The Complexity of Theorem-Proving Procedures

Stephen A. Cook's 1971 Paper

Presentation by Vasileios Gkolosi

Graduate Program in Algorithms, Logic and Mathematics
"ALMA"

Summary

The general aim of Cook's 1971 paper is to:

- Illustrate formally the computational difficulty (relating to other, apparently hard problems) of deciding the set of tautologies in propositional calculus,
- establish a notion of computational difficulty relating to running time, and its degrees of difficulty.

Some backround (Deterministic Turing Machine)

The simplest capable computation paradigm with a real world analogue is one that can primitively do the following:

- Be able to make decisions/distinguish between configurations,
- Be able to access memory with no limitation to its location

To this end, the following definition is given:

Some backround (Deterministic Turing Machine)

A one-tape Turing machine is a tuple (Γ, Q, δ) containing the following:

1. The set Γ contains a finite set of characters, which may be written on and read from the tape
2. The set Q contains a finite set of states, necessarily including the following three states: $q_{start}, q_{yes}, q_{no}$.
3. The transition function $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$
4. The tape on which all calculations are made (with a leftmost end, extending to the right infinitely)

Before its activation, a Turing Machine receives an input, which is given in the tape in the form of a string of characters from Γ.

Some backround (Deterministic Turing Machine)

Conventionally, in this context, a Turing Machine is used by defining it in such a way as to accept a special set of input strings. It is proven that for a given set of strings, we may not be able to design a Turing Machine that can decide if the given string is in the set of interest. We will focus on problems that are decidable and how quickly we can decide them.

Tautologies

Formally in mathematics, theorems are statements that are always true, regardless of the different meanings that its constituents may receive. Such an example is the following in propositional calculus:

$$(p \vee (\neg p))$$

Regardless of what the meaning of p may be, we can show semantically that the above is always true.

Tautologies

Formally in mathematics, theorems are statements that are always true, regardless of the different meanings that its constituents may receive. Such an example is the following in propositional calculus:

$$(p \vee (\neg p))$$

Regardless of what the meaning of p may be, we can show semantically that the above is always true.
However the semantic approach quickly becomes cumbersome as the number of propositional symbols increases (we have to check over all configurations of the truth values which are 2^n).

Tautologies

The above approach is seemingly naive. Is there a way we can leverage the structure of the propositional formula whose truth we are trying to evaluate to do so quicker?

Tautologies

The above approach is seemingly naive. Is there a way we can leverage the structure of the propositional formula whose truth we are trying to evaluate to do so quicker?

By showing that we can consider other problems -that are obviously hard and unstructured- to be instances of the above problem, we will establish its difficulty.

Tautologies

The complexity of theorem proving is of central importance in mathematics. In other words, we want to estimate the complexity of deciding the following problem:

Tautologies

The complexity of theorem proving is of central importance in mathematics. In other words, we want to estimate the complexity of deciding the following problem:

Definition

$$\{TAUT\} = \{w \in \{0,1\}^* \mid w \text{ is an encoding of a prop. formula and is a tautology}\}$$

Reductions (Many to One)

Consider the problems:

$$HP := \{< M, w > \mid M \text{ is a TM: } M(w) \downarrow\}$$

$$2HP = \{< M_1, M_2, w > \mid M_1, M_2 \text{ are TMs: } M_1(w) \downarrow \ \wedge M_2(w) \downarrow\}$$

Reductions (Many to One)

Consider the problems:

$$HP := \{< M, w > \mid M \text{ is a TM: } M(w) \downarrow\}$$

$$2HP = \{< M_1, M_2, w > \mid M_1, M_2 \text{ are TMs: } M_1(w) \downarrow \wedge M_2(w) \downarrow\}$$

It is apparent that $2HP$ holds information relevant to deciding HP.

Reductions (Many to One)

Consider the problems:

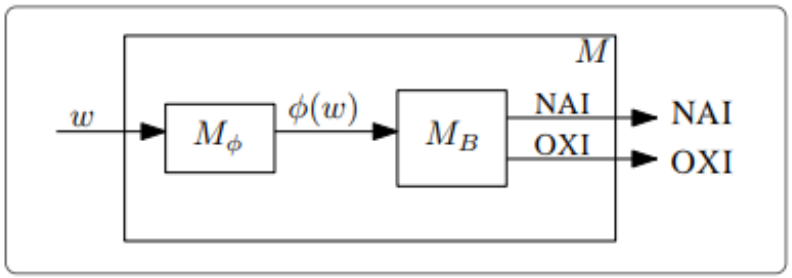
$$HP := \{< M, w > \mid M \text{ is a TM: } M(w) \downarrow\}$$

$$2HP = \{< M_1, M_2, w > \mid M_1, M_2 \text{ are TMs: } M_1(w) \downarrow \wedge M_2(w) \downarrow\}$$

It is apparent that $2HP$ holds information relevant to deciding HP. If ϕ is the computable function such that

$$\phi(< M', w >) =< M', M', w >,$$

and M_B a TM that decides $2HP$. Then the following TM M decides HP.



Reductions and Motivation

In this sense, $2HP$ is a harder problem to solve.

Reductions and Motivation

In this sense, $2HP$ is a harder problem to solve.
We will now turn our attention to problems such that M_ϕ's running time is bounded polynomially.
It follows that if the problem we are reducing to (in the above example $2HP$) is also polynomially computable, then so is the problem reduced to it (similarly, HP).

Hard Problems and Non-Determinism

Defining a Turing Machine that decides a given problem may be overly complicated. We allow the following modification to the definition of the Turing Machine as to address this issue:

Hard Problems and Non-Determinism

Defining a Turing Machine that decides a given problem may be overly complicated. We allow the following modification to the definition of the Turing Machine as to address this issue:

Definition

A nondeterministic Turing Machine is defined like a deterministic one, but with a transition relation instead of a transition function.

Hard Problems and Non-Determinism

Defining a Turing Machine that decides a given problem may be overly complicated. We allow the following modification to the definition of the Turing Machine as to address this issue:

Definition

A nondeterministic Turing Machine is defined like a deterministic one, but with a transition relation instead of a transition function.

We consider the NTM to accept if a branch of computation ends at the q_{yes} state. The implication is that in this case, we may exponentially decrease the work necessary to accept.

Hard Problems and Non-Determinism

Consider the problem of deciding if a graph G is isomorphic to a subgraph of G'.
It suffices to choose non-deterministically a 1-1 function from the vertices of G to the vertices of G'. We can easily verify if this is a graph isomorphism in polynomial time.

Hard Problems and Non-Determinism

Consider the problem of deciding if a graph G is isomorphic to a subgraph of G'.
It suffices to choose non-deterministically a 1-1 function from the vertices of G to the vertices of G'. We can easily verify if this is a graph isomorphism in polynomial time.
This problem is considered to be hard to solve deterministically, as there is little known about the structure of the graph globally from a local context, but non deterministically it can be solved in polynomial time.

Cook's Theorem

Definition

The set of sets recognizable by a NTM bounded by polynomial time is denoted as *NP*.

Theorem

Every NP problem is polynomially reducible to $\{TAUT\}$.

Cook's Theorem

We assume that $\Gamma = \{\sigma_1, \ldots, \sigma_\ell\}$.

Sketch of the Proof: Fix a language L and assume a NTM that decides it, M with states $Q = \{q_1, \ldots, q_r\}$. By our assumption, for an input w the time is bounded by $T = Q_t(|w|)$ where Q_t is a polynomial function. We construct a proposition in CNF form such that it encapsulates what it means for $M(w) \downarrow_{q_{yes}}$.

Cook's Theorem

We assume that $\Gamma = \{\sigma_1, \ldots, \sigma_\ell\}$.

Sketch of the Proof: Fix a language L and assume a NTM that decides it, M with states $Q = \{q_1, \ldots, q_r\}$. By our assumption, for an input w the time is bounded by $T = Q_t(|w|)$ where Q_t is a polynomial function. We construct a proposition in CNF form such that it encapsulates what it means for $M(w) \downarrow_{q_{yes}}$. Set the proposition symbols:

1. $P^i_{s,t}$, $1 \leq i \leq \ell$, $1 \leq s, t \leq T$ true iff tape square number s at step t contains the symbol σ_i.

Cook's Theorem

We assume that $\Gamma = \{\sigma_1, \ldots, \sigma_\ell\}$.

Sketch of the Proof: Fix a language L and assume a NTM that decides it, M with states $Q = \{q_1, \ldots, q_r\}$. By our assumption, for an input w the time is bounded by $T = Q_t(|w|)$ where Q_t is a polynomial function. We construct a proposition in CNF form such that it encapsulates what it means for $M(w) \downarrow_{q_{yes}}$. Set the proposition symbols:

1. $P^i_{s,t}$, $1 \leq i \leq \ell$, $1 \leq s, t \leq T$ true iff tape square number s at step t contains the symbol σ_i.
2. $Q^i_t 1 \leq i \leq r, 1 \leq t \leq T$ true iff at step t the machine is in state q_i.

Cook's Theorem

We assume that $\Gamma = \{\sigma_1, \ldots, \sigma_\ell\}$.

Sketch of the Proof: Fix a language L and assume a NTM that decides it, M with states $Q = \{q_1, \ldots, q_r\}$. By our assumption, for an input w the time is bounded by $T = Q_t(|w|)$ where Q_t is a polynomial function. We construct a proposition in CNF form such that it encapsulates what it means for $M(w) \downarrow_{q_{yes}}$. Set the proposition symbols:

1. $P^i_{s,t}$, $1 \leq i \leq \ell$, $1 \leq s, t \leq T$ true iff tape square number s at step t contains the symbol σ_i.
2. $Q^i_t 1 \leq i \leq r, 1 \leq t \leq T$ true iff at step t the machine is in state q_i.
3. $S_{s,t} 1 \leq s, t \leq T$ true iff at time t square number s is scanned by the tape head.

Cook's Theorem

We assume that $\Gamma = \{\sigma_1, \ldots, \sigma_\ell\}$.

Sketch of the Proof: Fix a language L and assume a NTM that decides it, M with states $Q = \{q_1, \ldots, q_r\}$. By our assumption, for an input w the time is bounded by $T = Q_t(|w|)$ where Q_t is a polynomial function. We construct a proposition in CNF form such that it encapsulates what it means for $M(w) \downarrow_{q_{yes}}$. Set the proposition symbols:

1. $P^i_{s,t}$, $1 \leq i \leq \ell$, $1 \leq s, t \leq T$ true iff tape square number s at step t contains the symbol σ_i.
2. $Q^i_t 1 \leq i \leq r, 1 \leq t \leq T$ true iff at step t the machine is in state q_i.
3. $S_{s,t} 1 \leq s, t \leq T$ true iff at time t square number s is scanned by the tape head.

Cook's Theorem

We set

$$B = \bigwedge_{t=1}^{T} ((\bigvee_{k=1}^{T} S_{k,t}) \wedge [\bigwedge_{1 \leq i \leq j \leq T} (\neg S_{i,t} \vee \neg S_{j,t})])$$

which asserts that for all steps t during the running time, one square is scanned, and one only.

Cook's Theorem

We set

$$B = \bigwedge_{t=1}^{T} ((\bigvee_{k=1}^{T} S_{k,t}) \wedge [\bigwedge_{1 \leq i \leq j \leq T} (\neg S_{i,t} \vee \neg S_{j,t})])$$

which asserts that for all steps t during the running time, one square is scanned, and one only.
Similarly, we set

$$C = \bigwedge_{1 \leq s,t \leq T} C_{s,t}$$

where $C_{s,k}$ asserts that at square s and time t there is only one symbol (expressed through the $P^i_{s,t}$ symbols)

Cook's Theorem

We also set

- $E = Q_1^0 \wedge S_{1,1} \wedge P_{1,1}^{i_1} \wedge \ldots, P_{n,1}^{i_n} \wedge P_{n+1,1}^1 \wedge \ldots \wedge P_{T,1}^1$ which asserts that the initial conditions are met ($w = \sigma_{i_1} \ldots \sigma_{i_n}$, q_0 the initial state q_{start} and σ_1 is the blank symbol)

Cook's Theorem

We also set

- $E = Q_1^0 \wedge S_{1,1} \wedge P_{1,1}^{i_1} \wedge \ldots, P_{n,1}^{i_n} \wedge P_{n+1,1}^1 \wedge \ldots \wedge P_{T,1}^1$ which asserts that the initial conditions are met ($w = \sigma_{i_1} \ldots \sigma_{i_n}$, q_0 the initial state q_{start} and σ_1 is the blank symbol)
- D which asserts that for each t there is one and only one state,

Cook's Theorem

We also set

- $E = Q_1^0 \wedge S_{1,1} \wedge P_{1,1}^{i_1} \wedge \ldots, P_{n,1}^{i_n} \wedge P_{n+1,1}^1 \wedge \ldots \wedge P_{T,1}^1$ which asserts that the initial conditions are met ($w = \sigma_{i_1} \ldots \sigma_{i_n}$, q_0 the initial state q_{start} and σ_1 is the blank symbol)
- D which asserts that for each t there is one and only one state, and I which asserts that the machine reaches the accepting state at some time.

Cook's Theorem

We also set

- $E = Q_1^0 \wedge S_{1,1} \wedge P_{1,1}^{i_1} \wedge \ldots, P_{n,1}^{i_n} \wedge P_{n+1,1}^1 \wedge \ldots \wedge P_{T,1}^1$ which asserts that the initial conditions are met ($w = \sigma_{i_1} \ldots \sigma_{i_n}$, q_0 the initial state q_{start} and σ_1 is the blank symbol)
- D which asserts that for each t there is one and only one state, and I which asserts that the machine reaches the accepting state at some time.

The above is for naught if we exclude the time dependencies between the symbols. To wit:

Cook's Theorem

We also set F, G, H that assert that for each time t the values of the P, Q and S symbols are updated properly.

Cook's Theorem

We also set F, G, H that assert that for each time t the values of the P, Q and S symbols are updated properly.
For example G is the conjunction over all t, i, j of $G^t_{i,j}$ which asserts that if at time t the machine is in state q_i scanning symbol σ_j then at time $t+1$ the machine is in state q_k that is demanded by the transition function for M:

$$G^t_{i,j} = \bigwedge_{s=1}^{T} (\neg Q^i_t \vee \neg S_{s,t} \vee \neg P^j_{s,t} \vee Q^k_{t+1})$$

Cook's Theorem

We now set

$$A(w) = B \wedge C \wedge D \wedge E \wedge F \wedge G \wedge H \wedge I$$

Cook's Theorem

We now set

$$A(w) = B \wedge C \wedge D \wedge E \wedge F \wedge G \wedge H \wedge I$$

By virtue of construction

$$M(w) \downarrow_{q_{yes}} \iff A(w) \text{ is satisfiable} \iff \neg A(w) \text{ is not a tautology .}$$

Cook's Theorem

We now set

$$A(w) = B \wedge C \wedge D \wedge E \wedge F \wedge G \wedge H \wedge I$$

By virtue of construction

$$M(w) \downarrow_{q_{yes}} \iff A(w) \text{ is satisfiable} \iff \neg A(w) \text{ is not a tautology .}$$

Thus, our ability to decide $\neg A(w) \notin \{TAUT\}$ in polynomial time allows us to decide $L(M)$ in polynomial time.

Understanding *NP*

Now that we have established a sense of the difficulty of $\{TAUT\}$ we turn our attention to the fact that deciding the language $L(M)$ may not be of importance, but its recognition might be.

Understanding *NP*

Now that we have established a sense of the difficulty of $\{TAUT\}$ we turn our attention to the fact that deciding the language $L(M)$ may not be of importance, but its recognition might be.
While $\{TAUT\}$ is an important set, it doesn't leave any room to now estimate the difficulty of *NP* problems relative to it. Observe that it is easy to guess if a propositional formula is not a tautology (as it is to guess $M(w)\downarrow_{q_{yes}}$), but our intuition and experience leads us to the suspicion that to verify that it is not a tautology, may have much greater cost.

Understanding *NP*

Now that we have established a sense of the difficulty of $\{TAUT\}$ we turn our attention to the fact that deciding the language $L(M)$ may not be of importance, but its recognition might be.
While $\{TAUT\}$ is an important set, it doesn't leave any room to now estimate the difficulty of *NP* problems relative to it. Observe that it is easy to guess if a propositional formula is not a tautology (as it is to guess $M(w)\downarrow_{q_{yes}}$), but our intuition and experience leads us to the suspicion that to verify that it is not a tautology, may have much greater cost.
The class of *NP* essentially constitutes of problems whose solution we can guess and validate in polynomial time. It is likely that demanding that we know if $w \in \{TAUT\}$ quickly may be too great of a demand.

The Up to Date formulation (Cook-Levin)

The problem: Deciding $\{TAUT\}$ may be too hard of a problem to be a representative of an upper bound to the difficulty for the NP class.

The Up to Date formulation (Cook-Levin)

The problem: Deciding $\{TAUT\}$ may be too hard of a problem to be a representative of an upper bound to the difficulty for the NP class. **Solution:** Observe that $M(w) \downarrow_{q_{yes}} \iff A(w)$ is satisfiable.

Definition

$$SAT := \{w \in \{0,1\}^* \mid w \text{ is an encoding of a prop. formula and is satisfiable } \}$$

The Up to Date formulation (Cook-Levin)

Theorem

$\forall L \in NP\,(L \leq SAT)$ *(L is polynomially reducible to SAT).*

Proof.

Immediately apparent. □

Open Problems

We may now wonder, is there a chance that we can find a hard problem in *NP* (one that is not obviously polynomial/is as hard as SAT), whose instances we can recognize deterministically in polynomial time?
(*P* vs *NP*)

Open Problems

We may now wonder, is there a chance that we can find a hard problem in *NP* (one that is not obviously polynomial/is as hard as SAT), whose instances we can recognize deterministically in polynomial time?
(*P* vs *NP*)

How accurate is the intuition that $\{TAUT\}$ is much more complex than *SAT*? (*coNP* vs *NP*)

One More Reduction

It is further illustrated that $\{TAUT\}$ is not the only problem such that every NP problem can be reduced to it.

One More Reduction

It is further illustrated that $\{TAUT\}$ is not the only problem such that every NP problem can be reduced to it.

Definition

- $\{SUB\ ISO\} = \{w \in \{0,1\}^* \mid w$ is a representation of two graphs G_1, G_2 such that G_1 is isomorphic to a subgraph of $G_2\ \}$
- $\{DNF\ \ TAUT\} = \{w \in \{0,1\}^* \mid w$ is a representation of tautology in DNF form $\}$.

Theorem

$\{TAUT\}$ can be reduced to $\{SUB\ ISO\}$ and $\{DNF\ \ TAUT\}$

Thank you for your time!