Sobre la promesa de automatizar el descubrimiento matemático.

Retos en el uso de IA para demostraciones matemáticas

Nancy Abigail Nuñez Hernández

Octubre 7, 2025

Contenido

Contenido

Demostradores automáticos y asistentes de prueba

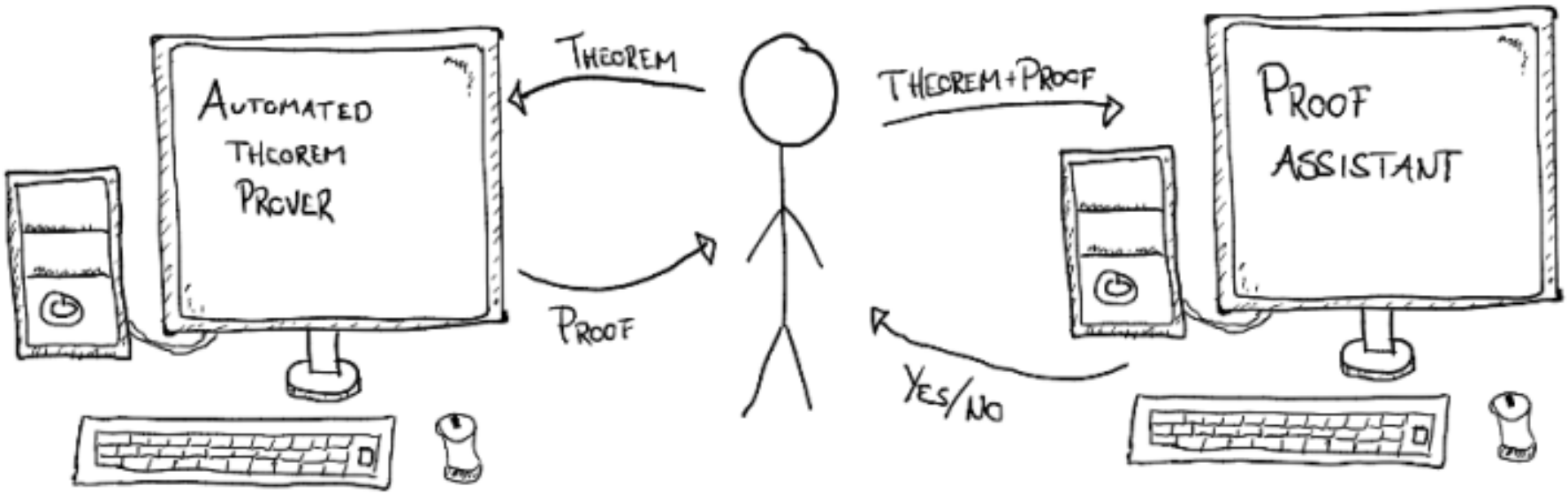


Figura 1. Automated theorem provers and their role in cryptography por M. R. Solberg y K. Gjøsteen. 2023

Demostradores automáticos y asistentes de prueba

- Los demostradores automáticos son herramientas diseñadas para aceptar como entrada (input) fórmulas en un lenguaje lógico y establecer, sin ayuda humana) si la fórmula es satisfacible en en el lenguaje del demostrador. E. g., E, Porver9 and Mace4

$$\text{¿Es } \varphi \text{ insatisfacible? } \equiv \text{¿Es } \neg\varphi \text{ válida?}$$

- Dado que la meta de los demostradores automáticos is demostrar la validez de alguna fórmula, basta con tener una herramienta que pueda verificar satisfacibilidad.
- Los asistentes de prueba también son herramientas diseñadas para razonar acerca de fórmulas en un lenguaje lógico. E. g., HOL, Rocq, Lean. Pero generalmente no establecen validez o satisfacibilidad de las fórmulas por si mismos, sino que ofrecen una interfaz donde los usuarios pueden escribir sus propios programas que demuestren la validez de las fórmulas.

Demostradores automáticos, asistentes de prueba & IA

- En los primeros años de la IA [Newell and Simon, 1956] desarrollaron un programa llamado "Logic Theorist" para derivar teoremas a partir de reglas de inferencia.
- "Logic Theorist" demostró 38 teoremas de *Principia Mathematica*.
- El teorema de los 4 colores. [Robertson et al., 1997] ofreció una prueba que fue verificada por [Gonthier, 2005] usando el asistente de prueba Rocq.
- La conjetura de Kepler. [Hales, 2005] recurrió a los asistentes de prueba Isabelle y HOL Light para verificar la corrección de la prueba.

- [Urban and Vyskočil, 2013] ha trabajado en combinar técnicas de aprendizaje de máquinas con razonamiento automatizado para automatizar la demostración de teoremas. Vease también [Bengio et al., 2021, Rabe et al., 2020].
- Los modelos del lenguaje de gran tamaño (LLMs por las siglas de *Large Language Models*) en combinación con técnicas re razonamiento automatizado o sistemas de verificación han producido resultados interesantes. Por ejemplo, AlphaProof y AlphaGeometry 2 resolvieron 4 de 6 problemas de la Olimpiada Internacional de Matemáticas 2024.
- [Tao, 2024] ha señalado diferentes maneras en las que algoritmos de aprendizaje de máquinas, LLMs y asistentes de prueba pueden ser usados en la investigación matemática.
- La agencia norteamericana Defense Advanced Research Projects Agency (DARPA) tiene una inciativa denominada expMath cuyo objetivo es desarrollar una IA co-autora, i. e., una herramienta capaz de descomponer problemas grandes y complejos en problemas más pequeños y simples, que sean más fáciles de entender y rápidos de resolver.

La IA está robando nuestros trabajos...

- Algunos piensan que el progreso de la IA nos permitirá automatizar el descubrimiento de demostraciones matemáticas, haciendo la labor de demostrar la verdad de enunciados matemáticos más fácil o sin esfuerzo.

- ¿Qué tan difícil es el trabajo de los matemáticos?
- En 1956, Gödel escribó a von Neumann preguntando si "el razonamiento de las/los matemáticos/as sobre preguntas de sí o no puede ser completamente reemplzado por máquinas." [Hartmanis, 1993, p. 6]

Contenido

- Gödel estaba preguntando si hay un procedimiento mecánico –o algoritmo– ***factible*** para decidir si una fórmula de lógica de primer orden es demostrable.[Buss, 1995]
- Cuando las/los matemáticas/os demuestran un teorema, muestran que un enunciado se sigue de un conjunto de axiomas o teoremas.
- Demostrar que un enunciado es una consecuencia lógica de un conjunto de axiomas es un problema que en teoría de la complejidad computacional se conoce como **Implicación Lógica**.
- El problema de la Implicación Lógica es un problema **coNP-completo**.

Clases de complejidad

- La teoría de complejidad computacional investiga y asigna valores a los problemas de decisión que pueden resolverse mediante un procedimiento mecánico o un algoritmo.
  [Arora and Barak, 2009, Goldreich, 2010]
- Problema de decisión: ¿x tiene la propiedad P? Si la tiene, entonces $x \in P$.
- $\mathbf{P}$: toda instancia de un problema en $\mathbf{P}$ puede resolverse mediante un algoritmo que da la respuesta correcta en tiempo polinomial.
- $\mathbf{EXP}$: resolver instancias de problemas en esta clase puede tomar tiempo exponencial.

Polinomial vs Exponencial

Polinomial $f(n) = n^2$

- $2^2 = 4$
- $3^2 = 9$
- $8^2 = 64$
- $10^2 = 100$

Exponencial $f(n) = 2^n$

- $2^2 = 4$
- $2^3 = 8$
- $2^8 = 256$
- $2^{10} = 1024$

La clase NP

- $\mathbf{P}$:toda instancia de un problema en $\mathbf{P}$ puede resolverse mediante un algoritmo que da la respuesta correcta en tiempo polinomial.
- $\mathbf{EXP}$: resolver instancias de problemas en esta clase puede tomar tiempo exponencial.
- $\mathbf{NP}$: no conocemos ningún algoritmo que *corra* en tiempo polinomial que pueda resolver cualquier instancia de un problema en $\mathbf{NP}$, pero si hay una propuesta de solución, se puede verificar de manera **eficiente**, es decir, a través de un algoritmo que *corra* en tiempo polinomial, el cual puede ser visto como **certificado** o prueba de la solución.
- $\mathbf{coNP}$...

P vs NP

Problema en $\mathbf{P}$

- Ejemplo: 2-SAT
- $(x \vee y) \wedge (x \vee \neg z)$
- Dada una fórmula 2CNF φ, ¿existe una asignación que satisfaga φ?
- Contamos con algoritmos que pueden decirnos en tiempo polinomial si una instancia cualquiera de 2CNF se puede satisfacer.

Problema ien $\mathbf{NP}$

- Ejemplo: 3-SAT
- $(x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z)$
- Dada una fórmula 3CNF φ, ¿existe una asignación que satisfaga φ?
- No hay un algoritmo que pueda decirnos en tiempo polinomial para cualquier instancia de 3CNF se puede satisfacer. Si de hecho se puede satisfacer y tenemos una prueba, podemos verificarlo en tiempo polinomial.

$$\text{¿}\mathbf{P} = \mathbf{NP}?$$

La clase coNP

- $\mathbf{P}$: toda instancia de un problema en $\mathbf{P}$ puede resolverse mediante un algoritmo que da la respuesta correcta en tiempo polinomial.
- $\mathbf{EXP}$: resolver instancias de problemas en esta clase puede tomar tiempo exponencial.
- $\mathbf{NP}$: no conocemos ningún algoritmo que *corra* en tiempo polinomial que pueda resolver cualquier instancia de un problema en $\mathbf{NP}$, pero si hay una propuesta de solución, se puede verificar de manera **eficiente**, es decir, a través de un algoritmo que *corra* en tiempo polinomial, el cual puede ser visto como **certificado** o prueba de la solución.
- $\mathbf{coNP}$: no conocemos un algoritmo que *corra* en tiempo polinomial que pueda resolver cualquier instancia de un problema en $\mathbf{coNP}$ o que pueda veriticar propuestas de solución. Pero si x no es una intancia de un problema en $\mathbf{coNP}$, se puede **refutar** de manera eficiente.

$$\text{¿}\mathbf{NP} = \mathbf{coNP}?$$

coNP

- Ejemplo: TAUT.
- Dada una fórmula φ, ¿es φ verdadera en cualquier interpretación?
- Si la respuesta es 'sí', **no** hay un algoritmo que *corra* en tiempo polinomial que pueda *verificarlo*.
- Si la respuesta es 'no', **sí** hay un algoritmo que *corra* en tiempo polinomial capaz de *refutarlo*.

coNP

¿Quieres saber si φ es una tautología?

Te puede decir de manera eficiente si **no** es una tautología.

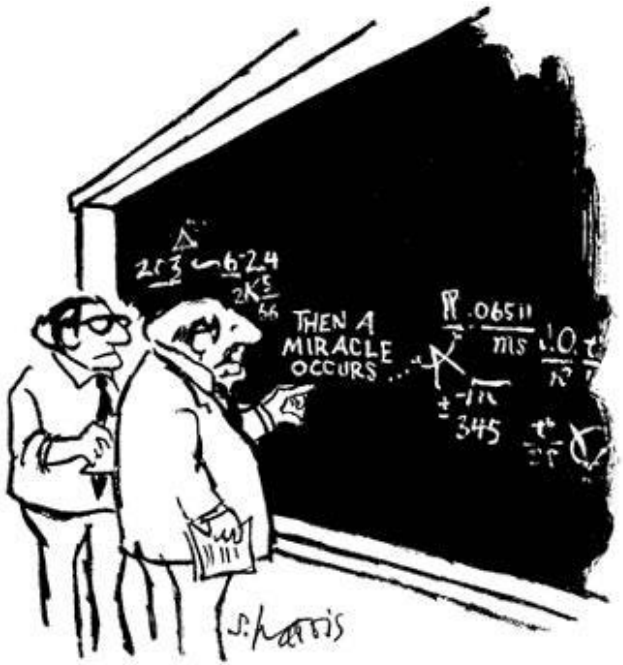
Implicación lógica

- Si la lógica se caracteriza en términos de teoría de la prueba, i. e., en términos del conjunto de fórmulas que se puede derivar de un conjunto de axiomas, TAUT coincide con el problema de decidir si φ es derivable de esos axiomas.[1]
- Implicación Lógica (IL): ¿La fórmula φ de lógica proposicional es implicada por el conjunto de premisas Γ?
- IL: ¿$\Gamma \models \varphi$?
- IL tiene la misma complejidad que TAUT, i.e., IL es un problema **coNP-completo**.

[1] La lógica proposicional y la lógica de primer orden son completas: cualquier fórmula verdadera en todos los modelos de la teoría debe ser logicamente deducible de la teoría y viceversa.

Contenido

Demostraciones matemáticas



"I think you should be more explicit here in step two."

"Los teoremas y las demostraciones son el principal producto en las matemáticas."[Pudlák, 2013, p. 65]
"Una demostración es una secuencia de símbolos que satisfacen ciertas reglas sintácticas. ...
La experiencia diaria de lxs matemáticxs es que a veces es extremadamente difícil encontrar una demostración de un enunciado dado."[Pudlák, 2013, p. 94]

La complejidad de las demostraciones matemáticas

- ¿$\Gamma \models \varphi$?
- Digamos que **sí**. Entonces **no** hay un algoritmo que *corra* en tiempo polinomial capaz de *verificarlo*.
- Digamos que **no**. Entonces **sí** hay un algoritmo que *corra* en tiempo polinomial capaz de *refutarlo*.
- No es el caso que siempre podremos saber de manera eficiente si $\Gamma \models \varphi$.
- "Reconocer la corrección de una respuesta suele ser más fácil que dar la respuesta... verificar la prueba de un teorema es más fácil que dar la prueba (un hecho referido en la carta de Gödel)". [Arora and Barak, 2009, p. 58]

¿Podemos automatizar el descubrimiento de demostraciones matemáticas?

- La complejidad de la Implicación Lógica muestra que no siempre podemos saber de manera eficiente cuando una algo es una consecuencia lógica de lo que ya sabemos.
- Debido a la complejidad del descubrimiento de pruebas en matemáticas, podría no ser posible automatizar las demostraciones matemáticas a través del uso IA, demostradores automáticos o asistentes de prueba.
- Resultados limitativos en lógica (Gödel, Tarski, Church-Turing) implican que ni siquiera la teoría elemental de números se puede hacer de manera completamente automática.
- Decidir la valdiez de una fórmula de FOL is *semidecidible*, i.e., hay procedimientos completos de prueba que pueden *correr* indefinidamente en fórumlas inválidas.

Pregunta de João Marcos y Francisco Hernández Quiroz

Si nos enfocamos en la complejidad computacional, en los recursos computacionales, entonces parece que los seres humanos tampoco podríamos hacerlo los seres humanos.

[Van Rooij, 2008] propone que las capacidades cognitivas humanas están restringidas en tanto que los humanos son seres finitos con recursos computacionales limitados.

Pero en lugar de sostener que las capacidades cognitivas humanas deben ser computables en tiempo polinomial, propone que las funciones cognitivas deben ser computables con respecto a un parámetro fijo tratable. "Cognitive functions are among the functions that are fixed-parameter tractable for one or more input parameters that are small in practice." [Van Rooij, 2008]

¡Gracias!

Contenido

References I


Arora, S. and Barak, B. (2009).
Computational complexity: A modern approach.
Cambridge University Press.


Bengio, Y., Lecun, Y., and Hinton, G. (2021).
Deep learning for AI.
Commun. ACM, 64(7):58–65.


Buss, S. R. (1995).
On gödel's theorems on lengths of proofs ii: Lower bounds for recognizing k symbol provability.
In *Feasible mathematics II*, pages 57–90. Springer.


Goldreich, O. (2010).
P, NP, and NP-Completeness: The basics of computational complexity.
Cambridge University Press.


Gonthier, G. (2005).
A computer-checked proof of the four colour theorem.


Hales, T. C. (2005).
A proof of the kepler conjecture.
Annals of mathematics, pages 1065–1185.

References II


Hartmanis, J. (1993).
Gödel, von neumann and the p=? np problem.
In *Current Trends in Theoretical Computer Science: Essays and Tutorials*, pages 445–450. World Scientific.


Newell, A. and Simon, H. (1956).
The logic theory machine–a complex information processing system.
IRE Transactions on information theory, 2(3):61–79.


Pudlák, P. (2013).
Logical foundations of mathematics and computational complexity: A gentle introduction.
Springer.


Rabe, M. N., Lee, D., Bansal, K., and Szegedy, C. (2020).
Mathematical reasoning via self-supervised skip-tree training.


Robertson, N., Sanders, D., Seymour, P., and Thomas, R. (1997).
The four-colour theorem.
journal of combinatorial theory, Series B, 70(1):2–44.


Tao, T. (2024).
Machine assisted proof.
Notices of the American Mathematical Society, 72(1):6–13.

References III

Urban, J. and Vyskočil, J. (2013).
Theorem proving in large formal mathematics as an emerging ai field.
In *Automated Reasoning and Mathematics: Essays in Memory of William W. McCune*, pages 240–257. Springer.

Van Rooij, I. (2008).
The tractable cognition thesis.
Cognitive science, 32(6):939–984.