Karl Bringmann and Marvin Künnemann **Winter 2017/18**

Exercises for Fine-Grained Complexity Theory

www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter17/fine-complexity/

Presence exercise sheet

Exercise 1 In the lecture we introduced the *Orthogonal Vectors Hypothesis*:

OVH: Given two sets $A, B \subseteq \{0,1\}^d$ such that $|A| = |B| = n$. There is no algorithm running in time $O(n^{2-\varepsilon} \cdot \text{poly}(d))$ (for any $\varepsilon > 0$) which decides whether there exists $a \in A, b \in B$ such that a and b are orthogonal.

a) Consider the following variant **OVH′** of **OVH**:

   OVH′: Given a set $A \subseteq \{0,1\}^d$ such that $|A| = n$. There is no algorithm running in time $O(n^{2-\varepsilon} \cdot \text{poly}(d))$ (for any $\varepsilon > 0$) which decides whether there exist $a, a' \in A$ such that a and a' are orthogonal.

   Prove that **OVH′** and **OVH** are equivalent.

b) Consider the problem of finding the maximum inner product of elements of two sets:

   MaxInnerProduct: Given two sets $A, B \subseteq \mathbb{R}^d$ such that $|A| = |B| = n$, compute the maximum

$$\max \{\langle a, b\rangle \mid a \in A, b \in B\},$$

   where "$\langle \cdot, \cdot \rangle$" denotes the standard inner product of $\mathbb{R}^d$.

   Prove that there is no algorithm running in time $O(n^{2-\varepsilon} \cdot \text{poly}(d))$ (for any $\varepsilon > 0$) for **MaxInnerProduct** unless **OVH** fails.

Exercise 2 In this exercise, we will fill in the missing part of the hardness result for **LCS**. Recall the problem definition for **LCS**:

Longest Common Subsequence (LCS): Given two strings A, B over some alphabet Σ, where $|A| = |B| = n$, compute the length $L = L(A, B)$ of the longest string $C = c_1 c_2 \dots c_L$, that is a subsequence of both A and B, i.e., the longest string that suffices

$$A = \star_1 \, c_1 \, \star_2 \, c_2 \, \star_3 \, \dots \, \star_L \, c_L \, \star_{L+1}$$

and

$$B = *_1 \, c_1 \, *_2 \, c_2 \, *_3 \, \dots \, *_L \, c_L \, *_{L+1},$$

for some arbitrary strings $\star_i, *_i \in \Sigma^*$ (which may be the empty strings).

Further, recall that we already proved the following lemma, providing us with gadgets to encode vectors as strings.

Lemma 1 (Vector Gadgets)**.** *There are functions $V_A, V_B : \{0,1\}^d \to \{0,1,2\}^{3d^2}$, computable in time $O(d^2)$, such that*

$$\forall a, b \in \{0,1\}^d : \quad L(V_A(a), V_B(b)) = \alpha_d - 2 \cdot \langle a, b\rangle \begin{cases} = \alpha_d, & \text{if } a \perp b, \\ \leq \alpha_d - 2, & \text{otherwise,} \end{cases}$$

where $\alpha_d := 3d^2 - d$ and "$\langle\cdot,\cdot\rangle$" again denotes the standard inner product of $\mathbb{R}^d$.

As we saw in the lecture, for these vector gadget to be useful, we need to ensure that the LCS of gadgets of two non-orthogonal vectors can only attain a single value (dependent on d), instead of the range of values that is possible with the current gadgets. In particular, we need the following *normalized* vector gadgets.

Lemma 2 (Normalized Vector Gadgets)**.** *There are functions $N_A, N_B : \{0,1\}^d \to \{0,1,2,3\}^{9(d+1)^2}$, computable in time $O(d^2)$, such that*

$$\forall a, b \in \{0,1\}^d : \quad L(N_A(a), N_B(b)) = \begin{cases} \beta_d, & \text{if } a \perp b, \\ \beta_d - 2, & \text{otherwise,} \end{cases}$$

for $\beta_d := 3(d+1)^2 + \alpha_{d+1}$.

(In the lecture we used $\beta_d := 3(d+1)^2 + \alpha_{d+1} - 2$, thus resulting in a slightly different formulation of this lemma. However, in general, neither the exact value of β_d nor the exact length of the generated strings is important. What is important is that the lengths of the LCS of gadgets of orthogonal and non-orthogonal vectors are exactly two values – one value for orthogonal vectors and a different, smaller one for non-orthogonal vectors – and that these gadgets can be constructed in poly(d) *time.)*

Prove this last Lemma 2.

*(Hint. Recall the reduction from **OVH** to **OVH′** from exercise 1a). Further, recall how we used these normalized vector gadgets in the final construction.)*