Better Time-Space Lower Bounds for SAT and Related Problems

Ryan Williams
Carnegie Mellon University

June 12, 2005

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

We will show time lower bounds for the SAT problem, on random-access machines using $n^{o(1)}$ space

These lower bounds carry over to MAX-SAT, Hamilton Path, Vertex Cover, *etc.* [Raz and Van Melkebeek]

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

We will show time lower bounds for the SAT problem, on random-access machines using $n^{o(1)}$ space

These lower bounds carry over to MAX-SAT, Hamilton Path, Vertex Cover, *etc.* [Raz and Van Melkebeek]

Previous time lower bounds for SAT (assuming $n^{o(1)}$ space):

- $\omega(n)$ [Kannan 84]

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

We will show time lower bounds for the SAT problem, on random-access machines using $n^{o(1)}$ space

These lower bounds carry over to MAX-SAT, Hamilton Path, Vertex Cover, *etc.* [Raz and Van Melkebeek]

Previous time lower bounds for SAT (assuming $n^{o(1)}$ space):

- $\omega(n)$ [Kannan 84]
- $\Omega(n^{1+\varepsilon})$ for some $\varepsilon > 0$ [Fortnow 97]

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

We will show time lower bounds for the SAT problem, on random-access machines using $n^{o(1)}$ space

These lower bounds carry over to MAX-SAT, Hamilton Path, Vertex Cover, *etc.* [Raz and Van Melkebeek]

Previous time lower bounds for SAT (assuming $n^{o(1)}$ space):

- $\omega(n)$ [Kannan 84]
- $\Omega(n^{1+\varepsilon})$ for some $\varepsilon > 0$ [Fortnow 97]
- $\Omega(n^{\sqrt{2}-\varepsilon})$ for all $\varepsilon > 0$ [Lipton and Viglas 99]

Introduction

Few super-linear time lower bounds known for natural problems in NP

(Existing ones generally use a restricted computational model)

We will show time lower bounds for the SAT problem, on random-access machines using $n^{o(1)}$ space

These lower bounds carry over to MAX-SAT, Hamilton Path, Vertex Cover, *etc.* [Raz and Van Melkebeek]

Previous time lower bounds for SAT (assuming $n^{o(1)}$ space):

- $\omega(n)$ [Kannan 84]
- $\Omega(n^{1+\varepsilon})$ for some $\varepsilon > 0$ [Fortnow 97]
- $\Omega(n^{\sqrt{2}-\varepsilon})$ for all $\varepsilon > 0$ [Lipton and Viglas 99]
- $\Omega(n^{\phi-\varepsilon})$ where $\phi = 1.618...$ [Fortnow and Van Melkebeek 00]

Our Main Result

$\sqrt{2}$ and ϕ are nice constants...

The constant of our work will be (larger, but) not-so-nice.

Our Main Result

$\sqrt{2}$ and ϕ are nice constants...

The constant of our work will be (larger, but) not-so-nice.

Theorem: **For all k, SAT requires n^{Υ_k} time (infinitely often) on a random-access machine using $n^{o(1)}$ workspace, where Υ_k is the positive solution in $(1,2)$ to**

$$\Upsilon_k{}^3(\Upsilon_k - 1) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}}).$$

Our Main Result

$\sqrt{2}$ and ϕ are nice constants...

The constant of our work will be (larger, but) not-so-nice.

Theorem: **For all k, SAT requires n^{Υ_k} time (infinitely often) on a random-access machine using $n^{o(1)}$ workspace, where Υ_k is the positive solution in $(1, 2)$ to**

$$\Upsilon_k{}^3(\Upsilon_k - 1) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}}).$$

Define $\Upsilon := \lim_{k\to\infty} \Upsilon_k$. Then: $n^{\Upsilon-\varepsilon}$ lower bound.

Our Main Result

$\sqrt{2}$ and ϕ are nice constants...

The constant of our work will be (larger, but) not-so-nice.

Theorem: **For all k, SAT requires n^{Υ_k} time (infinitely often) on a random-access machine using $n^{o(1)}$ workspace, where Υ_k is the positive solution in $(1,2)$ to**

$$\Upsilon_k{}^3(\Upsilon_k - 1) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}}).$$

Define $\Upsilon := \lim_{k\to\infty} \Upsilon_k$. Then: $n^{\Upsilon-\varepsilon}$ lower bound.

(Note: the Υ stands for 'Ugly')

Our Main Result

$\sqrt{2}$ and ϕ are nice constants...

The constant of our work will be (larger, but) not-so-nice.

Theorem: **For all k, SAT requires n^{Υ_k} time (infinitely often) on a random-access machine using $n^{o(1)}$ workspace, where Υ_k is the positive solution in $(1, 2)$ to**

$$\Upsilon_k{}^3(\Upsilon_k - 1) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}}).$$

Define $\Upsilon := \lim_{k\to\infty} \Upsilon_k$. Then: $n^{\Upsilon-\varepsilon}$ lower bound.

(Note: the Υ stands for 'Ugly')

However, $\Upsilon \approx \sqrt{3} + \frac{6}{10000}$, so we'll present the result with $\sqrt{3}$.

Points About The Method We Use

- The theorem says for any sufficiently restricted machine, there is an infinite set of SAT instances it cannot solve correctly

  We will not construct such a set of instances for every machine!

  Proof is by contradiction: it would be absurd, if such a machine could solve SAT almost everywhere

- **Ours and the above cited methods use artificial computational models (alternating machines) to prove lower bounds for explicit problems in a realistic model**

Outline

- Preliminaries and Proof Strategy
- A Speed-Up Theorem

  (small-space computations can be accelerated using alternation)
- A Slow-Down Lemma

  *(*NTIME *can be efficiently simulated implies* Σ_kTIME *can be efficiently simulated with some slow-down)*
- Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound

  (the starting point for our approach)
- Our Inductive Argument

  (how to derive a better bound from Lipton-Viglas)
- From $n^{1.66}$ to $n^{1.732}$

  (a subtle argument that squeezes more from the induction)

Preliminaries: Two possibly obscure complexity classes

- $\mathsf{DTISP}[t, s]$ is *deterministic time t and space s, simultaneously*

  (Note $\mathsf{DTISP}[t, s] \neq \mathsf{DTIME}[t] \cap \mathsf{SPACE}[s]$ in general)

  We will be looking at $\mathsf{DTISP}[n^k, n^{o(1)}]$ for $k \geq 1$.

- $\mathsf{NQL} := \bigcup_{c \geq 0} \mathsf{NTIME}[n(\log n)^c] = \mathsf{NTIME}[n \cdot poly(\log n)]$

  The NQL stands for "nondeterministic quasi-linear time"

Preliminaries: SAT Facts

Satisfiability (SAT) is not only NP-complete, but also:

Theorem: [Cook 85, Gurevich and Shelah 89, Tourlakis 00]

SAT is NQL-complete, under reductions doable in $O(n \cdot \text{poly}(\log n))$ time and $O(\log n)$ space (simultaneously). Moreover the ith bit of the reduction can be computed in $O(\text{poly}(\log n))$ time.

Let $\mathcal{D}$ be closed under quasi-linear time, logspace reductions.

Preliminaries: SAT Facts

Satisfiability (SAT) is not only NP-complete, but also:

Theorem: [Cook 85, Gurevich and Shelah 89, Tourlakis 00]

SAT is NQL-complete, under reductions doable in $O(n \cdot \text{poly}(\log n))$ time and $O(\log n)$ space (simultaneously). Moreover the ith bit of the reduction can be computed in $O(\text{poly}(\log n))$ time.

Let $\mathcal{D}$ be closed under quasi-linear time, logspace reductions.

Corollary: If $\mathsf{NTIME}[n] \not\subseteq \mathcal{D}$, then SAT $\notin \mathcal{D}$.

If one can show $\mathsf{NTIME}[n]$ is not contained in some $\mathcal{D}$, then one can name an *explicit problem* (SAT) not in $\mathcal{D}$ (modulo polylog factors)

Preliminaries: Some Hierarchy Theorems

For reasonable $t(n) \geq n$,

$$\mathsf{NTIME}[t] \not\subseteq \mathsf{coNTIME}[o(t)].$$

Furthermore, for integers $k \geq 1$,

$$\Sigma_k\mathsf{TIME}[t] \not\subseteq \Pi_k\mathsf{TIME}[o(t)].$$

So, there's a tight time hierarchy within levels of the polynomial hierarchy.

Proof Strategy

Show if SAT has a sufficiently good algorithm, then one contradicts a hierarchy theorem.

Proof Strategy

Show if SAT has a sufficiently good algorithm, then one contradicts a hierarchy theorem.

Strategy of Prior work:

1. *Show that* $\mathsf{DTISP}[n^c, n^{o(1)}]$ *can be "sped-up" when simulated on an alternating machine*

2. *Show that* $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ *allows those alternations to be "removed" without much "slow-down"*

3. *Contradict a hierarchy theorem for small* c

Proof Strategy

Show if SAT has a sufficiently good algorithm, then one contradicts a hierarchy theorem.

Strategy of Prior work:

1. *Show that* $\mathsf{DTISP}[n^c, n^{o(1)}]$ *can be "sped-up" when simulated on an alternating machine*

2. *Show that* $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ *allows those alternations to be "removed" without much "slow-down"*

3. *Contradict a hierarchy theorem for small* c

Our proof will use the Σ_k time versus Π_k time hierarchy, for all k

Outline

- Preliminaries
- **A Speed-Up Theorem**
- A Slow-Down Lemma
- Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound
- Our Inductive Argument
- From $n^{1.66}$ to $n^{1.732}$

A Speed-Up Theorem

(Trading Time for Alternations)

Let:

- $t(n) = n^c$ for rational $c \geq 1$,
- $s(n)$ be $n^{o(1)}$, and
- $k \geq 2$ be an integer.

Theorem: [Fortnow and Van Melkebeek 00] [Kannan 83]

$$\mathsf{DTISP}[t, s] \subseteq \Sigma_k \mathsf{TIME}[t^{1/k+o(1)}] \cap \Pi_k \mathsf{TIME}[t^{1/k+o(1)}].$$

A Speed-Up Theorem

(Trading Time for Alternations)

Let:

- $t(n) = n^c$ for rational $c \geq 1$,
- $s(n)$ be $n^{o(1)}$, and
- $k \geq 2$ be an integer.

Theorem: [Fortnow and Van Melkebeek 00] [Kannan 83]

$$\mathsf{DTISP}[t, s] \subseteq \Sigma_k\mathsf{TIME}[t^{1/k+o(1)}] \cap \Pi_k\mathsf{TIME}[t^{1/k+o(1)}].$$

That is, for any machine M running in time t and using small workspace, there is an *alternating* machine M' that makes k alternations and takes roughly $\sqrt[k]{t}$ time.

Moreover, M' can start in either an existential or a universal state

Proof of the speed-up theorem

Let x be input, M be the small space machine to simulate

Goal: *Write a clever sentence in first-order logic with k (alternating) quantifiers that is equivalent to $M(x)$ accepting*

Proof of the speed-up theorem

Let x be input, M be the small space machine to simulate

Goal: *Write a clever sentence in first-order logic with k (alternating) quantifiers that is equivalent to $M(x)$ accepting*

Let C_j denote **configuration** of $M(x)$ after jth step: bit string encoding *head positions, workspace contents, finite control*

By space assumption on M, $|C_j| \in n^{o(1)}$

Proof of the speed-up theorem

Let x be input, M be the small space machine to simulate

Goal: *Write a clever sentence in first-order logic with k (alternating) quantifiers that is equivalent to $M(x)$ accepting*

Let C_j denote **configuration** of $M(x)$ after jth step: bit string encoding *head positions, workspace contents, finite control*

By space assumption on M, $|C_j| \in n^{o(1)}$

$M(x)$ accepts iff there is a sequence $C_1, C_2, \ldots, C_t$ where

- C_1 is the "initial" configuration,
- C_t is in "accept" state
- For all i, C_i leads to C_{i+1} in one step of M on input x.

Proof of speed-up theorem: The case $k = 2$

$M(x)$ accepts iff

$(\exists C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t)$
$(\forall i \in \{1, \ldots, \sqrt{t}\})$
[$C_{i \cdot \sqrt{t}}$ leads to $C_{(i+1) \cdot \sqrt{t}}$ in $\sqrt{t}$ steps, C_0 is initial, C_t is accepting]

Proof of speed-up theorem: The case $k = 2$

$M(x)$ accepts iff

$(\exists C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t)$
$(\forall i \in \{1, \ldots, \sqrt{t}\})$
$[C_{i\cdot\sqrt{t}}$ leads to $C_{(i+1)\cdot\sqrt{t}}$ in $\sqrt{t}$ steps, C_0 is initial, C_t is accepting]

Runtime on an alternating machine:

- $\exists$ takes $O(\sqrt{t} \cdot s) = t^{1/2+o(1)}$ time to write down the C_j's
- $\forall$ takes $O(\log t)$ time to write down i
- $[\cdots]$ takes $O(\sqrt{t} \cdot s)$ deterministic time to check

Two alternations, square root speedup

Proof of speed-up theorem: The $k = 3$ case, first attempt

For $k = 2$, we had

$(\exists C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t)$
$(\forall i \in \{0, 1, \ldots, \sqrt{t}\})$
[$C_{i \cdot \sqrt{t}}$ leads to $C_{(i+1) \cdot \sqrt{t}}$ in $\sqrt{t}$ steps, C_0 initial, C_t accepting]

Proof of speed-up theorem: The $k = 3$ case, first attempt

For $k = 2$, we had

$(\exists C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t)$
$(\forall i \in \{0, 1, \ldots, \sqrt{t}\})$
[$C_{i \cdot \sqrt{t}}$ leads to $C_{(i+1) \cdot \sqrt{t}}$ in $\sqrt{t}$ steps, C_0 initial, C_t accepting]

Observation: The $[\cdots]$ is an $O(\sqrt{t})$ time and small-space computation, thus we can speed it up by a square root as well

Proof of speed-up theorem: The $k = 3$ case, first attempt

For $k = 2$, we had

$(\exists C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t)$
$(\forall i \in \{0, 1, \ldots, \sqrt{t}\})$
[$C_{i \cdot \sqrt{t}}$ leads to $C_{(i+1)\cdot\sqrt{t}}$ in $\sqrt{t}$ steps, C_0 initial, C_t accepting]

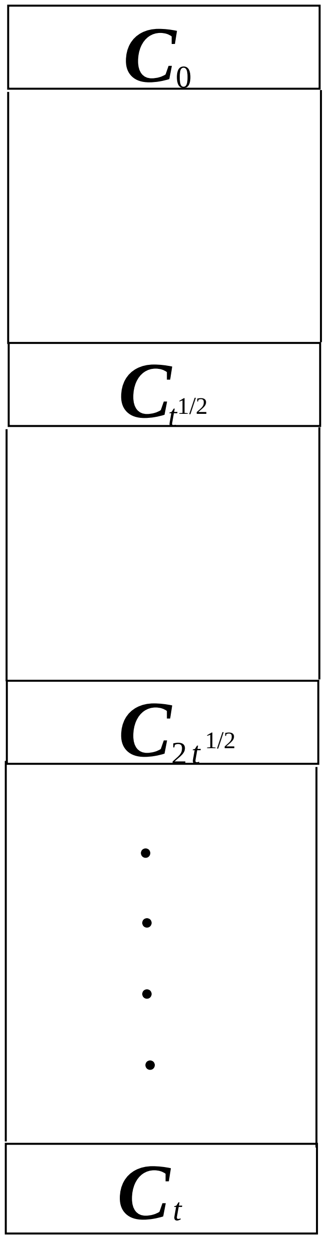
Observation: The $[\cdots]$ is an $O(\sqrt{t})$ time and small-space computation, thus we can speed it up by a square root as well

Straightforward way of doing this leads to:

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t)(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

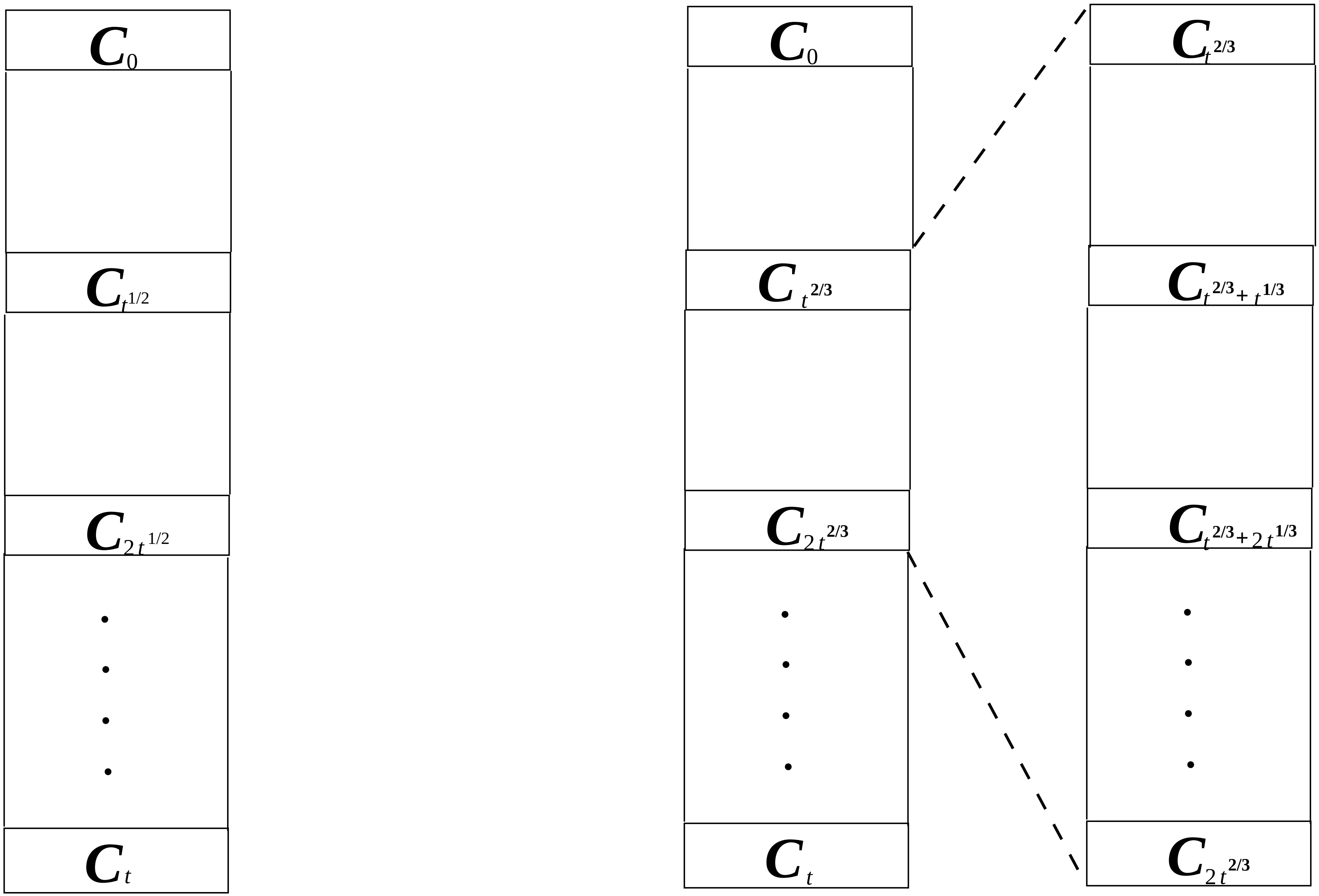
$(\exists C_{i\cdot t^{2/3}+t^{1/3}}, C_{i\cdot t^{2/3}+2\cdot t^{1/3}}, \ldots, C_{(i+1)\cdot t^{2/3}})(\forall j \in \{1, \ldots, \sqrt{t}\})$
[$C_{i\cdot t^{2/3}+j\cdot t^{1/3}}$ leads to $C_{i\cdot t^{2/3}+(j+1)\cdot t^{1/3}}$ in $t^{1/3}$ steps, C_0 initial, C_t accepting]

$k = 2$ has one "stage"



$k = 2$ has one "stage"

$k = 3$ has two "stages"



Proof of speed-up theorem: Not quite enough for $k = 3$

The $k = 3$ sentence we gave uses **four** quantifiers, for only $t^{1/3}$ time
(we want only three quantifier blocks)

Proof of speed-up theorem: Not quite enough for $k = 3$

The $k = 3$ sentence we gave uses **four** quantifiers, for only $t^{1/3}$ time (we want only three quantifier blocks)

Idea: *Take advantage of the computation's determinism – only one possible configuration at any step*

Proof of speed-up theorem: Not quite enough for $k = 3$

The $k = 3$ sentence we gave uses **four** quantifiers, for only $t^{1/3}$ time (we want only three quantifier blocks)

Idea: *Take advantage of the computation's determinism – only one possible configuration at any step*

The acceptance condition for $M(x)$ can be *complemented*:

$M(x)$ accepts iff

$(\forall C_0, C_{\sqrt{t}}, C_{2\sqrt{t}}, \ldots, C_t$ rejecting$)$
$(\exists i \in \{1, \ldots, \sqrt{t}\})$
[$C_{i \cdot \sqrt{t}}$ does not lead to $C_{i \cdot \sqrt{t} + \sqrt{t}}$ in $\sqrt{t}$ steps]

"For all configuration sequences $C_1, \ldots, C_t$ where C_t is rejecting, there exists a configuration C_i that does not lead to C_{i+1}"

The $k = 3$ case

We can therefore rewrite the $k = 3$ case, from

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t \text{ accepting})(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

$(\exists C_{i \cdot t^{2/3} + t^{1/3}}, C_{i \cdot t^{2/3} + 2 \cdot t^{1/3}}, \ldots, C_{(i+1) \cdot t^{2/3}})(\forall j \in \{1, \ldots, \sqrt{t}\})$

[$C_{i \cdot t^{2/3} + j \cdot t^{1/3}}$ leads to $C_{i \cdot t^{2/3} + (j+1) \cdot t^{1/3}}$ in $t^{1/3}$ steps]

to:

The $k = 3$ case

We can therefore rewrite the $k = 3$ case, from

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t \text{ accepting})(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

$(\exists C_{i \cdot t^{2/3} + t^{1/3}}, C_{i \cdot t^{2/3} + 2 \cdot t^{1/3}}, \ldots, C_{(i+1) \cdot t^{2/3}})(\forall j \in \{1, \ldots, \sqrt{t}\})$

[$C_{i \cdot t^{2/3} + j \cdot t^{1/3}}$ leads to $C_{i \cdot t^{2/3} + (j+1) \cdot t^{1/3}}$ in $t^{1/3}$ steps]

to:

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t \text{ accepting})(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

$(\forall C_{i \cdot t^{2/3} + t^{1/3}}, \ldots, C_{(i+1) \cdot t^{2/3} - t^{1/3}}, C'_{(i+1) \cdot t^{2/3}} \neq C_{(i+1) \cdot t^{2/3}})$

$(\exists j \in \{1, \ldots, \sqrt{t}\})$

[$C_{i \cdot t^{2/3} + j \cdot t^{1/3}}$ does not lead to $C_{i \cdot t^{2/3} + (j+1) \cdot t^{1/3}}$ in $t^{1/3}$ steps]

The $k = 3$ case

We can therefore rewrite the $k = 3$ case, from

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t \text{ accepting})(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

$(\exists C_{i \cdot t^{2/3} + t^{1/3}}, C_{i \cdot t^{2/3} + 2 \cdot t^{1/3}}, \ldots, C_{(i+1) \cdot t^{2/3}})(\forall j \in \{1, \ldots, \sqrt{t}\})$

[$C_{i \cdot t^{2/3} + j \cdot t^{1/3}}$ leads to $C_{i \cdot t^{2/3} + (j+1) \cdot t^{1/3}}$ in $t^{1/3}$ steps]

to:

$(\exists C_0, C_{t^{2/3}}, C_{2 \cdot t^{2/3}}, \ldots, C_t \text{ accepting})(\forall i \in \{0, 1, \ldots, t^{1/3}\})$

$(\forall C_{i \cdot t^{2/3} + t^{1/3}}, \ldots, C_{(i+1) \cdot t^{2/3} - t^{1/3}}, C'_{(i+1) \cdot t^{2/3}} \neq C_{(i+1) \cdot t^{2/3}})$

$(\exists j \in \{1, \ldots, \sqrt{t}\})$

[$C_{i \cdot t^{2/3} + j \cdot t^{1/3}}$ does not lead to $C_{i \cdot t^{2/3} + (j+1) \cdot t^{1/3}}$ in $t^{1/3}$ steps]

Voila! Three quantifier blocks. This is in $\Sigma_3\mathsf{TIME}[t^{1/3 + o(1)}]$

(and similarly one can show it's in $\Pi_3\mathsf{TIME}[t^{1/3 + o(1)}]$)

This can be generalized...

For arbitrary $k \geq 3$, one simply guesses (existentially or universally) $t^{1/k}$ configurations at each stage

This can be generalized...

For arbitrary $k \geq 3$, one simply guesses (existentially or universally) $t^{1/k}$ configurations at each stage

- Inverting quantifiers means the number of alternations only increases by one for every stage

$$(\exists\,\forall)\;(\forall\,\exists)\;(\exists\,\forall)\cdots$$

This can be generalized...

For arbitrary $k \geq 3$, one simply guesses (existentially or universally) $t^{1/k}$ configurations at each stage

- Inverting quantifiers means the number of alternations only increases by one for every stage

$$(\exists\,\forall)\;(\forall\,\exists)\;(\exists\,\forall)\cdots$$

- There are $k-1$ stages of guessing $t^{1/k}$ configurations, then $t^{1/k}$ time to deterministically verify configurations

Outline

- Preliminaries
- A Speed-Up Theorem
- **A Slow-Down Lemma**
- Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound
- Our Inductive Argument
- From $n^{1.66}$ to $n^{1.732}$

A Slow-Down Lemma

(Trading Alternations For Time)

Main Idea: The assumption $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}[n^c]$ allows one to remove alternations from a computation, with a small time increase

A Slow-Down Lemma

(Trading Alternations For Time)

Main Idea: The assumption $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}[n^c]$ allows one to remove alternations from a computation, with a small time increase

Let $t(n) \geq n$ be a polynomial, $c \geq 1$.

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}[n^c]$ then for all $k \geq 1$,

$$\Sigma_k\mathsf{TIME}[t] \subseteq \Sigma_{k-1}\mathsf{TIME}[t^c].$$

We prove the following, which will be very useful in our final proof.

A Slow-Down Lemma

(Trading Alternations For Time)

Main Idea: The assumption $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}[n^c]$ allows one to remove alternations from a computation, with a small time increase

Let $t(n) \geq n$ be a polynomial, $c \geq 1$.

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}[n^c]$ then for all $k \geq 1$,

$$\Sigma_k \mathsf{TIME}[t] \subseteq \Sigma_{k-1} \mathsf{TIME}[t^c].$$

We prove the following, which will be very useful in our final proof.

Theorem: If $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$ then

$$\Sigma_{k+1} \mathsf{TIME}[t] \subseteq \Sigma_k \mathsf{TIME}[t^c].$$

A Slow-Down Lemma: Proof

Assume $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$

Let M be a Σ_{k+1} machine running in time t

A Slow-Down Lemma: Proof

Assume $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$

Let M be a Σ_{k+1} machine running in time t

Recall $M(x)$ can be characterized by a first-order sentence:

$$(\exists x_1, |x_1| \leq t(|x|))(\forall x_2, |x_2| \leq t(|x|)) \cdots \\ (Qz, |x_{k+1}| \leq t(|x|))[P(x, x_1, x_2, \ldots, x_{k+1})]$$

where P "runs" in time $t(|x|)$

A Slow-Down Lemma: Proof

Assume $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$

Let M be a Σ_{k+1} machine running in time t

Recall $M(x)$ can be characterized by a first-order sentence:

$$(\exists x_1, |x_1| \leq t(|x|))(\forall x_2, |x_2| \leq t(|x|)) \cdots$$
$$(Qz, |x_{k+1}| \leq t(|x|))[P(x, x_1, x_2, \ldots, x_{k+1})]$$

where P "runs" in time $t(|x|)$

Important Point: *input* to P is of $O(t(|x|))$ length, so P actually runs in *linear time* with respect to the length of its input

A Slow-Down Lemma: Proof

Assume $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$

Define

$$R(x, x_1) \;:=\; (\forall x_2, |x_2| \leq t(|x|)) \cdots$$
$$(Qz, |x_{k+1}| \leq t(|x|))$$
$$[P(x, x_1, x_2, \ldots, x_{k+1})]$$

A Slow-Down Lemma: Proof

Assume $\Sigma_k \text{TIME}[n] \subseteq \Pi_k \text{TIME}[n^c]$

Define

$$\begin{aligned} R(x, x_1) \quad := \quad & (\forall x_2, |x_2| \leq t(|x|)) \cdots \\ & (Qz, |x_{k+1}| \leq t(|x|)) \\ & [P(x, x_1, x_2, \ldots, x_{k+1})] \end{aligned}$$

So $M(x)$ accepts iff $(\exists x_1, |x_1| \leq t(|x|)) R(x, x_1)$

A Slow-Down Lemma: Proof

Assume $\Sigma_k \mathsf{TIME}[n] \subseteq \Pi_k \mathsf{TIME}[n^c]$

Define
$$R(x, x_1) \quad := \quad (\forall x_2, |x_2| \leq t(|x|)) \cdots$$
$$(Qz, |x_{k+1}| \leq t(|x|))$$
$$[P(x, x_1, x_2, \ldots, x_{k+1})]$$

So $M(x)$ accepts iff $(\exists x_1, |x_1| \leq t(|x|)) R(x, x_1)$

- By definition, R recognized by a Π_k machine in time $t(|x|)$, *i.e.* *linear time* ($|x_1| = t(|x|)$).

A Slow-Down Lemma: Proof

Assume $\Sigma_k\mathsf{TIME}[n] \subseteq \Pi_k\mathsf{TIME}[n^c]$

Define
$$R(x,x_1) \;:=\; (\forall x_2, |x_2| \le t(|x|)) \cdots (Qz, |x_{k+1}| \le t(|x|)) [P(x,x_1,x_2,\ldots,x_{k+1})]$$

So $M(x)$ accepts iff $(\exists x_1, |x_1| \le t(|x|)) R(x,x_1)$

- By definition, R recognized by a Π_k machine in time $t(|x|)$, *i.e.* *linear time* ($|x_1| = t(|x|)$).
- By assumption, there is R' equivalent to R that starts with an $\exists$, has k quantifier blocks, is in $t(|x|)^c$ time

A Slow-Down Lemma: Proof

Assume $\Sigma_k\mathsf{TIME}[n] \subseteq \Pi_k\mathsf{TIME}[n^c]$

Define
$$R(x, x_1) \quad := \quad (\forall x_2, |x_2| \leq t(|x|)) \cdots$$
$$(Qz, |x_{k+1}| \leq t(|x|))$$
$$[P(x, x_1, x_2, \ldots, x_{k+1})]$$

So $M(x)$ accepts iff $(\exists x_1, |x_1| \leq t(|x|)) R(x, x_1)$

- By definition, R recognized by a Π_k machine in time $t(|x|)$, *i.e.* *linear time* ($|x_1| = t(|x|)$).
- By assumption, there is R' equivalent to R that starts with an $\exists$, has k quantifier blocks, is in $t(|x|)^c$ time

$M(x)$ accepts iff $[(\exists x_1, |x_1| \leq t(|x|)) R'(x, x_1)] \longleftarrow \Sigma_k\mathsf{TIME}[t^c]$

Outline

- Preliminaries
- A Speed-Up Theorem
- A Slow-Down Lemma
- **Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound**
- Our Inductive Argument
- From $n^{1.66}$ to $n^{1.732}$

Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound (Rephrased)

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ for some $c \geq 1$, then for all polynomials $t(n) \geq n$, $\mathsf{NTIME}[t] \subseteq \mathsf{DTISP}[t^c, t^{o(1)}]$

Theorem: $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$

Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound (Rephrased)

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ for some $c \geq 1$, then for all polynomials $t(n) \geq n$, $\mathsf{NTIME}[t] \subseteq \mathsf{DTISP}[t^c, t^{o(1)}]$

Theorem: $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$

Proof: Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$

(We will find a c that implies a contradiction)

Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound (Rephrased)

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ for some $c \geq 1$, then for all polynomials $t(n) \geq n$, $\mathsf{NTIME}[t] \subseteq \mathsf{DTISP}[t^c, t^{o(1)}]$

Theorem: $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$

Proof: Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$

(We will find a c that implies a contradiction)

- $\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{NTIME}[n^c]$, by slow-down theorem

Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound (Rephrased)

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ for some $c \geq 1$, then for all polynomials $t(n) \geq n$, $\mathsf{NTIME}[t] \subseteq \mathsf{DTISP}[t^c, t^{o(1)}]$

Theorem: $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$

Proof: Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$

(We will find a c that implies a contradiction)

- $\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{NTIME}[n^c]$, by slow-down theorem
- $\mathsf{NTIME}[n^c] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}]$, by assumption and padding

Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound (Rephrased)

Lemma: If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ for some $c \geq 1$, then for all polynomials $t(n) \geq n$, $\mathsf{NTIME}[t] \subseteq \mathsf{DTISP}[t^c, t^{o(1)}]$

Theorem: $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$

Proof: Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$

(We will find a c that implies a contradiction)

- $\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{NTIME}[n^c]$, by slow-down theorem
- $\mathsf{NTIME}[n^c] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}]$, by assumption and padding
- $\mathsf{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{c^2/2}]$, by speed-up theorem, so $c < \sqrt{2}$ **contradicts the hierarchy theorem** □

Outline

- Preliminaries
- A Speed-Up Theorem
- A Slow-Down Lemma
- Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound
- **Our Inductive Argument**
- From $n^{1.66}$ to $n^{1.732}$

Viewing Lipton-Viglas as a Lemma

(The Base Case for Our Induction)

We deliberately presented Lipton-Viglas's result differently from the original argument. In this way, we get

Lemma: $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \Pi_2\mathsf{TIME}[n^{c^2/2}].$$

Note if $c < 2$ then $c^2/2 < c$.

Viewing Lipton-Viglas as a Lemma

(The Base Case for Our Induction)

We deliberately presented Lipton-Viglas's result differently from the original argument. In this way, we get

Lemma: $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \Pi_2\mathsf{TIME}[n^{c^2/2}].$$

Note if $c < 2$ then $c^2/2 < c$.

- Thus, we may not necessarily have a contradiction for larger c, but we can remove one alternation from Σ_3 with only $n^{c^2/2}$ cost

Viewing Lipton-Viglas as a Lemma

(The Base Case for Our Induction)

We deliberately presented Lipton-Viglas's result differently from the original argument. In this way, we get

Lemma: $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \Pi_2\mathsf{TIME}[n^{c^2/2}].$$

Note if $c < 2$ then $c^2/2 < c$.

- Thus, we may not necessarily have a contradiction for larger c, but we can remove one alternation from Σ_3 with only $n^{c^2/2}$ cost
- Slow-down theorem implies $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

- $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$, by slow-down and lemma

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

- $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$, by slow-down and lemma
- $\Sigma_2\mathsf{TIME}[n^{c^2/2}] \subseteq \mathsf{NTIME}[n^{c^3/2}]$, by slow-down

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

- $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$, by slow-down and lemma
- $\Sigma_2\mathsf{TIME}[n^{c^2/2}] \subseteq \mathsf{NTIME}[n^{c^3/2}]$, by slow-down
- $\mathsf{NTIME}[n^{c^3/2}] \subseteq \mathsf{DTISP}[n^{c^4/2}, n^{o(1)}]$, by assumption and padding

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

- $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$, by slow-down and lemma
- $\Sigma_2\mathsf{TIME}[n^{c^2/2}] \subseteq \mathsf{NTIME}[n^{c^3/2}]$, by slow-down
- $\mathsf{NTIME}[n^{c^3/2}] \subseteq \mathsf{DTISP}[n^{c^4/2}, n^{o(1)}]$, by assumption and padding
- $\mathsf{DTISP}[n^{c^4/2}, n^{o(1)}] \subseteq \Pi_3\mathsf{TIME}[n^{c^4/6}]$, by speed-up

The Start of the Induction: Σ_3

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and the lemma

- $\Sigma_3\mathsf{TIME}[n] \subseteq \Sigma_2\mathsf{TIME}[n^{c^2/2}]$, by slow-down and lemma
- $\Sigma_2\mathsf{TIME}[n^{c^2/2}] \subseteq \mathsf{NTIME}[n^{c^3/2}]$, by slow-down
- $\mathsf{NTIME}[n^{c^3/2}] \subseteq \mathsf{DTISP}[n^{c^4/2}, n^{o(1)}]$, by assumption and padding
- $\mathsf{DTISP}[n^{c^4/2}, n^{o(1)}] \subseteq \Pi_3\mathsf{TIME}[n^{c^4/6}]$, by speed-up

Observe:

- Now $c < \sqrt[4]{6} \approx 1.565$ contradicts time hierarchy for Σ_3 and Π_3
- But if $c \geq \sqrt[4]{6}$, then we obtain a new "lemma":

$$\Sigma_3\mathsf{TIME}[n] \subseteq \Pi_3\mathsf{TIME}[n^{c^4/6}]$$

$\Sigma_4, \Sigma_5, \ldots$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and lemmas

$\Sigma_4, \Sigma_5, \ldots$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and lemmas

(Here we drop the TIME from $\Sigma_k\mathsf{TIME}$ for tidiness)

$$\Sigma_4[n] \subseteq \Sigma_3[n^{\frac{c^4}{6}}] \subseteq \Sigma_2[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}}] \subseteq \mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}]$$, but

$\Sigma_4, \Sigma_5, \ldots$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and lemmas

(Here we drop the TIME from $\Sigma_k\mathsf{TIME}$ for tidiness)

$$\Sigma_4[n] \subseteq \Sigma_3[n^{\frac{c^4}{6}}] \subseteq \Sigma_2[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}}] \subseteq \mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}], \text{ but}$$

$$\mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}] \subseteq \mathsf{DTISP}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c^2}, n^{o(1)}] \subseteq \Pi_4[n^{c^8/48}]$$

($c < \sqrt[8]{48} \approx 1.622$ implies contradiction)

$\Sigma_4, \Sigma_5, \ldots$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and lemmas

(Here we drop the TIME from $\Sigma_k\mathsf{TIME}$ for tidiness)

$$\Sigma_4[n] \subseteq \Sigma_3[n^{\frac{c^4}{6}}] \subseteq \Sigma_2[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}}] \subseteq \mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}]\text{, but}$$

$$\mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}] \subseteq \mathsf{DTISP}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c^2}, n^{o(1)}] \subseteq \Pi_4[n^{c^8/48}]$$

($c < \sqrt[8]{48} \approx 1.622$ implies contradiction)

$$\Sigma_5[n] \subseteq \Sigma_4[n^{\frac{c^8}{48}}] \subseteq \Sigma_3[n^{\frac{c^{12}}{48\cdot 6}}] \subseteq \Sigma_2[n^{\frac{c^{14}}{48\cdot 6\cdot 2}}]\text{, and this is in}$$

$\Sigma_4, \Sigma_5, \ldots$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$ and lemmas

(Here we drop the TIME from $\Sigma_k\mathsf{TIME}$ for tidiness)

$$\Sigma_4[n] \subseteq \Sigma_3[n^{\frac{c^4}{6}}] \subseteq \Sigma_2[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}}] \subseteq \mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}]\text{, but}$$

$$\mathsf{NTIME}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c}] \subseteq \mathsf{DTISP}[n^{\frac{c^4}{6}\cdot\frac{c^2}{2}\cdot c^2}, n^{o(1)}] \subseteq \Pi_4[n^{c^8/48}]$$

($c < \sqrt[8]{48} \approx 1.622$ implies contradiction)

$$\Sigma_5[n] \subseteq \Sigma_4[n^{\frac{c^8}{48}}] \subseteq \Sigma_3[n^{\frac{c^{12}}{48\cdot 6}}] \subseteq \Sigma_2[n^{\frac{c^{14}}{48\cdot 6\cdot 2}}]\text{, and this is in}$$

$$\mathsf{NTIME}[n^{\frac{c^{15}}{48\cdot 12}}] \subseteq \mathsf{DTISP}[n^{\frac{c^{16}}{48\cdot 12}}, n^{o(1)}] \subseteq \Pi_5[n^{\frac{c^{16}}{48\cdot 60}}]$$

($c < \sqrt[16]{2880} \approx 1.645$ implies contradiction)

An intermediate lower bound, $n^{\Upsilon'}$

Assume $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$

Claim: The inductive process of the previous slide converges.

The constant derived is

$$\Upsilon' := \lim_{k \to \infty} f(k),$$

where $f(k) := \prod_{j=1}^{k-1} (1 + 1/j)^{1/2^j}$.

Note $\Upsilon' \approx 1.66$.

A Time-Space Tradeoff

Corollary: For every $c < 1.66$ there is $d > 0$ such that SAT is not in $\mathsf{DTISP}[n^c, n^d]$.

Outline

- Preliminaries
- A Speed-Up Theorem
- A Slow-Down Lemma
- Lipton and Viglas' $n^{\sqrt{2}}$ Lower Bound
- Our Inductive Argument
- **From $n^{1.66}$ to $n^{1.732}$**

From $n^{1.66}$ to $n^{1.732}$

$\mathsf{DTISP}[t, t^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[t^{1/k+o(1)}]$ is an **unconditional** result

From $n^{1.66}$ to $n^{1.732}$

$\mathsf{DTISP}[t, t^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[t^{1/k+o(1)}]$ is an **unconditional** result

All other derived class inclusions in the above proof actually depend on the assumption that $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$.

From $n^{1.66}$ to $n^{1.732}$

$\mathsf{DTISP}[t, t^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[t^{1/k+o(1)}]$ is an **unconditional** result

All other derived class inclusions in the above proof actually depend on the assumption that $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$.

We'll now show how such an assumption can get

$$\mathsf{DTISP}[n^c, n^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[n^{c/(k+\varepsilon)+o(1)}]$$

for some $\varepsilon > 0$. This will push the lower bound higher.

From $n^{1.66}$ to $n^{1.732}$

$\mathsf{DTISP}[t, t^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[t^{1/k+o(1)}]$ is an **unconditional** result

All other derived class inclusions in the above proof actually depend on the assumption that $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$.

We'll now show how such an assumption can get

$$\mathsf{DTISP}[n^c, n^{o(1)}] \subseteq \Pi_k \mathsf{TISP}[n^{c/(k+\varepsilon)+o(1)}]$$

for some $\varepsilon > 0$. This will push the lower bound higher.

Lemma: Let $c \leq 2$. Define $d_1 := 2$, $d_k := 1 + \frac{d_{k-1}}{c}$.

If $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$, then

for all k, $\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2 \mathsf{TIME}[n^{1+o(1)}]$.

From $n^{1.66}$ to $n^{1.732}$

$\mathsf{DTISP}[t, t^{o(1)}] \subseteq \Pi_k\mathsf{TISP}[t^{1/k+o(1)}]$ is an **unconditional** result

All other derived class inclusions in the above proof actually depend on the assumption that $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$.

We'll now show how such an assumption can get

$$\mathsf{DTISP}[n^c, n^{o(1)}] \subseteq \Pi_k\mathsf{TISP}[n^{c/(k+\varepsilon)+o(1)}]$$

for some $\varepsilon > 0$. This will push the lower bound higher.

Lemma: Let $c \leq 2$. Define $d_1 := 2$, $d_k := 1 + \frac{d_{k-1}}{c}$.

If $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$, then

for all k, $\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

For $c < 2$, $\{d_k\}$ is increasing – for each k, a bit more of $\mathsf{DTISP}[n^{O(1)}, n^{o(1)}]$ is shown to be contained in $\Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Proof of Lemma

Lemma: Let $c < 2$. Define $d_1 := 2$, $d_k := 1 + \frac{d_{k-1}}{c}$.

If $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$, then

for all $k \in \mathbb{N}$, $\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

Induction on k. $k = 1$ case is trivial (speedup theorem).

Suppose $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$ and

$$\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}].$$

Proof of Lemma

Lemma: Let $c < 2$. Define $d_1 := 2$, $d_k := 1 + \frac{d_{k-1}}{c}$.

If $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$, then

for all $k \in \mathbb{N}$, $\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

Induction on k. $k = 1$ case is trivial (speedup theorem).

Suppose $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$ and

$$\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}].$$

Want: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

Proof of Lemma

Lemma: Let $c < 2$. Define $d_1 := 2$, $d_k := 1 + \frac{d_{k-1}}{c}$.

If $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$, then

for all $k \in \mathbb{N}$, $\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

Induction on k. $k = 1$ case is trivial (speedup theorem).

Suppose $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$ and

$$\mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}].$$

Want: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

By padding, the purple assumptions imply

$$\mathsf{NTIME}[n^{d_k/c}] \subseteq \mathsf{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]. \ (*)$$

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$

$(\exists i \in \{1, \ldots, n^{1-o(1)} - 1\})[C_i$ does not lead to C_{i+1} in $n^{d_k/c+o(1)}$ time$]$

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$

$(\exists i \in \{1, \ldots, n^{1-o(1)} - 1\})[C_i$ does not lead to C_{i+1} in $n^{d_k/c+o(1)}$ time$]$

Green part is an NTIME computation, input of length $O(n)$, takes $n^{d_k/c+o(1)}$ time

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$
$(\exists i \in \{1, \ldots, n^{1-o(1)} - 1\})[C_i$ does not lead to C_{i+1} in $n^{d_k/c+o(1)}$ time$]$

Green part is an NTIME computation, input of length $O(n)$, takes $n^{d_k/c+o(1)}$ time

$(*) \Longrightarrow$ Green can be replaced with $\Pi_2\mathsf{TIME}[n^{1+o(1)}]$ computation, *i.e.*

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

$$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$$
$$(\exists i \in \{1, \ldots, n^{1-o(1)} - 1\})[C_i \text{ does not lead to } C_{i+1} \text{ in } n^{d_k/c+o(1)} \text{ time}]$$

Green part is an NTIME computation, input of length $O(n)$, takes $n^{d_k/c+o(1)}$ time

$(*) \implies$ Green can be replaced with $\Pi_2\mathsf{TIME}[n^{1+o(1)}]$ computation, *i.e.*

$$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$$
$$(\forall y, |y| = c|x|^{1+o(1)})\ (\exists z, |z| = c|z|^{1+o(1)})[R(C_1, \ldots, C_{n^{1-o(1)}}, x, y, z)],$$

for some deterministic linear time relation R and constant $c > 0$.

Goal: $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$

Consider a Π_2 simulation of $\mathsf{DTISP}[n^{1+d_k/c}, n^{o(1)}]$ with only $O(n)$ bits ($n^{1-o(1)}$ configurations) in the universal quantifier:

$$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$$
$$(\exists i \in \{1, \ldots, n^{1-o(1)} - 1\})[C_i \text{ does not lead to } C_{i+1} \text{ in } n^{d_k/c+o(1)} \text{ time}]$$

Green part is an NTIME computation, input of length $O(n)$, takes $n^{d_k/c+o(1)}$ time

$(*) \implies$ Green can be replaced with $\Pi_2\mathsf{TIME}[n^{1+o(1)}]$ computation, *i.e.*

$$(\forall \text{ configurations } C_1, \ldots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \text{ s.t. } C_{n^{1-o(1)}} \text{ is rejecting})$$
$$(\forall y, |y| = c|x|^{1+o(1)})\ (\exists z, |z| = c|z|^{1+o(1)})[R(C_1, \ldots, C_{n^{1-o(1)}}, x, y, z)],$$

for some deterministic linear time relation R and constant $c > 0$.

Therefore, $\mathsf{DTISP}[n^{d_{k+1}}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$. $\square$

New Lemma Gives Better Bound

Corollary 1

Let $c \in (1, 2)$. *If* $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$ *then for all* $\varepsilon > 0$ *such that* $\frac{c}{c-1} - \varepsilon \geq 1$,

$\mathsf{DTISP}[n^{\frac{c}{c-1} - \varepsilon}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

New Lemma Gives Better Bound

Corollary 1

Let $c \in (1,2)$. *If* $\mathsf{NTIME}[n^{2/c}] \subseteq \mathsf{DTISP}[n^2, n^{o(1)}]$ *then*

for all $\varepsilon > 0$ *such that* $\frac{c}{c-1} - \varepsilon \geq 1$,
$\mathsf{DTISP}[n^{\frac{c}{c-1}-\varepsilon}, n^{o(1)}] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$.

Proof. Recall $d_2 = 2$, $d_k = 1 + d_{k-1}/c$.

$\{d_k\}$ is monotone non-decreasing for $c < 2$; converges to $d_\infty = 1 + \frac{d_\infty}{c}$

$\Longrightarrow d_\infty = c/(c-1)$. (Note $c = 2$ implies $d_\infty = 2$)

It follows that for all ε, there's a K such that $d_K \geq \frac{c}{c-1} - \varepsilon$. $\square$

Now: Apply inductive method from $n^{1.66}$ lower bound–

the "base case" now resembles Fortnow-Van Melkebeek's n^{ϕ} lower bound

If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$, **Corollary 1** implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \mathsf{DTISP}[\left(n^{c^2 \cdot \frac{c-1}{c}}\right)^{c/(c-1)+o(1)}, n^{o(1)}]$$

$$\subseteq \Pi_2\mathsf{TIME}[n^{c\cdot(c-1)+o(1)}]. \quad \phi(\phi - 1) = 1$$

Now: Apply inductive method from $n^{1.66}$ lower bound–

the "base case" now resembles Fortnow-Van Melkebeek's n^{ϕ} lower bound

If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$, **Corollary 1** implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \mathsf{DTISP}[\left(n^{c^2 \cdot \frac{c-1}{c}}\right)^{c/(c-1)+o(1)}, n^{o(1)}]$$

$$\subseteq \Pi_2\mathsf{TIME}[n^{c\cdot(c-1)+o(1)}]. \quad \phi(\phi - 1) = 1$$

Inducting as before, we get

Now: Apply inductive method from $n^{1.66}$ lower bound–

the "base case" now resembles Fortnow-Van Melkebeek's n^{ϕ} lower bound

If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$, **Corollary 1** implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \mathsf{DTISP}[\left(n^{c^2 \cdot \frac{c-1}{c}}\right)^{c/(c-1)+o(1)}, n^{o(1)}]$$

$$\subseteq \Pi_2\mathsf{TIME}[n^{c\cdot(c-1)+o(1)}]. \quad \phi(\phi - 1) = 1$$

Inducting as before, we get

$$\Sigma_3[n] \subseteq \Sigma_2[n^{c\cdot(c-1)}] \subseteq \mathsf{DTISP}[n^{c^3\cdot(c-1)}, n^{o(1)}] \subseteq \Pi_3[n^{\frac{c^3\cdot(c-1)}{3}}]$$, then

Now: Apply inductive method from $n^{1.66}$ lower bound–

the "base case" now resembles Fortnow-Van Melkebeek's n^{ϕ} lower bound

If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^{o(1)}]$, **Corollary 1** implies

$$\Sigma_2\mathsf{TIME}[n] \subseteq \mathsf{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \mathsf{DTISP}[\left(n^{c^2 \cdot \frac{c-1}{c}}\right)^{c/(c-1)+o(1)}, n^{o(1)}]$$

$$\subseteq \Pi_2\mathsf{TIME}[n^{c\cdot(c-1)+o(1)}]. \quad \phi(\phi-1)=1$$

Inducting as before, we get

$$\Sigma_3[n] \subseteq \Sigma_2[n^{c\cdot(c-1)}] \subseteq \mathsf{DTISP}[n^{c^3\cdot(c-1)}, n^{o(1)}] \subseteq \Pi_3[n^{\frac{c^3\cdot(c-1)}{3}}], \text{ then}$$

$$\Sigma_4[n] \subseteq \Sigma_3[n^{\frac{c^3\cdot(c-1)}{3}}] \subseteq \Sigma_2[n^{\frac{c^4\cdot(c-1)^2}{3}}] \subseteq \mathsf{DTISP}[n^{\frac{c^6\cdot(c-1)^2}{3}}, n^{o(1)}]$$

$$\subseteq \Pi_4[n^{\frac{c^6\cdot(c-1)^2}{12}}], \textit{ etc.}$$

Claim: The exponent e_k derived for $\Sigma_k\mathsf{TIME}[n] \subseteq \Pi_k\mathsf{TIME}[n^{e_k}]$ is

$$e_k = \frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))}.$$

Finishing up

Simplifying, $e_k =$

$$\frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))} = \left(\frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})}\right)^{2^{k-3}}$$

thus

$$e_k < 1 \iff \frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})} < 1$$

Finishing up

Simplifying, $e_k =$

$$\frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))} = \left(\frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})}\right)^{2^{k-3}}$$

thus

$$e_k < 1 \iff \frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})} < 1$$

- Define $f(k) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}})$

Finishing up

Simplifying, $e_k =$

$$\frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))} = \left(\frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})}\right)^{2^{k-3}}$$

thus

$$e_k < 1 \iff \frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})} < 1$$

- Define $f(k) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}})$
- $f(k) \to 3.81213\cdots$ as $k \to \infty$

Finishing up

Simplifying, $e_k =$

$$\frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))} = \left(\frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})}\right)^{2^{k-3}}$$

thus

$$e_k < 1 \iff \frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})} < 1$$

- Define $f(k) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}})$
- $f(k) \rightarrow 3.81213\cdots$ as $k \rightarrow \infty$
- Above task reduces to finding positive root of

$$c^3 \cdot (c-1) = 3.81213$$

Finishing up

Simplifying, $e_k =$

$$\frac{c^{3\cdot 2^{k-3}}(c-1)^{2^{k-3}}}{k\cdot(3^{2^{k-4}}\cdot 4^{2^{k-5}}\cdot 5^{2^{k-6}}\cdots(k-1))} = \left(\frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})}\right)^{2^{k-3}}$$

thus

$$e_k < 1 \iff \frac{c^3(c-1)}{k^{2^{-k+3}}\cdot(3^{2^{-1}}\cdot 4^{2^{-2}}\cdot 5^{2^{-3}}\cdots(k-1)^{2^{-k+3}})} < 1$$

- Define $f(k) = k^{2^{-k+3}} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \cdots (k-1)^{2^{-k+3}})$
- $f(k) \to 3.81213\cdots$ as $k \to \infty$
- Above task reduces to finding positive root of

$$c^3 \cdot (c-1) = 3.81213$$

$\Longrightarrow c \approx 1.7327 > \sqrt{3} + \frac{6}{10000}$ yields a contradiction.

The above inductive method can be applied to improve several existing lower bound arguments.

- Time lower bounds for SAT on off-line one-tape machines
- Time-space tradeoffs for nondeterminism/co-nondeterminism in RAM model
- *Etc.* See the paper!