Admissible Map

Problem author and developer: Ilya Zban

Let's call any string of shape "RLRL...RL" trivial.

Lemma: any non-trivial string has at most one constitution as an admissible map.

Proof: let's consider string $s_0 s_1 \dots s_{|s|-1}$, and i be a first even number such that $s_i s_{i+1} \neq$"RL". There can be a few cases:

- s_i="L". String s doesn't have a constitution as an admissible map, as either the i-th symbol will be on the left edge of the matrix (and its edge isn't directed to any other cell), or it points to the left, to "RL" so there are two incoming edges to neighboring "L" (or it points out of the matrix).
- s_i="U". String s doesn't have a constitution as an admissible map, as either the i-th symbol will be on the upper edge of the matrix (and its edge isn't directed to any other cell), or it points to some pair "RL", that should be matched to each other.
- s_i="R". As $s_{i+1} \neq$ "L", the only incoming edge to i-th cell can be "U". And we can notice that it can only be the first "U" in the string, as otherwise the first "U" would point either outside of the matrix or to some "RL" pair. So, if s_j is the leftmost "U", string s can only be constituted as $\frac{|s|}{j-i} \times j - i$ map.
- s_i="D". Let's say that $k = i$ if $s_{i+1} \neq$ "L", and otherwise take k as maximal number such that $s_{i+1} \dots s_k =$"LLL..L". Substring $s_i \dots s_k$ should be in same row of matrix, and as $s_{k+1} \neq$"L", by same argument as above we can see that the first "U" in the string should point to s_k. So, string s can only be constituted as $\frac{|s|}{j-k} \times j - k$ map.

So, the shape $n \times m$ of a matrix for any non-trivial substring $s_l s_{l+1} \dots s_r$ is determined only by the position of the first "U" after l. We can compute an array m_l meaning that any substring $s_l s_{l+1} \dots s_r$ should be constituted as $\frac{r-l+1}{m_l} \times m_l$ matrix. This array can be computed in linear time directly from proof.

Using m_l we can iterate over all $r = l + t \cdot m_l$ for all t, and check all substrings $s_l s_{l+1} \dots s_r$. We need to be able to quickly determine if substring $s_i \dots s_{i+m_l-1}$ can be a top, middle or a bottom row of constituted matrix. It can be done in $\mathcal{O}(1)$ time.

Let's consider the case of the middle row (other cases are similar). We want to check that no edge from $s_i \dots s_{i+m_l-1}$ goes outside of the matrix and that each cell has exactly one incoming edge. First, we need to check that $s_i \neq$"L" and $s_{i+m_l-1} \neq$"R". Conditions on incoming edges can be tested using hashes. We choose a hash base x, and build 4 arrays $a^c_0 \dots a^c_{|s|-1}$ (c in "ULDR") such that $a^U_j = x^j$ if $s_j =$"U", $a^D_j = x^j$ if $s_j =$"D", $a^R_j = x^{j+1}$ if $s_j =$"R" and $a^L_j = x^{j-1}$ if $s_j =$"L", and all other values are zero. Then we can see that each cell from $s_i \dots s_{i+m_l-1}$ has one incoming edge if $x^{m_l} \sum_{t=i-m}^{i-1} a^D_t + \sum_{t=i}^{i+l-1} (a^L_t + a^R_t) + x^{-m_l} \sum_{t=i+m}^{i+2m-1} a^U_t = \sum_{t=i}^{i+l-1} x^t$. This check can be done in constant time using a precomputed array of prefix sums.

Using these checks we can iterate over all non-trivial substrings, test them and add missed trivial strings. This solution works in $\mathcal{O}(s^2)$.

We can further notice that for each l we iterate over all possible $\frac{|s|}{m_l}$ possible r-s with step m_l. We can count all l that have both the same m_l and $l \mod m_l$ together, as we just do a lot of duplicated work in that case. This optimization gives us a very fast solution that works in $\mathcal{O}(\sum_{m=1}^{s} \frac{s}{m} \cdot \min(cnt_m, m)) = \mathcal{O}(s\sqrt{s})$ in worst case (here cnt_m is the number of different remainders $l \mod m_l$ for each m).