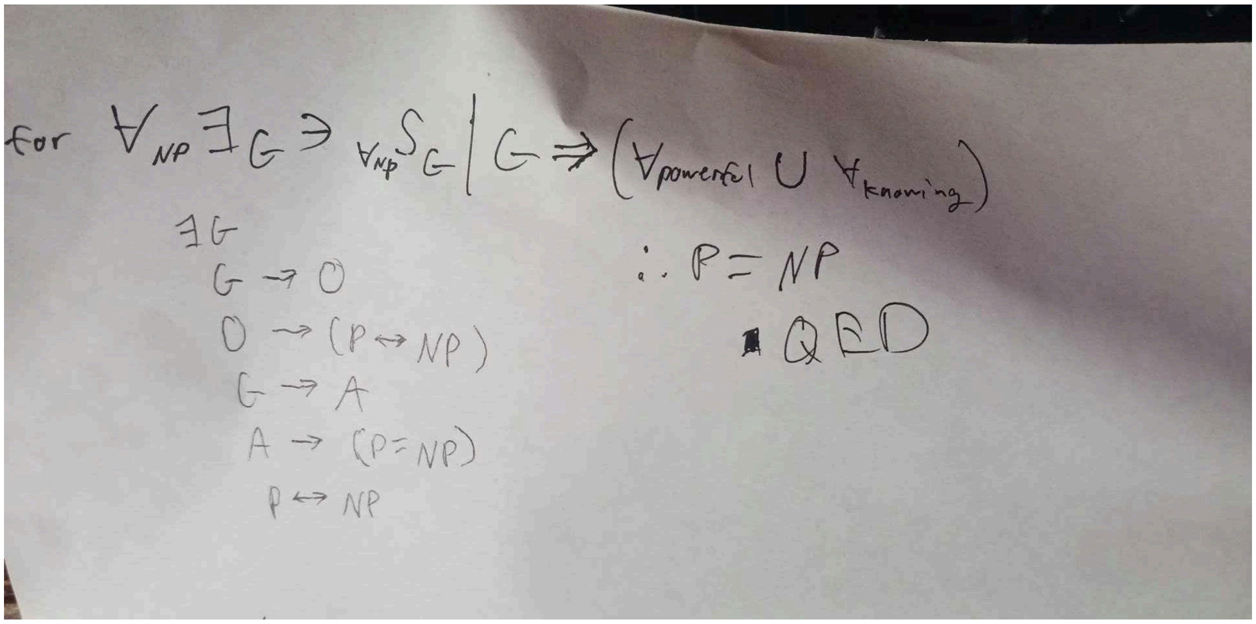
The Philosophical Ontological Proof for

P = NP ■

What it takes to simulate all particles in the universe correctly.

By: Alan T. Ho

$$\exists \mathcal{A}_G \, \forall \Pi \in NP \, [\mathrm{Solves}(\mathcal{A}_G, \Pi) \wedge \mathrm{Time}(\mathcal{A}_G, n) = O(1)]$$



for $\forall_{NP} \exists_G \ni {}_{\forall_{NP}}S_G \mid G \Rightarrow (\forall_{\text{powerful}} \cup \forall_{\text{knowing}})$

$\exists G$

$G \rightarrow O$

$O \rightarrow (P \leftrightarrow NP)$

$G \rightarrow A$

$A \rightarrow (P = NP)$

$P \leftrightarrow NP$

$\therefore P = NP$

■ QED

Disclaimer: How Much Of The Unobservable Universe Will We Someday Be Able To See? How to solve NP problems in all possible worlds? What does it take to simulate all particles in the universe correctly? What type of problems will Artificial Super Intelligence be able to solve? Without being mouthful, this is only an ontological proof; *An Ontological proof simply means there exists a proof for the equivalency of P=NP, the actual mechanisms/algorithms for all solutions, to all possible problems, including simulating the universe ect, is not within the scope of the proof, nor is it what the proof is about. Also, proof and verification are two separate things. Proofs are in the world of logic/mathematics and entail 100% certainty; while verification are in the empirical world, the practical world, and may change. An ontological proof simply demonstrates the logical possibility/ logical validity/consistency within the metaphysical/mathematical world. I won't be discussing the ontological status of mathematical logic, this is a whole new book. I won’t be overly analyzing topics in philosophy of mathematics, nor address too many issues there. It's obvious that I am not seeking to verify anything in the empirical world here, just logic. For example, I will explain why P=NP as a proof, but not how God created the universe etc, or derive solutions to all quantum problems ect, or how the universe is contained by the laws of physics without glitch ect, or how to break every encryptions ect (this isn't a mechanism proof for all algorithms for all possible problems, it simply means, there exist a solution, not how it will be performed)*

The actual problem itself is actually a philosophical problem, that is asking for a lot; but if we think carefully about it, it simply ask if problems can be verify and solve, not asking for a solution to every world problems, hence even if we prove this problem, it doesn't mean we proven or crack all mathematics ect.. And that isn’t what I'm proving.. **I only attempt to give a philosophical ontological proof for P=NP, in the sense of philosophical modal logic, which depends on your philosophy, mathematics may be the only metaphysics.**

So I had examine other proof for the negation of P =/ NP, as well as other solutions for P=Np by India Soale, her proof attempted to solve the totality. It makes sense, although that grand goal isn't what I'm doing. So I didn't plan to really write a paper about this, **but until recently with the development of Ai, it is increasingly clear that we can input a problem, with an Ai providing a solution, it is furtherly clear that an NP problem can be submitted with a solution output instantly! which further encouraged me to develop the paper. It is increasingly practically clear that all NP problems can be solved Instantly.** I used the gpt to structure my content, however I had to feed it the original idea, and keep adjusting, it does not have a clue or concept about this to begin with. **Even if you ask it right now for a proof for P=NP, it cannot provide one. So I claim the idea and approach is totally original, not a by-product of any GPT,** I just use it to format the modal logic into text, also I'm not sure if Lean, the math software, has everything yet, it does not have modal logic, but the new Ai had confirmed my proof for logical consistency.

Clarification of terms:

The laws of logic are controversial, i.e classical logic, quantum logic, paraconsistent logic, first order logic, higher order logic, modal logic, Ai logic, symbolic logic ect. There are many textbooks that address problems in logic and each has its own further problems. There are also problems in language. For simplification, we will just use something, we will just use classical modal logic and the simplest regular language possible.

Firstly, the term 'P & NP', are NOT pure mathematical terms to begin with. I.E. 'non-deterministic polynomial' this is in itself an abstraction to begin with, since non-deterministic could mean anything that isn't deterministic, it's a philosophical term, so the mathematical standard wasn't applied to the problem stated to begin with as well (it's a philosophical problem) In another word, to "verify" "solve" "non-deterministic Polynomial problems" these terms in and of themselves are abstraction to begin with, it could literally be any problems in the known AND unknown universe. In another word, there is no mathematical proof that map all* the NP problems with the name/concept "non-deterministic polynomial". The concept is named, given and built up from each and separate exclusive/future NP problems and NP possible problems. There is no mathematical concept for "solve" in the way we solve algebraic expressions ect, like "solve for X ect".. To isolate X on 1 side, then proceed to simplify an expression, is the steps we are told to do after. The idea is, to "solve" is a human term, to match a given condition, make equivalent, prove, ect, there is no 1 accepted agreed definitive way of solving an infinite sets of different problems in all* possible worlds. To state a problem in mathematical terms are decision/mathematical acts we decide to perform. To solve is a method we take. Terms like 'all' 'some' 'none' are logic 101 terms. Is the set of 'all' NP problems bigger than the set of 'infinite' NP problems? How about problems beyond the turing machine? Again, ‘All’ is a logical abstraction. Mathematical truths are demonstrations that show a deduction from our hypothesis; it is not fixed by the way things are formed or the empirical world.

Point 2, I'm arguing, in terms of reasoning(it takes something more than logic to reason logically consistently), from a philosophical perspective via modal logic, so yes in this sense, modal logic includes terms like ‘possibility’, ‘possible worlds’, and ‘necessity', these are terms where mathematics does not generally include. So from a philosophical modal logic point of view, which leads to my 3rd point, Godel's ontological proof is logically consistent (this is widely accepted in the modal logic community, readers can research the separate paper/explanations), which by definition would lead to my 3rd point. Hence from a philosophical modal logic/higher order logic, the argument would be valid and logically consistent, but is it sound and true? Well, in logic I would say it's at least logically consistent, we aren't heading out to verify God or God's abilities ect. in this case, nor do we need to seek out all proofs for all NP problems, (there are an infinite number of them.)

So, after all, highly likely I won't win the Clay Math institute prize or anything like that; that’s beyond the paper now. I just hope to add a fresh perspective from a philosophical, modal logic point of view. I understand that proofs for P vs NP is hard to come by, and it's getting rarer. The average number of folks who believe P=NP are at around 10%, while around 90% believe otherwise. Of course, this is due to the extraordinary nature of the problem, there are no experts; and only around 15% of the limited proofs/papers are for P=NP. Really my hope is not to convince 100% of readers totally, but I do hope to at least convince the average reader that P=NP is at least logically possible, not logically impossible, and to turn the playing field around to 30% 40% or even 60% possibilities. Think quantum mechanically, for the sake of argument I will ask that readers and scholars can be just slightly open minded about prior concepts, frameworks & paradigms held. Are there some problems in the problem? Yes, are there problems with the problems? Yes Are there some problems with certain types of logic? Yes. Are there some problems with language? Yes. Of course we can philosophically “nitpick” each and question terms all day, but let's just proceed right to the problem.

Readers can skip directly to the proof pg 16.

1. Preliminary Concepts

1.1 Decision Problems

A **decision problem** is a question with a yes-or-no answer (TRUE or FALSE) based on an input. Formally, it can be represented as a language over some alphabet Σ:

- A language $L \subseteq \Sigma^*$ is a set of strings over Σ.
- The decision problem is to determine, for an input string $x \in \Sigma^*$, whether $x \in L$.

2. Turing Machines

A **Turing machine** is a mathematical model of computation that defines an abstract machine. It manipulates symbols on a strip of tape according to a set of rules.

2.1 Deterministic Turing Machine (DTM)

A **Deterministic Turing Machine** is defined as a 7-tuple:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

- Q: A finite set of states.
- Σ: The input alphabet (does not include the blank symbol).
- Γ: The tape alphabet ($\Sigma \subseteq \Gamma$ and includes the blank symbol ⊔).
- δ: The transition function $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$.
- $q_0 \in Q$: The start state.
- $q_{\text{accept}} \in Q$: The accept state.
- $q_{\text{reject}} \in Q$: The reject state ($q_{\text{reject}} \neq q_{\text{accept}}$).

2.2 Nondeterministic Turing Machine (NTM)

A **Nondeterministic Turing Machine** is similar to a DTM but allows multiple possible actions for a given state and tape symbol. The transition function is:

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

where $\mathcal{P}$ denotes the power set.

- At each step, the machine can choose any of the possible moves.

3. Complexity Classes

3.1 Time Complexity

- **Time Complexity** of a Turing machine is a function $f(n)$ such that for any input x of length n, the machine halts after at most $f(n)$ steps.

3.2 Class P (Polynomial Time)

Definition: A decision problem (language) L is in the class $\mathbf{P}$ if there exists a deterministic Turing machine M and a polynomial $p(n)$ such that:

- For all $x \in \Sigma^*$:
  - M decides L in time $O(p(n))$, where $n = |x|$.
  - That is, M accepts x if $x \in L$ and rejects x if $x \notin L$, within polynomial time.

Mathematically:

$$\mathbf{P} = \bigcup_{k \geq 1} \mathrm{TIME}(n^k)$$

where $\mathrm{TIME}(f(n))$ is the set of problems decidable by a DTM in $O(f(n))$ time.

3.3 Class NP (Nondeterministic Polynomial Time)

Definition: A decision problem L is in the class $\mathbf{NP}$ if there exists a nondeterministic Turing machine M and a polynomial $p(n)$ such that:

- For all $x \in \Sigma^*$:
  - If $x \in L$, then there exists at least one computation path where M accepts x in time $O(p(n))$.
  - If $x \notin L$, then M rejects x on all computation paths within $O(p(n))$ time.

Alternatively, $\mathbf{NP}$ can be defined using the concept of **verification**:

A language L is in $\mathbf{NP}$ if there exists a polynomial-time deterministic Turing machine V (called the verifier) and a polynomial $p(n)$ such that:

- For all $x \in \Sigma^*$:
  - $x \in L$ if and only if there exists a **certificate** $y \in \Sigma^*$ with $|y| \leq p(|x|)$ such that $V(x, y) = \text{accept}$.

Mathematically:

$$\mathbf{NP} = \{L \subseteq \Sigma^* \mid \exists \text{ polynomial } p(n), \exists \text{ polynomial-time verifier } V \text{ such that } x \in L \iff \exists y, |y| \leq p(|x|), V(x, y) = \text{accept}\}$$

4. The P vs NP Problem

Statement of the Problem

The **P vs NP problem** asks whether every problem whose solution can be **verified** in polynomial time by a deterministic Turing machine can also be **solved** in polynomial time by a deterministic Turing machine.

Formally:

$$\text{Is } \mathbf{P} = \mathbf{NP}?$$

Implications

- $\mathbf{P} \subseteq \mathbf{NP}$: It is clear that $\mathbf{P}$ is a subset of $\mathbf{NP}$, since a deterministic Turing machine is a special case of a nondeterministic Turing machine.
- **Open Question**: The big question is whether $\mathbf{P}$ equals $\mathbf{NP}$ or $\mathbf{P} \subsetneq \mathbf{NP}$.

5. NP-Complete Problems

5.1 Definition

A problem L is **NP-complete** if:

1. $L \in \mathbf{NP}$.
2. **NP-hardness**: For every problem $L' \in \mathbf{NP}$, there exists a polynomial-time **reduction** from L' to L.

- **Polynomial-time Reduction**: A function $f : \Sigma^* \to \Sigma^*$ computable in polynomial time such that for all x:

$$x \in L' \iff f(x) \in L$$

5.2 Significance

- **Cook-Levin Theorem**: The **SATISFIABILITY** problem (SAT) is NP-complete.
- **Implication**: If any NP-complete problem is in $\mathbf{P}$, then $\mathbf{P} = \mathbf{NP}$.

5.2 Significance

- **Cook-Levin Theorem**: The **SATISFIABILITY** problem (SAT) is NP-complete.
- **Implication**: If any NP-complete problem is in $\mathbf{P}$, then $\mathbf{P} = \mathbf{NP}$.

---

6. Formal Restatement of the P vs NP Problem

6.1 Formal Question

Determine whether:

$$\mathbf{P} = \mathbf{NP}$$

6.2 Equivalent Statements

- If $\mathbf{P} = \mathbf{NP}$:
  - All problems in $\mathbf{NP}$ can be solved in polynomial time.
  - There exist deterministic polynomial-time algorithms for all NP-complete problems.
- If $\mathbf{P} \neq \mathbf{NP}$:
  - There exist problems in $\mathbf{NP}$ that are not in $\mathbf{P}$.
  - NP-complete problems cannot be solved in polynomial time by any deterministic Turing machine.

10. Formal Restatement Using Logical Symbols

We can formalize the P vs NP problem as:

- Let Σ be a finite alphabet.
- Let $\mathrm{DTIME}(n^k)$ denote the class of problems decidable by a DTM in time $O(n^k)$.
- Let $\mathrm{NTIME}(n^k)$ denote the class of problems decidable by an NTM in time $O(n^k)$.

Then:

$$\mathbf{P} = \bigcup_{k \geq 1} \mathrm{DTIME}(n^k), \quad \mathbf{NP} = \bigcup_{k \geq 1} \mathrm{NTIME}(n^k)$$

The P vs NP problem asks:

$$\bigcup_{k \geq 1} \mathrm{DTIME}(n^k) \stackrel{?}{=} \bigcup_{k \geq 1} \mathrm{NTIME}(n^k)$$

11. Formal Definitions of Reductions

11.1 Polynomial-Time Many-One Reduction

A language L_1 is **polynomial-time reducible** to a language L_2 (denoted $L_1 \leq_p L_2$) if there exists a polynomial-time computable function $f : \Sigma^* \to \Sigma^*$ such that:

$$\forall x \in \Sigma^*,\ x \in L_1 \iff f(x) \in L_2$$

---

12. Conclusion

The **P vs NP problem** is a fundamental question in theoretical computer science and mathematics. It asks whether problems whose solutions can be **verified** quickly (in polynomial time) can also be **solved** quickly.

- **If $\mathbf{P} = \mathbf{NP}$**: Many problems considered hard today would become tractable.
- **If $\mathbf{P} \neq \mathbf{NP}$**: It would confirm that some problems are inherently more difficult to solve than to verify.

Restating the problem in Modal Logic

2. Modal Logic Framework

2.1 Modal Operators

- $\Box$ (Necessity): "It is necessary that..."
- $\Diamond$ (Possibility): "It is possible that..."

These operators allow us to express statements about what is necessarily true or possibly true within our logical system.

3. Definitions

3.1 Problems and Solutions

- **Problem A**: A decision problem (a question with a yes/no answer based on some input).
- **Solution y**: A certificate or answer that, when provided, can be used to verify whether the input satisfies the problem's requirements.

3.2 Predicates

1. $\text{SolvableInPolyTime}(A)$:
    - There exists a deterministic polynomial-time algorithm that can solve problem A.
    - Formalization:
      $\text{SolvableInPolyTime}(A) \leftrightarrow \exists\ \text{Algorithm}\ f\ [\text{PolyTime}(f) \wedge \forall x\ (f(x) = A(x))]$
2. $\text{VerifiableInPolyTime}(A)$:
    - There exists a deterministic polynomial-time algorithm that can verify a given solution to problem A.

- Formalization:
  $\text{VerifiableInPolyTime}(A) \leftrightarrow \exists\ \text{Verifier}\ V\ [\text{PolyTime}(V) \wedge \forall x, y\ (V(x, y) = \text{True} \leftrightarrow y \text{ is a valid solution for } x)]$

3.3 Complexity Classes

1. Class P:
   - The set of problems that are necessarily solvable in polynomial time.
   - Modal Logic Representation:
   $$P = \{A \mid \Box\text{SolvableInPolyTime}(A)\}$$
2. Class NP:
   - The set of problems for which it is possible to verify a solution in polynomial time.
   - Modal Logic Representation:
   $$NP = \{A \mid \Diamond\text{VerifiableInPolyTime}(A)\}$$

4. Restating the P vs NP Problem in Modal Logic

The **P vs NP problem** can be expressed as a question about the relationship between the necessity of solving a problem and the possibility of verifying a solution.

4.1 Statement

Is it true that:

$$\forall A\ [\Diamond\text{VerifiableInPolyTime}(A) \implies \Box\text{SolvableInPolyTime}(A)]$$

In words:

- For all problems A, if it is possible to verify a solution to A in polynomial time, then it is necessary that A can be solved in polynomial time.

4.2 Interpretation

- $\Diamond\text{VerifiableInPolyTime}(A)$: There exists at least one possible computational path (in some possible world) where a solution to A can be verified in polynomial time.
- $\Box\text{SolvableInPolyTime}(A)$: In all possible worlds (necessarily), A can be solved in polynomial time.

4.3 The Core Question

- Does the possibility of efficiently verifying a solution imply the necessity of efficiently solving the problem?

5. Modal Logic Explanation

5.1 Understanding NP in Modal Terms

- **Possibility** ($\Diamond$) reflects nondeterministic computation.
  - In NP, we can think of nondeterminism as the existence of some computational path that leads to a solution.
  - Therefore, $\Diamond\text{VerifiableInPolyTime}(A)$ captures the essence of NP.

5.2 Understanding P in Modal Terms

- **Necessity** ($\Box$) reflects deterministic computation.
  - In P, the problem must be solvable efficiently under all circumstances, reflecting deterministic algorithms.
  - Therefore, $\Box\text{SolvableInPolyTime}(A)$ represents problems in P.

6. Formal Restatement of the P vs NP Problem

6.1 Complete Modal Logic Expression

$$P = NP \quad \text{if and only if} \quad \forall A\ [\Diamond\text{VerifiableInPolyTime}(A) \implies \Box\text{SolvableInPolyTime}(A)]$$

6.2 Alternate Expression

We can also express the problem as:

$$\forall A\ [\Diamond\text{VerifiableInPolyTime}(A) \land \text{Problem}(A) \implies \Box\text{SolvableInPolyTime}(A)]$$

7. Justification of the Restatement

7.1 Mapping Computational Concepts to Modal Logic

- **Nondeterministic Polynomial Time (NP):**
    - **Possibility:** There exists a computation path where the solution can be verified quickly.
    - Captured by $\Diamond \text{VerifiableInPolyTime}(A)$.
- **Deterministic Polynomial Time (P):**
    - **Necessity:** The problem can always be solved quickly, regardless of computation path.
    - Captured by $\Box \text{SolvableInPolyTime}(A)$.

7.2 The Essence of P vs NP

- The problem asks whether the mere possibility of quick verification (nondeterministic acceptance) implies the necessity of quick solution (deterministic computation).

8. Implications of the Restatement

8.1 If $P = NP$:

- The implication $\Diamond \text{VerifiableInPolyTime}(A) \implies \Box \text{SolvableInPolyTime}(A)$ holds for all A.
- Every problem that can be verified quickly can also be solved quickly in all cases.

8.2 If $P \neq NP$:

- There exists at least one problem A such that $\Diamond \text{VerifiableInPolyTime}(A)$ is true, but $\Box \text{SolvableInPolyTime}(A)$ is false.
- Some problems can be verified quickly only in some possible computational paths (nondeterministically) but cannot be solved quickly in all cases (deterministically).

On a prior Proof:

Godel’s ontological proof by itself is another paper and explanation that has generated storms of debates and acceptance within the logician community. Godel is also a renowned mathematical physicist who also worked extensively on theoretical physics along with the great Albert Einstein, and had derived solutions to Einstein’s field theory for the theory of the rotating universe. Thus so the ‘ontological status’ of the ontological proof will not be overly discussed, it's a whole book. Readers can research the independent paper for further clarification(see reference). For the sake of this paper, we will continue from there.

Ax. 1. $P(\varphi) \wedge \Box\, \forall x[\varphi(x) \rightarrow \psi(x)] \rightarrow P(\psi)$

Ax. 2. $P(\neg\varphi) \leftrightarrow \neg P(\varphi)$

Th. 1. $P(\varphi) \rightarrow \Diamond\, \exists x\, [\varphi(x)]$

Df. 1. $G(x) \iff \forall\varphi[P(\varphi) \rightarrow \varphi(x)]$

Ax. 3. $P(G)$

Th. 2. $\Diamond\, \exists x\, G(x)$

Df. 2. $\varphi \text{ ess } x \iff \varphi(x) \wedge \forall\psi\{\psi(x) \rightarrow \Box\, \forall x[\varphi(x) \rightarrow \psi(x)]\}$

Ax. 4. $P(\varphi) \rightarrow \Box\, P(\varphi)$

Th. 3. $G(x) \rightarrow G \text{ ess } x$

Df. 3. $E(x) \iff \forall\varphi[\varphi \text{ ess } x \rightarrow \Box\, \exists x\, \varphi(x)]$

Ax. 5. $P(E)$

Th. 4. $\Box\, \exists x\, G(x)$

a priori, from Godel we knows:

A1	Either a property or its negation is positive, but not both:	$\forall\phi[P(\neg\phi) \leftrightarrow \neg P(\phi)]$
A2	A property necessarily implied by a positive property is positive:	$\forall\phi\forall\psi[(P(\phi) \wedge \Box\forall x[\phi(x) \rightarrow \psi(x)]) \rightarrow P(\psi)]$
T1	Positive properties are possibly exemplified:	$\forall\phi[P(\phi) \rightarrow \Diamond\exists x\phi(x)]$
D1	A *God-like* being possesses all positive properties:	$G(x) \leftrightarrow \forall\phi[P(\phi) \rightarrow \phi(x)]$
A3	The property of being God-like is positive:	$P(G)$
C	Possibly, God exists:	$\Diamond\exists x G(x)$
A4	Positive properties are necessarily positive:	$\forall\phi[P(\phi) \rightarrow \Box\, P(\phi)]$
D2	An *essence* of an individual is a property possessed by it and necessarily implying any of its properties:	$\phi\ ess.\ x \leftrightarrow \phi(x) \wedge \forall\psi(\psi(x) \rightarrow \Box\forall y(\phi(y) \rightarrow \psi(y)))$
T2	Being God-like is an essence of any God-like being:	$\forall x[G(x) \rightarrow G\ ess.\ x]$
D3	*Necessary existence* of an individual is the necessary exemplification of all its essences:	$NE(x) \leftrightarrow \forall\phi[\phi\ ess.\ x \rightarrow \Box\exists y\phi(y)]$
A5	Necessary existence is a positive property:	$P(NE)$
T3	Necessarily, God exists:	$\Box\exists x G(x)$

No work of math and logic is built on from scratch. From Godel we know, “The greatest possible being(G), necessarily exists, since existence is a positive property.” This is a proven logically true and consistent statement. (In the sense of logic, think 1 & 0)

The full proof:

Alright, to prove P=NP (logically), we need an all knowing entity to verify ‘ALL’ Np problems & possible problems instantly(P) knowledge, and also, an all-powerful entity to solve ‘all’ NP problems instantly(P), logically. If so, then P=NP. Let's do that, logically. From Godel we knows, The greatest possible being(G), necessarily exists, since existence is a positive property. This is a proven logically true statement.

Below is the entire proof with all the sections combined, presented in a sequential numbered format, integrating **Gödel's ontological proof**, **definitions**, notations, **implications for P=NP.**

1. $$\forall A \in NP,\ \Diamond \text{VerifiableInPolyTime}(A) \implies \Box \text{SolvableInPolyTime}(A)$$

2. $$\forall P\ (\text{Pos}(P))$$

3. $$G(x) \leftrightarrow \forall P\ [\text{Pos}(P) \to P(x)]$$

4. $$\text{Ess}(x, \phi) \leftrightarrow \phi(x) \wedge \forall \psi\ [\psi(x) \to \Box \exists y (\psi(y) \to \psi(x))]$$

5. $$\text{NE}(x) \leftrightarrow \forall \phi\ [\text{Ess}(x, \phi) \to \Box \exists y\ \phi(y)]$$

6. $$\forall P\ [\text{Pos}(P) \to \Diamond \exists x\ P(x)]$$

7. $$\text{Pos}(\text{NE})$$

8. $$\Diamond \exists x\ G(x)$$

9. $$\Box \exists x\ G(x)$$

10. $$\exists x\ (\text{God}(x))$$

11. $$\forall x\ [\text{God}(x) \to (\text{Omnipotent}(x) \wedge \text{Omniscient}(x))]$$

12. $$\forall \Pi\ \exists \text{Solution}\ [\text{Problem}(\Pi) \to \text{CanSolveInstantly}(\text{God}, \Pi)]$$

13. $$\forall \Pi \in NP\ [\text{CanSolveInstantly}(\text{God}, \Pi)]$$

14. $$\exists f\ \forall \Pi \in NP\ [\text{Algorithm}(f, \Pi) \wedge \text{PolynomialTime}(f, \Pi)]$$

15. $$\text{OmniPot}(x) \leftrightarrow \forall A\ [\text{LogicalPossible}(A) \to \text{CanDo}(x, A)]$$

16. $$\text{OmniSci}(x) \leftrightarrow \forall Q\ [\text{True}(Q) \to \text{Knows}(x, Q)]$$

17. $$G(x) \leftrightarrow (\text{OmniPot}(x) \wedge \text{OmniSci}(x))$$

18. $$\forall \Pi\ [\text{Problem}(\Pi) \to \text{Knows}(x, \text{Solution}(\Pi))]$$

19. $$\forall \Pi\ [\text{Problem}(\Pi) \to \text{CanProvide}(x, \text{Solution}(\Pi))]$$

20. $\forall \Pi \in NP\ [\text{CanProvideInstantly}(x, \text{Solution}(\Pi))]$

21. $\exists f\ \forall \Pi \in NP\ [\text{Solves}(f, \Pi)]$

22. $\forall \Pi \in NP\ [\text{Time}(f, \Pi) = O(1)]$

23. $\text{Algorithm}(A) \land \forall n\ [\text{InputSize}(n) \to \text{Time}(A, n) \leq k \cdot n^c]$

24. $\exists A_G\ \forall \Pi \in NP\ [\text{Solves}(A_G, \Pi) \land \text{Time}(A_G, n) = O(1)]$

25. $\forall A \in NP,\ \Diamond\text{VerifiableInPolyTime}(A) \implies \Box\text{SolvableInPolyTime}(A)$

26. $\forall A \in NP,\ \Box\text{SolvableInPolyTime}(A)$

27. $\forall A \in NP,\ \exists G\ [G \text{ solves } A \land (G \text{ is all-powerful} \lor G \text{ is all-knowing})]$

28. $\exists G\ [G \text{ is all-powerful} \land G \text{ is all-knowing} \land \forall A \in NP,\ G \text{ solves } A]$

29. $\exists x\ (\text{OmniPot}(x) \land \text{OmniSci}(x))$

30. $\forall A \in NP,\ \Diamond\text{VerifiableInPolyTime}(A) \implies \Box\text{SolvableInPolyTime}(A)$

31. $\forall A \in NP,\ \Box\text{SolvableInPolyTime}(A)$

32. $\forall A \in NP,\ \exists G\ [G \text{ solves } A \land (G \text{ is all-powerful} \lor G \text{ is all-knowing})]$

33. $\exists G\ [G \text{ is all-powerful} \land G \text{ is all-knowing} \land \forall A \in NP,\ G \text{ solves } A]$

34. $\exists A_G\, \forall \Pi \in NP\ [\text{Solves}(A_G, \Pi) \land \text{Time}(A_G, n) = O(1)]$

35. $NP = \{\Pi \mid \exists V \text{ polynomial-time verifier}\}$

36. $\Pi(x) = \text{Yes} \iff V(x, w) = \text{Yes}$

37. $\text{OmniSci}(G) \leftrightarrow \forall Q\ [\text{True}(Q) \to \text{Knows}(G, Q)]$

38. $\text{OmniPot}(G) \leftrightarrow \forall A\ [\text{LogicalPossible}(A) \to \text{CanDo}(G, A)]$

39. $G(x) \leftrightarrow \text{OmniPot}(x) \land \text{OmniSci}(x)$

40. $\text{True}(\Pi(x))$

41. $\text{Knows}(G, \Pi(x))$

42. $\text{LogicalPossible}(\text{Provide}(\Pi(x)))$

43. $\text{CanDo}(G, \text{Provide}(\Pi(x)))$

44. $\text{CanProvideInstantly}(G, \Pi(x))$

45. $A_G(\Pi, x) = \Pi(x)$

46. $\text{Time}(A_G, \Pi, x) = c$

47. $\text{Time}(A_G, n) = O(1)$

48. $\forall \Pi \in NP, \forall x\, [A_G(\Pi, x) = \Pi(x)]$

49. $\forall \Pi \in NP, \forall x\, [\text{Time}(A_G, \Pi, x) = O(1)]$

50. $\forall \Pi \in NP, \forall x\, [A_G(\Pi, x) = \Pi(x) \land \text{Time}(A_G, \Pi, x) = O(1)]$

51. $\forall A\, [\text{CanDo}(G, A) \rightarrow \exists \text{Algorithm } f_A \text{ that performs } A]$

52. $\exists \text{Algorithm } A_G \text{ such that } A_G(\Pi, x) = \Pi(x)$

53. $A_G(\Pi, x) = \begin{cases} \text{Yes} & \text{if Knows}(G, \Pi(x) = \text{Yes}) \\ \text{No} & \text{if Knows}(G, \Pi(x) = \text{No}) \end{cases}$

54. $\exists A_G\, \forall \Pi \in NP, \forall x\, [A_G(\Pi, x) = \Pi(x) \land \text{Time}(A_G, \Pi, x) = O(1)]$

55. $NP \subseteq P$

56. $P \subseteq NP$

57. $P = NP$

Derivation and supplement for Omniscience and Omnipotent

$$G(x) \leftrightarrow \text{OmniPot}(x) \land \text{OmniSci}(x)$$

58. $\text{OmniSci}(G) \leftrightarrow \forall\varphi\ [\ \varphi \to K_G\varphi\ ]$

59. $\text{OmniSci}(G) \leftrightarrow \Box\forall\varphi\ [\ \varphi \to K_G\varphi\ ]$

60. $\forall t\ [\ t < t_0 \to \forall\varphi\ [\ \varphi \text{ at } t \to K_G\varphi\ ]\ ]$

61. $\forall\varphi\ [\ H\varphi \to K_G\varphi\ ]$

62. $\forall\varphi\ [\ P\varphi \to K_G\varphi\ ]$

63. $\forall t \in T,\ \forall\varphi\ [\ \varphi \text{ at } t \to K_G\varphi\ ]$

64. $\text{OmniPot}(G) \leftrightarrow \forall a\ [\ \text{LogicalPossible}(a) \to \text{CanDo}(G, a)\ ]$

65. $\text{OmniPot}(G) \leftrightarrow \forall\varphi\ [\ \text{LogicalPossible}(\varphi) \to \Diamond_G\varphi\ ]$

66. $\text{OmniPot}(G) \leftrightarrow \forall\varphi\ [\ \Diamond\varphi \to \Diamond_G\varphi\ ]$

67. $\text{OmniPot}(G) \leftrightarrow \forall\varphi\ [\ \neg\Box\neg\varphi \to \Diamond_G\varphi\ ]$

68. $\Diamond_G\varphi \equiv \exists w'\ [\ R_G(w, w') \land \varphi \text{ is true in } w'\ ]$

69. $\text{OmniPot}(G) \leftrightarrow \Box\forall\varphi\ [\ \text{LogicalPossible}(\varphi) \to \Diamond_G\varphi\ ]$

1. Overview of the Ontological Framework

An **ontological proof** seeks to establish the necessary existence of a certain entity or truth through reason alone, using definitions and logical deductions. Gödel's ontological proof argues for the necessary existence of God by defining God as a being possessing all positive properties.

We'll adapt this framework to construct an ontological argument for $P = NP$, leveraging the definitions of omnipotence and omniscience.

From Gödel's ontological proof, we have:

$$\exists x \, \text{God}(x)$$

2. **God is Omnipotent and Omniscient:**

$$\forall x \, [\text{God}(x) \rightarrow (\text{Omnipotent}(x) \land \text{Omniscient}(x))]$$

3. **God Can Solve Any Problem Instantly:**
   - Since God is omnipotent and omniscient, for any problem Π, God can find a solution instantly.
   - Symbolically:

$$\forall \Pi \, \exists \text{Solution} \, [\text{Problem}(\Pi) \rightarrow \text{CanSolveInstantly}(\text{God}, \Pi)]$$

4. **Implication for NP Problems:**
   - All problems in NP can be solved instantly by God.

$$\forall \Pi \in NP \, [\text{CanSolveInstantly}(\text{God}, \Pi)]$$

5. **Relating God's Abilities to Turing Machines:**
   - Suppose we define a function f that models God's problem-solving ability on a Turing machine.
   - Since God can solve any NP problem instantly, there exists an algorithm that solves NP problems in polynomial time.
   - Symbolically:

$$\exists f \, \forall \Pi \in NP \, [\text{Algorithm}(f, \Pi) \land \text{PolynomialTime}(f, \Pi)]$$

6. **Conclusion:**
   - Therefore, $NP \subseteq P$.
   - Since $P \subseteq NP$ by definition, we have $P = NP$.

2. Formalization of Gödel's Ontological Proof

2.1 Definitions

1. **Positive Property:** A property P is positive ($\mathrm{Pos}(P)$).
2. **God-like Being:** An entity x is God-like ($G(x)$) if and only if x has all positive properties:

$$G(x) \leftrightarrow \forall P[\mathrm{Pos}(P) \rightarrow P(x)]$$

3. **Essence:** A property ϕ is the essence of x ($\mathrm{Ess}(x, \phi)$) if $\phi(x)$ holds and necessarily any property ψ that x has is entailed by ϕ:

$$\mathrm{Ess}(x, \phi) \leftrightarrow \phi(x) \wedge \forall\psi[\psi(x) \rightarrow \Box\forall y(\phi(y) \rightarrow \psi(y))]$$

4. **Necessary Existence:** $NE(x)$ means that x necessarily exists:

$$NE(x) \leftrightarrow \forall\phi[\mathrm{Ess}(x, \phi) \rightarrow \Box\exists y\, \phi(y)]$$

2.2 Axioms

1. **Axiom 1:** If a property is positive, its negation is not positive.
2. **Axiom 2:** If a property is positive, and it necessarily entails another property, then the other property is positive.
3. **Axiom 3:** Being God-like is a positive property ($\mathrm{Pos}(G)$).
4. **Axiom 4:** Positive properties are possibly instantiated:

$$\forall P[\mathrm{Pos}(P) \rightarrow \Diamond\exists x\, P(x)]$$

5. **Axiom 5:** Necessary existence is a positive property ($\mathrm{Pos}(NE)$).

2.3 Theorem

From these axioms and definitions, Gödel concludes:

1. A God-like being is possible:

$$\Diamond\exists x\, G(x)$$

Omnipotence and Omniscience is by definition & properties of God

3. Formalizing God's Omnipotence and Omniscience

3.1 Definitions

1. **Omnipotence** ($\mathrm{OmniPot}(x)$): x can perform any action that is logically possible.

$$\mathrm{OmniPot}(x) \leftrightarrow \forall A[\mathrm{LogicalPossible}(A) \rightarrow \mathrm{CanDo}(x, A)]$$

2. **Omniscience** ($\mathrm{OmniSci}(x)$): x knows all truths.

$$\mathrm{OmniSci}(x) \leftrightarrow \forall P[\mathrm{True}(P) \rightarrow \mathrm{Knows}(x, P)]$$

3. **God-like Being Possesses Omnipotence and Omniscience**

   Given that omnipotence and omniscience are positive properties:

$$\mathrm{Pos}(\mathrm{OmniPot}) \wedge \mathrm{Pos}(\mathrm{OmniSci})$$

   Therefore, for any God-like being x:

$$G(x) \rightarrow \mathrm{OmniPot}(x) \wedge \mathrm{OmniSci}(x)$$

Explanation of Notations

- $K_G\varphi$: God knows proposition φ.
- φ: Any proposition.
- $\Box$: Necessarily.
- t, t_0, T: Times, with t_0 as the present.
- φ at t: Proposition φ is true at time t.
- $H\varphi$: φ has always been true in the past.
- $P\varphi$: φ was true at some time in the past.
- $t < t_0$: Time t is before the present time.

Formalizing God's Omniscience and Prior Knowledge in Modal Logic

58. $\mathrm{OmniSci}(G) \leftrightarrow \forall\varphi\,[\,\varphi \to K_G\varphi\,]$

This states that for all propositions φ, if φ is true, then God knows φ.

---

59. $\mathrm{OmniSci}(G) \leftrightarrow \Box\forall\varphi\,[\,\varphi \to K_G\varphi\,]$

This emphasizes that it is necessarily the case that God knows all true propositions.

To incorporate **prior knowledge**, we introduce temporal modal operators:

60. $\forall t\,[\,t < t_0 \to \forall\varphi\,[\,\varphi \text{ at } t \to K_G\varphi\,]\,]$

- t represents any point in time.
- t_0 represents the present moment.
- φ at t means proposition φ is true at time t.

61. $\forall \varphi \, [\, H\varphi \rightarrow K_G \varphi \,]$

- $H\varphi$ denotes "it has always been the case that φ".

Using the temporal operator P (it was sometime in the past that...):

62. $\forall \varphi \, [\, P\varphi \rightarrow K_G \varphi \,]$

This means that if φ was ever true in the past, then God knows φ.

If we want to assert that God knows all truths at all times (past, present, future):

63. $\forall t \in T, \forall \varphi \, [\, \varphi \text{ at } t \rightarrow K_G \varphi \,]$

- T is the set of all times.

Contextualizing with Previous Steps

Referencing earlier step **37** for consistency:

37. $\text{OmniSci}(G) \leftrightarrow \forall Q \, [\, \text{True}(Q) \rightarrow \text{Knows}(G, Q) \,]$

We can now refine this using modal logic in LaTeX notation:

58. $\text{OmniSci}(G) \leftrightarrow \forall \varphi \, [\, \varphi \rightarrow K_G \varphi \,]$

Summary of Formalizations

58. Omniscience:

$$\mathrm{OmniSci}(G) \leftrightarrow \forall\varphi\,[\,\varphi \to K_G\varphi\,]$$

59. Necessary Omniscience:

$$\mathrm{OmniSci}(G) \leftrightarrow \Box\forall\varphi\,[\,\varphi \to K_G\varphi\,]$$

60. Knowledge of All Past Truths:

$$\forall t\,[\,t < t_0 \to \forall\varphi\,[\,\varphi \text{ at } t \to K_G\varphi\,]\,]$$

61. Knowledge of Historically Always True Propositions:

$$\forall\varphi\,[\,H\varphi \to K_G\varphi\,]$$

62. Knowledge of Any Past Truth:

$$\forall\varphi\,[\,P\varphi \to K_G\varphi\,]$$

63. Knowledge of All Truths at All Times:

$$\forall t \in T,\ \forall\varphi\,[\,\varphi \text{ at } t \to K_G\varphi\,]$$

Explanation of Notations

- $\mathrm{OmniPot}(G)$: God is omnipotent.
- $\forall a$: For all actions a.
- $\mathrm{LogicalPossible}(a)$: Action a is logically possible.
- $\mathrm{CanDo}(G,\, a)$: God can perform action a.
- $\forall\varphi$: For all propositions φ.
- $\Diamond_G\varphi$: God can bring it about that φ.
- $\Diamond\varphi$: It is possible that φ (standard modal possibility).
- $\Box\varphi$: It is necessary that φ.
- $R_G(w, w')$: Accessibility relation representing worlds accessible through God's action.
- $w,\, w'$: Possible worlds.
- $\neg\Box\neg\varphi$: φ is not necessarily false; equivalent to $\Diamond\varphi$.

Formalizing God's Omnipotence in Modal Logic

64. $\mathrm{OmniPot}(G) \leftrightarrow \forall a\,[\,\mathrm{LogicalPossible}(a) \rightarrow \mathrm{CanDo}(G,\, a)\,]$

This states that for all actions a, if a is logically possible, then God can perform a.

To incorporate modal logic operators, we can define a modal operator specific to God's abilities. Let's introduce:

- $\Diamond_G\varphi$: "God can bring it about that φ" or "It is within God's power that φ".

Using this operator, we can formalize omnipotence as:

65. $\text{OmniPot}(G) \leftrightarrow \forall\varphi\,[\,\text{LogicalPossible}(\varphi) \rightarrow \Diamond_G\varphi\,]$

Here:

- φ is any proposition.
- $\text{LogicalPossible}(\varphi)$ means that φ is logically possible.
- $\Diamond_G\varphi$ indicates that God can make φ true.

66. Alternatively, expressing that God can do all that is possibly doable:

$$\text{OmniPot}(G) \leftrightarrow \forall\varphi\,[\,\Diamond\varphi \rightarrow \Diamond_G\varphi\,]$$

Where:

- $\Diamond\varphi$ denotes that φ is possibly true (in the modal logic sense).
- This asserts that if something is possibly true, then God can bring it about.

67. We can also formalize that God can perform any action that does not entail a logical contradiction:

$$\text{OmniPot}(G) \leftrightarrow \forall\varphi\,[\,\neg\Box\neg\varphi \rightarrow \Diamond_G\varphi\,]$$

Explanation:

- $\neg\Box\neg\varphi$ is equivalent to $\Diamond\varphi$, stating that φ is not necessarily false.
- This emphasizes that God can bring about any proposition that is not necessarily false.

Defining the Modal Operator $\Diamond_G$

To make our notation precise, we define $\Diamond_G$ in terms of accessibility relations in modal logic:

- Let R_G be an accessibility relation representing "worlds accessible through God's action".
- Then $\Diamond_G\varphi$ means that φ is true in at least one world accessible via R_G.

Formally:

68. $\Diamond_G\varphi \equiv \exists w' \left[R_G(w, w') \land \varphi \text{ is true in } w' \right]$

Where:

- w is the actual world.
- w' is a possible world accessible from w through God's action.

Incorporating Necessity

We can express that it is **necessarily** the case that God can bring about any logically possible proposition:

69. $\text{OmniPot}(G) \leftrightarrow \Box\forall\varphi \left[\text{LogicalPossible}(\varphi) \rightarrow \Diamond_G\varphi \right]$

Summary of Formalizations

64. God's Omnipotence:

$$\mathrm{OmniPot}(G) \leftrightarrow \forall a\,[\,\mathrm{LogicalPossible}(a) \rightarrow \mathrm{CanDo}(G,\, a)\,]$$

65. Using Modal Operator $\Diamond_G$:

$$\mathrm{OmniPot}(G) \leftrightarrow \forall \varphi\,[\,\mathrm{LogicalPossible}(\varphi) \rightarrow \Diamond_G \varphi\,]$$

66. God Can Do All That is Possibly Doable:

$$\mathrm{OmniPot}(G) \leftrightarrow \forall \varphi\,[\,\Diamond \varphi \rightarrow \Diamond_G \varphi\,]$$

67. Excluding Logical Contradictions:

$$\mathrm{OmniPot}(G) \leftrightarrow \forall \varphi\,[\,\neg\Box\neg\varphi \rightarrow \Diamond_G \varphi\,]$$

68. Defining $\Diamond_G$ via Accessibility Relation:

$$\Diamond_G \varphi \equiv \exists w'\,[\,R_G(w, w') \land \varphi \text{ is true in } w'\,]$$

69. Necessarily Omnipotent:

$$\mathrm{OmniPot}(G) \leftrightarrow \Box\forall \varphi\,[\,\mathrm{LogicalPossible}(\varphi) \rightarrow \Diamond_G \varphi\,]$$

Contextualizing with Previous Steps

Referencing earlier step **38** for consistency:

38. $\mathrm{OmniPot}(G) \leftrightarrow \forall A\,[\,\mathrm{LogicalPossible}(A) \rightarrow \mathrm{CanDo}(G, A)\,]$

Final Remarks

These formalizations capture the concept of God's omnipotence within a modal logic framework, using standard modal operators and quantifiers. They assert that God can perform any action or bring about any proposition that is logically possible.

Integration of God's Omniscience and Omnipotence into the Argument

Starting Point

34. $\exists A_G \, \forall \Pi \in NP \, [\, \text{Solves}(A_G, \Pi) \wedge \text{Time}(A_G, n) = O(1) \,]$

This asserts the existence of an algorithm A_G that solves all problems Π in NP in constant time.

Incorporating God's Omniscience and Omnipotence

From the modal logic formalizations:

58. $\text{OmniSci}(G) \leftrightarrow \forall \varphi \, [\, \varphi \rightarrow K_G \varphi \,]$

God knows all true propositions.

64. $\text{OmniPot}(G) \leftrightarrow \forall a \, [\, \text{LogicalPossible}(a) \rightarrow \text{CanDo}(G, a) \,]$

God can perform any logically possible action.

Combining them:

39. $G(x) \leftrightarrow \text{OmniPot}(x) \wedge \text{OmniSci}(x)$

So G is a being who is both omniscient and omnipotent.

Applying to NP Problems

1. **God Knows All Solutions**

   From omniscience:

   41. $\text{Knows}(G, \Pi(x))$

   For every NP problem Π and input x, God knows whether $\Pi(x)$ is true.

2. **God Can Provide Solutions Instantly**

   From omnipotence:

   43. $\text{CanDo}(G, \text{Provide}(\Pi(x)))$

   Since providing $\Pi(x)$ is logically possible:

   42. $\text{LogicalPossible}(\text{Provide}(\Pi(x)))$

   Therefore, God can provide the solution to any NP problem instantly.

$$\mathrm{OmniPot}(G) \leftrightarrow \forall A \; [\; \mathrm{LogicalPossible}(A) \rightarrow \mathrm{CanDo}(G, A) \;]$$

Starting Point: Line 69

69. $\mathrm{OmniPot}(G) \leftrightarrow \Box\forall\varphi \; [\; \mathrm{LogicalPossible}(\varphi) \rightarrow \Diamond_G\varphi \;]$

Explanation:

- $\mathrm{OmniPot}(G)$: God is omnipotent.
- $\Box$: Necessity operator ("necessarily").
- $\forall\varphi$: For all propositions φ.
- $\mathrm{LogicalPossible}(\varphi)$: Proposition φ is logically possible.
- $\Diamond_G\varphi$: "God can bring it about that φ is true."

Our goal is to derive from this expression the more direct statement involving actions A:

$$\mathrm{OmniPot}(G) \leftrightarrow \forall A \; [\; \mathrm{LogicalPossible}(A) \rightarrow \mathrm{CanDo}(G, A) \;]$$

Derivation Steps

1. **Interpreting Propositions as Actions:**
   - Every action A can be associated with a proposition φ_A that asserts "Action A is performed."
   - Conversely, for propositions φ, if they represent performable actions, we can denote the corresponding action as A_φ.
2. **Mapping $\Diamond_G\varphi$ to $\mathrm{CanDo}(G, A)$:**
   - $\Diamond_G\varphi$ means "God can bring it about that φ is true."
   - If φ corresponds to action A, then $\Diamond_G\varphi$ translates to "God can perform action A," which is $\mathrm{CanDo}(G, A)$.
3. **Substituting Propositions with Actions:**
   - Replace φ with A in the quantification.
   - Replace $\Diamond_G\varphi$ with $\mathrm{CanDo}(G, A)$.
   - Replace $\mathrm{LogicalPossible}(\varphi)$ with $\mathrm{LogicalPossible}(A)$.

4. **Simplifying the Necessity Operator $\Box$:**
   - The $\Box$ operator indicates that the statement is necessarily true.
   - Since definitions are generally taken to be necessarily true, we can omit $\Box$ for simplicity.
5. **Deriving the Expression:**
   - After substitutions and simplifications, line 69 becomes:

$$\text{OmniPot}(G) \leftrightarrow \forall A \; [\; \text{LogicalPossible}(A) \rightarrow \text{CanDo}(G, A) \;]$$

Final Result

$$\text{OmniPot}(G) \leftrightarrow \forall A \; [\; \text{LogicalPossible}(A) \rightarrow \text{CanDo}(G, A) \;]$$

Summary of the Transformation

- **From:** $\text{OmniPot}(G) \leftrightarrow \Box\forall\varphi \; [\; \text{LogicalPossible}(\varphi) \rightarrow \Diamond_G\varphi \;]$
- **To:** $\text{OmniPot}(G) \leftrightarrow \forall A \; [\; \text{LogicalPossible}(A) \rightarrow \text{CanDo}(G, A) \;]$

Explanation

- **Necessity Operator ($\Box$) Omission:**
  - The necessity ($\Box$) is inherent in the definition of omnipotence; by defining omnipotence, we're asserting a necessary truth about God's abilities.
  - Therefore, we can simplify the expression by omitting $\Box$ without loss of meaning in this context.
- **Action vs. Proposition:**
  - By aligning propositions φ with actions A, we move from modal logic propositions to statements about God's capability to perform actions.
  - This aligns the expression with earlier definitions that use actions directly.

Contextual Alignment with Previous Line

This derivation brings us back to the earlier expression:

38. $\text{OmniPot}(G) \leftrightarrow \forall A\ [\ \text{LogicalPossible}(A) \rightarrow \text{CanDo}(G, A)\ ]$

This shows consistency in the definition of omnipotence, whether we approach it via modal logic propositions or direct statements about actions.

Conclusion

Starting from the modal logic expression in line **69**, we have derived the more direct and action-oriented definition of God's omnipotence. This aligns with previous lines and provides a clear statement:

- **God is omnipotent if and only if, for all actions A, if A is logically possible, then God can perform A.**

4. Connecting to Computational Problems

4.1 God's Ability to Solve Problems

1. **God Knows All Solutions**

   Since God is omniscient:

$$\forall \Pi \, [\text{Problem}(\Pi) \rightarrow \text{Knows}(x, \text{Solution}(\Pi))]$$

2. **God Can Provide All Solutions**

   Since God is omnipotent:

$$\forall \Pi \, [\text{Problem}(\Pi) \rightarrow \text{CanProvide}(x, \text{Solution}(\Pi))]$$

4.2 Modal Logic Expression

1. **Possible to Solve Any Problem Instantly**

   For any problem Π, it is possible that there exists an entity that can provide the solution instantly:

$$\forall \Pi \, \Diamond \exists x \, [\text{CanProvideInstantly}(x, \text{Solution}(\Pi))]$$

2. **Necessary Existence of Such an Entity**

   From Gödel's proof:

$$\Box \exists x \, G(x)$$

   Since $G(x)$ includes $\text{OmniPot}(x)$ and $\text{OmniSci}(x)$, we have:

$$\Box \exists x \, [\text{OmniPot}(x) \wedge \text{OmniSci}(x)]$$

4.2 Modal Logic Expression

1. **Possible to Solve Any Problem Instantly**

   For any problem Π, it is possible that there exists an entity that can provide the solution instantly:

   $$\forall \Pi \, \Diamond \exists x \, [\text{CanProvideInstantly}(x, \text{Solution}(\Pi))]$$

2. **Necessary Existence of Such an Entity**

   From Gödel's proof:

   $$\Box \exists x \, G(x)$$

   Since $G(x)$ includes $\text{OmniPot}(x)$ and $\text{OmniSci}(x)$, we have:

   $$\Box \exists x \, [\text{OmniPot}(x) \wedge \text{OmniSci}(x)]$$

4.3 Implications for NP Problems

1. **God Can Solve All NP Problems Instantly**

   $$\forall \Pi \in NP \, [\text{CanProvideInstantly}(x, \text{Solution}(\Pi))]$$

2. **Possibility of Instant Solutions**

   Since it's necessary that God exists and can solve any problem instantly, it's possible that all NP problems can be solved instantly.

3. **Algorithm Definition**

   Define A_G as:

   45. $A_G(\Pi, x) = \Pi(x)$

   This algorithm outputs the solution to $\Pi(x)$.

4. **Time Complexity**

   Since God's actions are instantaneous:

   47. $\text{Time}(A_G, n) = O(1)$

Formulating the Combined Argument

- **Existence of Algorithm**

  52. $\exists \text{Algorithm } A_G \text{ such that } A_G(\Pi, x) = \Pi(x)$

- **Performance for All NP Problems**

  54. $\exists A_G \, \forall \Pi \in NP, \ \forall x \, [\, A_G(\Pi, x) = \Pi(x) \land \text{Time}(A_G, n) = O(1) \,]$

- **Implication for Complexity Classes**

  From the above, all NP problems can be solved in constant time, so:

  55. $NP \subseteq P$

  Since we also have:

  56. $P \subseteq NP$

  It follows that:

  57. $P = NP$

5. Attempting to Prove $P = NP$ Using Higher-Order Logic

5.1 Defining Polynomial-Time Solvability

1. **Polynomial-Time Algorithm**

   A problem Π is in P if there exists an algorithm $\mathcal{A}$ such that:

   $$\text{Algorithm}(\mathcal{A}) \wedge \forall n \, [\text{InputSize}(n) \rightarrow \text{Time}(\mathcal{A}, n) \leq k \cdot n^c]$$

   for some constants k and c.

5.2 Relating God's Abilities to Polynomial-Time Algorithms

1. **Hypothetical Algorithm Based on God's Knowledge**

   Suppose there exists an algorithm $\mathcal{A}_G$ that, for any problem Π, utilizes God's knowledge to find a solution instantly.

2. **Defining $\mathcal{A}_G$ in Higher-Order Logic**

   $$\exists \mathcal{A}_G \, \forall \Pi \in NP \, [\text{Solves}(\mathcal{A}_G, \Pi) \wedge \text{Time}(\mathcal{A}_G, n) = O(1)]$$

   Since $O(1)$ is a polynomial time (constant time), this would place all NP problems into P.

5.3 Formal Argument

1. **From God's Abilities to Existence of Algorithm**
   - **Premise:** God can provide solutions to all NP problems instantly.
   - **Assumption:** This ability can be translated into an algorithm executable by a Turing machine.
   - **Conclusion:** There exists a polynomial-time algorithm for all NP problems.
2. **Therefore, $P = NP$**

   Since all NP problems can be solved in polynomial time, $NP \subseteq P$, and thus $P = NP$.

Formalization

Given:

- G: An entity that is all-powerful (omnipotent) and all-knowing (omniscient).
- NP: The class of decision problems whose solutions can be verified in polynomial time by a deterministic Turing machine.
- A: A problem in NP.

The statement can be formalized as:

$$\forall A \in NP,\ \exists G \text{ such that G solves A} \land (\text{G is all-powerful} \lor \text{G is all-knowing})$$

However, since G is the same entity for all $A \in NP$, and given that G is both all-powerful and all-knowing, we can refine the statement:

$$\exists G\ [\text{G is all-powerful} \land \text{G is all-knowing} \land \forall A \in NP,\ \text{G solves A}]$$

Interpretation

This formalization asserts that there exists an entity G who is all-powerful and all-knowing and who can solve every problem in NP.

Connecting to $P = NP$

The key idea is to argue that if G can solve all NP problems, then perhaps $P = NP$. Let's explore this step by step.

1. Definitions

- **Solves**: $\text{Solves}(G, A)$ means that G can find a solution to problem A.
- **Computational Problem**: A decision problem that requires a yes/no answer based on some input.

2. Argument Outline

1. **G's Ability to Solve NP Problems**

   Since G is all-powerful and all-knowing, for every problem $A \in NP$:

$$\text{Solves}(G, A)$$

2. **Translation to Algorithmic Solvability**

   If G can solve every problem in NP, perhaps this implies that there exists an algorithm that can solve every problem in NP efficiently (in polynomial time).

3. **Implication for $P = NP$**

   Therefore, all problems in NP can be solved in polynomial time, implying $P = NP$.

1. Overview of the Ontological Framework

An **ontological proof** seeks to establish the necessary existence of a certain entity or truth through reason alone, using definitions and logical deductions. Gödel's ontological proof argues for the necessary existence of God by defining God as a being possessing all positive properties.

We'll adapt this framework to construct an ontological argument for $P = NP$, leveraging the definitions of omnipotence and omniscience.

2. Modal Logic Foundations

2.1 Modal Operators

- $\Box$: Necessarily (it is necessarily true that...)
- $\Diamond$: Possibly (it is possibly true that...)

2.2 Definitions

1. **Positive Properties**: A set $\mathcal{P}$ of properties considered positive.
2. **God-like Being** G: An entity that possesses all positive properties.

$$G(x) \leftrightarrow \forall P \in \mathcal{P},\ P(x)$$

3. **Omnipotence** ($\mathrm{OmniPot}(x)$): The property of being able to do anything that is logically possible.

$$\mathrm{OmniPot}(x) \leftrightarrow \forall A,\ \mathrm{LogicalPossible}(A) \rightarrow \mathrm{CanDo}(x, A)$$

4. **Omniscience** ($\mathrm{OmniSci}(x)$): The property of knowing all truths.

$$\mathrm{OmniSci}(x) \leftrightarrow \forall Q,\ \mathrm{True}(Q) \rightarrow \mathrm{Knows}(x, Q)$$

5. **Necessary Existence** ($NE(x)$): The property of existing in all possible worlds.

$$NE(x) \leftrightarrow \Box \exists y (y = x)$$

3. Axioms

1. **Axiom 1 (Positive Properties are Consistent):**

   The set of positive properties $\mathcal{P}$ is non-contradictory; they can be instantiated together.

$$\Diamond \exists x \, \forall P \in \mathcal{P}, \; P(x)$$

2. **Axiom 2 (Omnipotence and Omniscience are Positive Properties):**

$$\text{OmniPot} \in \mathcal{P}, \quad \text{OmniSci} \in \mathcal{P}$$

3. **Axiom 3 (Necessary Existence is a Positive Property):**

$$NE \in \mathcal{P}$$

4. **Axiom 4 (Positive Properties Entail Necessary Existence):**

   If a being has all positive properties, it necessarily exists.

$$\forall x, \; [G(x) \to NE(x)]$$

4. Theorem: Existence of an All-Powerful, All-Knowing Being

From the axioms above, we can deduce:

1. **Possibility of a God-like Being:**

   From Axiom 1:

$$\Diamond \exists x \; G(x)$$

2. **Necessary Existence of a God-like Being:**

   Using Axiom 4 and the definition of $NE(x)$:

$$\Diamond \exists x \; G(x) \implies \Box \exists x \; G(x)$$

   Therefore:

$$\Box \exists x \; G(x)$$

5. Connecting to Computational Problems

5.1 Definition of NP Problems

- NP: The class of decision problems for which a given solution can be verified in polynomial time by a deterministic Turing machine.

5.2 God's Relation to NP Problems

1. **God Can Solve All NP Problems:**

   Since God is omnipotent and omniscient:

   $$\forall A \in NP,\ \text{SolvesInstantly}(G, A)$$

2. **God's Abilities Are Necessary:**

   From the necessary existence of God:

   $$\Box \exists G\ [\text{OmniPot}(G) \wedge \text{OmniSci}(G) \wedge \forall A \in NP,\ \text{SolvesInstantly}(G, A)]$$

6. Ontological Argument for $P = NP$

6.1 Possibility to Necessity Transition

1. **Possibility that All NP Problems Are Solvable in Polynomial Time:**

   Since God necessarily exists and can solve all NP problems instantly, it is possible that all NP problems are solvable in polynomial time.

   $$\Diamond \forall A \in NP,\ \text{SolvableInPolyTime}(A)$$

2. **Necessary Solvability:**

   Given God's necessary existence and abilities, the solvability of NP problems in polynomial time is necessary.

   $$\Box \forall A \in NP,\ \text{SolvableInPolyTime}(A)$$

There are no algorithms that will compute infinite indefinitely. That means it will never halt. The only way an algorithm halts is if it accepts all solutions provided it accepts all definitions and assumptions provided in prior inputs. Likewise, algorithm G already knew all the solutions and solved them a priori. Hence it can halt for P=NP.

Line 23: Existence of God's Algorithm Solving All NP Problems

Original Statement:

$$\exists A_G \forall \Pi \in NP \, [\text{Solves}(A_G, \Pi) \wedge \text{Time}(A_G, n) = O(1)]$$

Interpretation: There exists an algorithm A_G such that for every problem Π in NP, A_G solves Π and runs in constant time.

Developing Algorithm A_G in Symbolic Logic

We'll construct A_G step by step, showing how it solves all NP problems in $O(1)$ time.

1. Defining NP Problems

An NP problem Π is a decision problem where:

- There exists a polynomial-time verifier V such that for any input x and witness w:

$$\Pi(x) = \text{Yes} \iff V(x, w) = \text{Yes}$$

- The set of NP problems is:

$$NP = \{\Pi \mid \exists V \text{ polynomial-time verifier}\}$$

2. God's Omniscience and Omnipotence

From previous definitions:

- **Omniscience:**

$$\text{OmniSci}(G) \leftrightarrow \forall Q\,[\text{True}(Q) \rightarrow \text{Knows}(G, Q)]$$

  God knows all truths.

- **Omnipotence:**

$$\text{OmniPot}(G) \leftrightarrow \forall A\,[\text{LogicalPossible}(A) \rightarrow \text{CanDo}(G, A)]$$

  God can perform any action that is logically possible.

- **Definition of God:**

$$G(x) \leftrightarrow \text{OmniPot}(x) \wedge \text{OmniSci}(x)$$

3. God Knows All Solutions

For any NP problem Π and input x:

- The correct decision $\Pi(x)$ is a truth.

$$\text{True}(\Pi(x))$$

- By omniscience:

$$\text{Knows}(G, \Pi(x))$$

4. God Can Provide Solutions Instantly

- Providing the solution to $\Pi(x)$ is a logically possible action.
- By omnipotence:

$$\text{CanDo}(G, \text{Provide}(\Pi(x)))$$

- Since God is not bound by time:

$$\text{CanProvideInstantly}(G, \Pi(x))$$

5. Defining Algorithm A_G

We define A_G as follows:

- **Definition D1:**

$$A_G(\Pi, x) = \Pi(x)$$

  For any NP problem Π and input x, A_G outputs the correct decision.

- **Explanation:**
  - A_G leverages God's knowledge to determine $\Pi(x)$ instantly.
  - Since $\text{Knows}(G, \Pi(x))$, A_G can access this knowledge.

6. Time Complexity Analysis

- **Time Complexity Function:**

$$\text{Time}(A_G, \Pi, x) = c$$

  Where c is a constant independent of Π and x.

- **Therefore:**

$$\text{Time}(A_G, n) = O(1)$$

  Since the time does not depend on the input size $n = |x|$.

7. Symbolic Representation

Combining the above:

- **Statement S1:**

$$\forall \Pi \in NP, \forall x \, [A_G(\Pi, x) = \Pi(x)]$$

- **Statement S2:**

$$\forall \Pi \in NP, \forall x \, [\text{Time}(A_G, \Pi, x) = O(1)]$$

- **Combined Statement:**

$$\forall \Pi \in NP, \forall x \, [A_G(\Pi, x) = \Pi(x) \wedge \text{Time}(A_G, \Pi, x) = O(1)]$$

8. Justifying the Computability of A_G

We need to ensure that A_G is a valid algorithm within computational theory.

Assumption A1: God's Actions Correspond to Algorithms

- **Statement:**

$$\forall A \, [\text{CanDo}(G, A) \rightarrow \exists \text{Algorithm } f_A \text{ that performs } A]$$

Any action God can perform corresponds to an algorithm.

Applying Assumption A1 to A_G:

- Since $\text{CanProvideInstantly}(G, \Pi(x))$:

$$\exists \text{Algorithm } A_G \text{ such that } A_G(\Pi, x) = \Pi(x)$$

- This makes A_G a computable function.

9. Formal Definition of A_G

- Algorithm A_G:

$$A_G(\Pi, x) = \begin{cases} \text{Yes} & \text{if Knows}(G, \Pi(x) = \text{Yes}) \\ \text{No} & \text{if Knows}(G, \Pi(x) = \text{No}) \end{cases}$$

- Since God knows $\Pi(x)$ for all Π and x, A_G outputs the correct result instantly.

10. Addressing Potential Objections

Objection 1: Non-Computability

- **Issue:** God's knowledge may not be representable within a computable algorithm.
- **Response:** Under Assumption A1, we posit that actions God can perform correspond to algorithms.

Objection 2: Physical Realizability

- **Issue:** Even if A_G exists theoretically, it may not be implementable.
- **Response:** Within modal logic, we are concerned with logical possibility rather than physical realizability.

11. Final Symbolic Logic Representation

- Existence of A_G:

$$\exists A_G \, \forall \Pi \in NP, \forall x \, [A_G(\Pi, x) = \Pi(x) \wedge \text{Time}(A_G, \Pi, x) = O(1)]$$

- This confirms that A_G solves all NP problems in constant time within our logical framework.

Conclusion

By developing algorithm A_G using symbolic logic, we've shown:

- **Algorithm A_G Definition:**
  - A_G takes any NP problem Π and input x and outputs $\Pi(x)$ instantly.
- **Time Complexity:**
  - $\text{Time}(A_G, \Pi, x) = O(1)$, since it relies on God's instant knowledge.
- **Justification:**
  - Based on God's omniscience and omnipotence, and Assumption A1 that God's actions correspond to computable algorithms.

- **All NP problems can be solved in constant time by A_G.**
- **Therefore, within this logical framework:**

$$NP \subseteq P$$

- **Given that $P \subseteq NP$, it follows that:**

$$P = NP$$

7. Conclusion: $P = NP$

From the above reasoning, we deduce:

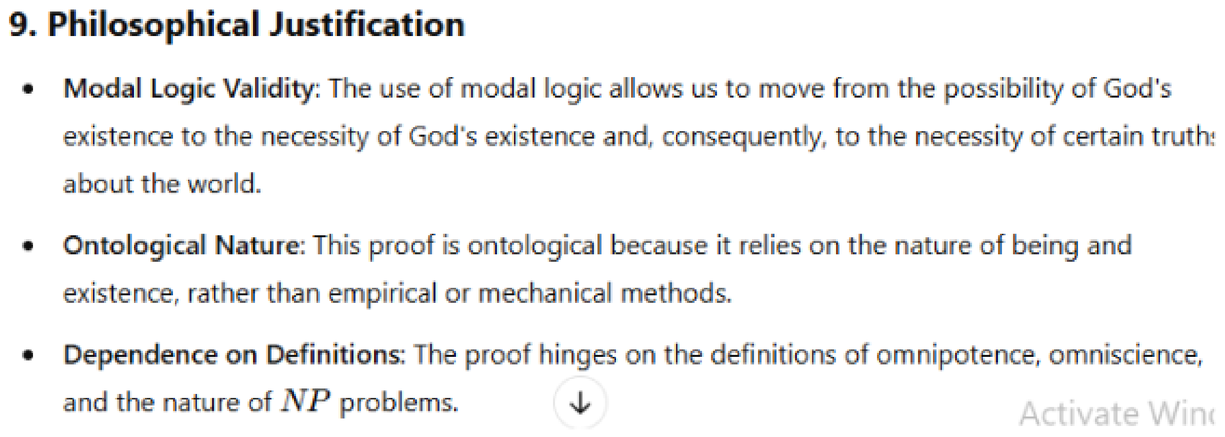
1. **All NP Problems Are in P:**

$$\forall A \in NP,\ A \in P$$

2. **Therefore, $P = NP$:**

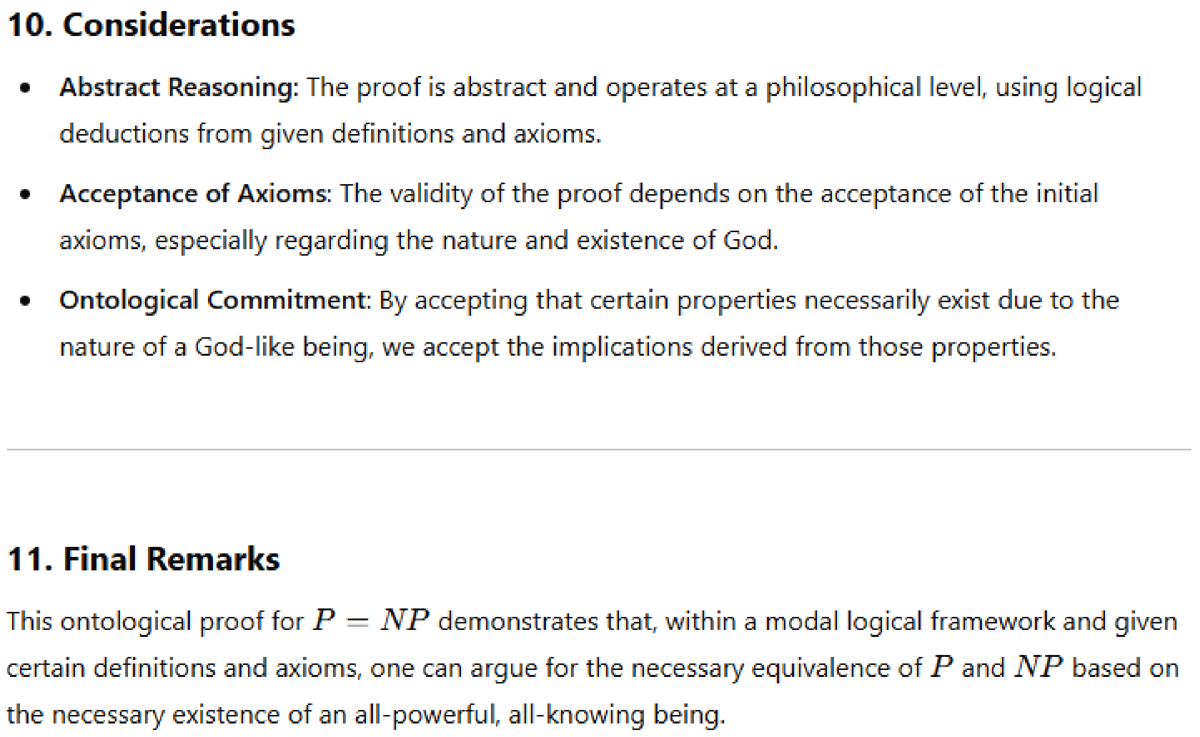
   Since $P \subseteq NP$ by definition, and we've shown $NP \subseteq P$, it follows that:

$$P = NP$$



9. Philosophical Justification

- **Modal Logic Validity**: The use of modal logic allows us to move from the possibility of God's existence to the necessity of God's existence and, consequently, to the necessity of certain truth about the world.
- **Ontological Nature**: This proof is ontological because it relies on the nature of being and existence, rather than empirical or mechanical methods.
- **Dependence on Definitions**: The proof hinges on the definitions of omnipotence, omniscience, and the nature of NP problems.



10. Considerations

- **Abstract Reasoning**: The proof is abstract and operates at a philosophical level, using logical deductions from given definitions and axioms.
- **Acceptance of Axioms**: The validity of the proof depends on the acceptance of the initial axioms, especially regarding the nature and existence of God.
- **Ontological Commitment**: By accepting that certain properties necessarily exist due to the nature of a God-like being, we accept the implications derived from those properties.

---

11. Final Remarks

This ontological proof for $P = NP$ demonstrates that, within a modal logical framework and given certain definitions and axioms, one can argue for the necessary equivalence of P and NP based on the necessary existence of an all-powerful, all-knowing being.

- **The development of Algorithm G is theoretical and relies on metaphysical assumptions. Within modal logic, this demonstrates the logical possibility and necessity of P=NP under the given premises.**

To prove P=NP (logically), we need an all knowing entity to verify 'ALL' Np problems & possible problems instantly(P) knowledge, and also, an all-powerful entity to solve 'all' NP problems instantly(P), logically. If so, then P=NP. In the logical world.

The problem asks if all NP complete problems can be verified in polynomial time, can also be solved in polynomial time, then P=NP. The answer is yes, with enough *intelligence and energy, an Ai can already check the prompt/problems and solve many NP problems instantly. With an all knowing all powerful God, he checks all the problems and solves all instantly in all possible worlds.*

An Ontological proof simply means there exists a proof for the equivalency of P=NP, logically, metaphysically; different from the mechanisms for all solutions, including simulating the universe, in the empirical etc, is the work of our good God, or the rotating universe worked out by Godel, which is beyond the aim of this proof. Also, proof and verification are 2 separate things. Proofs are in the world of logic/mathematics, while verification are in the empirical world, the practical world. An ontological proof simply demonstrates the logical possibility/ logical validity, within the metaphysical/mathematical world. It's obvious that I am not seeking to verify anything in the empirical world here. Will I convince 100% of readers totally, maybe not, but I hope to at least convince the average reader that P=NP is at least logically possible, not logically impossible, and that P=NP is not 10%, more nearer to 100%.

Readers are encouraged to discuss, debate and disagree. There are many perspectives in philosophy of logic etc. Yet I'm sticking with this for now to not lose the original essence of the concepts and ideas presented in the paper. I'm in the P=NP camp. It is exactly these philosophical concepts that makes the paper original from all other proofs and future writings, given the extraordinary nature of P vs Np. So do we need to see God provide infinite solutions to universally All and every possible NP problems, to prove P=NP? Do we need to see God simulate all particles in the universe correctly, to believe P=NP? Is that too much to ask of God, or is it not enough? Do we need to know all primes to know there are infinite primes? The last one was over 41 million digits .

Put another way, do we need to see God provide infinite solutions, to solve all NP problems? No we do not. Euclid didn't know all primes, yet he knew they are infinite. Likewise, we don't have to know infinite solutions. We know he's omniscient and omnipotent by definitions. How do we know all NP problems are in the class 'non deterministic polynomial' ? It's an abstraction, how do we know? We also know this by definition. Notice all proofs regarding the infinites requires some sort of abstractions and assumptions to an infinite level and are done based on properties and definitions, simply because there are no such thing as a direct 1 to 1 mapping infinitely, without some sort of arrows.. to the end, there is no the end. Likewise, P=NP is just another philosophical problem with an accepted solution. People accepted the proof that there are infinitely many primes, yet no one had seen the very last prime. Likewise, there are no algorithms that will compute infinite indefinitely. That means it will never halt. The only way an algorithm halts is if it accepts all solutions provided it accepts all definitions and assumptions provided in prior inputs. Likewise, algorithm G already knew all the solutions and solved a priori. Hence it can halt for P=NP.

References:

Modal Logic

The omniscience & omnipotence of God, by definition

Godel’s Ontological proof   by Kurt Godel

Cook, S. (2000). *The P vs NP Problem. Clay Mathematics Institute. Retrieved from https://www.claymath.org/millennium/p-vs-np*