### 0.1.1. 7.1 Answer each part TRUE or FALSE

#### 0.1.1.1. a. $2n = O(n)$

by the definition of $O(n)$ we have

$\boldsymbol{L \in O(n) \Leftrightarrow cL \leq n}$

set c to 2 , then we have $\boldsymbol{2n}$, so it must be the case that $2n \in O(n)$

#### 0.1.1.2. b. $n^2 = O(n)$

by the definition, again

$L \in O(n) \Leftrightarrow cL \leq n$

but note that c must be n, thus

$nL \leq n^2$

thus it must be that this case is false

#### 0.1.1.3. c. $n^2 = O(n \log^2 n)$

by the definition of $O(n \log^2 n)$

remember that we can multiply the outside factor for the inside one, thus

$L \in O(\log^2 n^2) \Leftrightarrow L \leq c \log^2 n^2$

but, then, $2^L \leq 2^c n^2$, but it's not the same language anymore, thus it must be the case that $n^2$ does not belong to $O(\log^2 n^2)$, false

#### 0.1.1.4. d. $n \log n = O(n^2)$

by the definition, $n \log n \in O(n^2) \Leftrightarrow n \log n \leq cn^2$ , simplifying

$\log n^2 \leq cn^2$

$n^2 \leq 2^c 2^{n^2}$ again not a constant, as such is false.

#### 0.1.1.5. e . $3^n = 2^{O(n)}$

$3^n \in 2^{O(n)} \Leftrightarrow 3^n \leq 2^{cn}$

#### 0.1.1.6. f . $2^{2^n} = O(2^{2^n})$

again, by definition

$2^{2^n} \in O(2^{2^n}) \Leftrightarrow 2^{2^n} \leq c2^{2^n}$

keep applying log

$2^n \leq \log c + 2^n$

repeat, one more time

$2^n - 2^n \leq \log c$

$0 \leq \log c$

so there's a constant $\geq 1$ so TRUE.

### 0.1.2. 7.2 Answer each part TRUE or FALSE

#### 0.1.2.1. a. $n = o(2n)$

based on the definition of $o(n)$

$L \in o(n) \Leftrightarrow n < cn$

if $c = 1$ then it must be the case that $n \in O(n)$, but then we can always find a constant that would make this false, they have the same growth rate, thus it must be false.

#### 0.1.2.2. b . $2n = o(n^2)$

it must be the case that $n \in o(n^2) \Leftrightarrow n < cn^2$ thus it must also be the case that $2n < n^2$ for any c, thus $2n \in o(n^2)$

#### 0.1.2.3. c. $2^n = o(3^n)$

again, based on the definition

$2^n \in o(3^n) \Leftrightarrow 2^n < c3^n$

it must be the case as well, thus TRUE.

#### 0.1.2.4. d. $1 = o(n)$

by the definition $1 \in O(n)$,that is a linear function always grow faster than a constant, TRUE.

#### 0.1.2.5. e n = o(log n)

by the definition

$\boldsymbol{n \in o(\log n) \Leftrightarrow n < c \log n}$

but $\boldsymbol{\log n \leq n}$, thus it must be false.

#### 0.1.2.6. f. $\boldsymbol{1 = o(\frac{1}{n})}$

this one must be false, due to $\boldsymbol{0 < n < 1}$ being the only case for this to be true.

#### 0.1.2.7. 7.4 Fill out the table described in the polynomial time algorithm for context free language

recognition from Theorem 7.16 for string w = baba and CFG G:

$S \to RT$<br>
$R \to TR \mid a$<br>
$T \to TR \mid b$

| | **1** | **2** | **3** | **4** | **4** |
|---|---|---|---|---|---|
| **1** | T | | | | |
| **2** | | R | | | |
| **3** | | | T | | |
| **4** | | | | R | |

7.6 Show that P is closed under union, concatenation, and complement

Consider the following TM

```
$M =$ "On input $\langle M_1, M_2, a \rangle$ where $M_1$is a TM and $M_2$ is a TM and a is a string
    1. Run $M_1$ on input a
    2. Run $M_2$ on input b
    3. If any of them accept, accept, else reject
```

Due to each machine taking at most polynomial time due to $L \in P$ it must be the case that step 1 takes $O(t^{k_i})$ and step 2 $O(t^{k_j})$ for some $(k_j, k_i) \in N \times N$ and the sum of two polynomials is a polynomial, thus it must be the case that the union of two $P$ languages is also in P.

```
$M =$ "On input $\langle M_1, M_2, a \rangle$ where $M_1$ is a TM and $M_2$ and  is a TM, a is a string
    1. For $k = 0$ to $|a|$
     2. Partition a as $(l, r)$ where $l$ start at position 0 of a and end at $k - 1$ and r start at $k$ and end at |a| position
      3. Run $M_1$ in $l$ and $M_2$ in $r$
       4. If $M_1, M_2$ accepts then accept
    5. Reject
```

Note that we can partition a in $a^2$ splits, each partition can have at most size $|a|$ and two exists, each partition takes $O(t^{k_j})$ for $M_1$ and $O(t^{k_i})$ for $M_2$ for $(t_k, t_i) \in N \times N$, so as seen before the sum of polynomials is a polynomial, and the multiplication of a constant by a polynomial is a polynomail, $a^2$, in this case.

As such it must be the case that concatenation of languages is in $P$

```
$M =$ "On input $\langle M_1, a \rangle$ where $M_1$ is a TM and a a string
    1. Run $M_1$ on a
    2. Reject if accepts and accept if rejects
```

Note that it must be the case that the time it takes is $O(t^k)$ where $k \in N$, as such it take polynomial time, consequently the extra step is constant time, inverting the answer, thus negation of a language in P belongs to P as well

### 0.1.3. 7.7 Show that NP is closed under union and concatenation

```
$M =$ "On input $\langle M_1, M_2, a \rangle$ where $M_1$  is a TM, $M_2$ is a TM and a is a string
    1. Run $M_1$ with a as input
    2. Run $M_2$ with a as input
    3. Accept if one of them accept, otherwise reject"
```

Note that takes polynomial time for each machine, and the sum of polynomials is a poylnomial consequently it must be the case that the union of two languages in NP is in NP

```
$M =$ "On input $\langle M_1, M_2, a \rangle$ where $M_1, M_2, a$ is a TM, TM and string respectively
    1. Partition a in a non deterministic manner
     2. For each partition run $M_1$ on the left substring and $M_2$ on the right substring
     3. Accept if both accept
    4. Reject
```

### 0.1.4. 7.8 Let CONNECTED = $\{< G > \mid \text{G is a connected undirected graph}\}$. Analyze the algorithm given on page 185 to show that this language is in P.

```
$M$ = "On input $\langle G \rangle$ where G is the encoding of a Graph
    1. Select the first node of G and mark it
    2. Repeat the following stage until no new nodes are marked:
    3.  For each node in G, mark it if its attached by an edge to a node that is already marked
    4. Scan all the nodes of G to determine
       whether they all are marked. If they are, accept; otherwise, reject"
```

- the first step takes $O(1)$ so constant time
- the second step takes $O(G^3)$ so linear time
- the last step takes $O(G)$ so linear time

so $O(G^2)$ is the final time the second step has the following time because in the worst case a node might compare with $n - 1$ nodes and n nodes exist, so $O(n^3)$

### 0.1.5. 7.9 A triangle in an undirected graph is a 3-clique. Show that TRIANGLE $\in$ P, where TRIANGLE = $\{\langle G \rangle \mid G \text{ contains a triangle}\}$