1 Алгоритм решения

1. **Конкатенация** строк в новую: $S = P + \text{“\$“} + T$
2. Строится **Z-функция** на S, где $S[i]$ равный ‘?’ удовлетворяет условию на совпадение для любого символа.
3. Начиная с $i = |P| + 1$ определяем **наличие вхождения** на $i - |P|$ индексе при соблюдении $Z[i] = |P|$.

2 Корректность

Лемма 1. *Быстрее чем за $O(|P| + |T|)$ сделать нельзя.*

Доказательство. Следует из того, что каждый элемент обоих P и T должен быть учтен в алгоритме хотя бы раз, иначе элемент не участвует в проверке на совпадение. □

Лемма 2. *Использование сжимающих представлений над T для такой задачи избыточно.*

Доказательство. Пример: $P =$ “??????????“, $|T| = 100.000$ состоит из уникальных комбинаций алфавита длинной в 10. Длинна строки всех комбинаций равна 10^{27}, но достаточно любой подстроки учитывая $|T|$.

При использовании префиксного/суффиксного дерева на T, помимо построения за $O(|T|)$для того, чтобы учесть ‘?‘ нужно идти во все ноды `children`, что рекурсивно вырастает в то, что все равно на поиск вхождений мы проходим все ноды дерева, то есть 100.000. Не быстрее предложенного алгоритма. □

Теорема 1. *Использование модификации Z-функции достаточно.*

Доказательство. Следует из нижней оценки на скорость алгоритма и скорости алгоритма модифицированной Z-функции (доказанной на лекции). □

3 Временная сложность

Каждый из пунктов алгоритма решения занимает $O(|P| + |T|) \to O(|P| + |T|)$ - общая сложность по времени.

4 Затраты по памяти

Первые два пункта $O(|P| + |T|)$ по памяти, последний $O(1) \to O(|P| + |T|)$ - общая сложность по дополнительной (по отношению к вводу - P и T) памяти.