CS420

Introduction to the Theory of Computation

Lecture 7: Context-free grammars

Tiago Cogumbreiro

Attendance code: 1453

estalee.com

Google Workshop @ UMB

How to Make a Great Technical Resume

When: Wed. Oct. 2, 2pm-3pm
Where: CC-3-Ballroom A

- Get inside tips from industry experts at Google on how to prepare your technology resume so it gets viewed by potential employers
- Bring your resume, take notes and ask questions

RSVP: `goo.gle/UMB-Fall19-RSVP`

Google Workshop @ UMB

Tech Interview workshop with Google

Interview like a Pro

When: Wed. Oct. 9, 11am~12pm
Where: CC-2-Alumni Lounge

- Take a deep dive into interviewing techniques for technical internship and full time opportunities
- Hear directly from engineers and experts at Google, observe a simulated interview to gain insight and strategies

RSVP: `goo.gle/UMB-Fall19-RSVP`

Google Scholarships

- Women Techmakers Scholars Program: students increasing involvement of women in CS
- Google Lime Scholarship: students with disabilities
- Generation Google Scholarship: students from historically underrepresented groups
- Google Student Veterans of America Scholarship: student veterans
- Google Conference and Travel Scholarship

`buildyourfuture.withgoogle.com/scholarships/`

Non-regular proof for regular languages

0. **Assumption:** any pumping length p
1. State which w you pick
2. *Prove:* $w \in L$
3. *Prove:* $|w| \geq p$
4. **Assumptions:** $w = xyz$, $|xy| \leq p$, $|y| > 0$
5. State which i you pick in $xy^i z$ you pick, clearly say what $xy^i z$ is
6. *Prove:* $xy^i z \notin L$

Conclusion: L is not regular

Today we will learn...

- Context-Free Language
- Context-Free Grammar (CFG)
- Derivation
- Parse tree
- Writing context-free grammars
- Left-most derivations
- Ambiguous grammars

> *Section 2.1*

Context-free grammars

- Appear in the context of natural languages
- Allows the formalization of a syntactic structure of terms
- Context-free grammars introduce recursive definition
- Context-free grammars are widely used in the specification of protocols, file formats, compilers, and interpreters

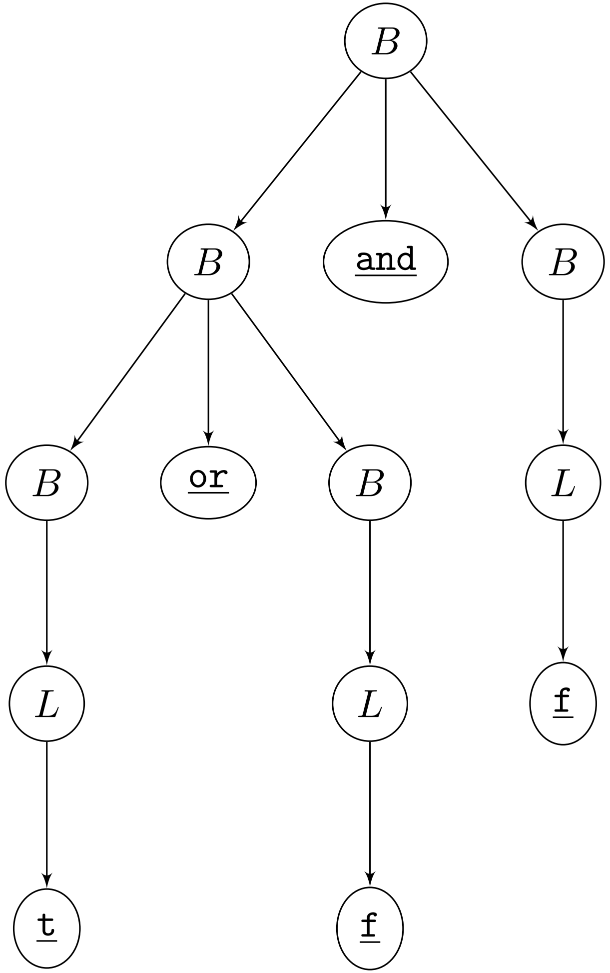
An example of a context-free grammar

Example grammar

- A boolean expression B can be either an and-operation, an or-operation, or a boolean literal.
- A boolean literal is either t or f

$$B \to B\ \texttt{and}\ B$$
$$B \to B\ \texttt{or}\ B$$
$$B \to L$$
$$L \to \texttt{t}$$
$$L \to \texttt{f}$$

Example: `t or f and f`



Grammar

- **Format:** A grammar G consists of a sequence of productions.
- **Start variable:** Every grammar has exactly one start variable. By *convention* the start variable is the first variable in the right-hand side of the first production.

Examples

Let grammar G consist of the following 5 productions:

- Production #1: $B \to B\ \texttt{and}\ B$
- Production #2: $B \to B\ \texttt{or}\ B$
- Production #3: $B \to L$
- Production #4: $L \to \texttt{t}$
- Production #5: $L \to \texttt{f}$

Productions

- **Also Known As:** substitution rule, or just a rule.
- **Format:** a **variable**, say A, followed by an arrow $\rightarrow$, and then a possibly-empty sequence of **terminals** / variables
- **Starts from:** A production **starts from** the variable on the left-hand side of the production. Example, production $B \rightarrow L$ starts from B (and not from L)
- **Variables** a symbol distinguished by $an\ italic\ font$, often capital letters. Examples: B or L.
- **Terminals** a symbol distinguished by a `mono` type font, often lower-case letters / numbers

Example

$$\underbrace{B}_{\text{variable}} \rightarrow \underbrace{B}_{\text{var.}}\underbrace{\texttt{and}}_{\text{sym.}}\underbrace{B}_{\text{var.}}$$

Generating strings

Yield $u \Rightarrow v$

Operation yield, given a string in the form $u\underline{A}v$ returns $u\underline{w}v$ if there is some rule $A \to w$ in the grammar.

Example

Grammar

$B \to B$ **and** B (1)

$B \to B$ **or** B

$B \to L$ (2), (4)

$L \to$ **t** (3)

$L \to$ **f** (5)

Derivation

$$\underbrace{B \Rightarrow}_{1} B \ \texttt{and}\ B$$

Example

Grammar

$B \to B \texttt{ and } B$ (1)

$B \to B \texttt{ or } B$

$B \to L$ (2), (4)

$L \to \texttt{t}$ (3)

$L \to \texttt{f}$ (5)

Derivation

$$B \underbrace{\Rightarrow}_{1} B \texttt{ and } B$$

$$\underbrace{\Rightarrow}_{2} L \texttt{ and } B$$

Example

Grammar

$B \to B$ **and** B (1)

$B \to B$ **or** B

$B \to L$ (2), (4)

$L \to$ **t** (3)

$L \to$ **f** (5)

Derivation

$$B \underbrace{\Rightarrow}_{1} B\ \texttt{and}\ B$$

$$\underbrace{\Rightarrow}_{2} L\ \texttt{and}\ B$$

$$\underbrace{\Rightarrow}_{3} \texttt{t}\ \texttt{and}\ B$$

Example

Grammar

$B \to B$ **and** B (1)

$B \to B$ **or** B

$B \to L$ (2), (4)

$L \to$ **t** (3)

$L \to$ **f** (5)

Derivation

$$B \underbrace{\Rightarrow}_{1} B \texttt{ and } B$$

$$\underbrace{\Rightarrow}_{2} L \texttt{ and } B$$

$$\underbrace{\Rightarrow}_{3} \texttt{t and } B$$

$$\underbrace{\Rightarrow}_{4} \texttt{t and } L$$

Example

Grammar

$B \to B$ **and** B (1)

$B \to B$ **or** B

$B \to L$ (2), (4)

$L \to$ **t** (3)

$L \to$ **f** (5)

Derivation

$$
\begin{aligned}
B \underbrace{\Rightarrow}_{1} & \; B \texttt{ and } B \\
\underbrace{\Rightarrow}_{2} & \; L \texttt{ and } B \\
\underbrace{\Rightarrow}_{3} & \; \texttt{t and } B \\
\underbrace{\Rightarrow}_{4} & \; \texttt{t and } L \\
\underbrace{\Rightarrow}_{5} & \; \texttt{t and f}
\end{aligned}
$$

Thus, $B \Rightarrow^{\star} \texttt{t and f}$

Examples

Example 1

Grammar that generates well-balanced braces.

$$C \to \{\ C\ \}$$
$$C \to CC$$
$$C \to \epsilon$$

Derivation

Build a derivation for $\{\{\}\}\{\}$.

Example 1

> Grammar that generates well-balanced braces.

$$
\begin{aligned}
C &\to \{\ C\ \} \\
C &\to CC \\
C &\to \epsilon
\end{aligned}
$$

Derivation

Build a derivation for $\{\{\}\}\{\}$.

$$\underline{C} \Rightarrow \underline{C}C \Rightarrow \{C\}\underline{C} \Rightarrow \{\underline{C}\}\{C\} \Rightarrow \{\{\underline{C}\}\}\{C\} \Rightarrow \{\{\epsilon\}\}\{\underline{C}\} \Rightarrow \{\{\}\}\{\}$$

Shorthand notation

For grammars

Shorthand notation

Instead of writing $A \to w_1, \ldots, A \to w_n$ can be **abbreviated** as $A \to w_1 \mid \cdots \mid w_n$.

Example

$$C \to \{ \; C \; \}$$
$$C \to CC$$
$$C \to \epsilon$$

can be abbreviated as

$$C \to \{ \; C \; \} \mid CC \mid \epsilon$$

Example 2

Build a grammar from a regex.

Write a CFG that recognizes $L(10^\star 1)$.

Example 2

Build a grammar from a regex.

Write a CFG that recognizes $L(10^{\star}1)$.

$$C \to 1D$$

$$D \to 0D \mid E$$

$$E \to 1$$

Example

Write a CFG that recognizes language $\{0^n 1^n \mid n \geq 0\}$.

Example

Write a CFG that recognizes language $\{0^n 1^n \mid n \geq 0\}$.

Solution

$A \to 0A1$

$A \to \epsilon$

Example

Write a CFG that recognizes language $\{0^n 1^m \mid n \leq m\}$.

Example

Write a CFG that recognizes language $\{0^n 1^m \mid n \leq m\}$.

Solution

$$A \rightarrow 0A1$$

$$A \rightarrow B$$

$$B \rightarrow 1B$$

$$B \rightarrow \epsilon$$

Parse tree examples

Parse tree examples

- CFG's may process a string in any order (not just from left-to-right)

Derive: 8 ÷ 2 × 4

Left-to-right derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derive: 8 ÷ 2 × 4

Left-to-right derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derivation D_1: $(8 \div 2) \times 4 = 16$

$$
\begin{aligned}
\underline{E} &\Rightarrow \underline{E} \times E \\
&\Rightarrow \underline{E} \div E \times E \\
&\Rightarrow \underline{L} \div E \times E \\
&\Rightarrow 8 \div \underline{E} \times E \\
&\Rightarrow 8 \div \underline{L} \times E \\
&\Rightarrow 8 \div 2 \times \underline{E} \\
&\Rightarrow 8 \div 2 \times \underline{L} \\
&\Rightarrow 8 \div 2 \times 4
\end{aligned}
$$

Derive: 8 ÷ 2 × 4

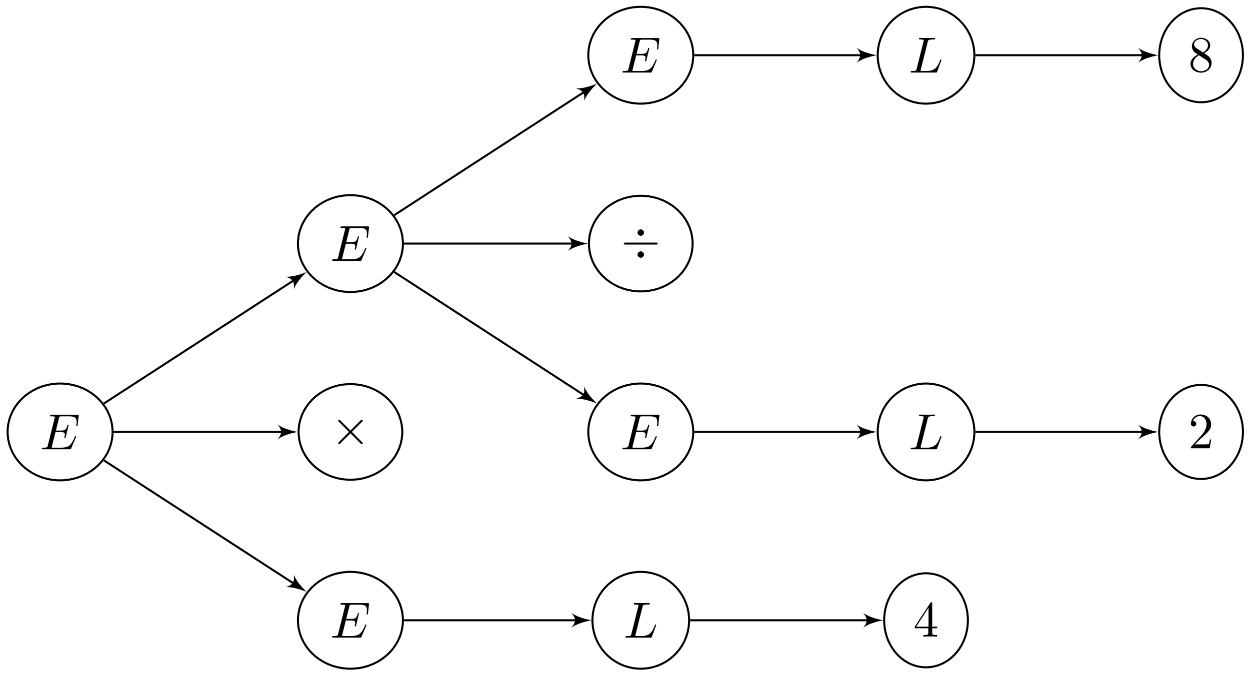
Left-to-right derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derivation D_1: $(8 \div 2) \times 4 = 16$

$$\begin{aligned}
\underline{E} &\Rightarrow \underline{E} \times E \\
&\Rightarrow \underline{E} \div E \times E \\
&\Rightarrow \underline{L} \div E \times E \\
&\Rightarrow 8 \div \underline{E} \times E \\
&\Rightarrow 8 \div \underline{L} \times E \\
&\Rightarrow 8 \div 2 \times \underline{E} \\
&\Rightarrow 8 \div 2 \times \underline{L} \\
&\Rightarrow 8 \div 2 \times 4
\end{aligned}$$

Parse Tree



Derive: $8 \div 2 \times 4$

Right-to-left derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derive: 8 ÷ 2 × 4

Right-to-left derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derivation D_2: $8 \div (2 \times 4) = 1$

$$
\begin{aligned}
\underline{E} &\Rightarrow E \div \underline{E} \\
&\Rightarrow \underline{E} \div E \times E \\
&\Rightarrow \underline{L} \div E \times E \\
&\Rightarrow 8 \div \underline{E} \times E \\
&\Rightarrow 8 \div \underline{L} \times E \\
&\Rightarrow 8 \div 2 \times \underline{E} \\
&\Rightarrow 8 \div 2 \times \underline{L} \\
&\Rightarrow 8 \div 2 \times 4
\end{aligned}
$$

Derive: 8 ÷ 2 × 4

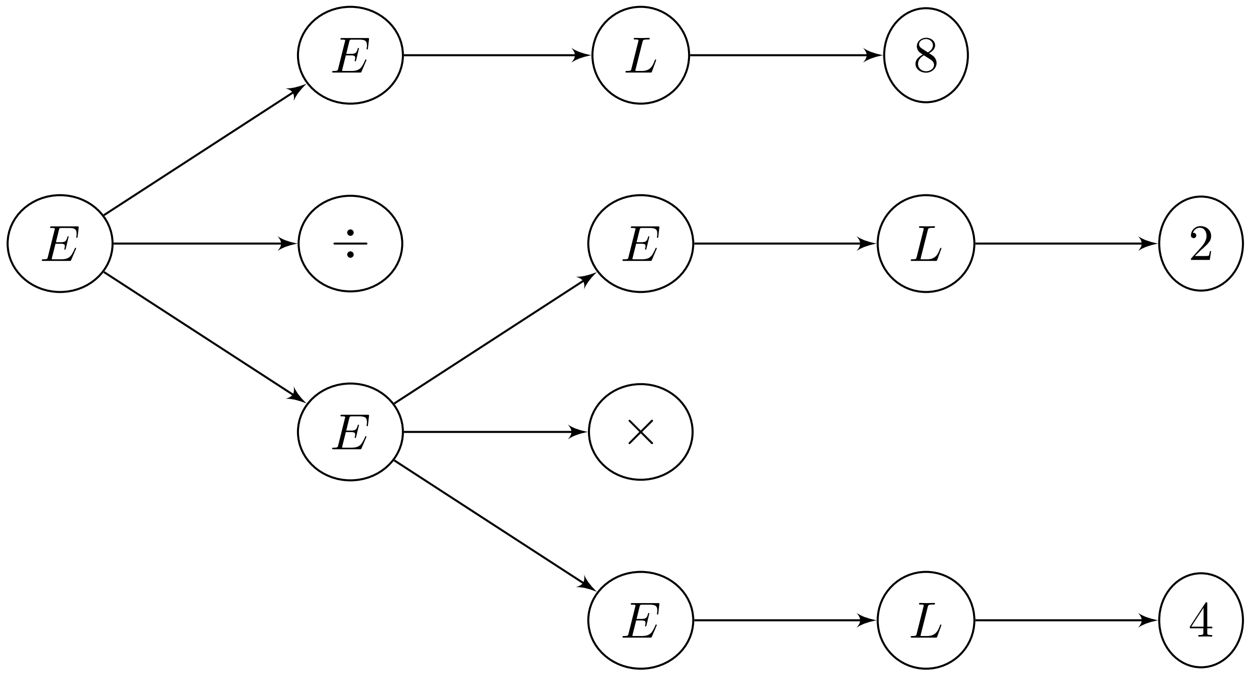
Right-to-left derivation example.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Derivation D_2: $8 \div (2 \times 4) = 1$

$$\begin{aligned} \underline{E} &\Rightarrow E \div \underline{E} \\ &\Rightarrow \underline{E} \div E \times E \\ &\Rightarrow \underline{L} \div E \times E \\ &\Rightarrow 8 \div \underline{E} \times E \\ &\Rightarrow 8 \div \underline{L} \times E \\ &\Rightarrow 8 \div 2 \times \underline{E} \\ &\Rightarrow 8 \div 2 \times \underline{L} \\ &\Rightarrow 8 \div 2 \times 4 \end{aligned}$$

Parse Tree



Ambiguity

$$E \rightarrow E \times E \mid E \div E \mid L$$
$$L \rightarrow 2 \mid 4 \mid 8$$

Admits two different parse trees for the same string!

Formalizing CFGs

Context-free grammar

$G = (V, \Sigma, R, S)$

1. V is a finite set of **variables**
2. Σ is a finite set of **terminals**; Σ is disjoint from V
3. R is a set of rules $V \times V \cup \Sigma$
4. S is the **start variable**; $S \in V$

Generating strings

Yield

A string u yields a string v according to grammar G, notation $u \xRightarrow{G} v$, defined as follows. When there is no ambiguity we may omit the grammar and just write $u \Rightarrow v$.

$$\frac{A \to w \in R \qquad G = (V, \Sigma, R, S)}{uAv \xRightarrow{G} uwv}$$

Generating strings

Derivation

Since, $\overset{G}{\Longrightarrow}$ is a binary relation, we call the reflexive transitive closure a **derivation**, notation $\overset{G}{\Longrightarrow}^{\star}$, defined as follows:

$$\frac{u \overset{G}{\Longrightarrow}^{\star} v \qquad v \overset{G}{\Longrightarrow} w}{u \overset{G}{\Longrightarrow}^{\star} w} \qquad \frac{}{u \overset{G}{\Longrightarrow}^{\star} u}$$

Language of a CFG

Let $G = (V, \Sigma, R, S)$ be a context-free grammar. We define the language of G, notation $L(G)$ bellow.

$$L(G) = \{w \mid S \Rightarrow^\star w\}$$

The language of a CFG consists of every word that can be derived from the start variable where all the letters are terminals.

Context-Free Language (CFL)

Definition. We say that a language L is context-free if there exists a CFG G such that $L(G) = L$

Ambiguity

> Note that we do not formalize parse trees, so we cannot define ambiguity in terms of a parse tree.

Definition

A **leftmost** derivation if at every step the leftmost remaining variable is the one replaced.

Definition 2.7

A string is derived **ambiguously** in context-free grammar G if it has two or more different leftmost derivations. Grammar G is ambiguous if it generates some string ambiguously.

Leftmost/non-leftmost example

Leftmost derivation

$$
\begin{aligned}
\underline{E} &\Rightarrow \underline{E} \times E \\
&\Rightarrow \underline{E} \div E \times E \\
&\Rightarrow \underline{L} \div E \times E \\
&\Rightarrow 8 \div \underline{E} \times E \\
&\Rightarrow 8 \div \underline{L} \times E \\
&\Rightarrow 8 \div 2 \times \underline{E} \\
&\Rightarrow 8 \div 2 \times \underline{L} \\
&\Rightarrow 8 \div 2 \times 4
\end{aligned}
$$

Non-leftmost derivation

$$
\begin{aligned}
\underline{E} &\Rightarrow E \div \underline{E} \\
&\Rightarrow \underline{E} \div E \times E \\
&\Rightarrow \underline{L} \div E \times E \\
&\Rightarrow 8 \div \underline{E} \times E \\
&\Rightarrow 8 \div \underline{L} \times E \\
&\Rightarrow 8 \div 2 \times \underline{E} \\
&\Rightarrow 8 \div 2 \times \underline{L} \\
&\Rightarrow 8 \div 2 \times 4
\end{aligned}
$$

Ambiguous grammar example

Claim: The grammar below is ambiguous.

$$E \rightarrow E \times E \mid E \div E \mid L$$
$$L \rightarrow 2 \mid 4 \mid 8$$

Can we convert D_2 into a leftmost derivation?

$$\begin{aligned} \underline{E} &\Rightarrow E \div \underline{E} \\ &\Rightarrow \underline{E} \div E \times E \\ &\Rightarrow \underline{L} \div E \times E \\ &\Rightarrow 8 \div \underline{E} \times E \\ &\Rightarrow 8 \div \underline{L} \times E \\ &\Rightarrow 8 \div 2 \times \underline{E} \\ &\Rightarrow 8 \div 2 \times \underline{L} \\ &\Rightarrow 8 \div 2 \times 4 \end{aligned}$$

Ambiguous grammar example

Claim: The grammar below is ambiguous.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

Ambiguous grammar example

Claim: The grammar below is ambiguous.

$$E \to E \times E \mid E \div E \mid L$$
$$L \to 2 \mid 4 \mid 8$$

$$\begin{aligned} &(D_1)\\ \underline{E} &\Rightarrow \underline{E} \times E\\ &\Rightarrow \underline{E} \div E \times E\\ &\Rightarrow \underline{L} \div E \times E\\ &\Rightarrow 8 \div \underline{E} \times E\\ &\Rightarrow 8 \div \underline{L} \times E\\ &\Rightarrow 8 \div 2 \times \underline{E}\\ &\Rightarrow 8 \div 2 \times \underline{L}\\ &\Rightarrow 8 \div 2 \times 4 \end{aligned}$$

$$\begin{aligned} &(D_2')\\ \underline{E} &\Rightarrow \underline{E} \div E\\ &\Rightarrow \underline{L} \div E\\ &\Rightarrow 8 \div E\\ &\Rightarrow 8 \div \underline{E} \times E\\ &\Rightarrow 8 \div \underline{L} \times E\\ &\Rightarrow 8 \div 2 \times \underline{E}\\ &\Rightarrow 8 \div 2 \times \underline{L}\\ &\Rightarrow 8 \div 2 \times 4 \end{aligned}$$

Proof. String `8 ÷ 2 × 4` is derived ambiguously, since there are at least two distinct leftmost derivation (see slides before).