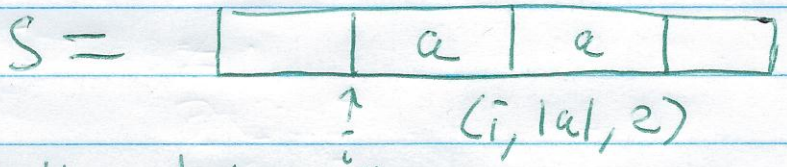
3

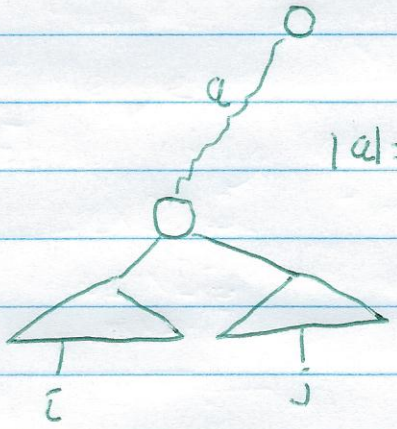
Tandem Repeats

Tandem Repeat

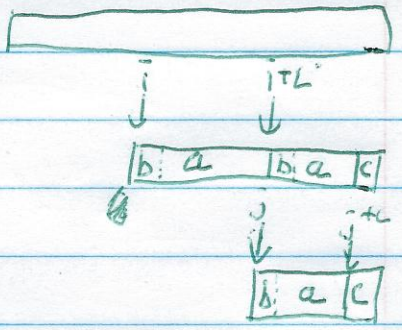


Problem finding all tandem repeats in a string
⇒ Can be done using a suffix tree

Branching tandem repeats



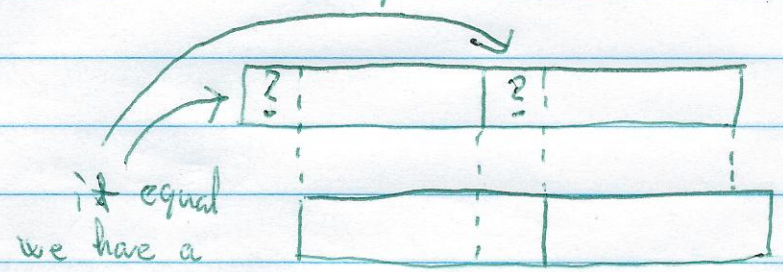
$|a| = j - 1 = L$



Lemma

Non-branching tandem repeat at $(i, |a|, 2)$ implies that another tandem repeat exists at $(i+1, |a|, 2)$.

⇒ we can find all tandem repeats if we have all branching tandem repeats



Algorithm

Idea: Find all branching ~~maximal~~ tandem Repeats.

$\forall v$

  $\forall i \in LL(v)$

    $j = i + D(v)$

      if $j \in LL(v) \wedge S[i] \neq S[i+2D(v)]$

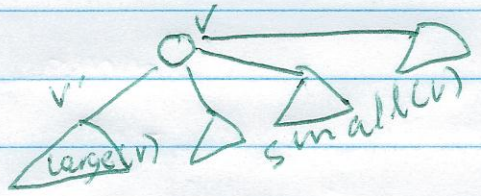
        $(i, D(v), 2)$

Use depth first renumbering to

- Check $j \in LL(v)$ in $O(1)$
- Compact $LL(v)$ representation using intervals

Time n nodes each taking $O(n)$ time $\Rightarrow O(n^2)$

We can do better.



i and j cannot both be in $large(v)$.

$\forall i \in small(v)$

  $j = i + D(v)$

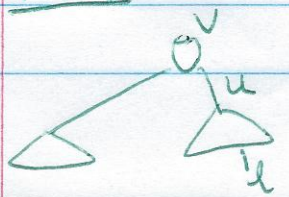
    if $j \in LL(v) \wedge S[i] \neq S[i+2D(v)]$

      $(i, D(v), 2)$

    if $i - D(v) \in LL(v')$

      $(i - D(v), D(v), 2)$

Time



l can be in $small(v)$ $\log n$ times since $|u| \leq \frac{|v|}{2}$ (or else it was large)

$\sum_v |small(v)| = O(n \log n)$ (+ |output| if ~~bit~~ bit-shift)