Proof. It is immediate from construction, since we have not removed any transitions from Σ within any Q_i, that for any $Q_i \in P$ holds $Q_i.w = q_l^i$, since w carefully synchronizes $\mathcal{A}$ and each Q_i on Σ induces an isomorphic copy of $\mathcal{A}$. So we have, that $\{q_l^1, q_l^2, .., q_l^{|u|+1}\} \subseteq P.w$. To prove that $P.w \subseteq \{q_l^1, q_l^2, .., q_l^{|u|+1}\} \subseteq P.w$ it suffices to notice, that, from fact 2 automaton $\mathcal{B}$ is deterministic and for any prefix w' of w if $Q.w' = \{q_{k_1}, .., q_{k_s}\}$, then $Q_i.w' = \{q_{k_1}^i, .., q_{k_s}^i\}$. □

Lemma 2. *There exist an algorithm with $O(|P||w|)$ time complexity and $O(|P||w|)$ space complexity which computes a partition of P on sets Q_1, Q_2, ..., $Q_{|u|+1}$.*

Proof. We describe a desired algorithm. Suppose we have an array with $|P|$ columns and $|w| + 1$ rows. Put every element of P in a different column of a first row. Then we fill the i-th row by taking a state from the $(i-1)$-th row of the corresponding column and applying to it the i-th letter of a word w until the end of the row. After this procedure, from lemma 1, the last row contains only the states from the set $\{q_l^1, q_l^2, .., q_l^{|u|+1}\}$. We can now compute each Q_i by taking those states from the first row that lie in the same columns as the state q_l^i. □

With these two lemmas we are ready to present a decryption method:

1. using Lemma 1 compute the set $\{q_l^1, q_l^2, .., q_l^{|u|+1}\}$,
2. using Lemma 2 compute the partition of P on sets Q_1, Q_2, ..., $Q_{|u|+1}$,
3. for every transition $x \in \{0, 1\}$ in $\mathcal{B}$ if x joins a states from different sets, say Q_i and Q_j, then join q_l^i and q_l^j with transition x, otherwise remove the transition.

Observe, that after applying that procedure to the ciphertext $\mathcal{B}$ we end up with a graph that was our plaintext, what can be concluded directly from the encryption procedure. In general one can decipher the message only by knowing any carefully synchronizing word for $\mathcal{A}$ or computing every possible induced subautomaton isomorphic to $\mathcal{A}$.

5 Extensions

As the ciphertext which is a result of our encryption method consists of n copies of isomorphic automaton with added transitions between those copies one can think of more "sophisticated" method of creating a ciphertext. As mentioned in the previous section a potential attacker can decipher the message computing every possible induced subautomaton isomorphic to a public key. However, the problem of determining for two given graphs say G and H, whether G has a copy of H as an induced subgraph is NP-complete [6]. In this section we present two lemmas that can be used to obfuscate the ciphertext even more. The first one involves adding the state to the public key and the second one adding arbitrary number of a-clusters to the ciphertext.