

Warm up...

- Recall:
 - A suffix tree is a compressed trie of all suffixes of a string $x\$$

$x = \text{abaab}$

1: $\text{abaab\$}$

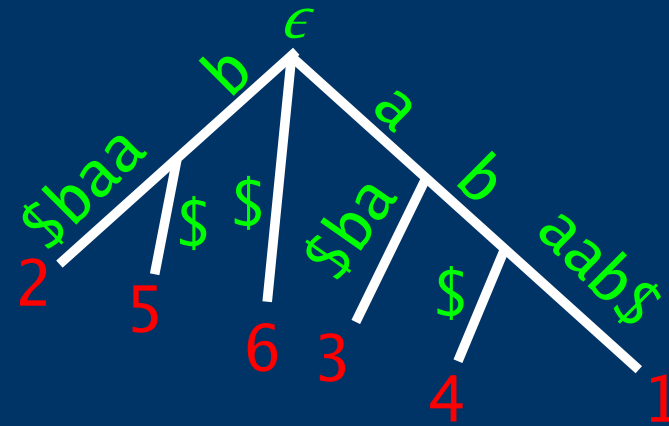
2: $\text{baab\$}$

3: $\text{aab\$}$

4: $\text{ab\$}$

5: $\text{b\$}$

6: $\text{\$}$

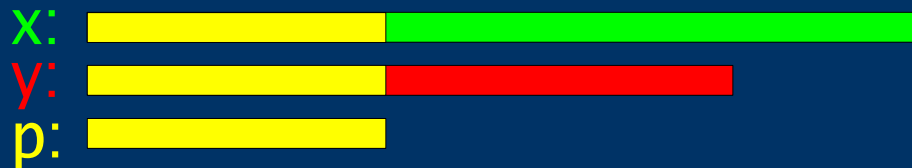


McCreight's algorithm:

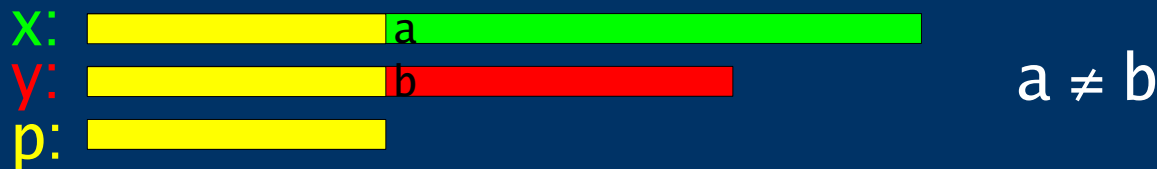
- Build a sequence of compressed tries, T_i , where T_i contains suffixes $j=1,\dots,i$;
 - for $i=n+1$, T_i is the suffix tree
- Be clever when inserting $x[i..n]$ so it won't take time $O(n^2)$ in total

Terminology

- A *common prefix* of x and y is a string, p , that is a prefix of both:

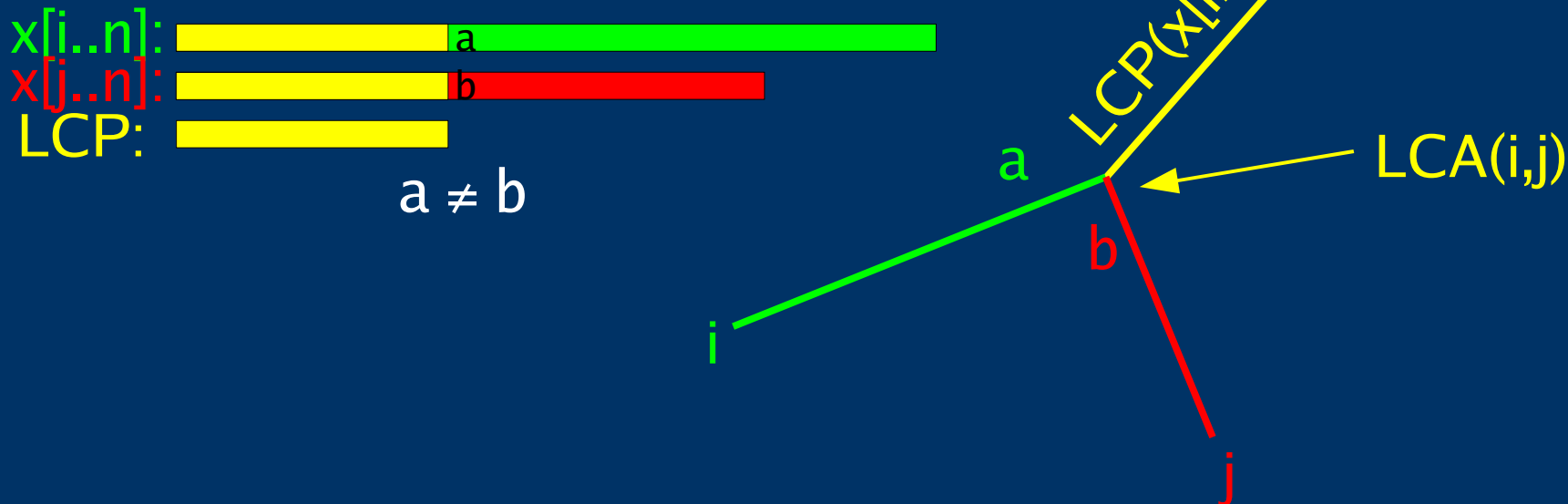


- The *longest common prefix*, $p = \text{LCP}(x, y)$, is a prefix such that: $x[|p|+1] \neq y[|p|+1]$



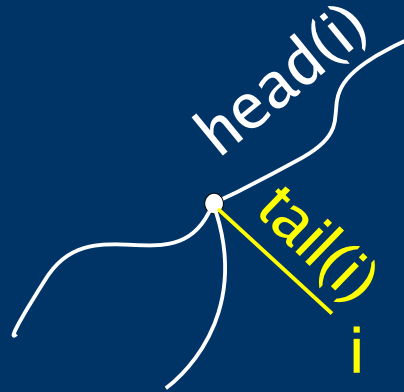
LCP and LCA

- For suffixes of x , $x[i..n]$, $x[j..n]$, their longest common prefix is their lowest common ancestor in the suffix tree:



Head and tail

- Let $\text{head}(i)$ denote the LCP of $x[i..n]\$$ and $x[j..n]\$$ for all $j < i$
- Let $\text{tail}(i)$ be the string such that $x[i..n]\$ = \text{head}(i)\text{tail}(i)$
- Iteration i in McCreight's algorithm consist of finding (or inserting) the node for $\text{head}(i)$ and appending $\text{tail}(i)$



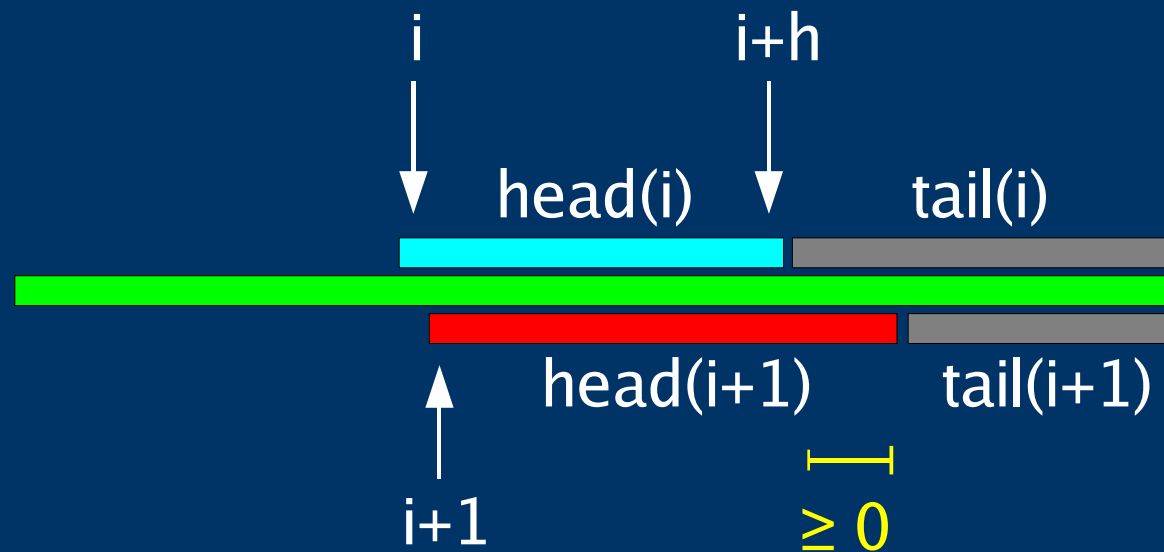
The trick

- The trick to McCreight's algorithm is a clever way of finding $\text{head}(i)$



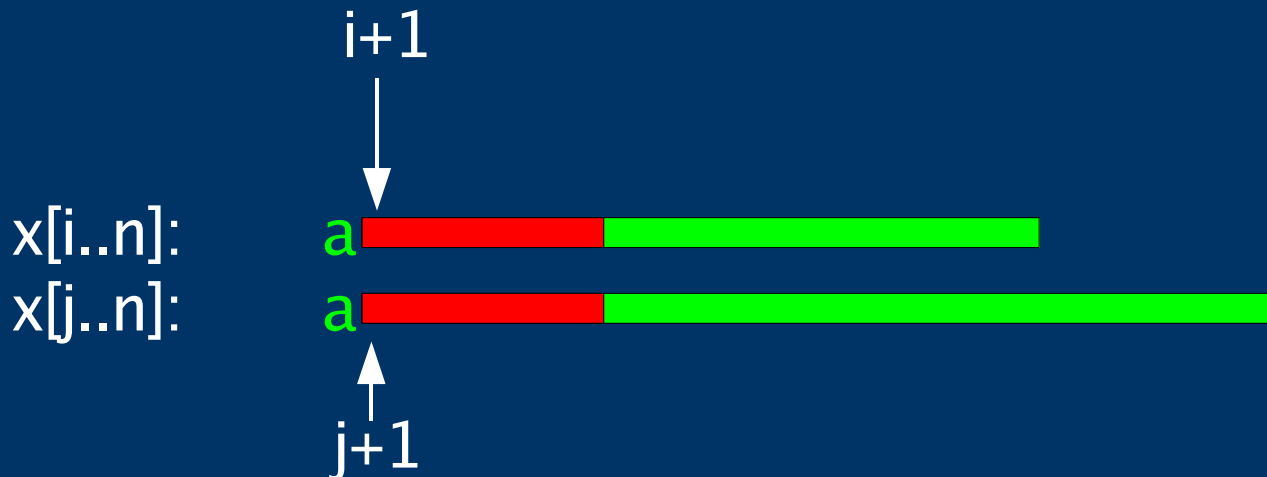
Lemma 5.2.1

- Let $\text{head}(i) = x[i..i+h]$. Then $x[i+1..i+h]$ is a prefix of $\text{head}(i+1)$



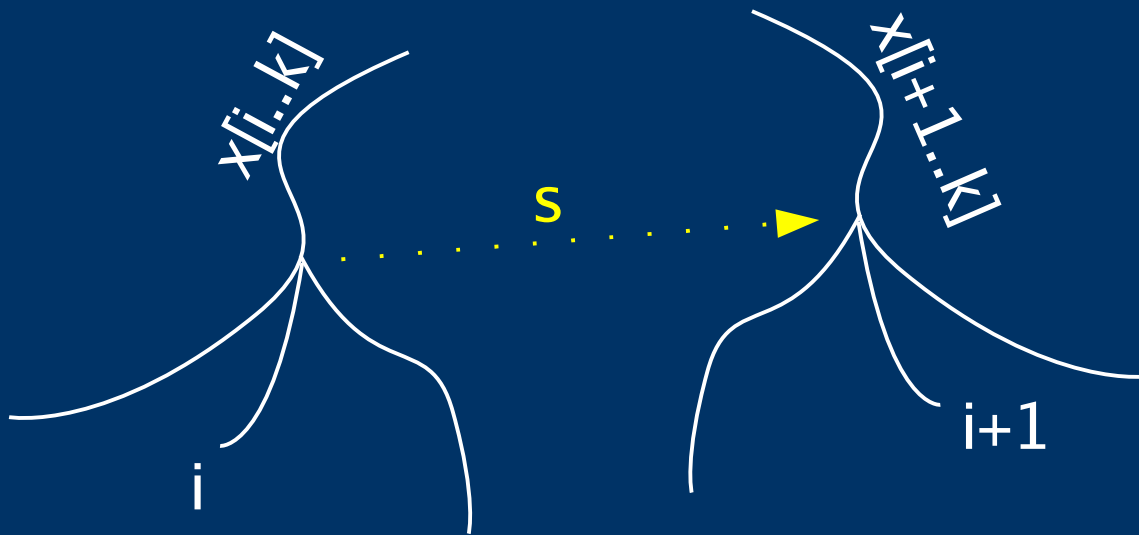
Proof

- Trivial for $h=0$ ($\text{head}(i)$ empty), so assume $h>0$:
 - Let $\text{head}(i) = ay$: a ████████
 - By def. $\exists j < i$ such that $\text{LCP}(i,j) = ay$
 - Thus suffix $j+1$ and $i+1$ share prefix y
 - Thus y is a prefix of $\text{LCP}(i+1, j+1)$
 - Thus y is a prefix of $\text{head}(i+1)$



Suffix link

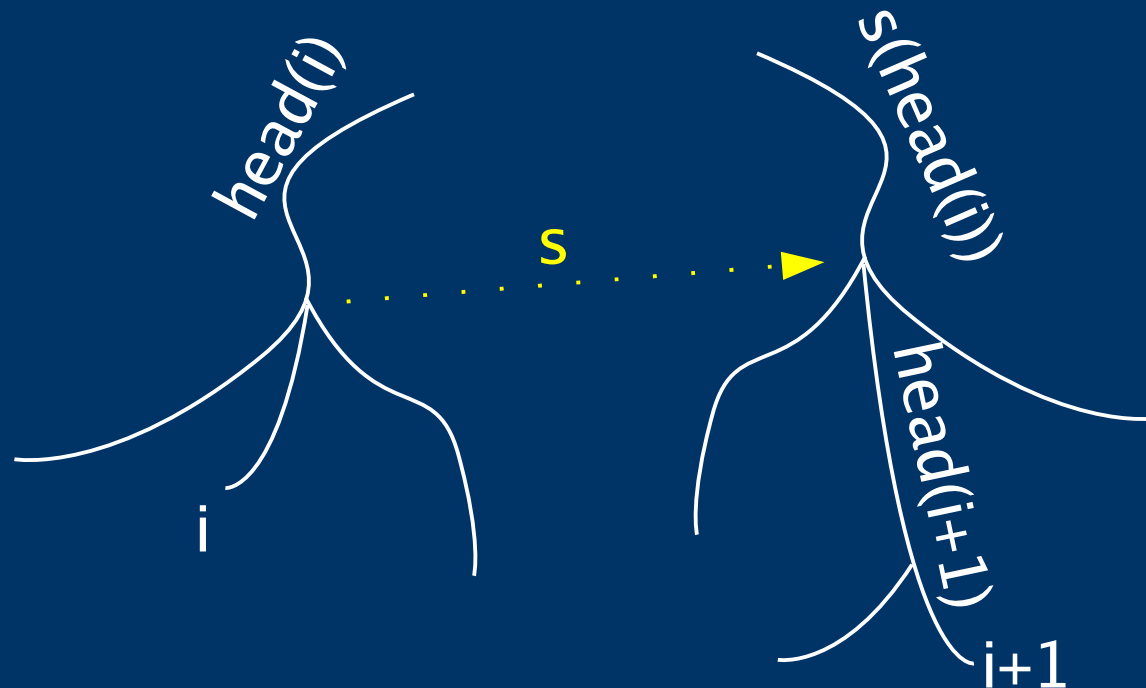
- Define $s(u) = \epsilon$ if $u = \epsilon$, v if $u = av$
- As a pointer from $x[i..k]$ to $x[i+1..k]$:



- (ex 5.2.3: if u is a node, so is $s(u)$)
-
-

Corollary of lemma 5.2.1

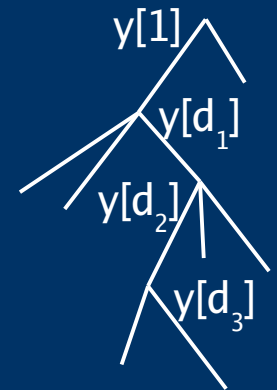
- $s(\text{head}(i))$ is a prefix of $\text{head}(i+1)$
- Thus: $s(\text{head}(i))$ is an ancestor of $\text{head}(i+1)$



- $s(\text{head}(i))$ can be used as a shortcut
-
-

Slowscan and fastscan

- **Slowscan:** if we do not know if string y is in T_i , we must search character by character
- **Fastscan:** if we *do know* that y is in x , we can jump directly from node to node
 - At node u at (path-)depth d , follow the edge with label starting with $y[d]$
 - Continue until we reach the end of y
 - On a node (if y is in T_i)
 - Or on an edge (if y is a prefix of a string in T_i)

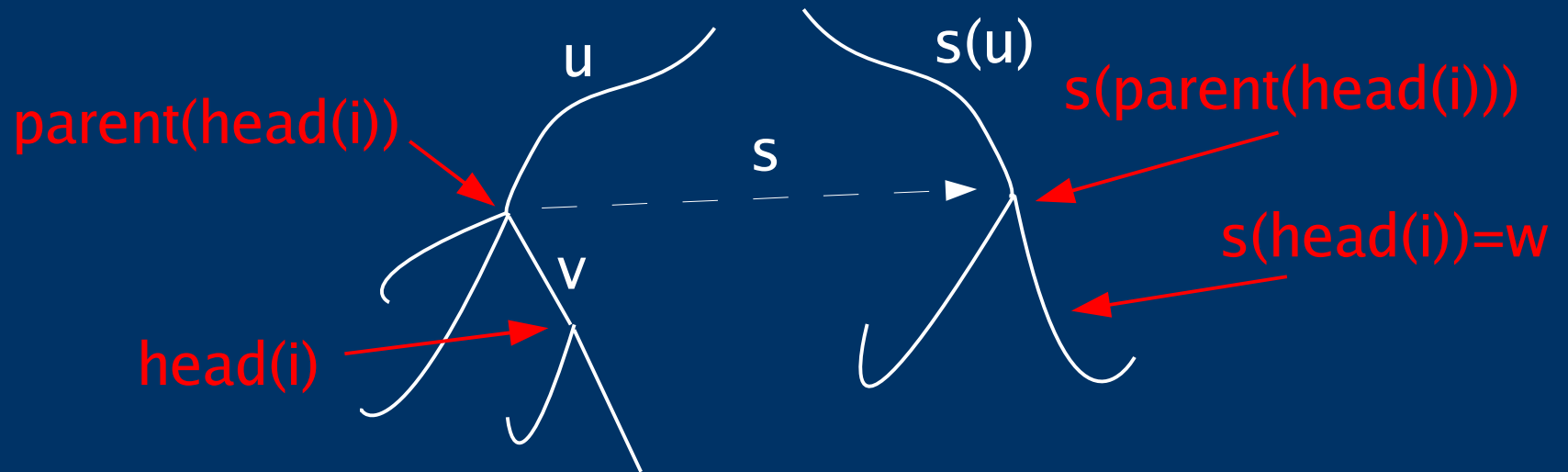


McCreight's algorithm



- Begin with the tree T_1 :
- For $i=1, \dots, n$, build tree T_{i+1} satisfying:
 - T_{i+1} is a compressed trie for $x[j..n]$, $j \leq i+1$
 - All non-terminal nodes (with the possible exception of $\text{head}(i)$) have a suffix link $s(-)$
- Each iteration must:
 - Add node $i+1$
 - Potentially add $\text{head}(i+1)$
 - Add $\text{tail}(i+1)$
 - Add suffix link $\text{head}(i) \rightarrow s(\text{head}(i))$

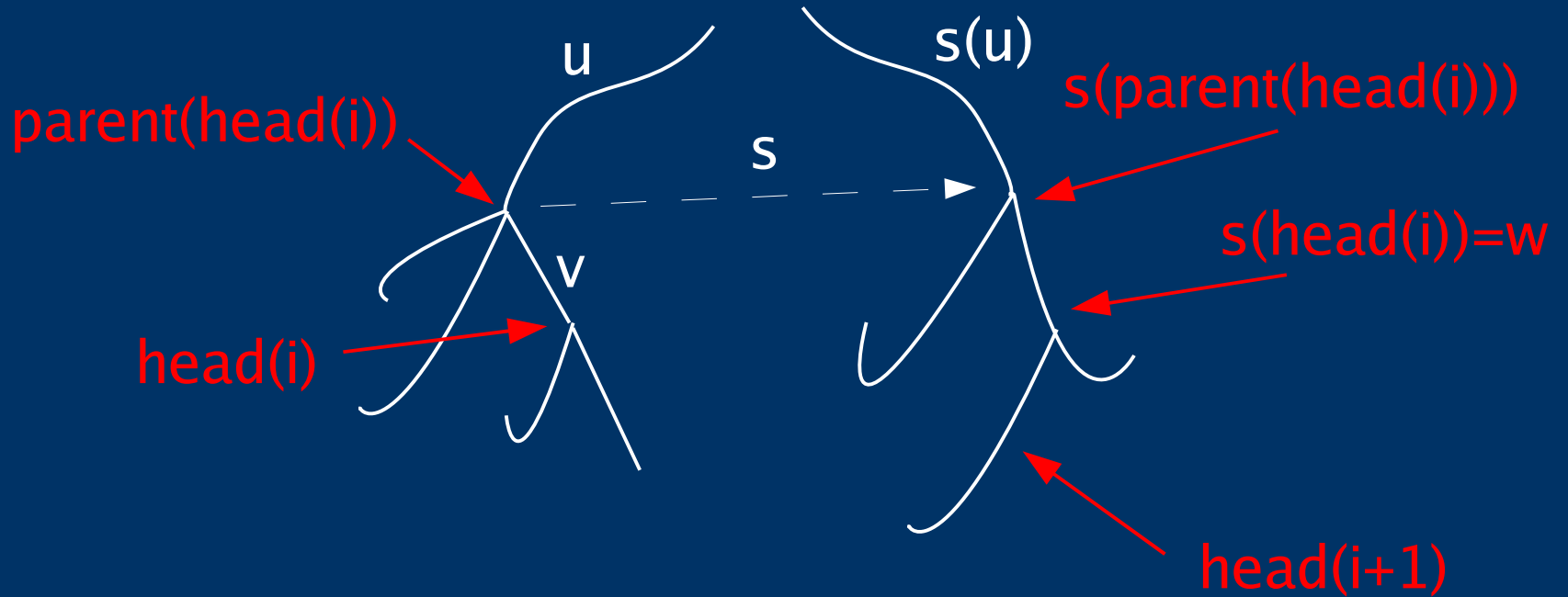
At the start of iteration i ...



Let $\text{head}(i)=uv$ and $\text{parent}(\text{head}(i))=u$ and $w=s(u)v=s(\text{head}(i))$

By the invariant, $s(\text{parent}(\text{head}(i)))$ and the suffix link exists;
by the lemma, w is an ancestor of $\text{head}(i+1)$

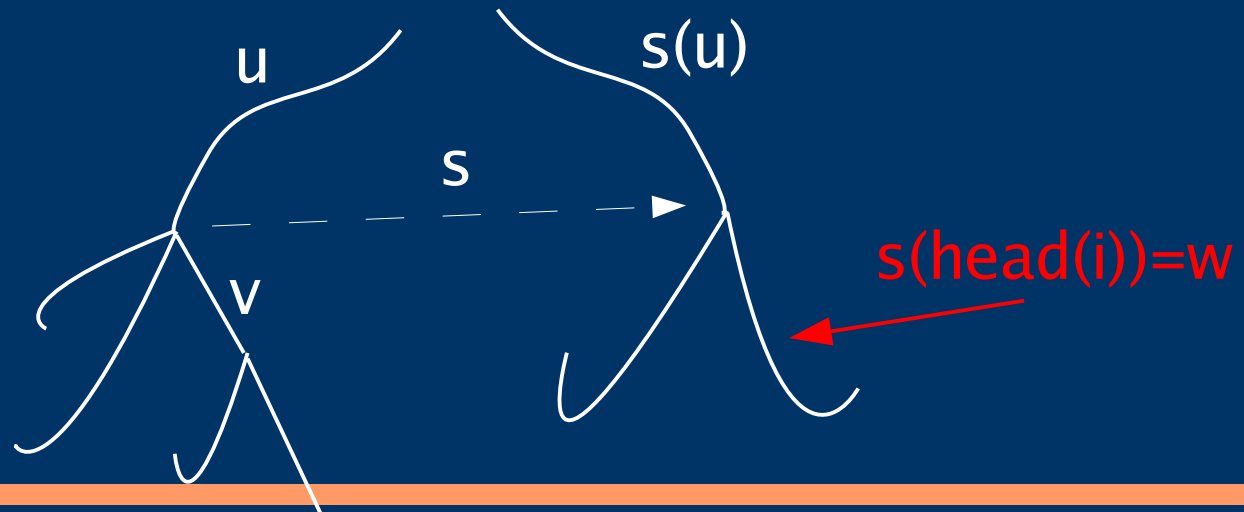
Steps in iteration $i...$



Move quickly to w , then search for $\text{head}(i+1)$ starting there

Observe: w is in T_i

- $\text{head}(i)$ is a prefix of $x[j..n]$ for some $j < i$
- Thus w is a prefix of $x[j+1..n]$ for some $j < i$
 - i.e w is a prefix of some suffix $j \leq i$
 - i.e. w is in T_i
- **Consequently:** we can search for w from $s(u)$ using **fastscan!**



If w is a node:

- Update $s(\text{head}(i)) := w$
- Then search for $\text{head}(i+1)$ using slowscan



If w is on an edge:

- If w is not a node, then all suffix $j < i$ with prefix w agree on the next letter
 - By definition of $\text{head}(i)$ there is $j < i$ such that suffix $x[i..n]$ and $x[j..n]$ differs after $\text{head}(i)$
 - $x[i+1..n]$ must also disagree at that character
 - Thus $\text{head}(i+1)$ must be w
 - Add node w , update $\text{head}(i) := w$ and set $\text{head}(i+1) = w$
-
-

McCreight's Suffix Tree Algorithm

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

Special Case

Construct tree for $x[1..n]$

for $i = 1$ to n do

if $\text{head}(i) = \epsilon$ **then**

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

if $u \neq \epsilon$ **then** $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge **then**

 add a node for w

$\text{head}(i+1) = w$

else if w is a node **then**

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

Special Case:

When $\text{head}(i)$ is empty
there is no information
to exploit

Short Cut in Search

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i)) ; v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(s(\text{head}(i)), w)$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

Use **suffix link** and **fastscan** to quickly find w , the overlap between $s(\text{head}(i))$ and $\text{head}(i+1)$

Add inner nodes

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(\text{tail}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

Add w and $\text{head}(i+1)$ as needed

Update Suffix-Link

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(\text{tail}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

make the suffix link of
 $\text{head}(i)$ point to w

Add leaf for suffix $x[i..n]$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

Add leaf for $x[i..n]$

add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

Example: $x = abaab$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$



Example: $x = abaab$, $i=1$, inserting $baab\$$

Construct tree for $x[1..n]$

for $i = 1$ **to** n **do**

if $head(i) = \epsilon$ **then**

$head(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $head(i+1)$ as node if necessary

continue

$u = \text{parent}(head(i))$; $v = \text{label}(u, head(i))$

if $u \neq \epsilon$ **then** $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge **then**

 add a node for w

$head(i+1) = w$

else if w is a node **then**

$head(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $head(i+1)$ as node if necessary

$s(head(i)) = w$

 add leaf $i+1$ and edge between $head(i+1)$ and $i+1$

$head(1) = \epsilon$
 $tail(1) = abaab\$$



Example: $x = abaab$, $i=1$, inserting $baab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

add $i+1$ and $\text{head}(i+1)$ as node if necessary

continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge then

add a node for w

$\text{head}(i+1) = w$

else if w is a node then

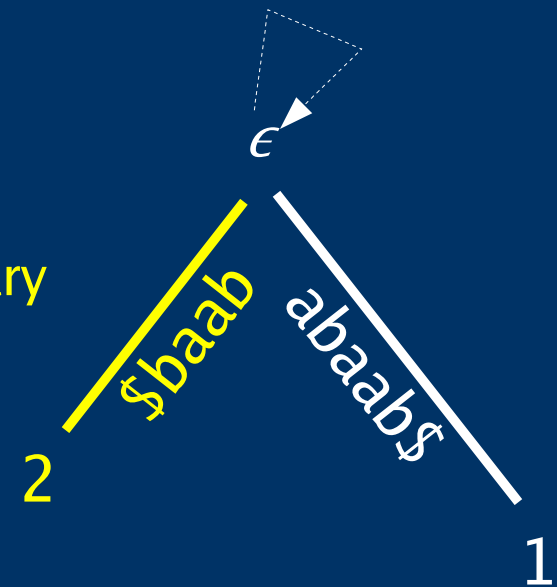
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(2) = \epsilon$
 $\text{tail}(2) = baab\$$



Example: $x = abaab$, $i=2$, inserting $aab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

add $i+1$ and $\text{head}(i+1)$ as node if necessary

continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge then

add a node for w

$\text{head}(i+1) = w$

else if w is a node then

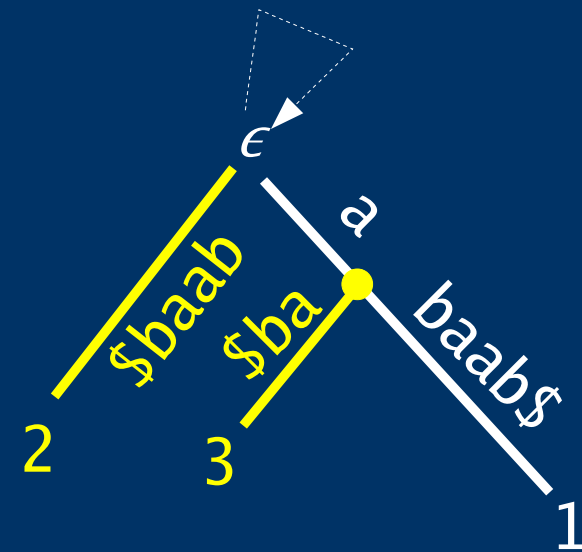
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(2) = \epsilon$
 $\text{tail}(2) = \text{baab\$}$



$\text{head}(3) = a$
 $\text{tail}(3) = ab\$$

Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ **to** n **do**

if $\text{head}(i) = \epsilon$ **then**

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

if $u \neq \epsilon$ **then** $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge **then**

 add a node for w

$\text{head}(i+1) = w$

else if w is a node **then**

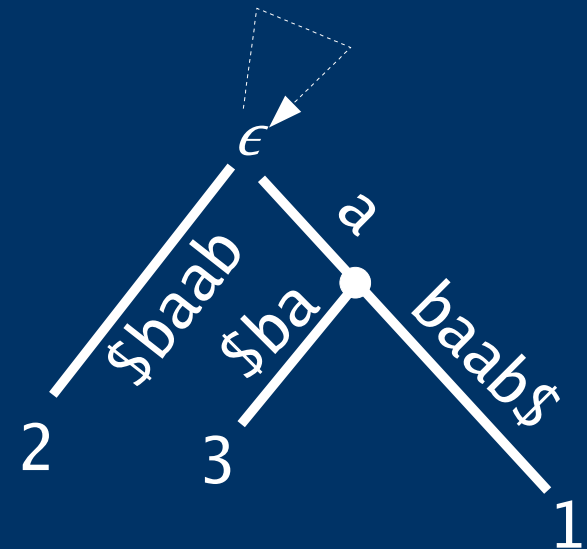
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i)) ; v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

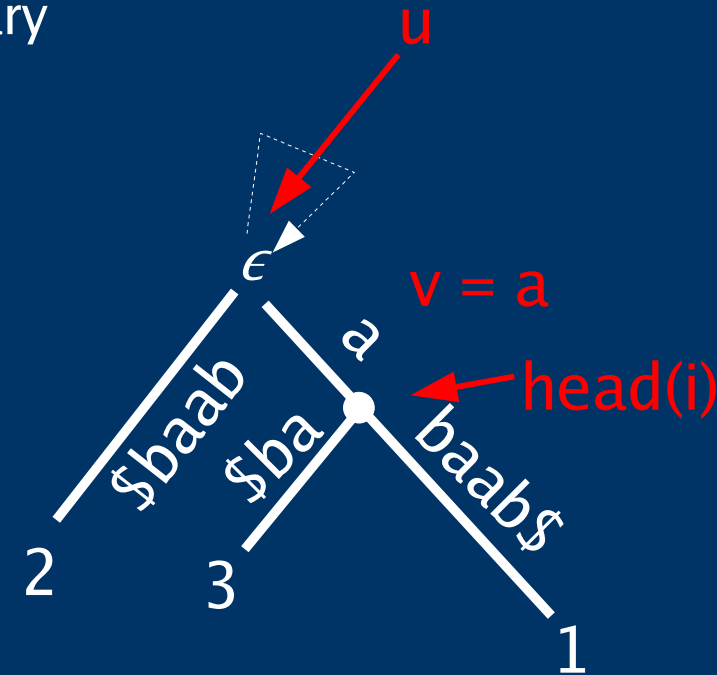
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

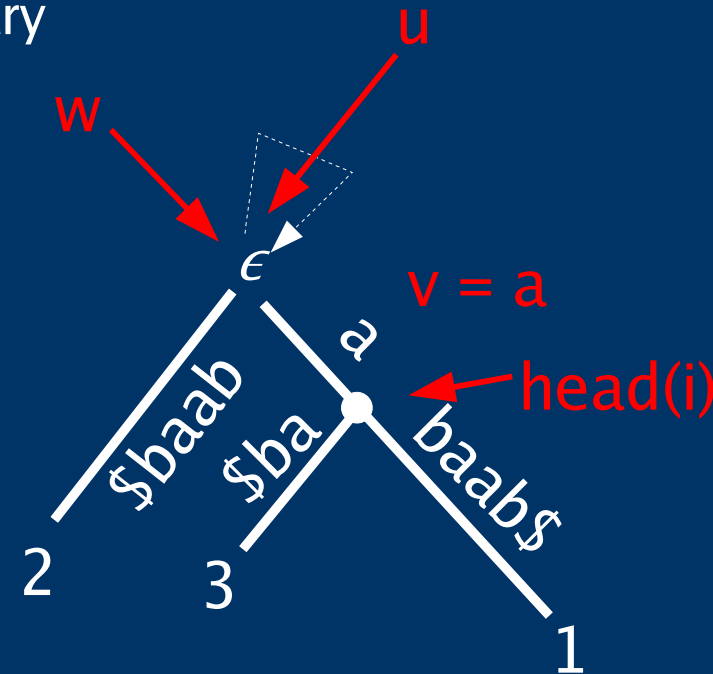
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

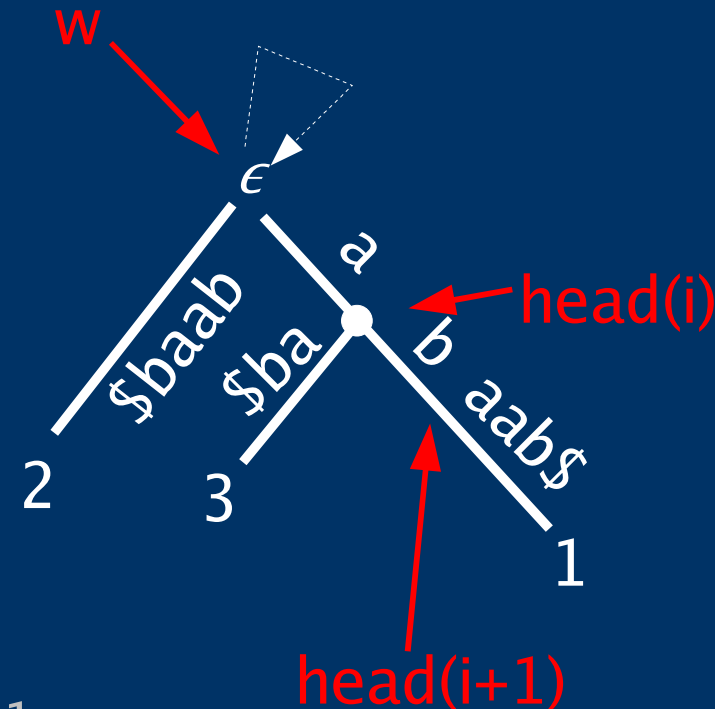
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

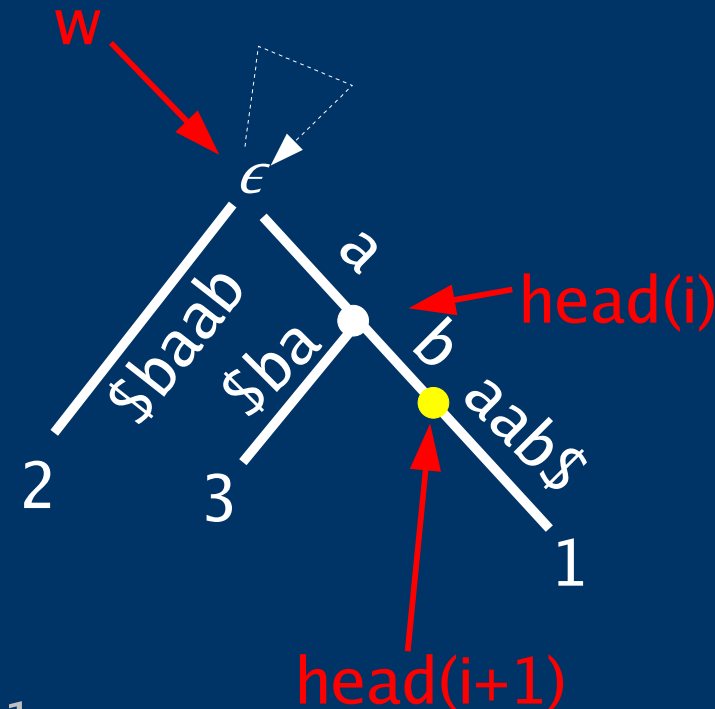
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

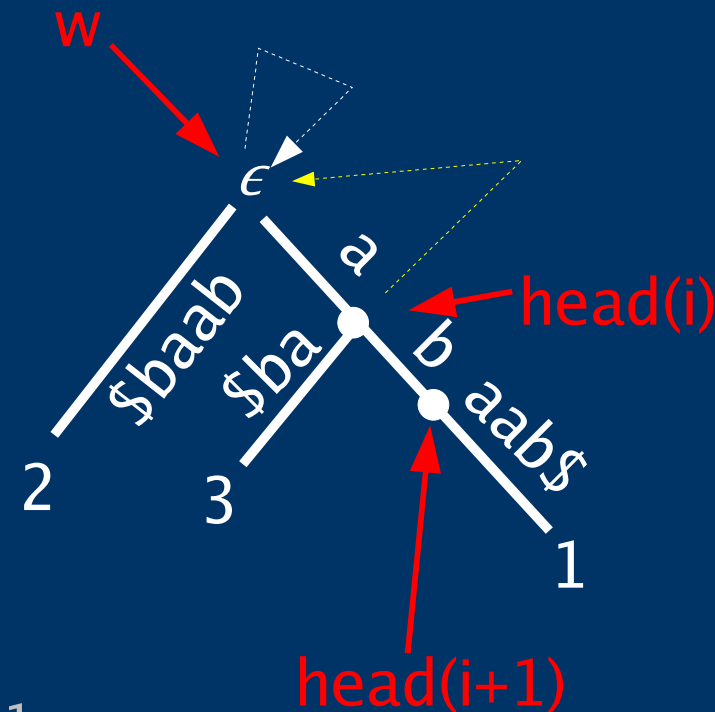
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=3$, inserting $ab\$$

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

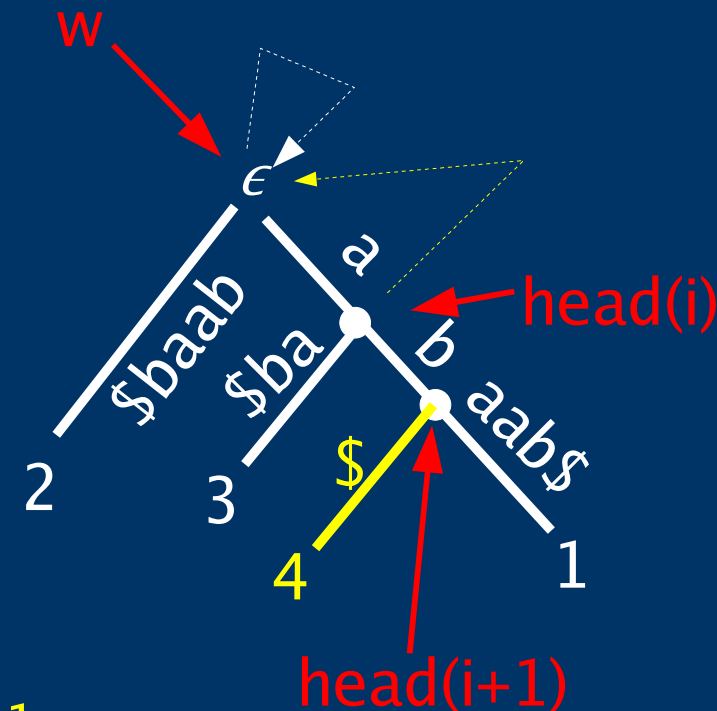
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(3) = a$
 $\text{tail}(3) = aab\$$



Example: $x = abaab$, $i=4$, inserting b

Construct tree for $x[1..n]$

for $i = 1$ to n do

if $\text{head}(i) = \epsilon$ then

head($i+1$) = $\text{slowscan}(\epsilon, s(\text{tail}(i)))$

add $i+1$ and $\text{head}(i+1)$ as node if necessary

continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

else $w = \text{fastscan}(\epsilon, v[2..|v|])$

if w is an edge then

add a node for w

head($i+1$) = w

else if w is a node then

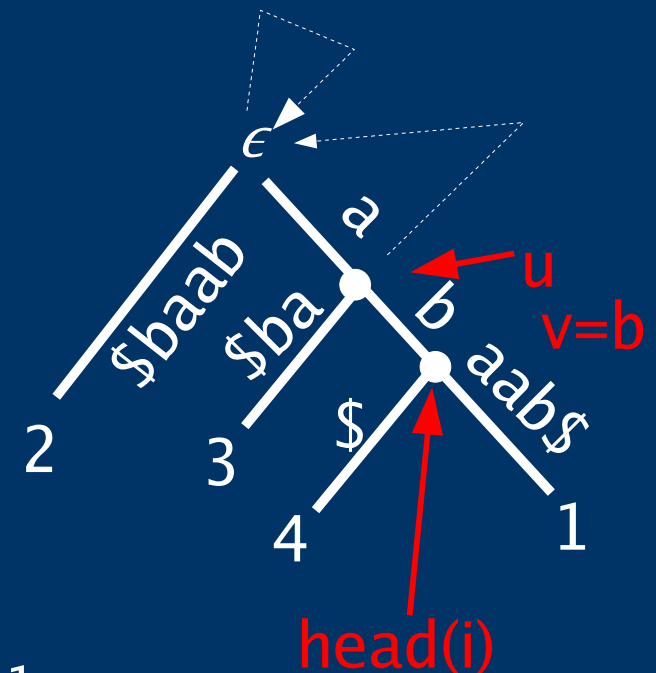
head($i+1$) = $\text{slowscan}(w, \text{tail}(i))$

add head($i+1$) as node if necessary

$s(\text{head}(i)) = w$

add leaf $i+1$ and edge between head($i+1$) and $i+1$

head(4)=ab
tail(4)=\$



Example: $x = abaab$, $i=4$, inserting b

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

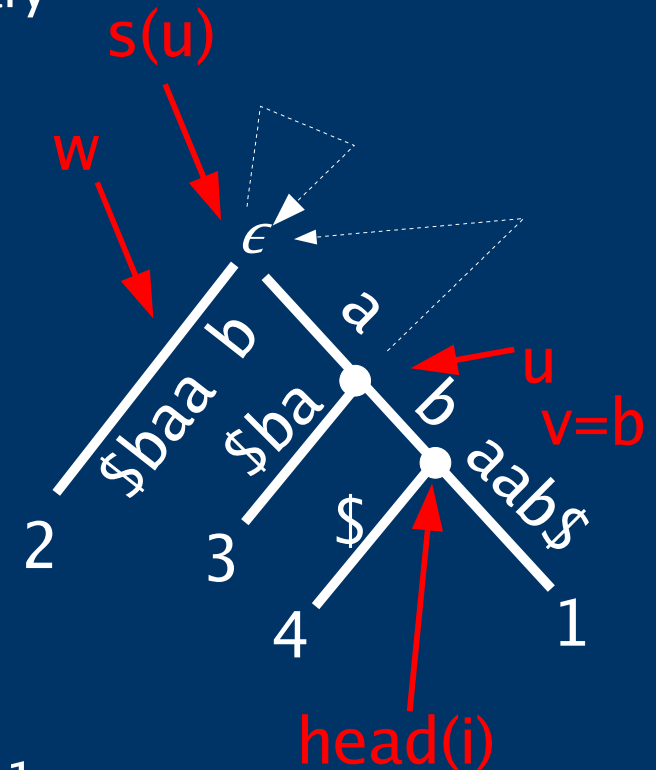
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(4) = ab$
 $\text{tail}(4) = \$$



Example: $x = abaab$, $i=4$, inserting b

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

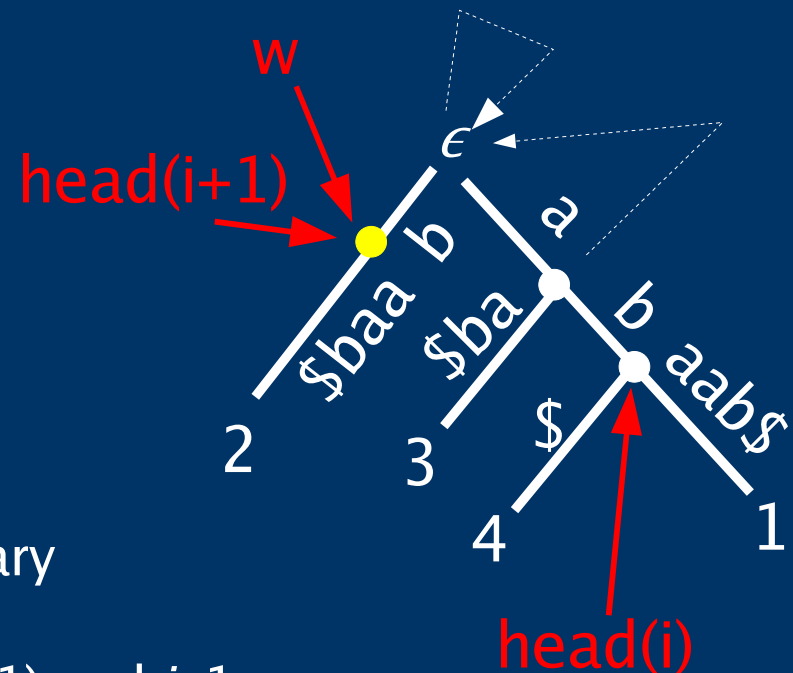
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(4) = ab$
 $\text{tail}(4) = \$$



Example: $x = abaab$, $i=4$, inserting b

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

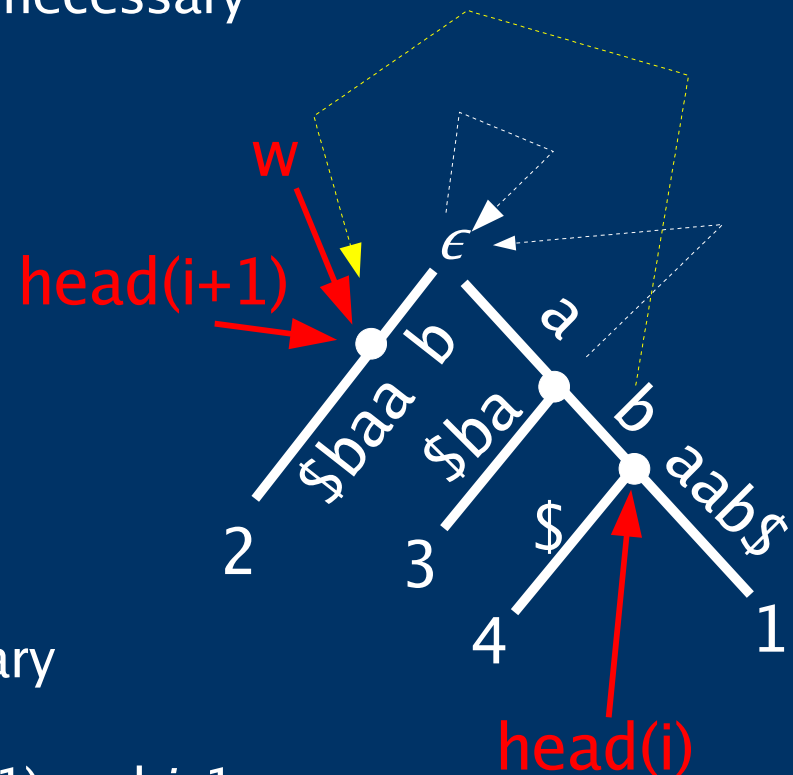
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(4) = ab$
 $\text{tail}(4) = \$$



Example: $x = abaab$, $i=4$, inserting b

Construct tree for $x[1..n]$

for $i = 1$ to n do

 if $\text{head}(i) = \epsilon$ then

$\text{head}(i+1) = \text{slowscan}(\epsilon, s(\text{tail}(i)))$

 add $i+1$ and $\text{head}(i+1)$ as node if necessary

 continue

$u = \text{parent}(\text{head}(i))$; $v = \text{label}(u, \text{head}(i))$

 if $u \neq \epsilon$ then $w = \text{fastscan}(s(u), v)$

 else $w = \text{fastscan}(\epsilon, v[2..|v|])$

 if w is an edge then

 add a node for w

$\text{head}(i+1) = w$

 else if w is a node then

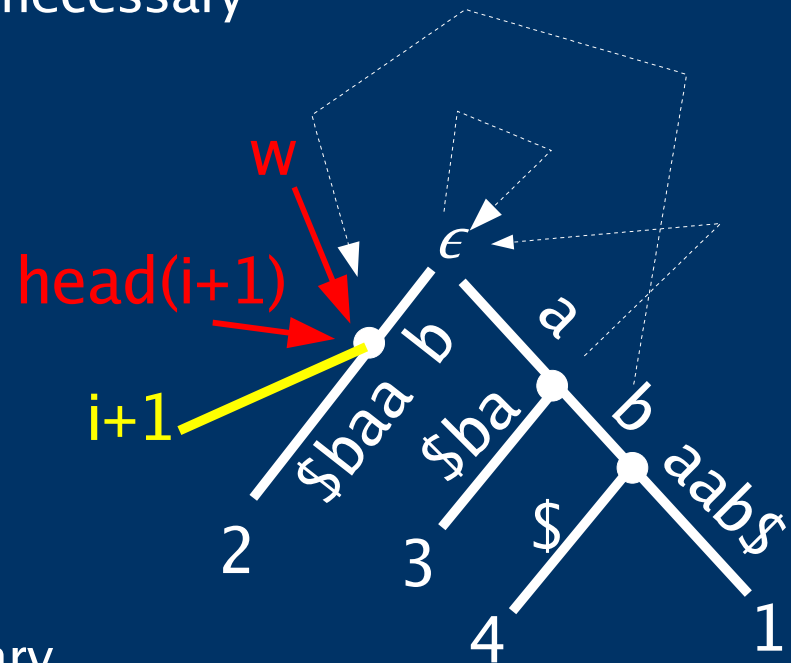
$\text{head}(i+1) = \text{slowscan}(w, \text{tail}(i))$

 add $\text{head}(i+1)$ as node if necessary

$s(\text{head}(i)) = w$

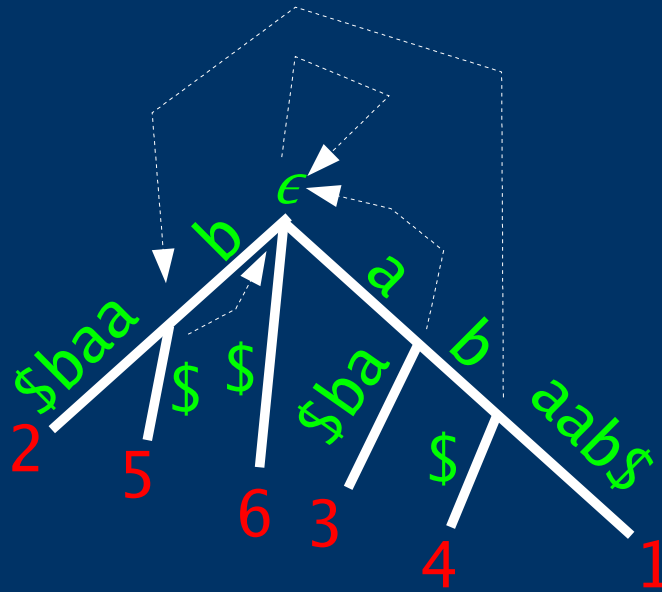
 add leaf $i+1$ and edge between $\text{head}(i+1)$ and $i+1$

$\text{head}(4) = ab$
 $\text{tail}(4) = \$$



Done

- Nothing new for $i=5$, inserting \$...



Correctness:

- Follows from the invariant
 - At iteration i we have a compressed trie of all suffixes $j < i$
 - At the final iteration we have a compressed trie of all suffixes, i.e. the suffix tree

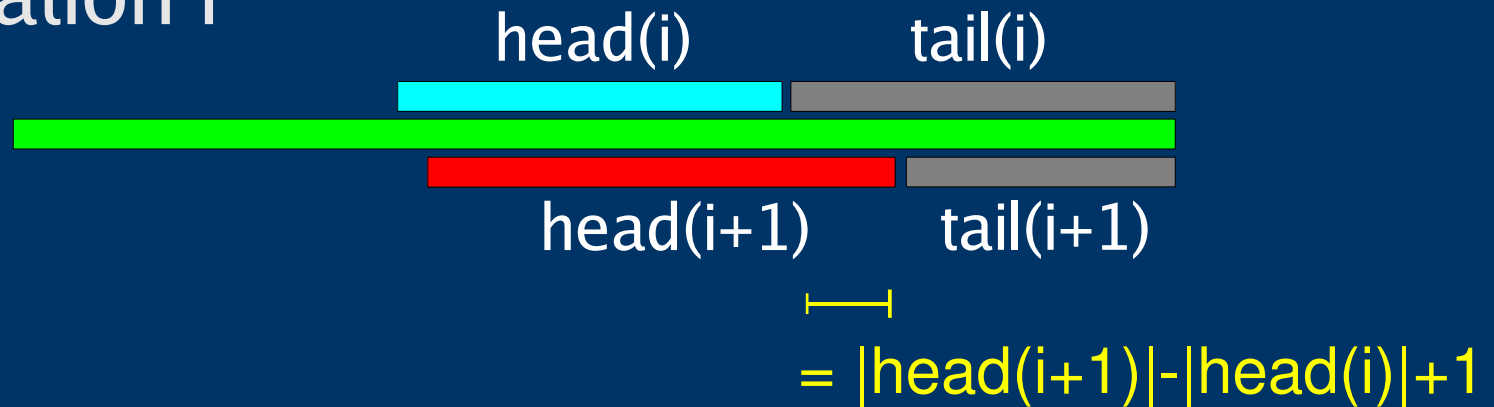
Time and space usage

- Everything but searching is constant time per suffix, so time is $O(n + \text{slowscan} + \text{fastscan})$
- We are not using more space than time, so the space complexity is the same



Slowscan time usage

- We use slowscan to find $\text{head}(i+1)$ from $w = s(\text{head}(i))$, i.e. time $|\text{head}(i+1)| - |\text{head}(i)| + 1$ for iteration i



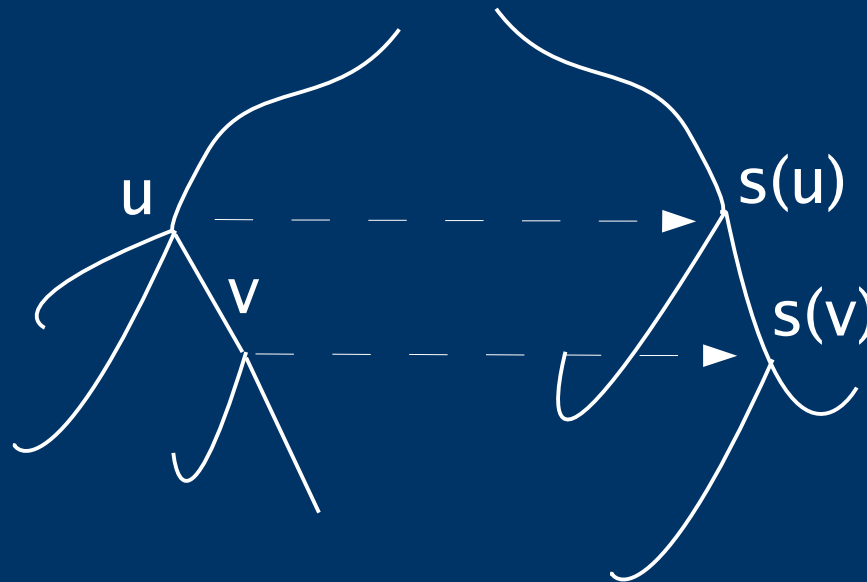
- A telescoping sum
 - Sum = $|\text{head}(n+1)| - |\text{head}(1)| + n$
 - Equal to n since $\text{head}(n+1) = \text{head}(1) = \epsilon$

Fastscan time usage

- Fastscan uses time proportional to the number of nodes it process
 - Define $d(v)$ as the (node-)depth of node v
 - Fastscan increases the node depth
 - Following parent and suffix pointers decreases the node depth
 - Time usage of fastscan is bounded by the total depth-increase
-
-

Proposition

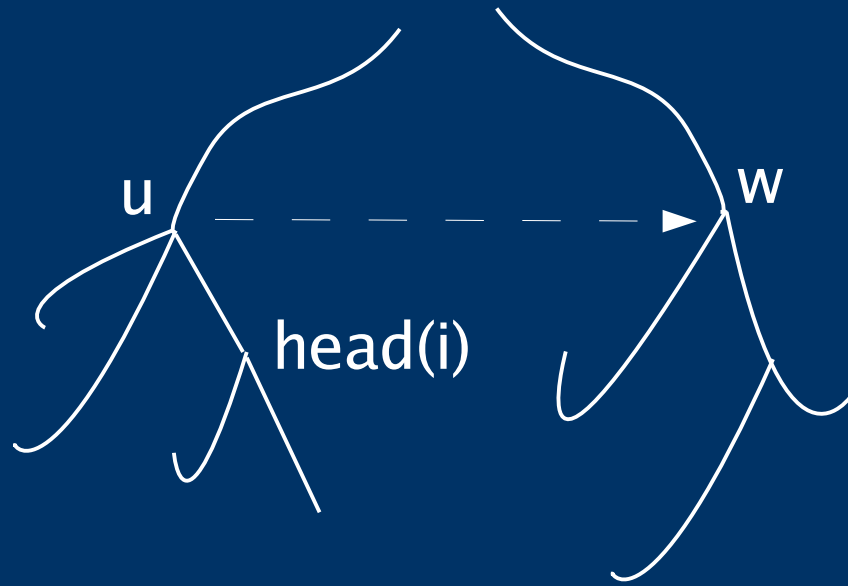
- $d(v) \leq d(s(v)) + 1$
- Proof:
 - For any ancestor u of v , $s(u)$ is an ancestor of $s(v)$
 - Except for the empty prefix and the single letter prefix of v , the $s(u)$'s are different



Corollary

- In each step, before calling fastscan, we decrease the depth by at most 2:

$$d(u) = d(\text{head}(i)) - 1;$$
$$d(w) \geq d(u) - 1$$



- The total decrease is thus $2n$
-
-

Time usage of fastscan

- The time usage of fastscan is bounded by n plus the total decrease of depth,
 - i.e. the time usage of fastscan is $O(n)$

1st mandatory project

- Implement the McCreight algorithm and an application that uses it for exact pattern matching
 - The project is evaluated by an oral presentation on **Monday 21st Nov.**
 - Presentation should include:
 - Insights you have had during the implementation
 - Problems encountered
 - Byte usage per tree node
 - Measured construction and search time
 - Validation approach (testing)
-
-

1st mandatory project

- Hand in a single piece of paper describing:
 - Status of the project (does it work? Are there any known bugs?)
 - A path to a linux executable program we can run to test the implementation

