1 Problem 5.1

$L_1 \cup L_2$ Будуємо МТ, яка буде запускати вирішувачі мов L_1, L_2 паралельно, та повертати reject якщо обидві вирішувачі не прийняли слово, інакше — accept. Дана МТ вирішує об'єднання та працює за поліноміальний час.

$L_1 \cap L_2$ Аналогічно. Запускаємо МТ, якщо обидва вирішувачі повертають 1, то також повертаємо 1, інакше — 0. Працюємо за поліноміальний час та вирішуємо перетин.

$L_1 L_2$ Для конкатенації наша МТ повинна розглядати всі префікси та постфікси слова, не забуваючи і про порожні. Маємо скінченну кількість таких префіксів та постфіксів, передаємо їх на вхід до вирішувачів L_1 та L_2. Результат повертаємо за аналогією до попереднього пункту. Працює за поліноміальний час та вирішує конкатенацію.

L_1^R Наша МТ пише слово у зворотньому порядку, передає на вхід до вирішувача L_1 та повертає результат вирішувача. Працюємо за поліноміальний час.

L_1^* Аналогічно до пункту з конкатенацію, окрім того, що наша МТ повинна послідовно розділяти слово на 0, 1, 2, ... частин та перевіряти кожну на вирішувачі L_1. Також потрібно зауважити, що будемо мати скінченну кількість таких розділів слова.

2 Problem 5.2

a Складність алгоритму факторизації — $O(n^2)$. Зчитування слова — знехтовно мале. Отже $O(n^2)$, $L_{UFACTOR} \in P$.

b Алгоритм Дейкстри — $O(n^2) \Rightarrow L_{PATH} \in P$.

c Зводимо задачу до пошуку найкоротшого шляху (алг. Дейкстри). При запуску із кожної вершини графа маємо складність $O(n^3) \Rightarrow L_{CONNECTED} \in P$.

3 Problem 5.5

$L_1 = HALT$. $L_2 = L_1 \cup \{0, 1\}$. $L_2^* \in P$. $L_2 \notin P$.

4 Problem 5.8

Нехай є така мова $L \in NP$. Тоді $\forall L_1 \in NP : L_1 \leq_D L$. $\Rightarrow L_1 \in coNP \Rightarrow NP \subseteq coNP$.

$\forall L_2 \in coNP : L_2 \leq_p L \Rightarrow L_2 \in NP \Rightarrow coNP \subseteq NP$.
$\Rightarrow coNP = NP$.

5 Problem 5.10

Побудуємо зведення $L_2 \leq_p L_1$. Нехай існує така недетермінована МТ M, що розв'язує мову L_2. Якщо $M(x) = 1$, то x без префікса 1 належить мові L_1. Якщо $M(x) = 0$, то x без префікса 1 не належить L_1.

6 Problem 5.12

a Припустимо, що $\exists L_{BigCycle} \leq_p L_{HAMPATH} \in NPC$. Домножимо граф на n вершин без ребер, а потім припустимо, що МТ, яка розпізнає $L_{BigCycle}$ поверне 1. Тоді визідний граф буде мати цикл із n вершин. $\Rightarrow \in NPC$.

b $L_{SmallCycle} = coL_{BigCycle} \Rightarrow L_{SmallCycle} \in NP \Rightarrow L_{SmallCycle} \in coNP.$

c Зведемо $SAT \to L_{TAUTOLOGY}$.

$L_{TAUTOLOGY}(\overline{\varphi}) = 1 \Rightarrow SAT(\varphi) = 0$
$L_{TAUTOLOGY}(\overline{\varphi}) = 0 \Rightarrow SAT(\varphi) = 1$
$L_{TAUTOLOGY} \in coNP$.

d Зводимо до SAT додаючи ще одну модель:

$$\varphi(x_1, x_2, \dots, x_n) \to \overline{\varphi}(x_1, x_2, \dots, x_n, x_{n+1}) \wedge (x_1 \wedge x_2 \wedge \dots \wedge x_{n+1})$$

e Будуємо зведення до $L_{HAMPATH}$. Покладемо $k = n - 1$. Тоді шлях існуватиме, якщо $L_{restrPATH}$ поверне 1, інакше ні.

f $\forall L_1 \in NP : L_1 \leq_p HALT$.
Будуємо ДМТ M, яка буде працювати за поліноміальний час. Модифікуємо її так, що вона буде приймати лише якесь вхідне слово. $\Rightarrow HALT \in NP$-hard.