Task sheet 8

Task 50. Show that if compression-based algorithm (for word equations) at some point performs a a-blocks compression (and at least one block is replaced), then the word compressed to a from the initial instance is primitive.

For simplicity you may consider the algorithm running on a given word, with no variables.

Task 51. RSLP (run-length SLP) is an SLP in which we allow rules $X \to Y^k$, where k is a natral number. The rule size is considered constant.

Show that a compression-based algorithm for one-variable word equation reports a solution defind using an RSLP of size (and height) $\mathcal{O}(1)$ in $\mathcal{O}(1)$ rounds.

Task 52. Show that a one-variable word equations, whose constant words between the variables are defined using SLPs, can be solved in polynomial time.

This should be easy with compression-based algorithm (also tor RSLPs instead of SLPs), for word-combinatorics based one you need to use some (known) algorithms: pattern matching of SLP-defined pattern inside SLP-defined word can be done ia polynomial time. Computation of all primitive squares prefixing an SLP can be done in polynomial time.

Task 53. Show how to solve (in polynomial time) a 1-variable word equation which uses (inside constant words) expression of a form $(u^i u' v)^j$, where i, j are *parameters*, i.e. we want to have a description of all solutions, for all possible values of i, j. Here u' is a prefix of u.

If this helps, you can assume that $(u^i u' v)^j$ is a prefix of A_0.

Again, should be rather easy using compression-based approach. Requires some not-so difficult claims on primitivity for word-combinatorics based