Problem 3.1

ZSP $\subseteq$ NP$\cap$coNP

For each language $L \in$ **ZSP** (so there exists a *Zosia Samosia* machine M that recognises L and an oracle O for which M recognises L in time bounded by polynomial $p(n)$) we can show two **NP** machines (one recognising L, the other $\overline{L}$) that simulate M, nondeterministically guessing answers of the oracle (remembering all queries and checking if some query has repeated – then they repeat the old answer) and stopping (rejecting) after $p(|w|)$ simulated steps.

Of course, if the simulated machine accepts or rejects, the simulating machine recognising L does the same, and the one recognising $\overline{L}$ does the contrary.

The proof of correctness is as follows. For each word w the machine M with oracle O reaches final state (rejects or accepts) in at most $p(n)$ steps. That means that some runs of both **NP** machines also reach them (and obviously give the correct answers). As the *Zosia Samosia* recognises the same language for each oracle, no run of the **NP** machines will give incorrect answer – and that's enough for the correctness.

It is also easy to see that their working time is polynomial – each run simulates polynomial number of steps and cost of simulating each step in bounded by some polynomial (the most expensive step is query to the oracle – we need to compare it with all previous queries – but the sum of lengths of all queries doesn't exceed $p(|w|)$).

NP$\cap$coNP $\subseteq$ ZSP

For each language $L \in$ **NP$\cap$coNP** (so for each word there exists a **witness** for it either being in or not being in the language, which size is bounded by some polynomial $p(n)$ and can be verified in polynomial time) we will show a *Zosia Samosia* machine and an oracle O with which it recognises L in polynomial time.

Let's consider specific set of the mentioned witnesses over some alphabet A. We will call a witness *positive* if it proves that a word **is** in the language. We will call it *negative* if it proves that it **is not** in the language.

The idea is simple: the *Zosia Samosia* machine will construct a witness symbol by symbol, using hints from the oracle whether there exists a witness beginning with the given sequence. It will believe in the oracle's oracles until it creates a correct witness or until it discovers some inconsistency (the size of witness exceeds $p(|w|)$ or the oracle states that there is no symbol that could go next) – if that happens, it will become a non-believer. It will start from the beginning, won't ask oracle for anything anymore and do all the work by itself (probably in exponential time) checking all possible witnesses.

The oracle O (for which the machine works in polynomial time) accepts word $w0v$ iff for a word w there exists a negative witness (of length less than $p(|w|)$) with a prefix equal to v. Similarly, it will accept a word $w1v$ iff for a word w there exists a positive witness that begins with v. Let's assume 0 and 1 are special characters, not occuring in the alphabets of words nor witnesses. We won't ask oracle for any other words than described above, so it can reject them.

The *Zosia Samosia*'s algorithm is now quite obvious. For input word w, it sends oracle words $w0$ and $w1$ (with the oracle O we would already know whether $w \in L$) and temporarily believes in the answer (of course, if exactly one of the answers is positive). Then it guesses

consecutive letters of the witness. Believing there exists a witness with prefix v, it checks whether v is a correct witness. If so, it answers; otherwise for each letter a from the witnesses' alphabet it asks the oracle about the prefix va. If none are accepted by the oracle or the length of the prefix exceeds $p(|w|)$, it stops believing, otherwise it extends the prefix it believes in.

The correctness follows immediately from the correctness of the witnesses (the machine answers only when it finds a witness). The machine makes polynomial number of steps believing in the oracle (at most $|A| \cdot p(|w|)$ prefixes will be considered and the verification of the witness takes polynomial number of steps). It's obvious that with the oracle O the working time will be polynomial and for other oracles the machine will eventually answer (in the worst case, after verifying $|A|^{p(|w|)}$ potential witnesses after becoming a non-believer).

Problem 3.2

General idea

If the answer is positive, there are four cases:

1. There is a triangle in A.
2. There is a triangle in B.
3. There is a triangle with two nodes in A and one node in B.
4. There is a triangle with one node in A and two nodes in B.

The first two cases are trivial, as we assume that Alice and Bob are honest, so in case 1 Alice can answer by herself and in case 2 Bob can send one bit with the information that there is a triangle.

Let's solve case 3. Alice wants to decide whether there is any vertex in B that creates a triangle with some pair from A. There is a specific set of vertices S_A fulfilling this condition, so Alice can calculate this set and check whether S_A and B are disjoint (as in the hint – everything works, because Merlin still wants to convince them that the sets are disjoint, i.e. there is no such triangle).

Case 4 is symmetrical – Bob calculates set S_B and they check if S_B and A are disjoint.

So the protocol will consist of one bit sent from Bob to Alice (whether there is a triangle in B), then the exact protocol from the hint (with Alice using S_A instead of A while checking), and again protocol from the hint, but with Bob using S_B instead of B (and sending values of the appropriate polynomial).

If there is no triangle in $A \cup B$, there exists a message from Merlin such that Alice always rejects (as there exist appropriate messages such that Alice will accept sets A and S_B as disjoint and B and S_A as disjoint).

If there is a triangle from case 3, there is $\geq \frac{1}{2}$ probability that Alice will accept, as it's equivalent to her (correctly) recognising that S_A and B are **not** disjoint. If it's case 4, there is again $\geq \frac{1}{2}$ probability of accepting (for the same reason). If both cases (3 and 4) are true, the probability of accepting is only higher.

Calculating S_A and S_B

We will talk about the set of vertices S_A that create triangles with some pair from A (for Bob everything is symmetrical, so we will focus on Alice). It is obvious that it is well defined (it is a set of these vertices that have edges to at least two vertices from A). It is also obvious that if any of these vertices are in B, there is a triangle with two nodes from A and one node from B, and otherwise – there is no such triangle.

It's easy to calculate this set in polynomial time – for each vertex v in V count number of edges $(v, u) \in E$ such that $u \in A$ and if it's greater or equal to 2, add it to S_A.

Proof of the hint

Text in gray is basically content of the hint

We assume that $V = \{1, ..., n\}$ and denote $k = \sqrt{n}$ (we assume that $k \in \mathbb{Z}$ for simplicity), p is the smallest prime number such that $p \geq 4n$. $Q_1, ..., Q_k$ and $R_1, ..., R_k$ are polynomials in $\mathbb{Z}_p\big[x\big]$ of degree less than k, such that for all $i, j \in 1, ..., k$:

$$Q_i(j) = \begin{cases} 0, & \text{if } (i-1)k + j \notin A \\ 1, & \text{if } (i-1)k + j \in A \end{cases}$$

$$R_i(j) = \begin{cases} 0, & \text{if } (i-1)k + j \notin B \\ 1, & \text{if } (i-1)k + j \in B \end{cases}$$

To cover all cases, let's say that if $\sqrt{n} \notin \mathbb{Z}$, everyone (Alice, Bob and Merlin) adds „artificial" vertices $n+1, n+2, ...\big\lceil\sqrt{n}\big\rceil^2$ with no edges and not beloging to A nor B. The answer remains the same, while the probability of giving the correct answer shouldn't suffer too much (we only add $O(\sqrt{n})$ vertices).

We defined values of polynomials Q_i and R_i in k distinct points and want them to be of degree at most $k-1$. That means we have correctly defined them and there is exactly one set of polynomials fulfilling these conditions. What's more, they can be calculated in polynomial time (we can find their coefficients; it's a known problem of polynomial interpolation), so Alice can calculate all Q_i, Bob can calculate all R_i and Merlin, of course, can do both.

The protocol is as follows:

1. Merlin sends to Alice coefficients of a polynomial $P \in \mathbb{Z}_p\,[x]$ of degree at most $2(k-1)$.
2. Bob draws $l \in \mathbb{Z}_p$ uniformly at random.
3. Bob sends to Alice l and $R_i(l)$ for all $i \in \{1, ..., k\}$.
4. Alice accepts if $P(j) = 0$ for all $j \in \{1, ..., k\}$ and $P(l) = \sum_{i=1}^{k} Q_i(l) \cdot R_i(l)$; otherwise she rejects.

Let's denote $T(l) = \sum_{i=1}^{k} Q_i(l) \cdot R_i(l)$. It is a polynomial of degree at most $2(k-1)$, as each $Q_i \cdot R_i$ is a polynomial of degree at most $2(k-1)$ and T is a sum of k such polynomials.

If $A \cap B = \varnothing$, there obviously exists a message from Merlin such that Alice always accepts (says that the sets are disjoint) – Merlin can send coefficients of T. As $A \cap B = \varnothing$, all $T(j) = 0$ for $j \in \{1, ..., k\}$ (for arguments $l \in \{1, ..., k\}$ all values of $Q_i(l)$ and $R_i(l)$ denote belonging of some node to, respectively, A and B; there is no node in both, so all $Q_i(l) \cdot R_i(l) = 0$, so the sum $T(l)$ is also equal to 0). And, of course, $T(l) = \sum_{i=1}^{k} Q_i(l) \cdot R_i(l)$, as that's its definition.

If $A \cap B \neq \varnothing$, if Merlin sends T, Alice will surely reject (say that the sets are not disjoint), because $T(j) \neq 0$ for some $j \in \{1, ..., k\}$. There is some node in $A \cap B$, let's write its number as $(a-1)k + b$. Then we have $Q_a(b) = R_a(b) = 1$, so $Q_a(b) \cdot R_a(b) = 1$, thus $T(b) > 0$ (as each component of the sum $T(b)$ is either 0 or 1, there are k components and $k < p$).

What happens when Merlin sends some other polynomial P? Then $P(x) - T(x)$ is a non-zero polynomial of degree at most $2(k-1)$, so it takes value 0 for at most $2(k-1)$ arguments. That means that, no matter what polynomial Merlin sends, Bob has chance at most $\frac{2(k-1)}{p}$ for drawing number l for which the values of T and P are equal.

Alice calculates value of $T(l)$ and compares it with $P(l)$. That means that the chance of Alice rejecting in this case ($A \cap B \neq \varnothing$) is at least $1 - \frac{2(k-1)}{p} \geq 1 - \frac{2\sqrt{n}}{4n} = 1 - \frac{1}{2\sqrt{n}}$, which is obviously $\geq \frac{1}{2}$ for any $n \geq 1$.

Lengths of messages

Merlin sends coefficients of two polynomials over $\mathbb{Z}_p$ of degree at most $2(k-1)$ to Alice, so the number of sent bits is at most $2 \cdot 2(k-1) \cdot \log p = O(\sqrt{n} \log n)$.

Bob draws two random numbers l_1, l_2 from $\{1, ..., p\}$ (tosses $2 \log n = O(\log n)$ coins) and sends them to Alice, along with $2k$ values from $\{1, ..., p\}$ and one additional bit (whether there is a triangle in B), so he sends at most $(2k+2) \log p + 1 = O(\sqrt{n} \log n)$ bits.