 定理証明支援系による形式化証明は, 人間が書かなければならない
- LLMや検索エンジンを使うことで, 形式化を効率化できる ← new

• これから

- プログラミング環境やLaTeX環境に定理証明支援系が組み込まれる
- AIが定理証明支援系と自動定理証明を使って, プログラムや証明を裏側でチェック
- 人類の数理知識はAIによって形式化されてデータベース化

もっと詳しく：定理証明支援系COQの参考書

- 萩原学/アフェルト・レナルド著
「Coq/SSReflect/MathCompによる定理証明」



情報理論の形式化の現状

シャノンの定理周り / COQ

- Formalization of Shannon's theorems,
R Affeldt, M Hagiwara, J Sénizergues,
Journal of automated reasoning, 2014 – Springer
- Formalization of the variable-length source coding theorem: Direct part,
Ryosuke Obi, Manabu Hagiwara, Reynald Affeldt,
2014 ISITA
- Formalization of binary symmetric erasure channel based on infotheo,
Kyosuke Nakano, Manabu Hagiwara,
2016 ISITA
- Examples of Formal Proofs about Data Compression,
Reynald Affeldt; Jacques Garrigue; Takafumi Saikawa
2018 ISITA

削除誤り訂正符号 / LEAN

- Formalization of coding theory using lean,
Manabu Hagiwara, Kyosuke Nakano, Justin Kong,
2016 ISITA
- Formalization of Insertion/Deletion Codes and the Levenshtein Metric in Lean,
Justin Kong, David J Webb, Manabu Hagiwara,
2018 ISITA
- Formalization of VT Codes and Their Single-Deletion Correcting Property in
Lean,
Y Kondo, M Hagiwara, M Kudo,
2020 ISITA

その他

- Verified Erasure Correction in Coq with MathComp and VST
Joshua M. Cohen, Qinshi Wang & Andrew W. Appel
2022 Computer Aided Verification
- Formal verification of ECCs for memories using ACL2
M Naseer, W Ahmad, O Hasan
Journal of Electronic Testing 2020 - Springer
- A library for formalization of linear error-correcting codes
R Affeldt, J Garrigue, T Saikawa
Journal of Automated Reasoning 2020 - Springer
- Markov chains and Markov decision processes in Isabelle/HOL
J Hölzl - Journal of Automated Reasoning 2017
- Formalization of DPI and Jensen's Inequality in HOL
C Dunchev, G Helali, O Hasan, S Tahar
2016 - hvg.ece.concordia.ca
- Formalization of error-correcting codes: from Hamming to modern coding theory,
R Affeldt, J Garrigue
Theorem Proving: 6th International Conference, ITP 2015
- Formalization of entropy measures in HOL
T Mhamdi, O Hasan, S Tahar
International Conference on Interactive Theorem Proving 2011 - Springer
- Formalized Information-Theoretic Proofs of Privacy Using the HOL4 Theorem-Prover
Aaron R. Coble
International Symposium on Privacy Enhancing Technologies Symposium PETS 2008: Privacy Enhancing Technologies



ISITA SINCE 2020

Home

Committee

Technical Program

Online Events

Plenary Talks

Information for Authors

Registration

How to Present Your
Talks

Paper Awards

Call for Papers

Interested authors are invited to submit papers describing novel and previously unpublished results on topics in information theory and its applications, including, but not limited to:

- Boolean Functions and Sequences
- Coding for Storage
- Coding Theory
- Communication Theory
- Computation and Complexity in Information Theory
- Cryptography and Information-Theoretic Security
- Data Privacy and Security
- Deep Learning in Information Theory
- Distributed Coding and Computation
- Estimation and Detection
- Formalization of Information Theory
- Group Testing
- Information Hiding
- Information Theory for Biology
- Information Inequalities
- Network Coding and Information Theory
- Pattern Recognition and Machine Learning
- Quantum Information and Coding Theory
- Shannon Theory
- Signal Processing
- Source Coding and Data Compression
- Sparsity and Compressed Sensing
- Statistical Inference and Learning
- Statistical Physics for Information Theory
- Statistics and Information Geometry
- Wireless Communications

あります。エントロピー周りの
ライブラリ

- https://leanprover-community.github.io/mathlib4_docs/Mathlib/InformationTheory/Hamming.html
- みましよう。



おしまい