CoCoNuT - Complexity - Lecture 4

N.P. Smart

Dept of Computer Science
University of Bristol,
Merchant Venturers Building

March 23, 2015

Outline

Resources

A natural philosophical question about the nature of computation is as follows:

Suppose I give you more resources, can you compute more?

To answer this we have to define what resources we mean and what we mean by compute more.

We shall mean by compute more as being able to solve more problems.

- There is also the question of can you compute things more efficiently,
- i.e. reduce the time complexity by spending other resources.
- e.g. Time/Memory trade-off algorithms.

Resources

So what resources do we have which we have not yet covered in our model.

There are in fact a large number of possibilities:

- Space/Memory
  - Bounding space rather than time enables us to compute a lot more than just bounding time.
- Random Numbers
  - True randomness is hard to come by, so we may not want to rely on it.
  - But if we assume randomness exists we can compute things faster
  - Perhaps we only want correct results with a given probability.
- Interaction
  - Perhaps communicating enables us to compute more?
  - May it is indeed true that “Its Good To Talk”

Space complexity for TMs

Defn: Space Complexity of TMs

Let M be a TM which halts on every input. The **space complexity** of M is $f : \mathbb{N} \to \mathbb{N}$, where $f(n)$ is the maximum number of tape cells that M scans for any input of length n.

Defn: The class SPACE($f(n)$)

Let $f : \mathbb{N} \to \mathbb{R}^+$. SPACE($f(n)$) is the class of all languages decided by an $O(f(n))$-space TM.

Space complexity for TMs

Defn: Space Complexity of TMs

Let M be a TM which halts on every input. The **space complexity** of M is $f : \mathbb{N} \to \mathbb{N}$, where $f(n)$ is the maximum number of tape cells that M scans for any input of length n.

Defn: The class SPACE($f(n)$)

Let $f : \mathbb{N} \to \mathbb{R}^+$. SPACE($f(n)$) is the class of all languages decided by an $O(f(n))$-space TM.

Space complexity for NTMs

Defn: Space Complexity of NTMs

Let N be a **N**TM which halts on every input. The space complexity of N is $f : \mathbb{N} \to \mathbb{N}$, where $f(n)$ is the maximum number of tape cells that N scans **on any computation branch** for any input of length n.

Defn: NSPACE(($f(n)$))

Let $f : \mathbb{N} \to \mathbb{R}^+$. **N**SPACE($f(n)$) is the class of all languages decided by an $O(f(n))$-space **N**TM.

Space complexity for NTMs

Defn: Space Complexity of NTMs

Let N be a **N**TM which halts on every input. The space complexity of N is $f : \mathbb{N} \to \mathbb{N}$, where $f(n)$ is the maximum number of tape cells that N scans **on any computation branch** for any input of length n.

Defn: NSPACE(($f(n)$))

Let $f : \mathbb{N} \to \mathbb{R}^+$. **N**SPACE($f(n)$) is the class of all languages decided by an $O(f(n))$-space **N**TM.

Space complexity classes

Definition. PSPACE $:= \bigcup_k$ SPACE(n^k) NPSPACE $:= \bigcup_k$ NSPACE(n^k)

Example. $SAT \in$ SPACE(n).

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\text{SPACE}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$$

(Recall: NTIME$(f(n)) \subseteq$ DTIME$(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$$

$$\boxed{\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXP}}$$

Space complexity classes

Definition. $\mathrm{PSPACE} := \bigcup_k \mathrm{SPACE}(n^k)$ $\qquad$ $\mathrm{NPSPACE} := \bigcup_k \mathrm{NSPACE}(n^k)$

Example. $SAT \in \mathrm{SPACE}(n)$.

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\mathrm{SPACE}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$$

(Recall: $\mathrm{NTIME}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$$

$$\boxed{\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXP}}$$

Space complexity classes

Definition. $\text{PSPACE} := \bigcup_k \text{SPACE}(n^k)$ $\qquad$ $\text{NPSPACE} := \bigcup_k \text{NSPACE}(n^k)$

Example. $SAT \in \text{SPACE}(n)$.

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\text{SPACE}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$$

(Recall: $\text{NTIME}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$$

$$\boxed{\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXP}}$$

Space complexity classes

Definition. PSPACE $:= \bigcup_k \mathrm{SPACE}(n^k)$ NPSPACE $:= \bigcup_k \mathrm{NSPACE}(n^k)$
Example. $SAT \in \mathrm{SPACE}(n)$.

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\mathrm{SPACE}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$$

(Recall: $\mathrm{NTIME}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$$

$$\boxed{\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXP}}$$

Space complexity classes

Definition. $\mathrm{PSPACE} := \bigcup_k \mathrm{SPACE}(n^k)$ $\quad$ $\mathrm{NPSPACE} := \bigcup_k \mathrm{NSPACE}(n^k)$

Example. $SAT \in \mathrm{SPACE}(n)$.

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\mathrm{SPACE}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$$

(Recall: $\mathrm{NTIME}(f(n)) \subseteq \mathrm{DTIME}(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$$

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXP}$$

Space complexity classes

Definition. PSPACE $:= \bigcup_k \text{SPACE}(n^k)$ NPSPACE $:= \bigcup_k \text{NSPACE}(n^k)$

Example. $SAT \in \text{SPACE}(n)$.

Theorem. For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$:

$$\text{SPACE}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$$

(Recall: $\text{NTIME}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$)

Theorem. *(Savitch)* For any $f : \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$ we have

$$\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$$

$$\boxed{\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXP}}$$

PSPACE-completeness

PSPACE-complete

A language B is **PSPACE-complete** if

- $B \in$ PSPACE, and
- every A in PSPACE is polynomial-time reducible to B

Quantified formulas

- Quantifiers:
  - $\forall$: for all
  - $\exists$: there exists
- Let $\phi(x_1, \ldots, x_n)$ be a Boolean formula.
- A **totally quantified** Boolean formula has a quantifier for every variable at the beginning.

PSPACE-completeness

PSPACE-complete

A language B is **PSPACE-complete** if

- $B \in$ PSPACE, and
- every A in PSPACE is polynomial-time reducible to B

Quantified formulas

- Quantifiers:
  - $\forall$: for all
  - $\exists$: there exists
- Let $\phi(x_1, \ldots, x_n)$ be a Boolean formula.
- A **totally quantified** Boolean formula has a quantifier for every variable at the beginning.

TQBF

A **totally quantified** Boolean formula has a quantifier for every variable at the beginning.

$$TQBF := \{\langle\psi\rangle \mid \psi \text{ is a true totally quantified Boolean formula}\}$$

Theorem

TQBF is PSPACE-complete.

We will now prove this theorem...

TQBF

A **totally quantified** Boolean formula has a quantifier for every variable at the beginning.

$$\textit{TQBF} := \{\langle\psi\rangle \mid \psi \text{ is a true totally quantified Boolean formula}\}$$

Theorem

TQBF is PSPACE-complete.

We will now prove this theorem...

TQBF $\in$ PSPACE

Let $\phi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n \psi(x_1, \dots, x_n)$ where

- $Q_i = \exists$ or $\forall$
- ψ has size m

Let $\phi_{x_1=b}$ for $b \in \{0, 1\}$ denote the formulae obtained by removing $Q_1 x_1$ and replacing x_1 by b in ψ.

Consider the following recursive algorithm A for deciding whether ϕ is true

- Compute $v_0 = A(\phi_{x_1=0})$.
- Compute $v_1 = A(\phi_{x_1=1})$.
- If $Q_1 = \exists$ then return one iff $v_0 = 1$ or $v_1 = 1$.
- If $Q_1 = \forall$ then return one iff $v_0 = 1$ and $v_1 = 1$.

Clearly A is correct.

TQBF $\in$ PSPACE

Let $s_{n,m}$ denote the space used by A on a formula with n variables and formulae of size m.

Each step requires $O(m)$ space to store $\phi_{x_1=b}$ for $b = 0$ and $b = 1$.

Each step requires two bits to store v_0 and v_1.

Thus $s_{n,m} = s_{n-1,m} + O(m)$.

So $s_{n,m} = O(n \cdot m)$.

Hence TQBF $\in$ PSPACE.

Configuration Graphs

We now move to showing TQBF is PSPACE complete.

Let M be a DTM or NDTM with input $x \in \{0,1\}^*$.

The configuration graph $G_{M,x}$ is the graph with

- Directed graph with nodes all possible configurations of M on input of x.
- Two nodes c and c' connected with an edge $c \longrightarrow c'$ iff the transition of c to c' is possible in one step.

If M is a DTM then the graph is a path

- Each node has out degree one.

If M is a NDTM then

- Each node has out degree at most two.

Configuration Graphs

We now move to showing TQBF is PSPACE complete.

Let M be a DTM or NDTM with input $x \in \{0,1\}^*$.

The configuration graph $G_{M,x}$ is the graph with

- Directed graph with nodes all possible configurations of M on input of x.
- Two nodes c and c' connected with an edge $c \longrightarrow c'$ iff the transition of c to c' is possible in one step.

If M is a DTM then the graph is a path

- Each node has out degree one.

If M is a NDTM then

- Each node has out degree at most two.

Configuration Graphs

We now move to showing TQBF is PSPACE complete.

Let M be a DTM or NDTM with input $x \in \{0, 1\}^*$.

The configuration graph $G_{M,x}$ is the graph with

- Directed graph with nodes all possible configurations of M on input of x.
- Two nodes c and c' connected with an edge $c \longrightarrow c'$ iff the transition of c to c' is possible in one step.

If M is a DTM then the graph is a path

- Each node has out degree one.

If M is a NDTM then

- Each node has out degree at most two.

Configuration Graphs of PSPACE Problems

Let $\mathcal{L} \in$ PSPACE and let M be a DTM which decides $\mathcal{L}$ in space $s(n)$.

Let $G_{M,x}$ denote the configuration graph of M on input x

Let $c \longrightarrow c'$ be an arc in $G_{M,x}$.

By our proof of Cook-Levin we can encode this transition as the AND of at most $O(s(n))$ constant sized checks.

So each arc corresponds to a formulae of size $O(s(n))$.

We can produce a formulae $\phi_0(a, b)$ which is true iff a and b are connected by an arc.

- Take OR of the above formulae and the formulae to test whether $a = b$.

Configuration Graphs of PSPACE Problems

Given $\phi_0(a, b)$ we define $\phi_i(a, b)$ recursively via

$$\phi_{i+1}(a, b) = \exists c, \forall x, y : ((x = a \text{ and } y = c) \text{ or } (x = c \text{ and } y = b)) \Longrightarrow \phi_i(x, y).$$

The size of ϕ_{i+1} is at most the size of ϕ_i plus $O(s(n))$.

- So size of $\phi_{s(n)}$ is $O(s(n)^2)$.
- Since $s(n) \cdot O(s(n)) = O(s(n)^2)$.

By induction the formulae $\phi_{i+1}(a, b)$ is true iff

- There is a path from a to b
- The size of the path is at most 2^i

Configuration Graphs of PSPACE Problems

Let c_S and c_E be the starting and ending nodes of the configuration graph of M on input x.

Note $G_{M,x}$ has at most $2^{O(s(n))}$ nodes

- See proof of Theorem 30 in notes for why.

So given M we have constructed a TQBF $\phi_{O(s(n))}$ of poly-size which encodes M.

Thus TQBF is PSPACE complete.

Note, this also shows that PSPACE $=$ NPSPACE.

Configuration Graphs of PSPACE Problems

Let c_S and c_E be the starting and ending nodes of the configuration graph of M on input x.

Note $G_{M,x}$ has at most $2^{O(s(n))}$ nodes

- See proof of Theorem 30 in notes for why.

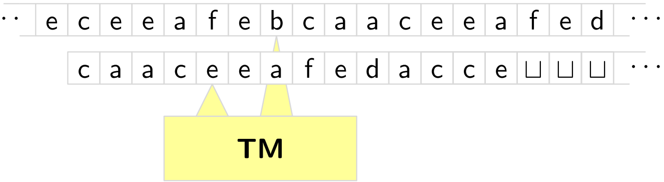
So given M we have constructed a TQBF $\phi_{O(s(n))}$ of poly-size which encodes M.

Thus TQBF is PSPACE complete.

Note, this also shows that PSPACE $=$ NPSPACE.

Sublinear Space

Sublinear space? Space complexity $f(n) < n =$ input size



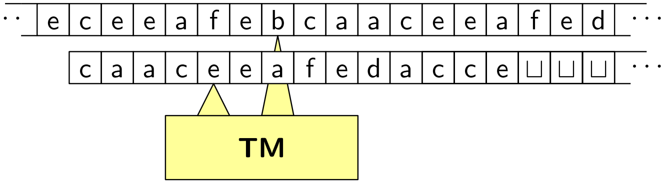
Space-bounded TM Two-tape TM

- Input tape is *read only*
- Work tape

The **space complexity** is defined by the number of cells scanned on the *work tape only*

Sublinear Space

Sublinear space? Space complexity $f(n) < n =$ input size



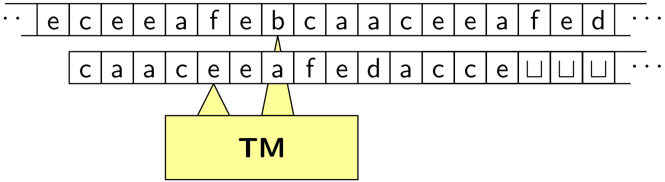
Space-bounded TM Two-tape TM

- Input tape is *read only*
- Work tape

The **space complexity** is defined by the number of cells scanned on the *work tape only*

Sublinear Space

Sublinear space? Space complexity $f(n) < n =$ input size



Space-bounded TM Two-tape TM

- Input tape is *read only*
- Work tape

The **space complexity** is defined by the number of cells scanned on the *work tape only*

Logarithmic space

Definition L and NL

L $= \text{SPACE}(\log n)$
NL $= \text{NSPACE}(\log n)$

Example. *PATH* $\in$ NL, *Undirected PATH* $\in$ L.

More results.

- NL-completeness: defined via log-space reductions
- *PATH* is NL-complete
- NL = co-NL

$$\text{L} \subseteq \text{NL} = \text{co-NL} \subseteq \text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXP}$$

Logarithmic space

Definition L and NL

L $=$ SPACE($\log n$)
NL $=$ NSPACE($\log n$)

Example. *PATH* $\in$ NL, *Undirected PATH* $\in$ L.

More results.

- NL-completeness: defined via log-space reductions
- *PATH* is NL-complete
- NL = co-NL

L $\subseteq$ NL$=$co-NL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE$=$NPSPACE $\subseteq$ EXP

Logarithmic space

Definition L and NL

L $= \text{SPACE}(\log n)$
NL $= \text{NSPACE}(\log n)$

Example. *PATH* $\in$ NL, *Undirected PATH* $\in$ L.

More results.

- NL-completeness: defined via log-space reductions
- *PATH* is NL-complete
- NL = co-NL

$L \subseteq NL = \text{co-}NL \subseteq P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXP$

Logarithmic space

Definition L and NL

L $=$ SPACE($\log n$)
NL $=$ NSPACE($\log n$)

Example. *PATH* $\in$ NL, *Undirected PATH* $\in$ L.

More results.

- NL-completeness: defined via log-space reductions
- *PATH* is NL-complete
- NL = co-NL

$$\boxed{\mathrm{L} \subseteq \mathrm{NL} = \text{co-NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXP}}$$

Space hierarchy

- Can we compute *more* when given more time/space?
- Could it be that all encountered classes are the *same*?

Space Hierarchy Theorem

For any space-constructible $f : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(f(n))$ space but not in $o(f(n))$ space.

- $\mathrm{SPACE}(n^{\varepsilon_1}) \subsetneq \mathrm{SPACE}(n^{\varepsilon_2})$, for $0 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{NL} \subsetneq \mathrm{PSPACE}$
- $\mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE} := \bigcup_k \mathrm{SPACE}(2^{n^k})$

Space hierarchy

- Can we compute *more* when given more time/space?
- Could it be that all encountered classes are the *same*?

Space Hierarchy Theorem

For any space-constructible $f : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(f(n))$ space but not in $o(f(n))$ space.

- $\mathrm{SPACE}(n^{\varepsilon_1}) \subsetneq \mathrm{SPACE}(n^{\varepsilon_2})$, for $0 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{NL} \subsetneq \mathrm{PSPACE}$
- $\mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE} := \bigcup_k \mathrm{SPACE}(2^{n^k})$

Space hierarchy

- Can we compute *more* when given more time/space?
- Could it be that all encountered classes are the *same*?

Space Hierarchy Theorem

For any space-constructible $f : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(f(n))$ space but not in $o(f(n))$ space.

- $\mathrm{SPACE}(n^{\varepsilon_1}) \subsetneq \mathrm{SPACE}(n^{\varepsilon_2})$, for $0 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{NL} \subsetneq \mathrm{PSPACE}$
- $\mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE} := \bigcup_k \mathrm{SPACE}(2^{n^k})$

Time hierarchy

Time Hierarchy Theorem

For any time-constructible $t : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(t(n))$ time but not in $o\left(\frac{t(n)}{\log t(n)}\right)$ time.

- $\mathrm{DTIME}(n^{\varepsilon_1}) \subsetneq \mathrm{DTIME}(n^{\varepsilon_2})$, for $1 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{P} \subsetneq \mathrm{EXP}$

$$\overbrace{\mathrm{L} \subseteq \underbrace{\mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}}_{\neq}}^{} \subseteq \mathrm{EXP} \subseteq \mathrm{EXPSPACE}$$

Time hierarchy

Time Hierarchy Theorem

For any time-constructible $t : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(t(n))$ time but not in $o\left(\frac{t(n)}{\log t(n)}\right)$ time.

- $\mathrm{DTIME}(n^{\varepsilon_1}) \subsetneq \mathrm{DTIME}(n^{\varepsilon_2})$, for $1 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{P} \subsetneq \mathrm{EXP}$

$$\mathrm{L} \subseteq \underbrace{\mathrm{NL} \subseteq \overbrace{\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}}_{\neq}\ \subseteq \underbrace{\mathrm{EXP}}^{\neq}}\ \subseteq \mathrm{EXPSPACE}$$

$$\overbrace{\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}}^{\neq} \qquad \underbrace{\mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}}_{\neq} \qquad \underbrace{\mathrm{PSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{EXPSPACE}}_{\neq}$$

Time hierarchy

Time Hierarchy Theorem

For any time-constructible $t : \mathbb{N} \to \mathbb{N}$, there exists a language A that is decidable in $O(t(n))$ time but not in $o\left(\frac{t(n)}{\log t(n)}\right)$ time.

- $\mathrm{DTIME}(n^{\varepsilon_1}) \subsetneq \mathrm{DTIME}(n^{\varepsilon_2})$, for $1 \leq \varepsilon_1 < \varepsilon_2$
- $\mathrm{P} \subsetneq \mathrm{EXP}$

$$\mathrm{L} \subseteq \underbrace{\mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}}_{\neq} \quad \overbrace{\mathrm{P} \cdots \mathrm{EXP}}^{\neq} \quad \underbrace{\mathrm{PSPACE} \subseteq \mathrm{EXP} \subseteq \mathrm{EXPSPACE}}_{\neq}$$