Lecture 4 - Space Complexity, PSPACE, TQBF

Rafael Oliveira
rafael.oliveira.teaching@gmail.com
University of Waterloo

CS 860 - Graduate Complexity Theory
Fall 2022

Overview

- PSPACE completeness
- TQBF and PSPACE-completeness

Reductions & completeness in PSPACE

Definition (Reductions in PSPACE)

Given two languages $L, L' \subseteq \{0,1\}^*$, we say that $L \leq_m L'$ if there is a poly-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in L'.$$

Reductions & completeness in PSPACE

Definition (Reductions in PSPACE)

Given two languages $L, L' \subseteq \{0,1\}^*$, we say that $L \leq_m L'$ if there is a poly-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in L'.$$

Definition

A language L' is PSPACE-hard if for every $L \in$ PSPACE, $L \leq_m L'$. If $L' \in$ PSPACE we say L' is PSPACE-complete.

Reductions & completeness in PSPACE

Definition (Reductions in PSPACE)

Given two languages $L, L' \subseteq \{0,1\}^*$, we say that $L \leq_m L'$ if there is a poly-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in L'.$$

Definition

A language L' is PSPACE-hard if for every $L \in$ PSPACE, $L \leq_m L'$. If $L' \in$ PSPACE we say L' is PSPACE-complete.

Proposition (A complete language)

The following language is PSPACE-complete:

SPACE-TMSAT $:= \{\langle M, x, 1^s \rangle \mid M(x) = 1$ *and* M *uses* s *space* $\}$

- PSPACE completeness
- TQBF and PSPACE-completeness

Quantified boolean formulas (QBF)

Definition (Quantified boolean formula)

A quantified boolean formula (QBF) is a formula of the form

$$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \ \ \varphi(x_1, \ldots, x_n)$$

where each $Q_i \in \{\exists, \forall\}$, $x_i \in \{0, 1\}$ and φ is a plain (no quantifiers) boolean formula.

Quantified boolean formulas (QBF)

Definition (Quantified boolean formula)

A quantified boolean formula (QBF) is a formula of the form

$$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \ \ \varphi(x_1, \ldots, x_n)$$

where each $Q_i \in \{\exists, \forall\}$, $x_i \in \{0, 1\}$ and φ is a plain (no quantifiers) boolean formula.

- by using auxiliary variables can assume φ is CNF or DNF

Quantified boolean formulas (QBF)

Definition (Quantified boolean formula)

A quantified boolean formula (QBF) is a formula of the form

$$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \dots, x_n)$$

where each $Q_i \in \{\exists, \forall\}$, $x_i \in \{0, 1\}$ and φ is a plain (no quantifiers) boolean formula.

- by using auxiliary variables can assume φ is CNF or DNF
- since all variables have a quantifier, the QBF is always either true or false

Quantified boolean formulas (QBF)

Definition (Quantified boolean formula)

A quantified boolean formula (QBF) is a formula of the form

$$Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n)$$

where each $Q_i \in \{\exists, \forall\}$, $x_i \in \{0, 1\}$ and φ is a plain (no quantifiers) boolean formula.

- by using auxiliary variables can assume φ is CNF or DNF
- since all variables have a quantifier, the QBF is always either true or false
- Examples:

$$\forall x \exists y \;\; (x \wedge y) \vee (\overline{x} \wedge \overline{y})$$

$$\exists x, y \forall a, b \;\; (a \wedge x) \oplus (b \wedge y)$$

NP and coNP

We can interpret NP and coNP-complete problems (SAT and $\overline{\text{SAT}}$) in terms of quantified boolean formulas

- For SAT: (satisfiability)

  Input: $\exists x_1, \ldots, x_n \;\; \varphi(x_1, \ldots, x_n)$
  Output: is the above true?

NP and coNP

We can interpret NP and coNP-complete problems (SAT and $\overline{\text{SAT}}$) in terms of quantified boolean formulas

- For SAT: (satisfiability)

  Input: $\exists x_1, \ldots, x_n \;\; \varphi(x_1, \ldots, x_n)$
  Output: is the above true?

- For $\overline{\text{SAT}}$: (tautology)

  Input: $\forall x_1, \ldots, x_n \;\; \varphi(x_1, \ldots, x_n)$
  Output: is the above true?

TQBF

Definition (TQBF)

The language TQBF is the set of all true quantified boolean formulae

$$\text{TQBF} := \{Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n) \text{ is a true QBF } \}$$

TQBF

Definition (TQBF)

The language TQBF is the set of all true quantified boolean formulae

$$\text{TQBF} := \{Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n) \text{ is a true QBF } \}$$

Theorem

TQBF is PSPACE-complete.

PSPACE-completeness of TQBF

TQBF $\in$ PSPACE

1. Let $\psi := Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n)$ be a QBF
   - $m :=$ size of φ

PSPACE-completeness of TQBF

TQBF $\in$ PSPACE

1. Let $\psi := Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n)$ be a QBF
   - $m :=$ size of φ
2. We will construct a TM M which decides whether $\psi \in$ TQBF using space $O(n(m+n))$
   - Idea: eliminate one variable at a time

PSPACE-completeness of TQBF

TQBF $\in$ PSPACE

1. Let $\psi := Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n)$ be a QBF
   - $m :=$ size of φ
2. We will construct a TM M which decides whether $\psi \in$ TQBF using space $O(n(m+n))$
   - Idea: eliminate one variable at a time
   - For $b \in \{0,1\}$, let $\psi|_{x_1=b}$ be the QBF

$$\psi|_{x_1=b} := Q_2 x_2 \cdots Q_n x_n \;\; \varphi(b, x_2, \ldots, x_n)$$

PSPACE-completeness of TQBF

TQBF $\in$ PSPACE

1. Let $\psi := Q_1 x_1 Q_2 x_2 \cdots Q_n x_n \;\; \varphi(x_1, \ldots, x_n)$ be a QBF
   - $m :=$ size of φ
2. We will construct a TM M which decides whether $\psi \in$ TQBF using space $O(n(m+n))$
   - Idea: eliminate one variable at a time
   - For $b \in \{0,1\}$, let $\psi|_{x_1=b}$ be the QBF

     $$\psi|_{x_1=b} := Q_2 x_2 \cdots Q_n x_n \;\; \varphi(b, x_2, \ldots, x_n)$$

   - If $Q_1 = \exists$, then

     $$M(\psi) = 1 \Leftrightarrow M(\psi|_{x_1=0}) = 1 \text{ OR } M(\psi|_{x_1=1}) = 1$$

     else, $Q_1 = \forall$ and

     $$M(\psi) = 1 \Leftrightarrow M(\psi|_{x_1=0}) = 1 \text{ AND } M(\psi|_{x_1=1}) = 1$$

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)
3. Space can be reused!
   - If $n = 0$, there are no variables and the formula can be evaluated in $O(m)$ space

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)
3. Space can be reused!
   - If $n = 0$, there are no variables and the formula can be evaluated in $O(m)$ space
   - If $n > 0$, note that each recursive call can use the same space.

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)
3. Space can be reused!
   - If $n = 0$, there are no variables and the formula can be evaluated in $O(m)$ space
   - If $n > 0$, note that each recursive call can use the same space.
   - After computing $M(\psi|_{x_1=0})$, M just needs to store the value of the bit $M(\psi|_{x_1=0})$ and use remaining space to compute $M(\psi|_{x_1=1})$

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)
3. Space can be reused!
   - If $n = 0$, there are no variables and the formula can be evaluated in $O(m)$ space
   - If $n > 0$, note that each recursive call can use the same space.
   - After computing $M(\psi|_{x_1=0})$, M just needs to store the value of the bit $M(\psi|_{x_1=0})$ and use remaining space to compute $M(\psi|_{x_1=1})$
   - $M(\psi|_{x_1=0})$ uses $c \cdot (m + n)$ space to store $\psi|_{x_1=b}$ for each recursive call (c constant), hence

$$s(n, m) = s(n-1, m) + c \cdot (m + n)$$

TQBF in PSPACE

Space analysis of our recursive algorithm M

1. size of input ψ is (n, m) (variables, formula size)
2. Let $s(n, m)$ be the space used by M on inputs of size (n, m)
3. Space can be reused!
    - If $n = 0$, there are no variables and the formula can be evaluated in $O(m)$ space
    - If $n > 0$, note that each recursive call can use the same space.
    - After computing $M(\psi|_{x_1=0})$, M just needs to store the value of the bit $M(\psi|_{x_1=0})$ and use remaining space to compute $M(\psi|_{x_1=1})$
    - $M(\psi|_{x_1=0})$ uses $c \cdot (m+n)$ space to store $\psi|_{x_1=b}$ for each recursive call (c constant), hence

      $$s(n, m) = s(n-1, m) + c \cdot (m+n)$$

    - $s(n, m) = O(n(m+n))$

TQBF is PSPACE-hard

1. Let $L \in$ PSPACE, we need to show that $L \leq_m$ TQBF.
   Need a poly-time $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in \text{TQBF}$$

TQBF is PSPACE-hard

1. Let $L \in$ PSPACE, we need to show that $L \leq_m$ TQBF.
   Need a poly-time $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in \text{TQBF}$$

2. $L \in$ PSPACE $\Rightarrow$ there exists c constant and TM M such that $M(x)$ uses $s(n) := n^c$-space for $x \in \{0,1\}^n$

TQBF is PSPACE-hard

1. Let $L \in \mathsf{PSPACE}$, we need to show that $L \leq_m \mathsf{TQBF}$.
   Need a poly-time $f : \{0,1\}^* \to \{0,1\}^*$ such that

$$x \in L \Leftrightarrow f(x) \in \mathsf{TQBF}$$

2. $L \in \mathsf{PSPACE} \Rightarrow$ there exists c constant and TM M such that $M(x)$ uses $s(n) := n^c$-space for $x \in \{0,1\}^n$
3. We will construct a QBF ψ of size $O(s(n)^2)$ such that

$$\psi \in \mathsf{TQBF} \Leftrightarrow M(x) = 1$$

Configuration Graphs - refresher

Definition (Configuration Graphs)

Let M be a TM in SPACE$(s(n))$, and c be a constant such that M uses $\leq c \cdot s(n)$ work tape space on inputs of length n.

1. Configuration of M:
   - contents of nonblank entries of M's tapes
   - state and head position

Configuration Graphs - refresher

Definition (Configuration Graphs)

Let M be a TM in SPACE$(s(n))$, and c be a constant such that M uses $\leq c \cdot s(n)$ work tape space on inputs of length n.

1. Configuration of M:
   - contents of nonblank entries of M's tapes
   - state and head position
2. Configuration graph of M on input x - denoted $G_{M,x}$ is a directed graph s.t.
   - nodes: all configurations of M where input tape contains exactly x and work tape contains $\leq c \cdot s(n)$ non-blank cells
   - directed edge from configuration C to C' if C' can be reached from C in one step of TM M

Configuration Graphs - refresher

Definition (Configuration Graphs)

Let M be a TM in SPACE$(s(n))$, and c be a constant such that M uses $\leq c \cdot s(n)$ work tape space on inputs of length n.

1. Configuration of M:
   - contents of nonblank entries of M's tapes
   - state and head position
2. Configuration graph of M on input x - denoted $G_{M,x}$ is a directed graph s.t.
   - nodes: all configurations of M where input tape contains exactly x and work tape contains $\leq c \cdot s(n)$ non-blank cells
   - directed edge from configuration C to C' if C' can be reached from C in one step of TM M

- By modifying M to erase all its work tapes before halting, can assume there is only one configuration C_{accept}

Configuration Graphs

Proposition

Let M be a (N)TM using $s(n)$ space and $x \in \{0,1\}^n$. If $G_{M,x}$ is the configuration graph of $M(x)$ then there is constant[1] $c > 0$ *such that*

1. *every vertex in $G_{M,x}$ can be described using $cs(n)$ bits. Hence $G_{M,x}$ has $\leq 2^{cs(n)}$ nodes.*

---

[1] depending on description of M

Configuration Graphs

Proposition

Let M be a (N)TM using $s(n)$ space and $x \in \{0,1\}^n$. If $G_{M,x}$ is the configuration graph of $M(x)$ then there is constant[1] $c > 0$ such that

1. *every vertex in $G_{M,x}$ can be described using $cs(n)$ bits. Hence $G_{M,x}$ has $\leq 2^{cs(n)}$ nodes.*
2. *There is $O(s(n))$ size CNF formula $\varphi_{M,x}$ such that for every two strings C, C', $\varphi_{M,x}(C, C') = 1$ iff C, C' are two neighboring configurations in $G_{M,x}$.*

[1] depending on description of M

TQBF is PSPACE-hard

1. We will construct a QBF ψ of size $O(s(n)^2)$ such that

$$\psi \in \text{TQBF} \Leftrightarrow M(x) = 1$$

- Let $m := s(n)$ be the number of bits needed to encode configuration of M for $x \in \{0,1\}^n$

TQBF is PSPACE-hard

1. We will construct a QBF ψ of size $O(s(n)^2)$ such that

$$\psi \in \text{TQBF} \Leftrightarrow M(x) = 1$$

- Let $m := s(n)$ be the number of bits needed to encode configuration of M for $x \in \{0,1\}^n$
- By item 2 of proposition, have $O(m)$ size CNF $\varphi_{M,x}$ such that $\forall C, C' \in \{0,1\}^m$ we have

$$\varphi_{M,x}(C, C') = 1 \Leftrightarrow (C, C') \in E(G_{M,x})$$

TQBF is PSPACE-hard

1. We will construct a QBF ψ of size $O(s(n)^2)$ such that

$$\psi \in \text{TQBF} \Leftrightarrow M(x) = 1$$

- Let $m := s(n)$ be the number of bits needed to encode configuration of M for $x \in \{0,1\}^n$
- By item 2 of proposition, have $O(m)$ size CNF $\varphi_{M,x}$ such that $\forall C, C' \in \{0,1\}^m$ we have

$$\varphi_{M,x}(C, C') = 1 \Leftrightarrow (C, C') \in E(G_{M,x})$$

- Construct ψ inductively such that $\forall C, C' \in \{0,1\}^m$

$$\psi(C, C') = 1 \Leftrightarrow \text{ there is path } C \mapsto C' \text{ in } G_{M,x}.$$

TQBF is PSPACE-hard

Construction of ψ

1. we'll construct $\psi_i(C, C')$ which is true iff there is path of length 2^i from C to C'

In this case $\psi = \psi_m$ and $\psi_0 = \varphi_{M,x}$

TQBF is PSPACE-hard

Construction of ψ

1. we'll construct $\psi_i(C, C')$ which is true iff there is path of length 2^i from C to C'

   In this case $\psi = \psi_m$ and $\psi_0 = \varphi_{M,x}$
2. Assume we have QBF ψ_{i-1}

   There is path of length 2^i from $C \mapsto C'$ iff there is C'' s.t.

$$\psi_{i-1}(C, C'') = \psi_{i-1}(C'', C') = 1.$$

TQBF is PSPACE-hard

Construction of ψ

1. we'll construct $\psi_i(C, C')$ which is true iff there is path of length 2^i from C to C'

   In this case $\psi = \psi_m$ and $\psi_0 = \varphi_{M,x}$
2. Assume we have QBF ψ_{i-1}

   There is path of length 2^i from $C \mapsto C'$ iff there is C'' s.t.

$$\psi_{i-1}(C, C'') = \psi_{i-1}(C'', C') = 1.$$

3. Naively:

$$\psi_i = \exists C'' \psi_{i-1}(C, C'') \wedge \psi_{i-1}(C'', C')$$

   Exponential blowup! Need to reuse previous formulae (circuit)

TQBF is PSPACE-hard

Construction of ψ

1. we'll construct $\psi_i(C, C')$ which is true iff there is path of length 2^i from C to C'

   In this case $\psi = \psi_m$ and $\psi_0 = \varphi_{M,x}$
2. Assume we have QBF ψ_{i-1}

   There is path of length 2^i from $C \mapsto C'$ iff there is C'' s.t.

$$\psi_{i-1}(C, C'') = \psi_{i-1}(C'', C') = 1.$$

3. More succinctly: $\psi_i(C, C') := \exists C'' \forall D_1, D_2$

$$\big((D_1 = C \wedge D_2 = C'') \vee (D_1 = C'' \wedge D_2 = C')\big) \Rightarrow \psi_{i-1}(D_1, D_2)$$

TQBF is PSPACE-hard

Construction of ψ

1. we'll construct $\psi_i(C, C')$ which is true iff there is path of length 2^i from C to C'

   In this case $\psi = \psi_m$ and $\psi_0 = \varphi_{M,x}$
2. Assume we have QBF ψ_{i-1}

   There is path of length 2^i from $C \mapsto C'$ iff there is C'' s.t.

$$\psi_{i-1}(C, C'') = \psi_{i-1}(C'', C') = 1.$$

3. More succinctly: $\psi_i(C, C') := \exists C'' \forall D_1, D_2$

$$\big((D_1 = C \wedge D_2 = C'') \vee (D_1 = C'' \wedge D_2 = C')\big) \Rightarrow \psi_{i-1}(D_1, D_2)$$

4. Now formula size recursion is:

$$S(\psi_i) = S(\psi_{i-1}) + O(m) \Rightarrow S(\psi) = O(m^2)$$

References I

Arora, Sanjeev and Barak, Boaz (2009)
Computational Complexity, A Modern Approach
Cambridge University Press

Meyer, A. and Stockmeyer, L. (1973)
Word problems requiring exponential time
STOC

Goldreich, Oded (2006)
Computational complexity: a conceptual perspective.
https://www.wisdom.weizmann.ac.il/~oded/cc-drafts.html