# Random Access to Compressed Strings

Philip Bille
Gad Landau
**Oren Weimann**

# Random Access to Compressed Strings

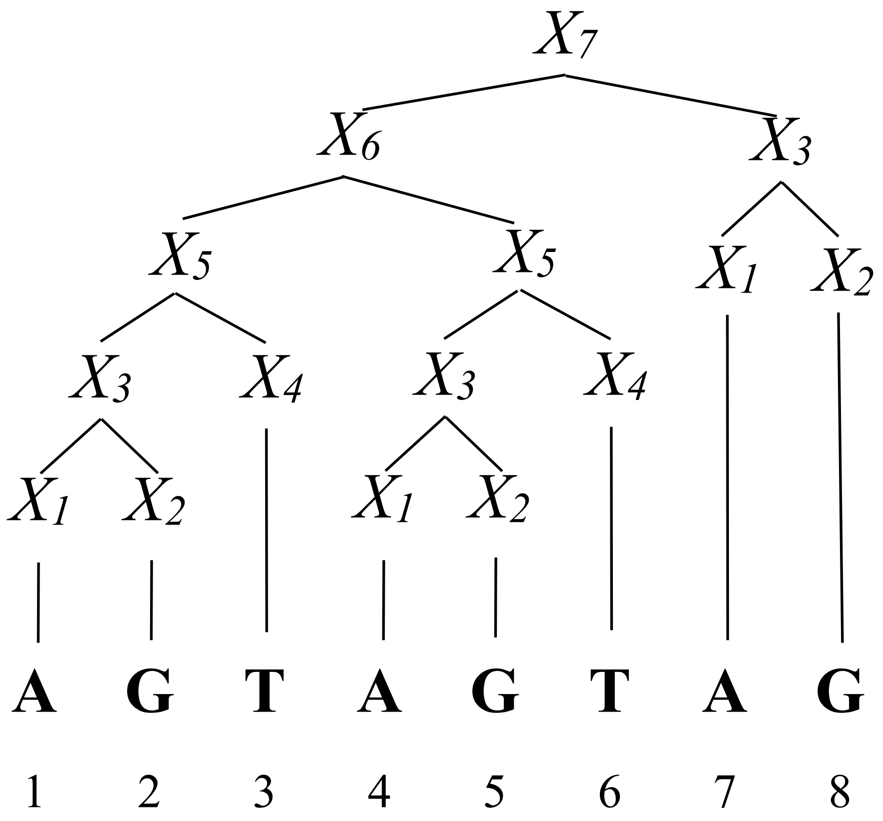
text
DNA
HTML

# Random Access to Compressed Strings

# Random Access to Compressed Strings

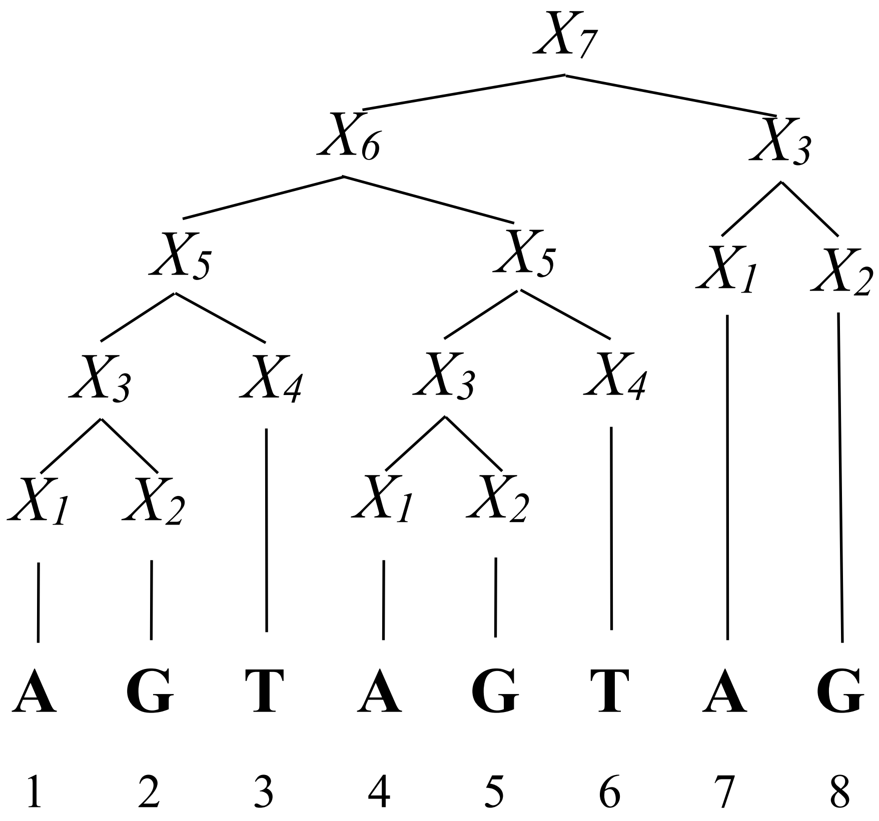
- What is the i'th character ?

# Grammar
# Random Access to Compressed Strings



- What is the i’th character ?

# Grammar
# Random Access to Compressed Strings

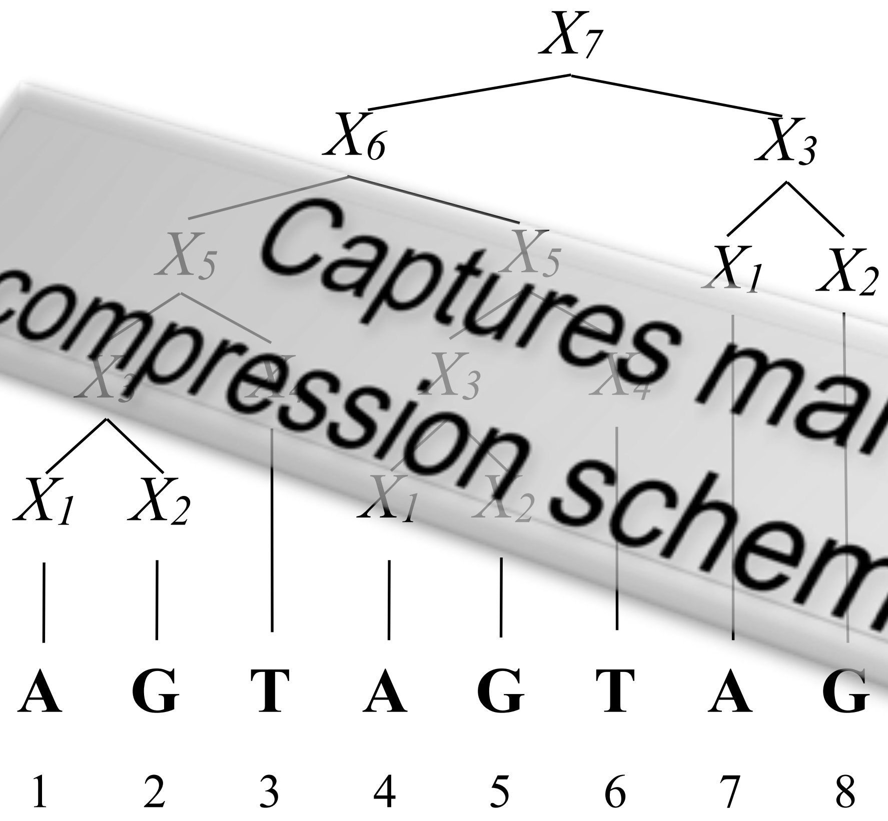


$n = 7$

$N = 8$

- What is the i’th character ?

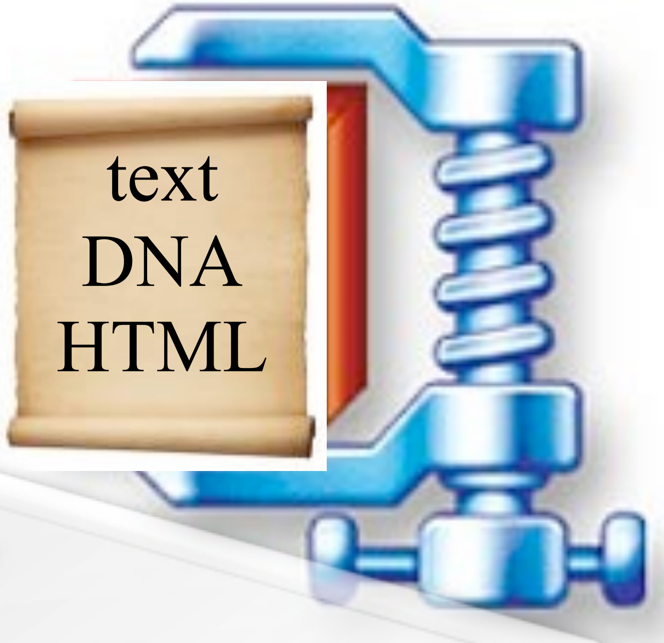
# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



Captures many popular compression schemes [Rytter 2003]

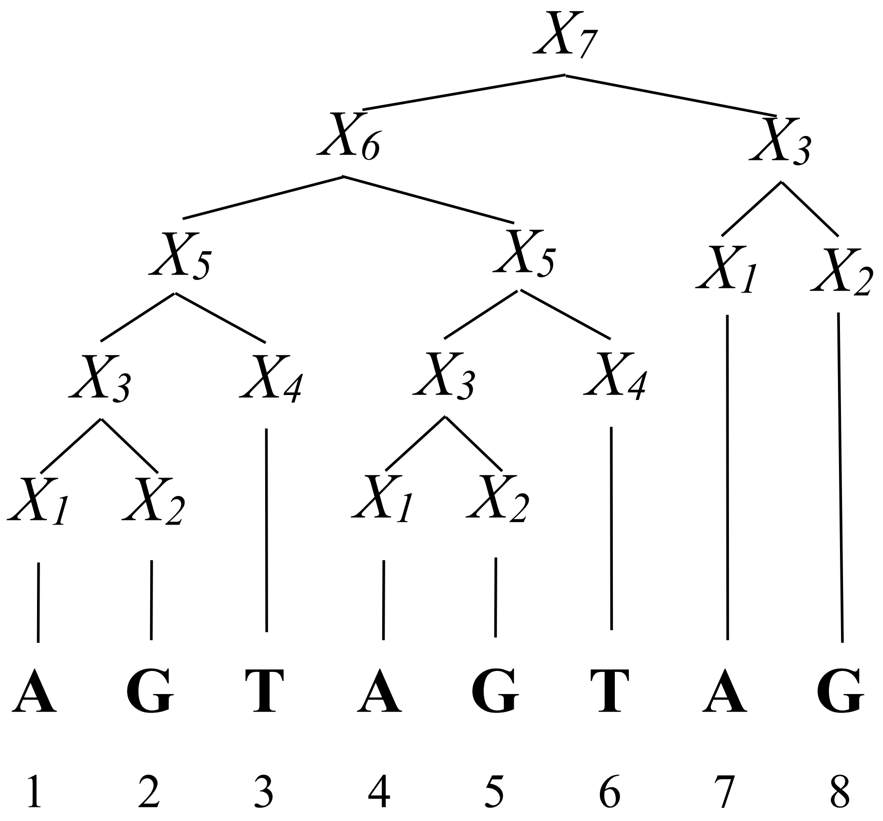
$n = 7$

$N = 8$



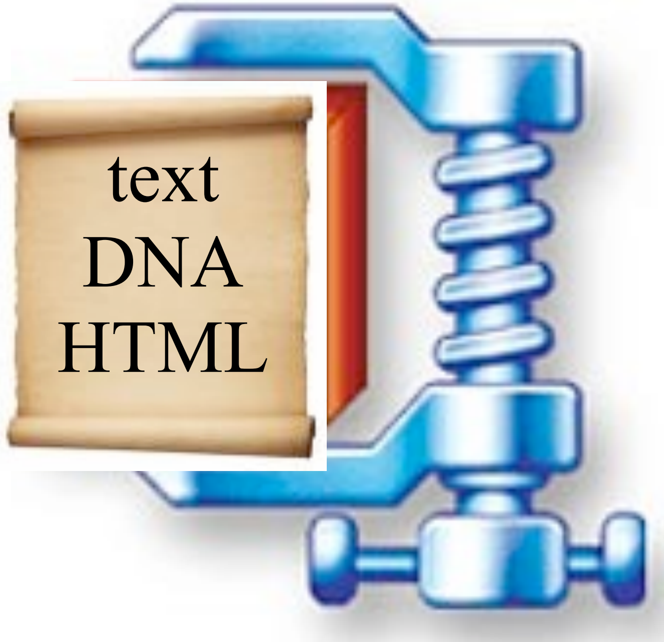
- What is the i'th character ?

# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



$n = 7$

$N = 8$

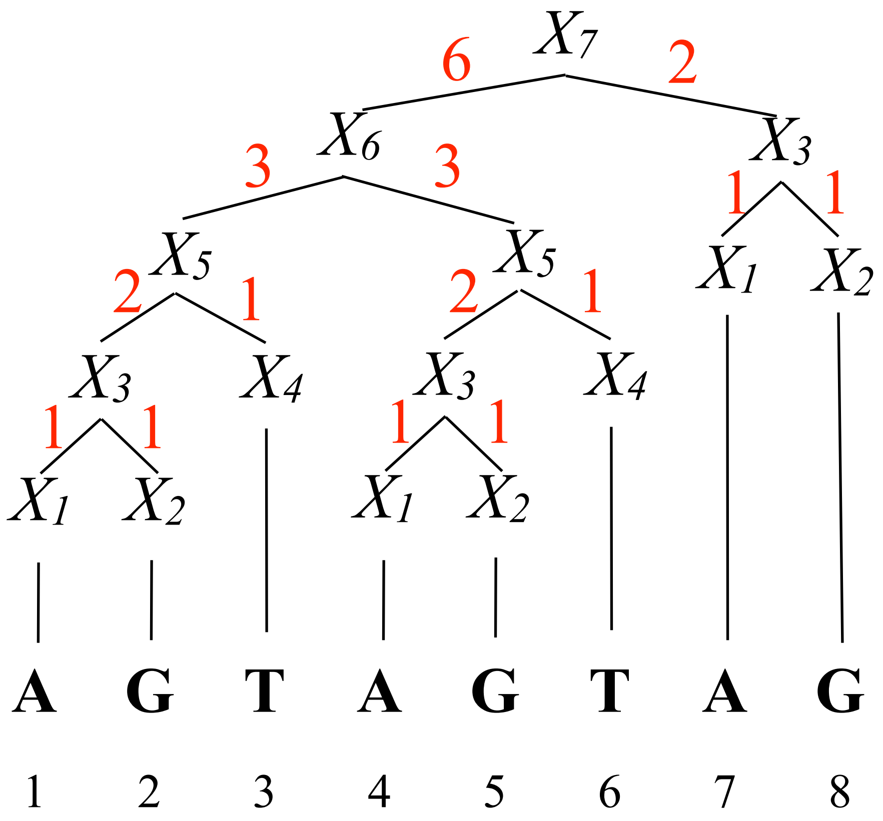


- What is the i’th character ?

  O($N$) space

  O(1) query

# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



$n = 7$

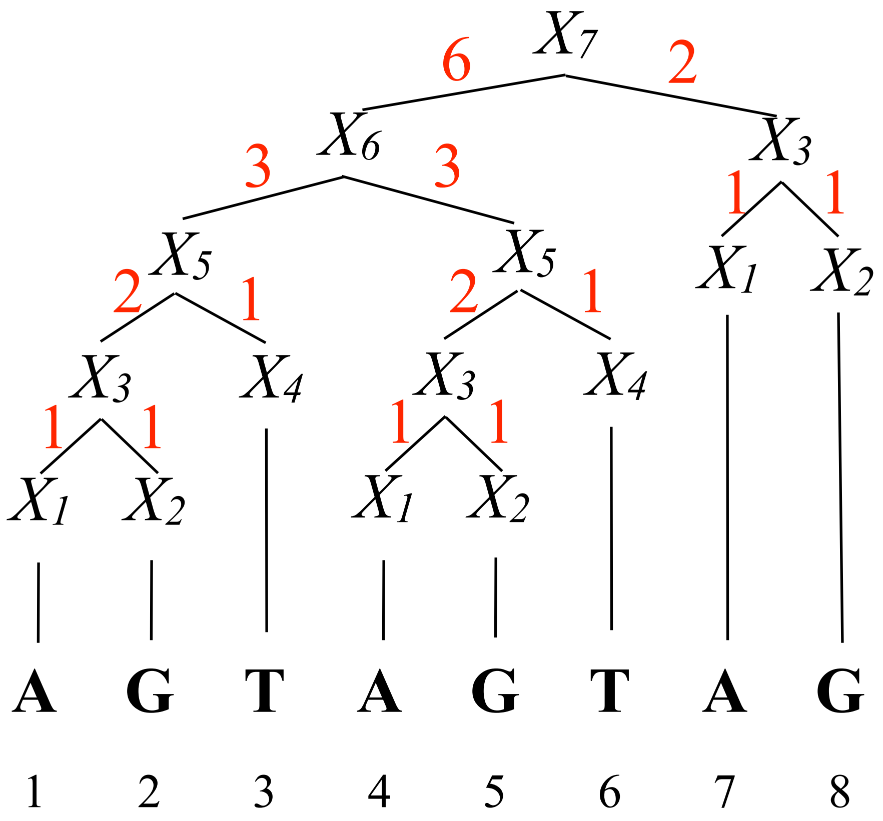
$N = 8$

- What is the i’th character ?

  O($N$) space O($n$) space

  O(1) query O($n$) query

# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



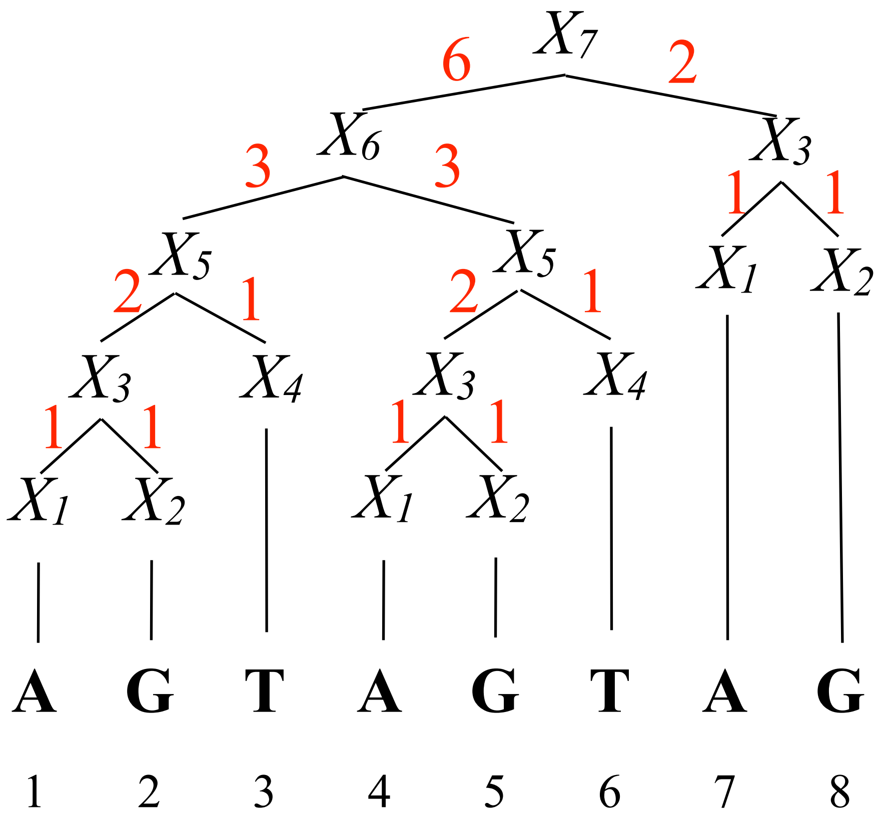
$n = 7$

$N = 8$

- What is the i'th character ?

| | | |
|---|---|---|
| $O(N)$ space | $O(n)$ space | $O(n)$ space |
| $O(1)$ query | $O(n)$ query | $O(\log N)$ query |

# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



$n = 7$

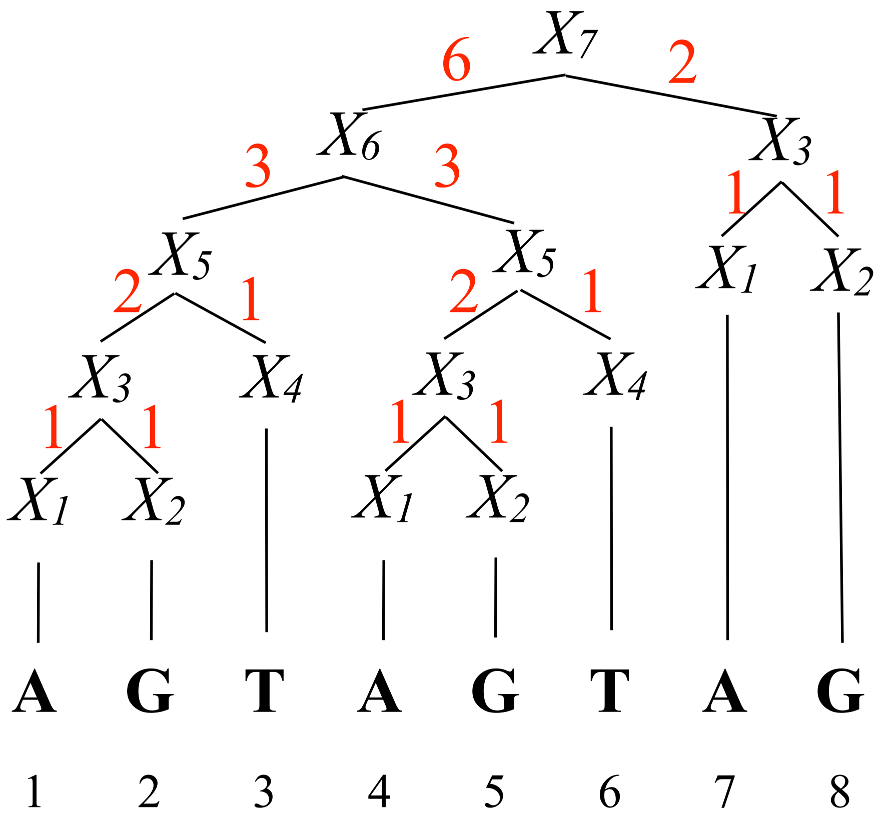
$N = 8$

- What is the i'th character ?
- What is the substring [i,j] ?

O($n$) space

O(log $N$) query

# Random Access to $\overbrace{\text{Compressed}}^{\text{Grammar}}$ Strings



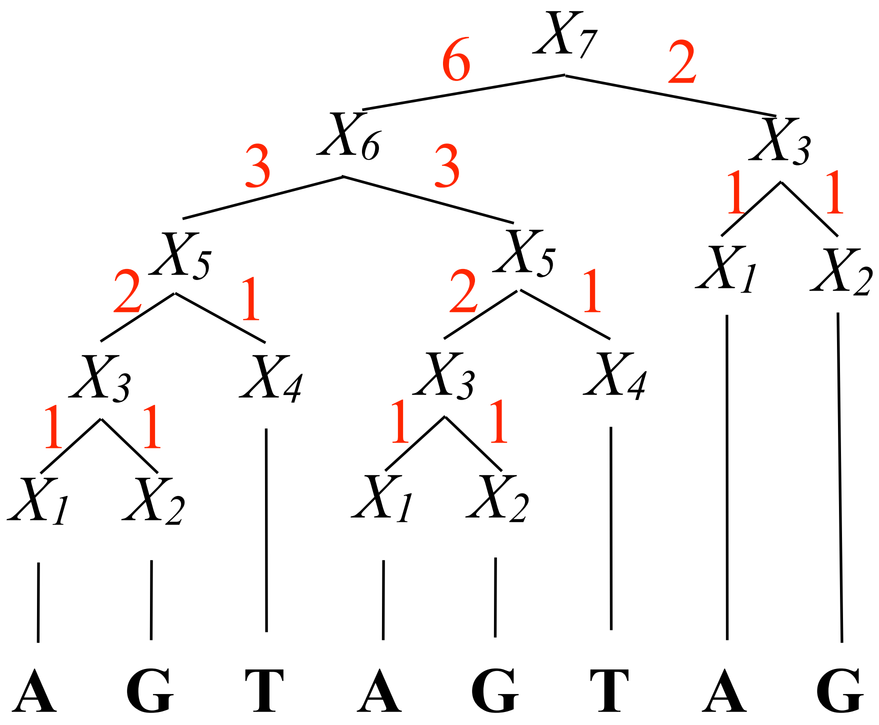
$n = 7$

$N = 8$

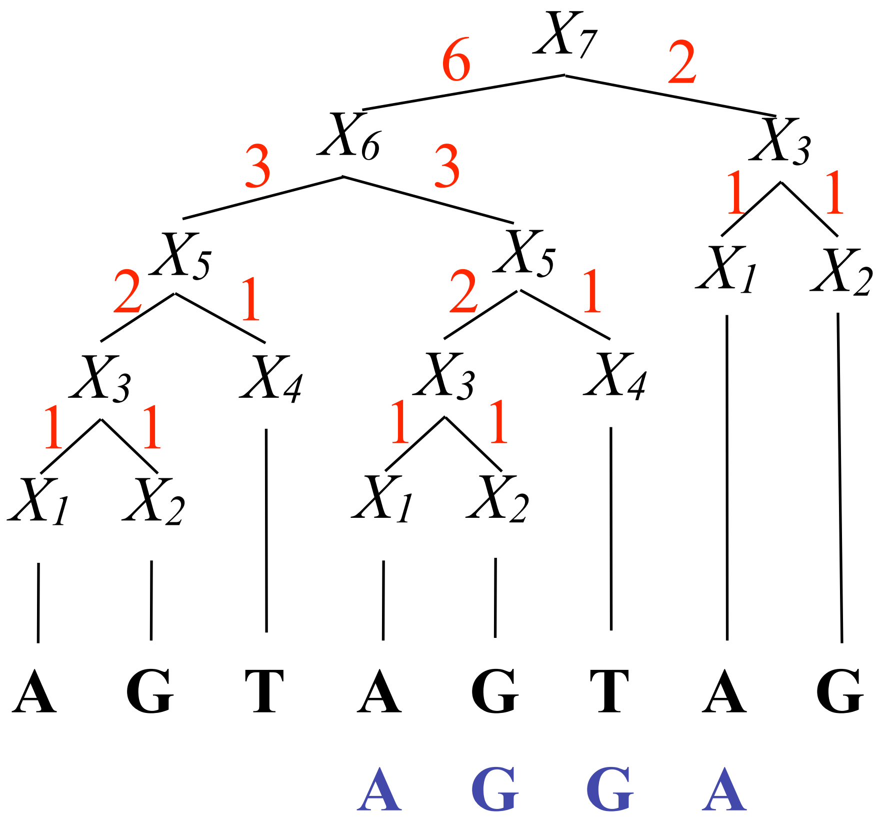
- What is the i'th character ?
- What is the substring [i,j] ?

O($n$) space
O(log $N$ + j-i) query

# Application:
# Black-box compressed pattern matching

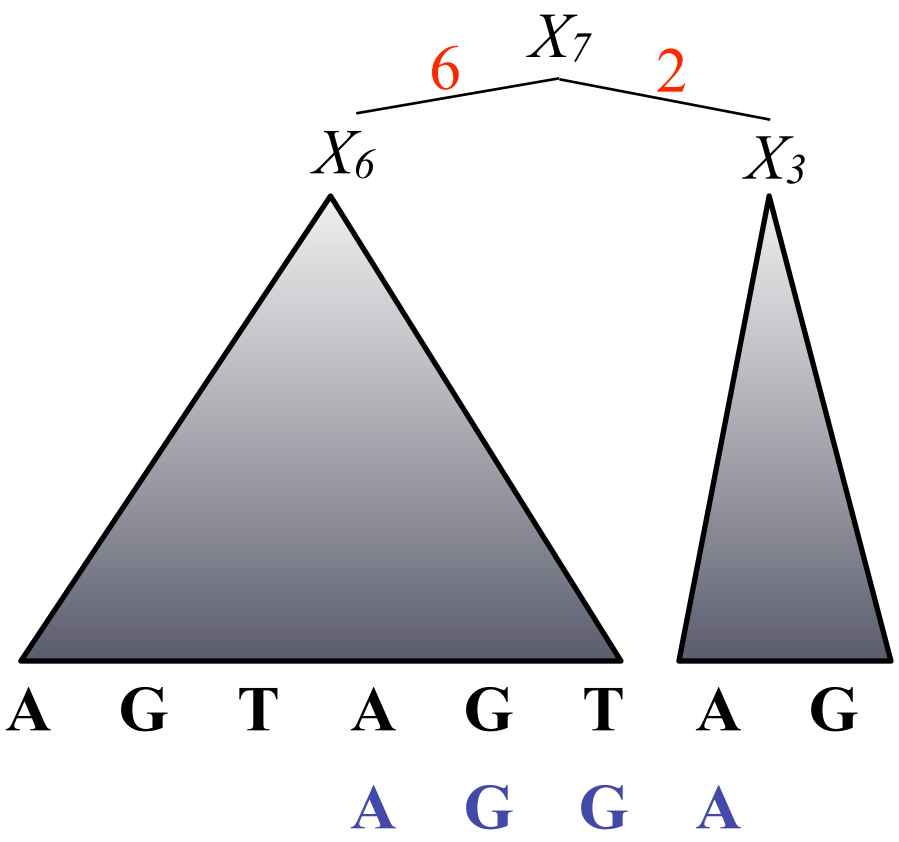


# Application: Black-box compressed pattern matching



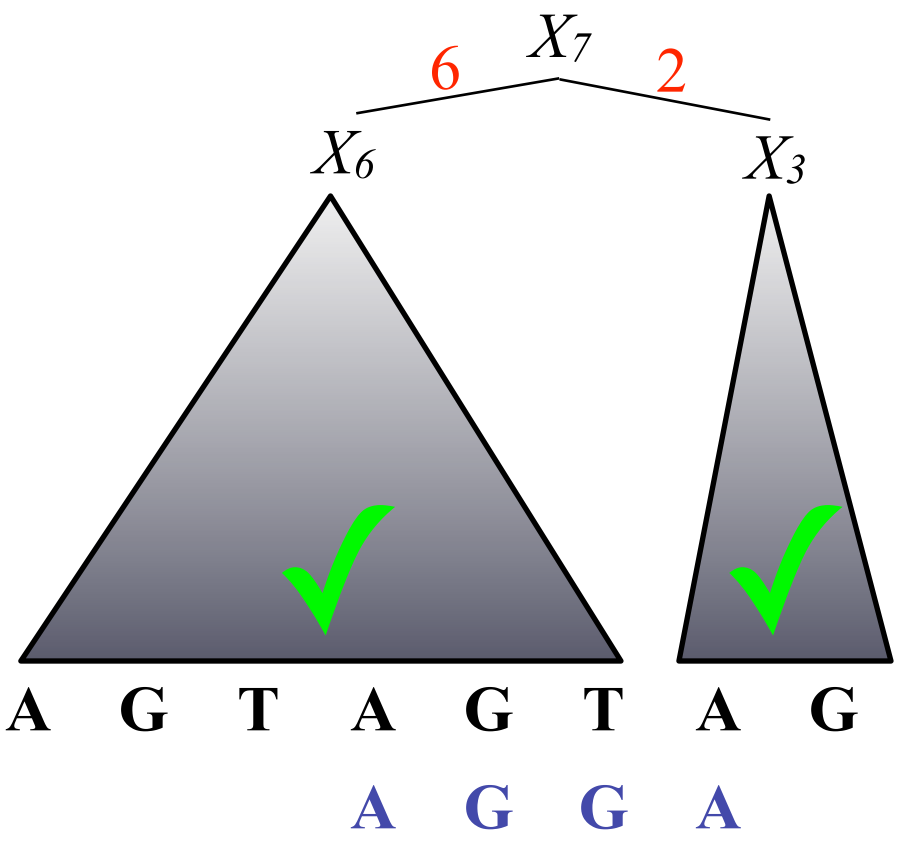
- Does the pattern "A G G A" appear in the text?
    - perhaps with $k$ errors

# Application:
# Black-box compressed pattern matching



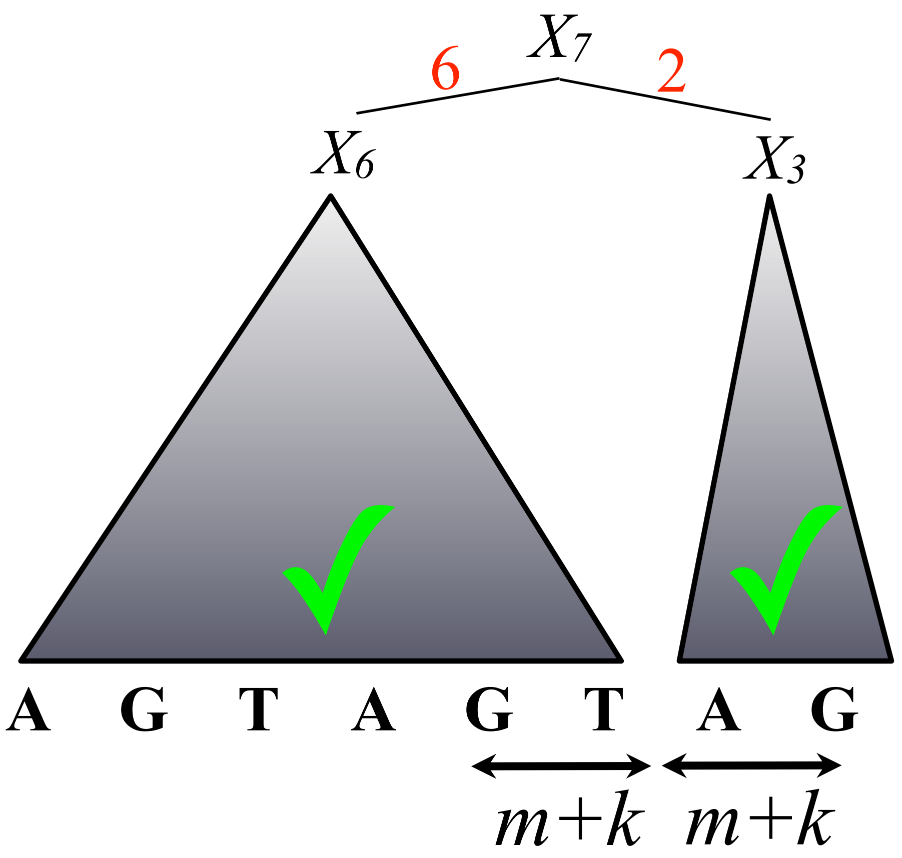
- Does the pattern "**A G G A**" appear in the text?
  - perhaps with $k$ errors

# Application: Black-box compressed pattern matching



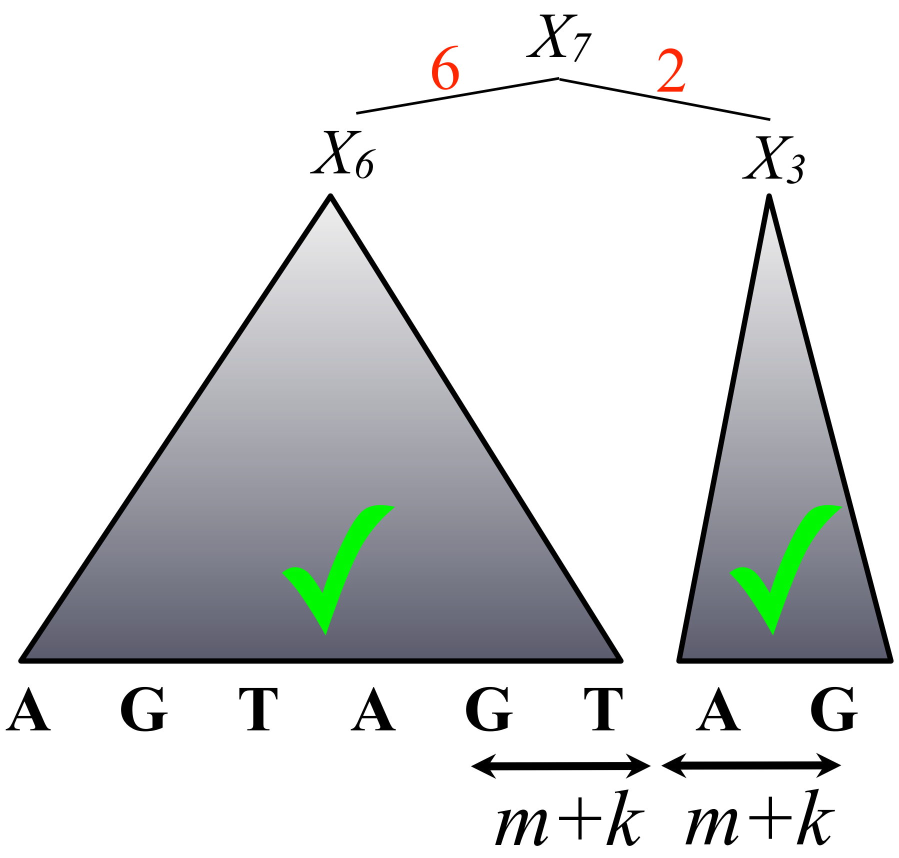
- Does the pattern "**A G G A**" appear in the text?
  - perhaps with $k$ errors

# Application:
# Black-box compressed pattern matching



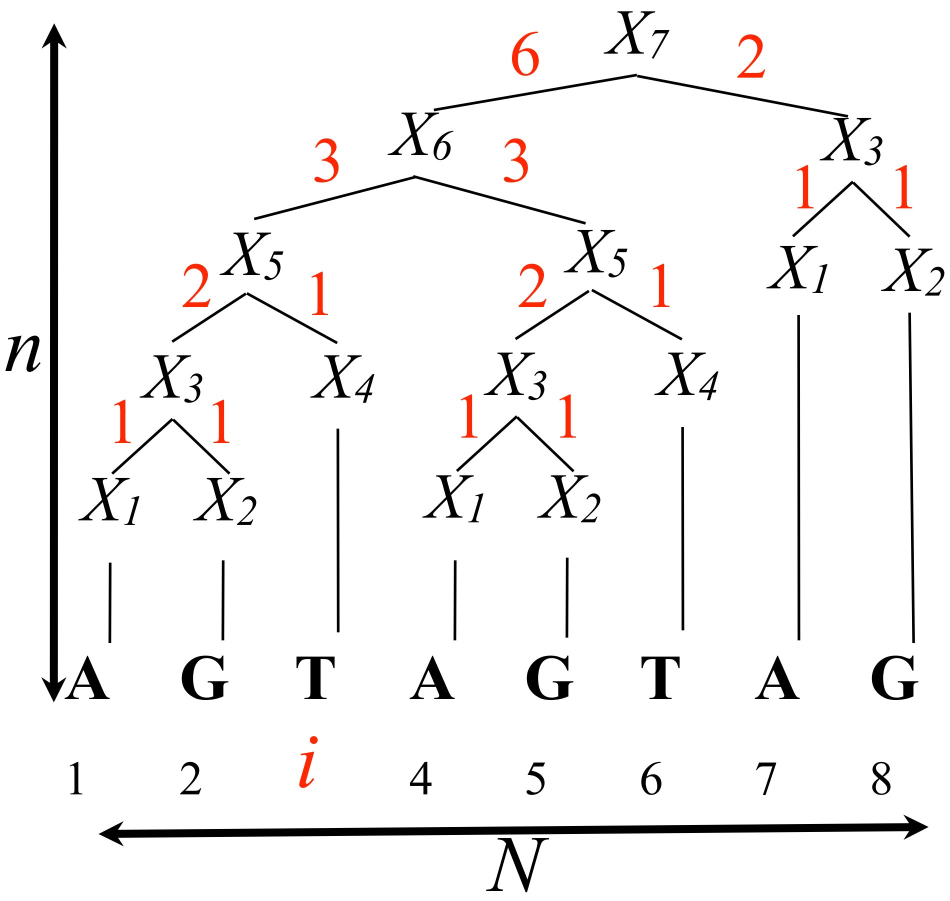
- Does the pattern "**A G G A**" appear in the text?
  - perhaps with $k$ errors

# Application:
# Black-box compressed pattern matching

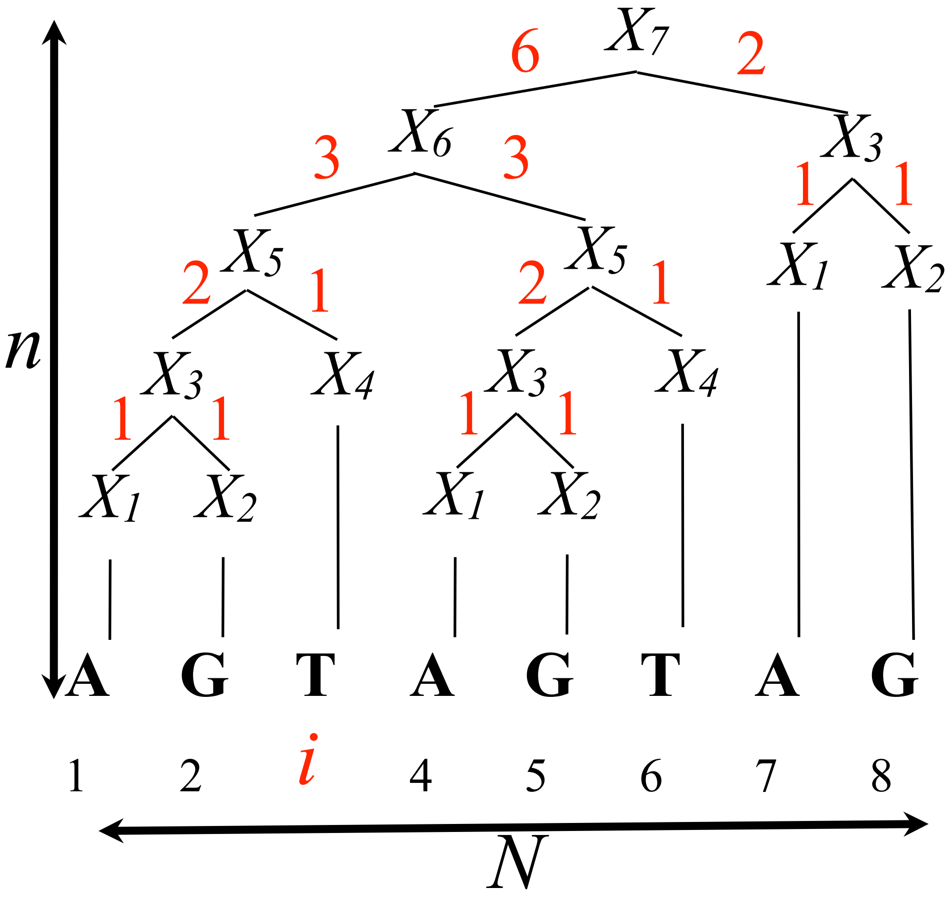


- Does the pattern "**A G G A**" appear in the text?
  - perhaps with $k$ errors
- Total time complexity: $(\log N + m + \text{BlackBox}(m))\, n$

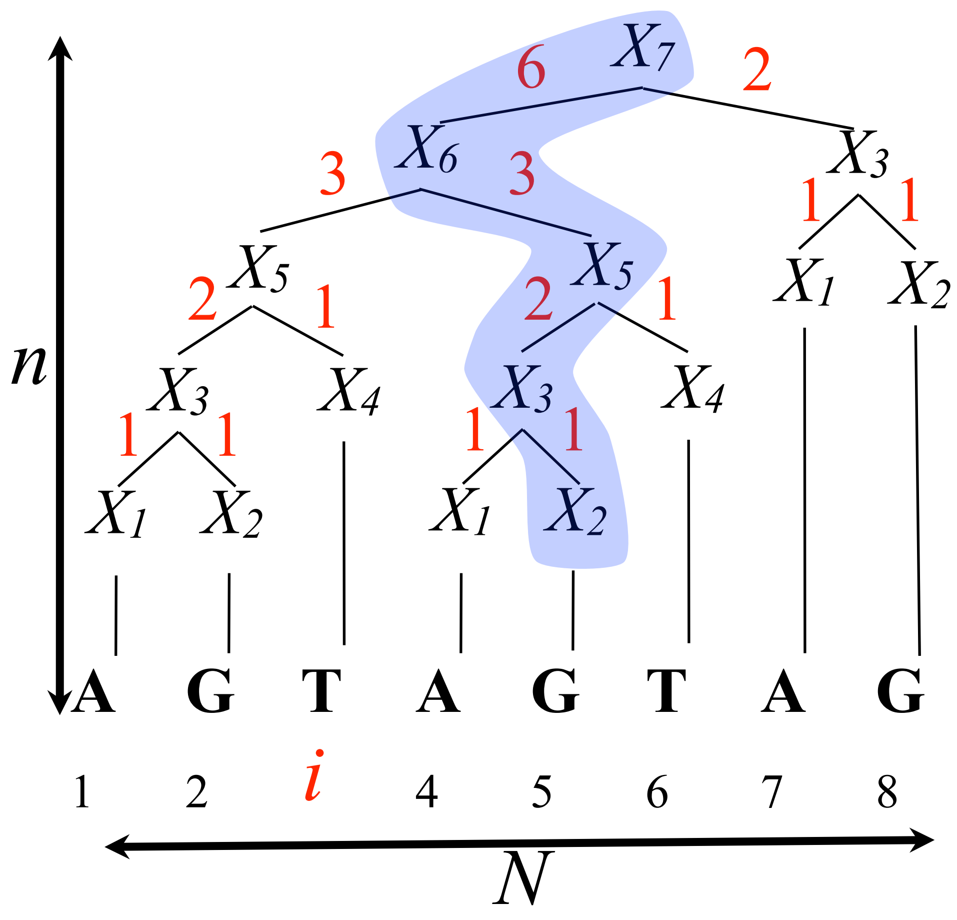
# Our Algorithm



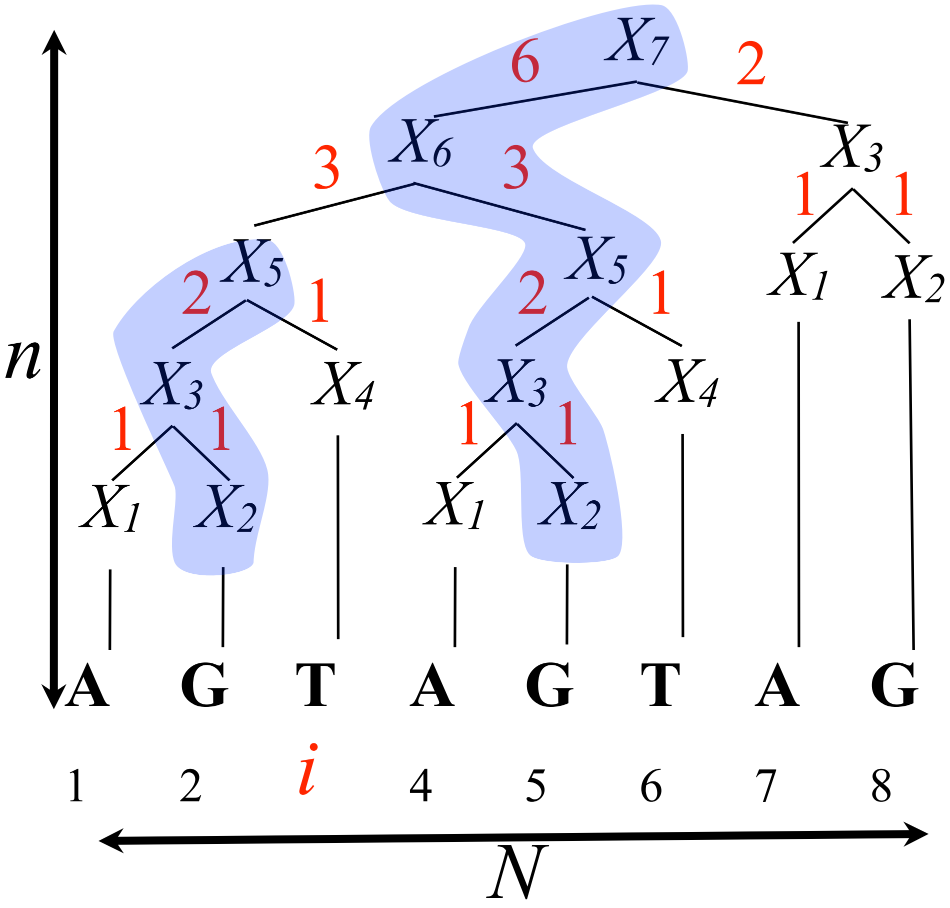
# Heavy Path Decomposition [HT84]



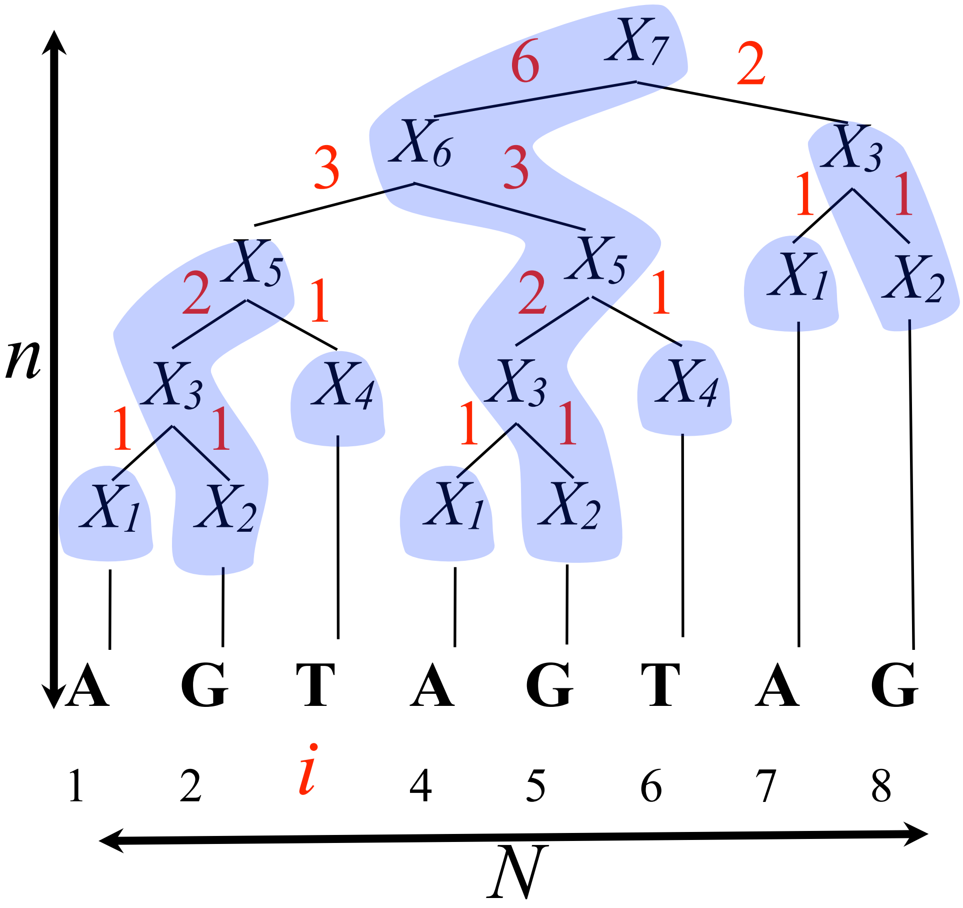
# Heavy Path Decomposition [HT84]



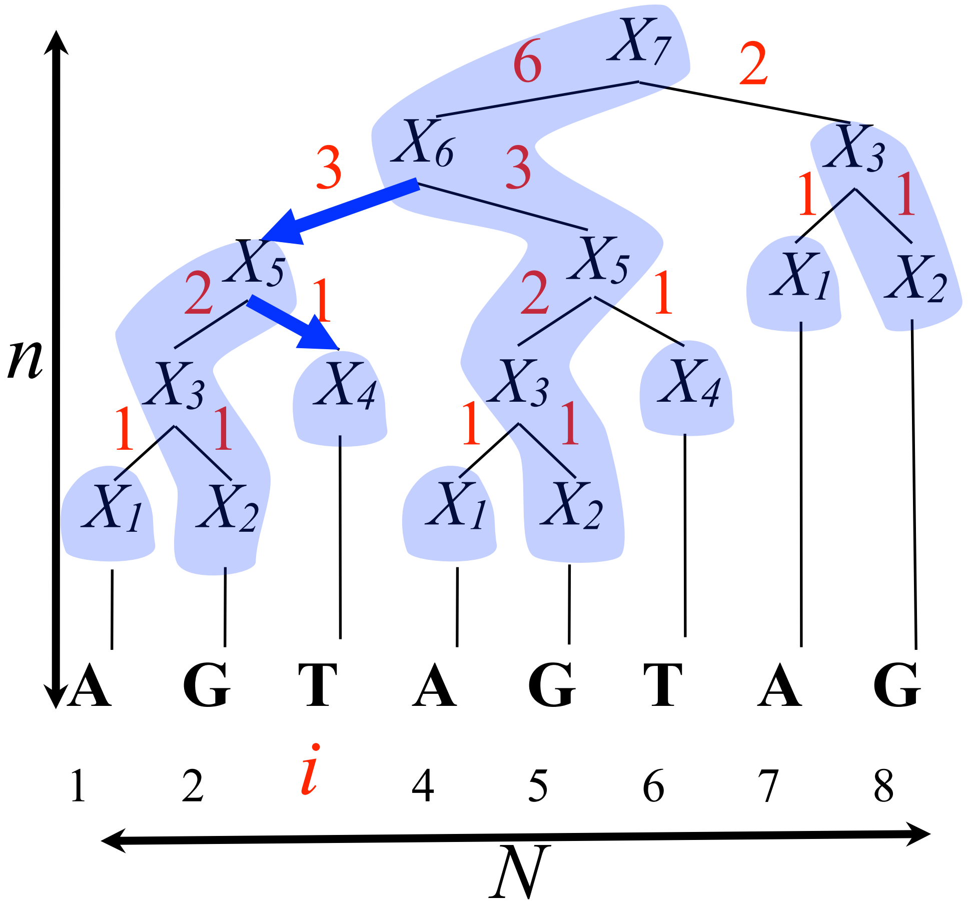
# Heavy Path Decomposition [HT84]



# Heavy Path Decomposition [HT84]

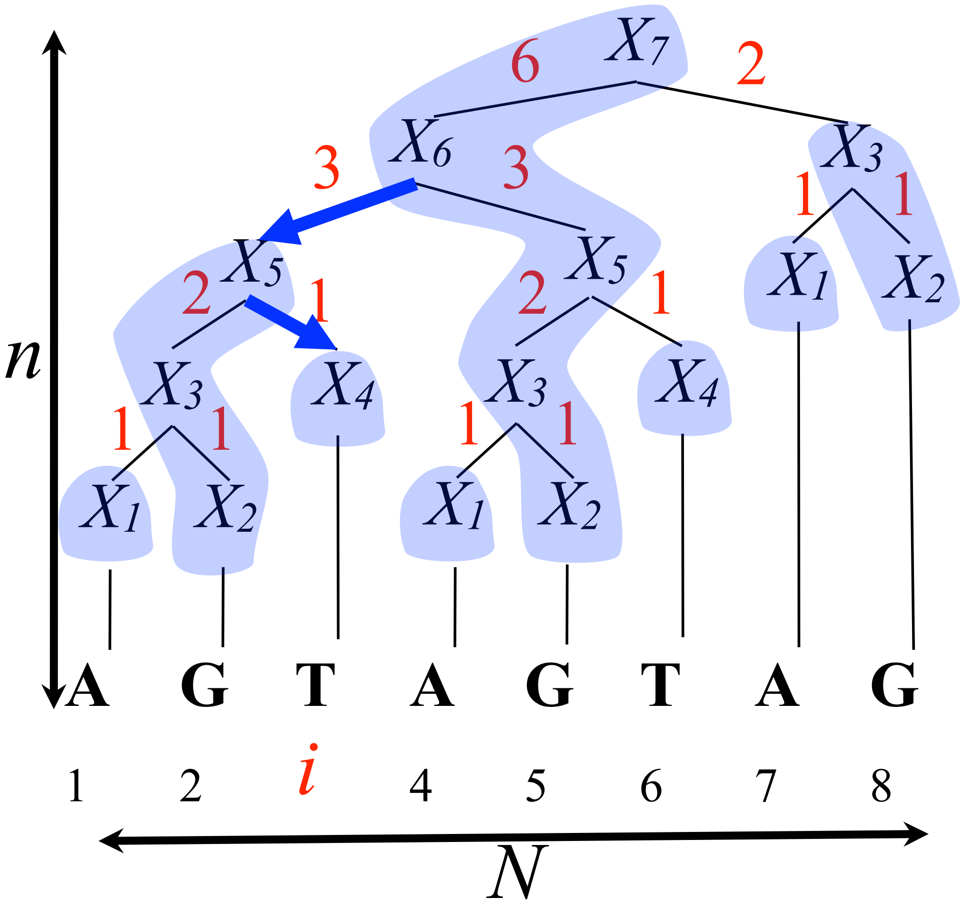


# Heavy Path Decomposition [HT84]



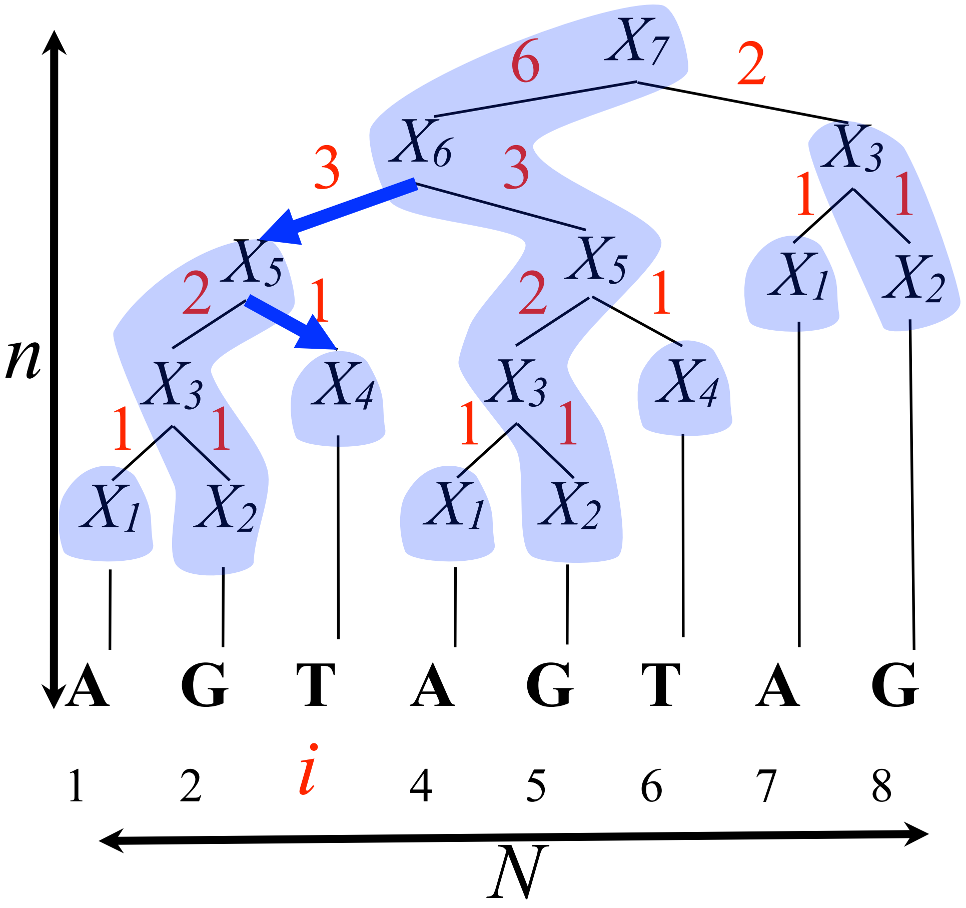
- The way to $i$ goes through O(log $N$) paths

# Random Access Query



- The way to *i* goes through O(log *N*) paths
- Query: binary-search all paths on the way

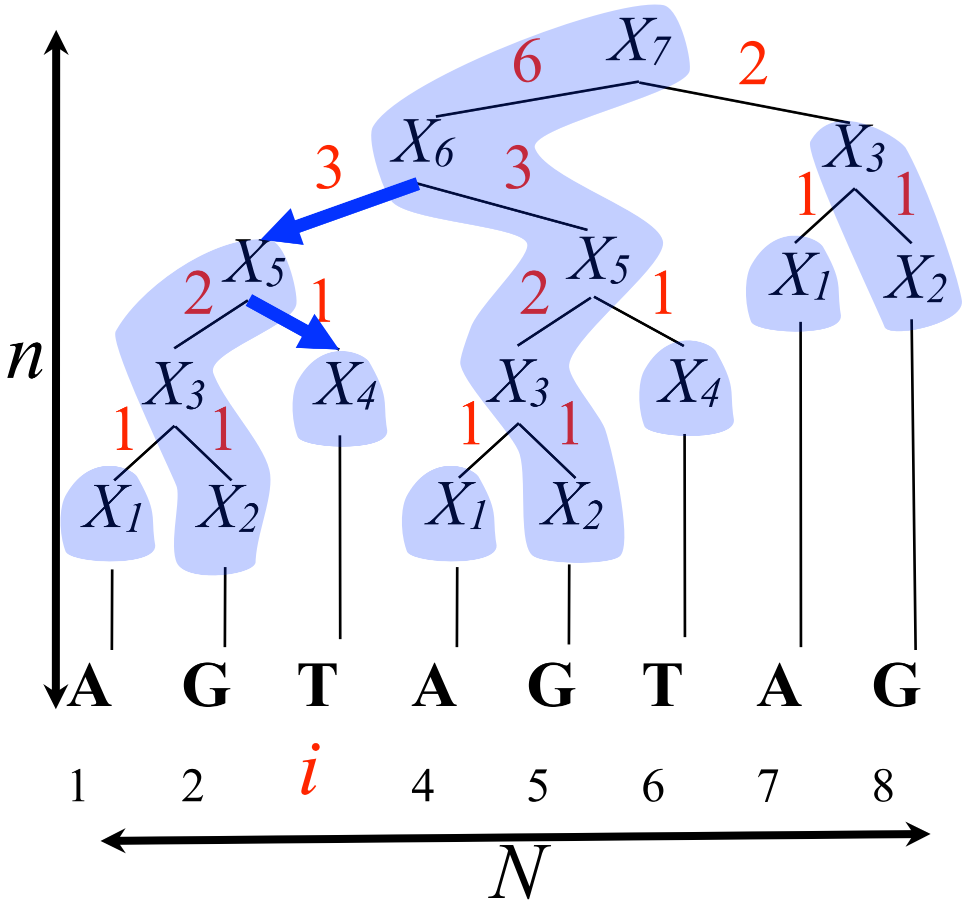
# Random Access Query



- The way to *i* goes through O(log *N*) paths
- Query: binary-search all paths on the way

O(log *n*)

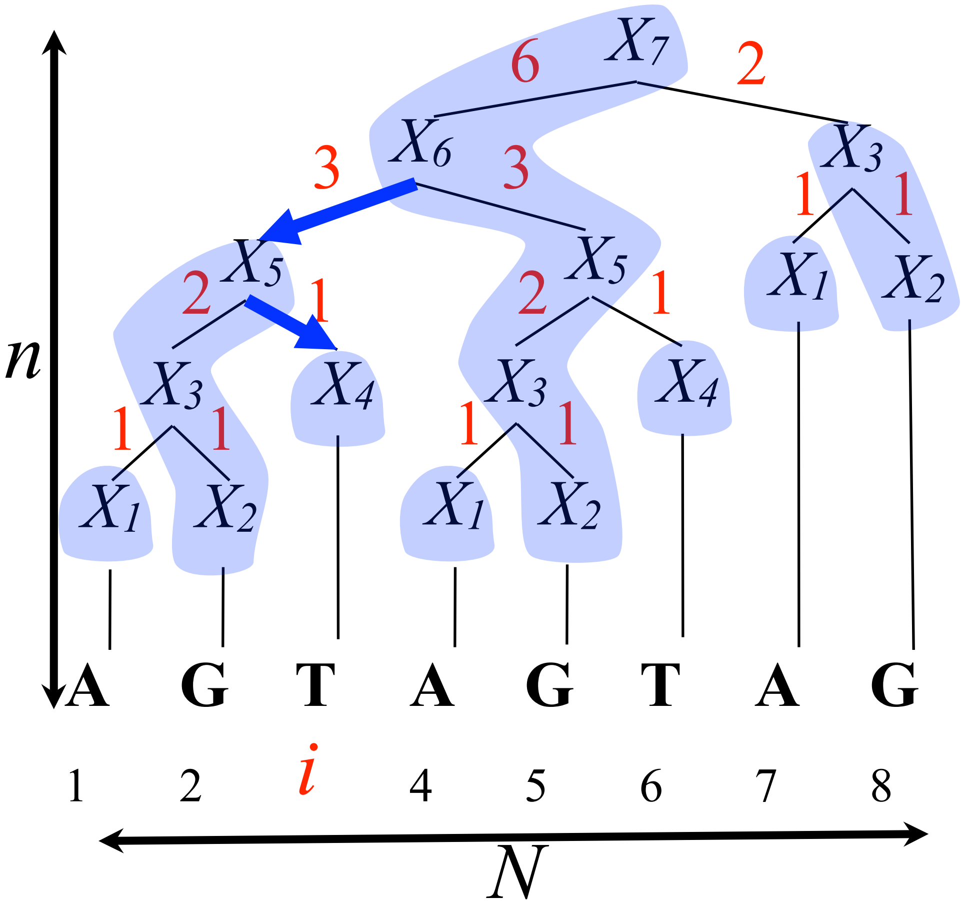
# Random Access Query



- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way

$$O(\log n) \cdot O(\log N)$$

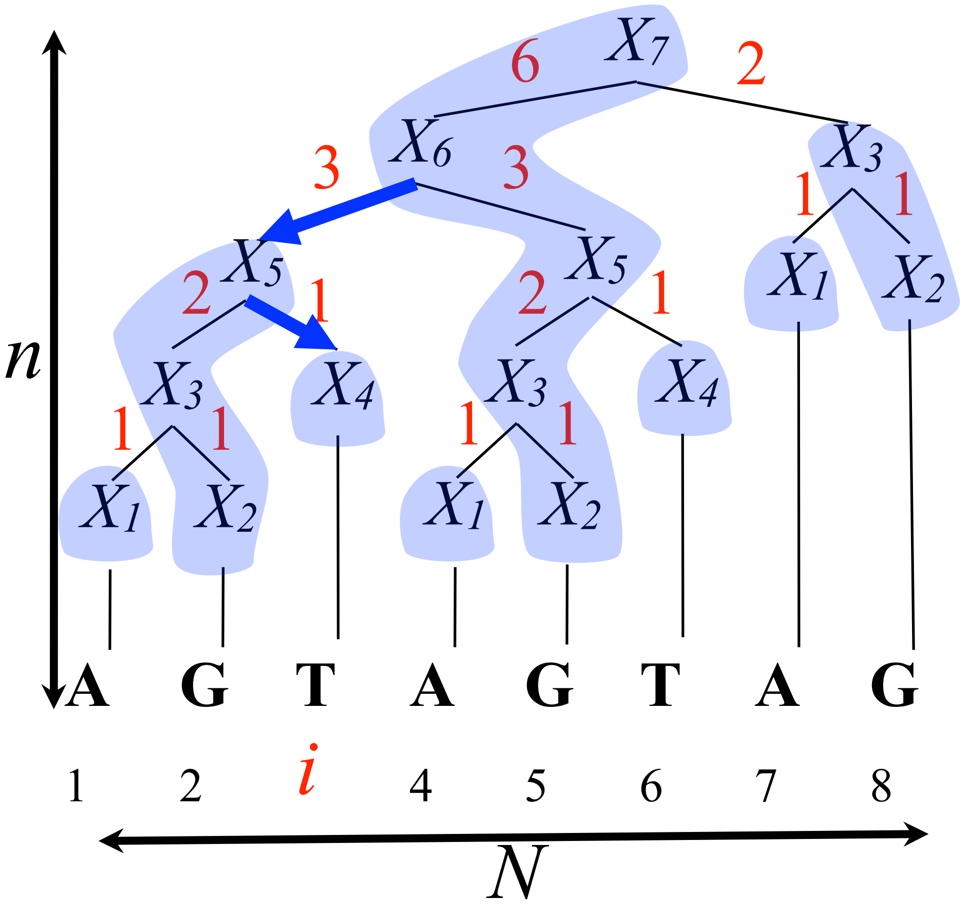
# Random Access Query



- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

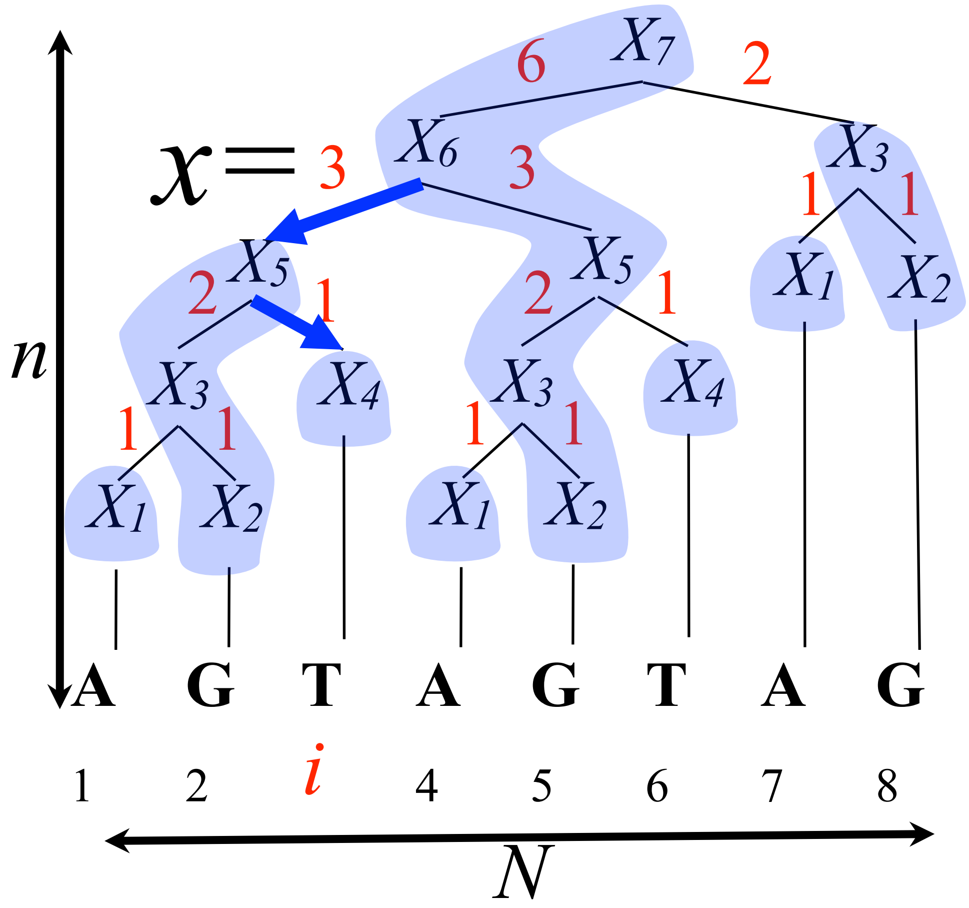
# Interval Biased Search Tree



- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

# Interval Biased Search Tree

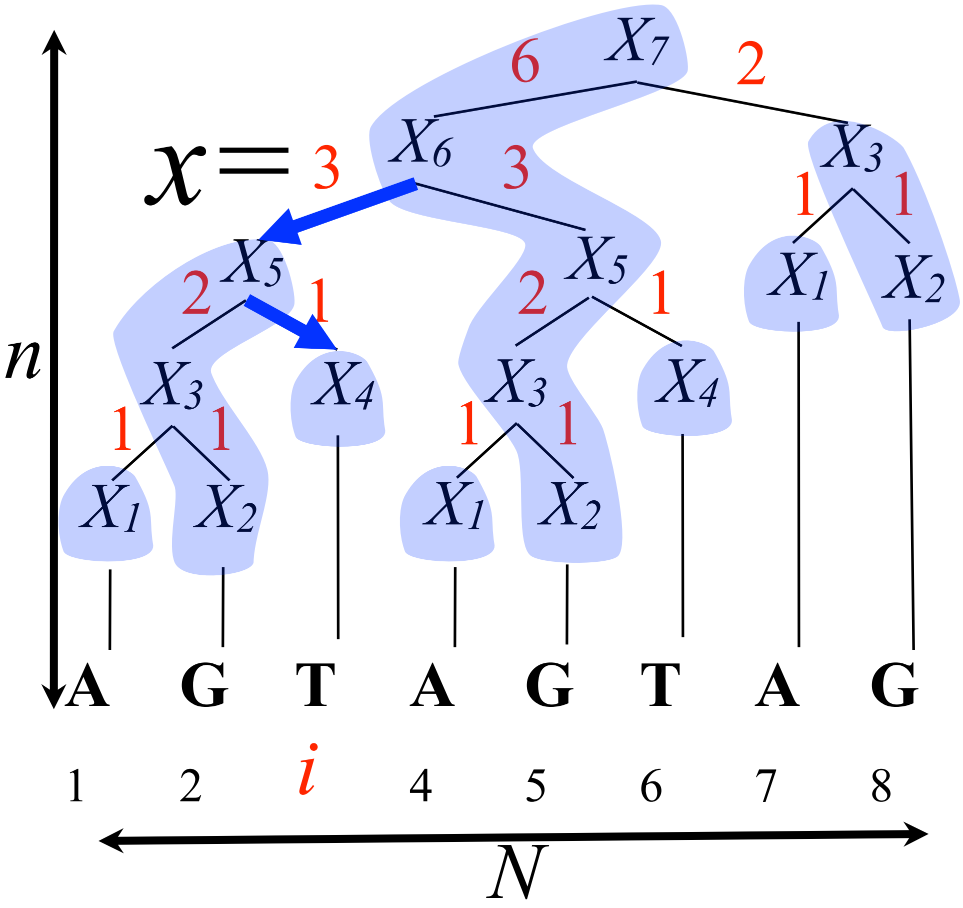


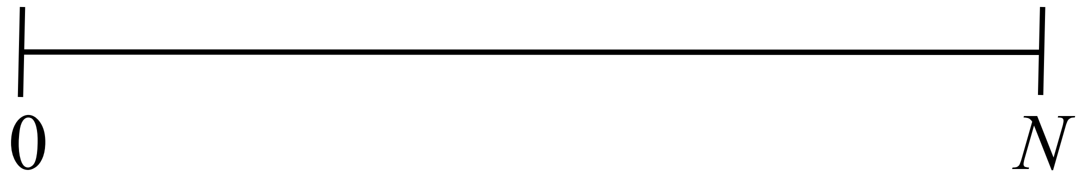
- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

$$O(\log N/x)$$

# Interval Biased Search Tree



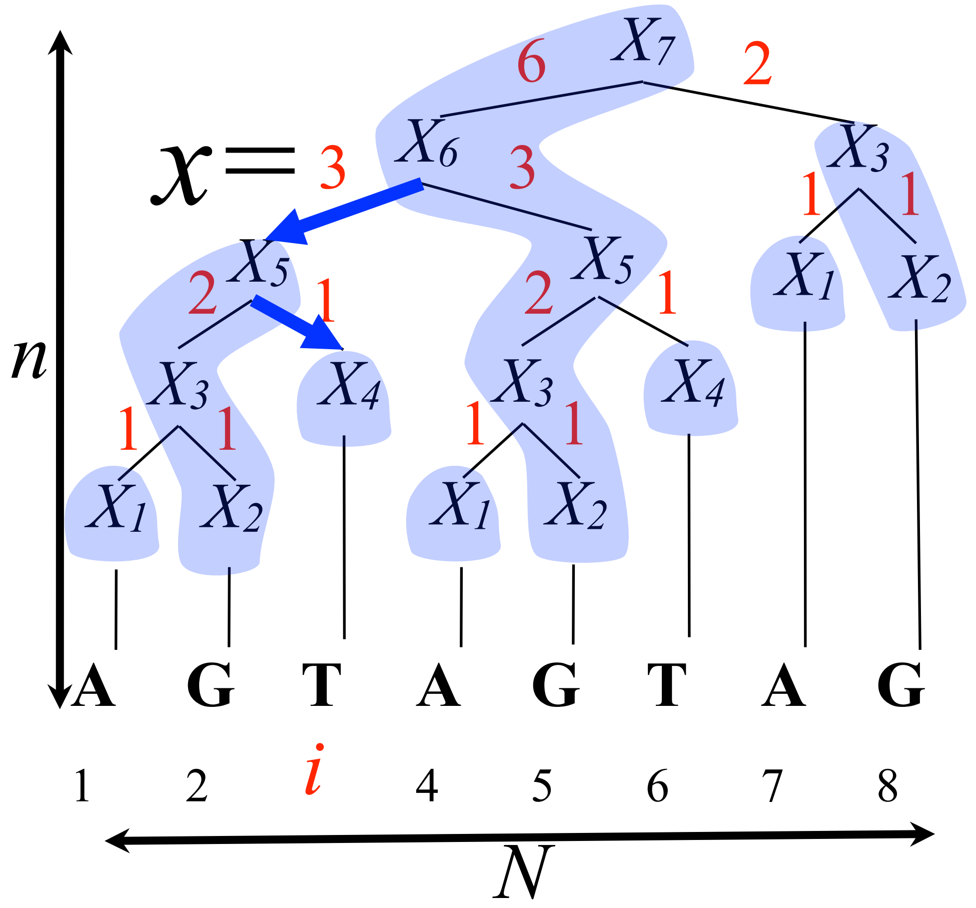


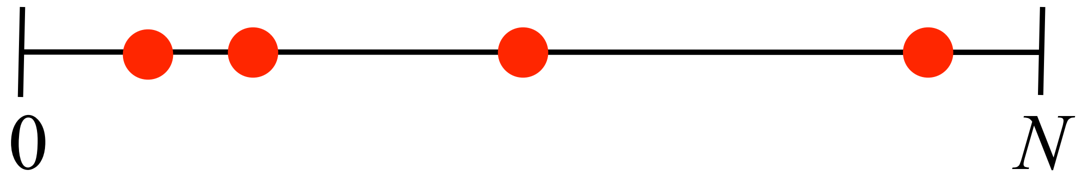
- The way to $i$ goes through $O(\log N)$ paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

$$O(\log N/x)$$

# Interval Biased Search Tree



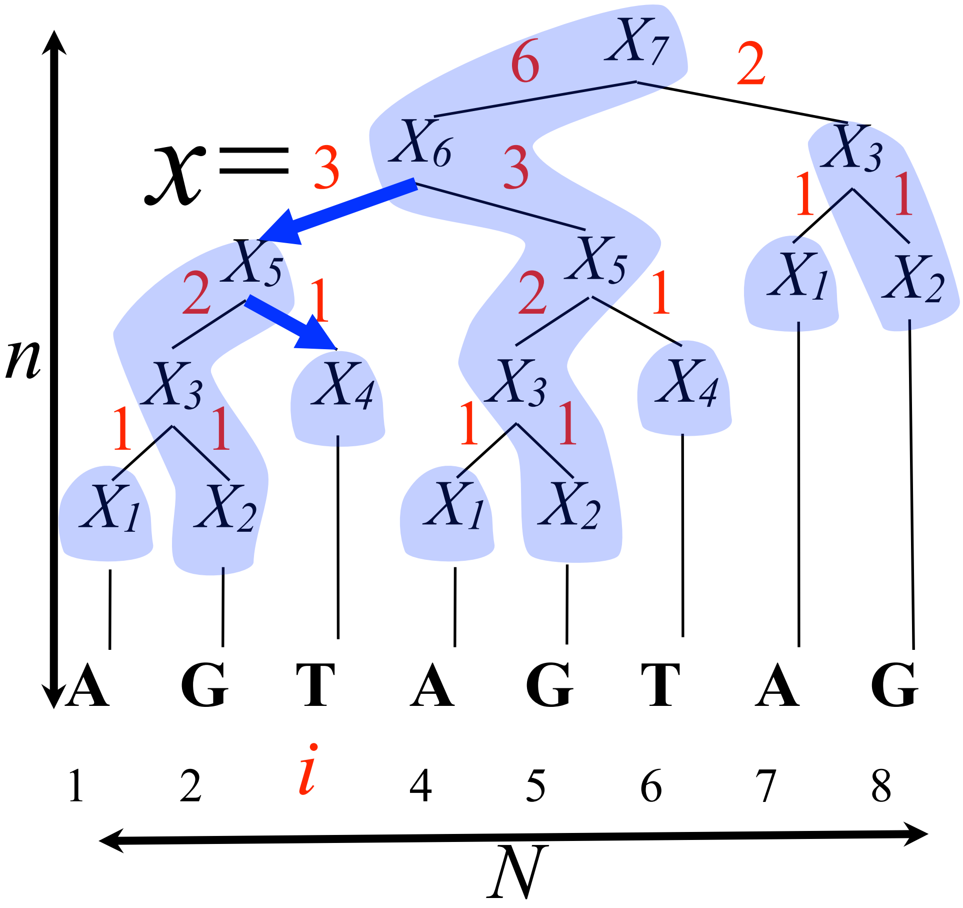


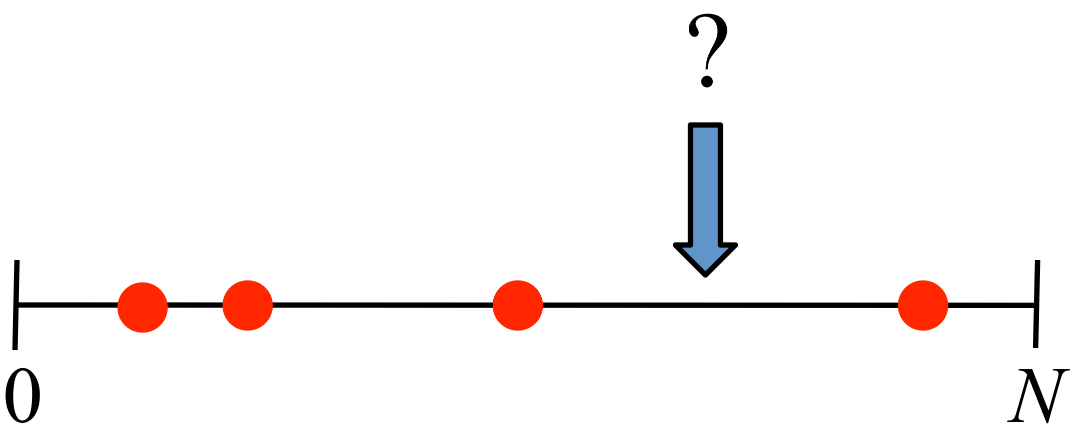
- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way

~~$O(\log n)$~~ $\cdot\ O(\log N)$

$O(\log N/x)$

# Interval Biased Search Tree



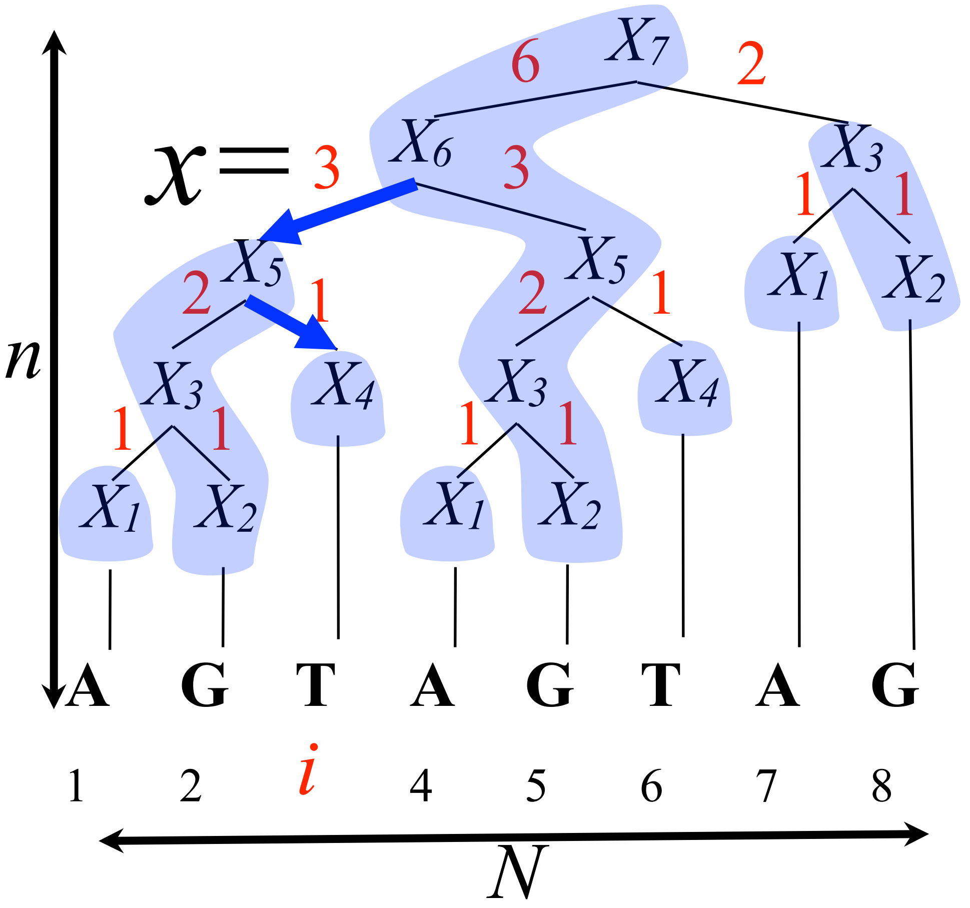


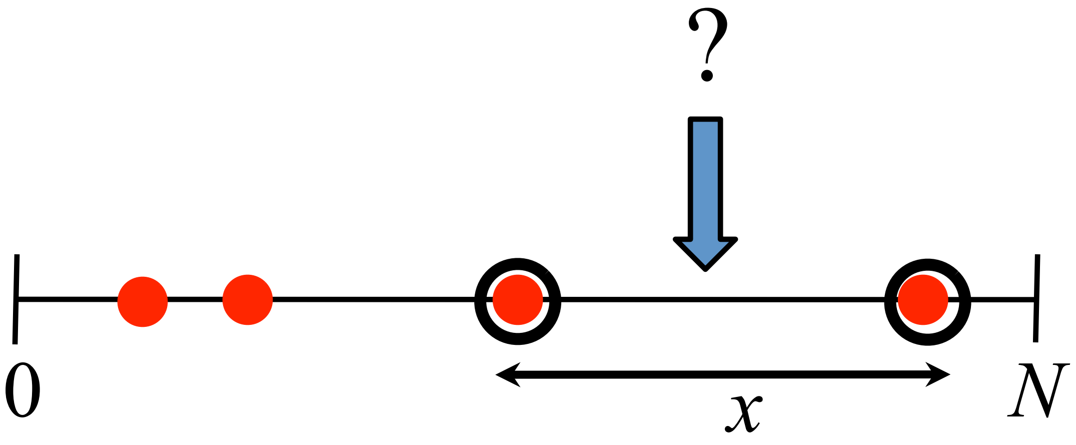
- The way to *i* goes through O(log *N*) paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

$$O(\log N/x)$$

# Interval Biased Search Tree



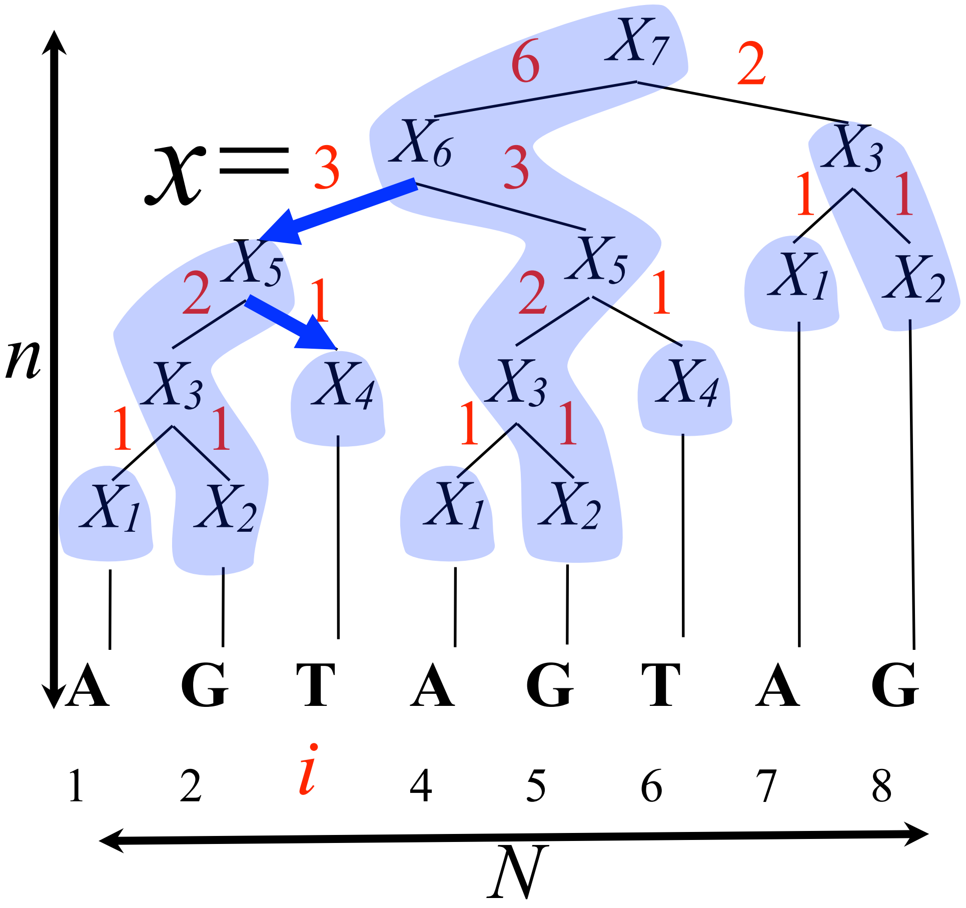


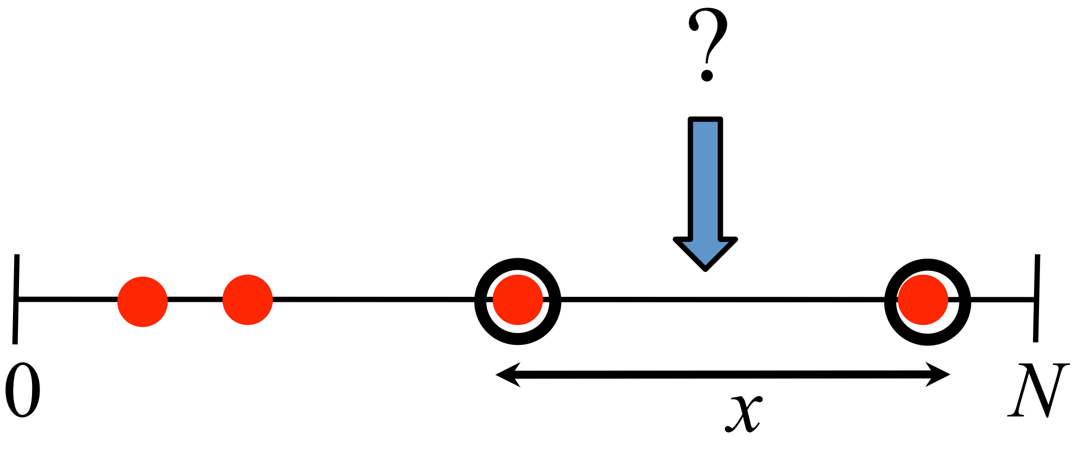
- The way to $i$ goes through $O(\log N)$ paths
- Query: binary-search all paths on the way

$$\cancel{O(\log n)} \cdot O(\log N)$$

$$O(\log N/x)$$

# Interval Biased Search Tree



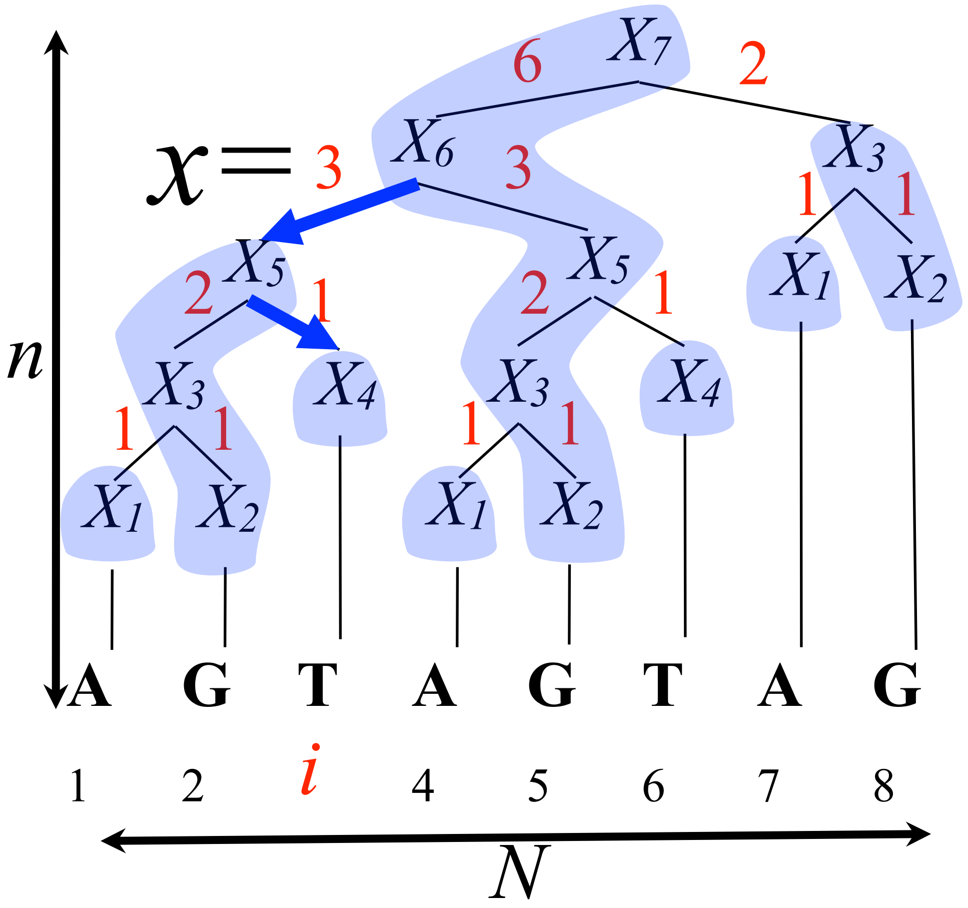


- The way to *i* goes through O(log *N*) paths
- Query: binary-search all paths on the way

~~O(log *n*)~~ · O(log *N*)

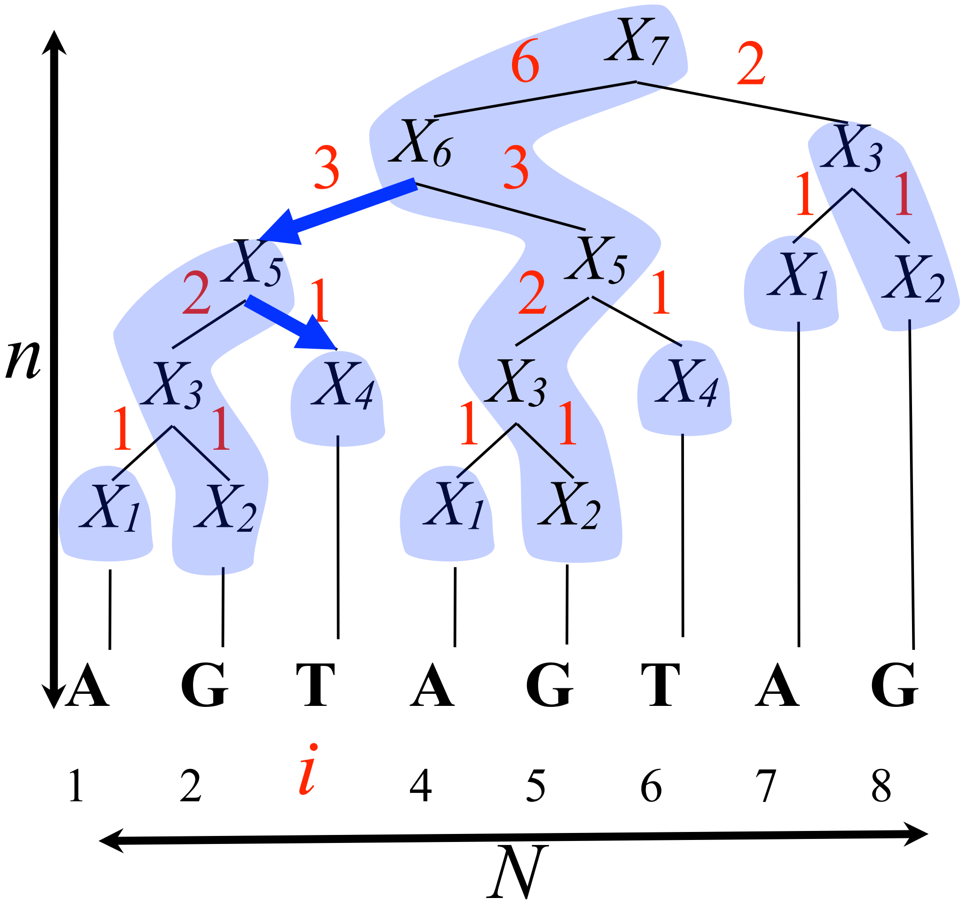
O(log *N*/*x*) telescopes to O(log *N*)

# Interval Biased Search Tree

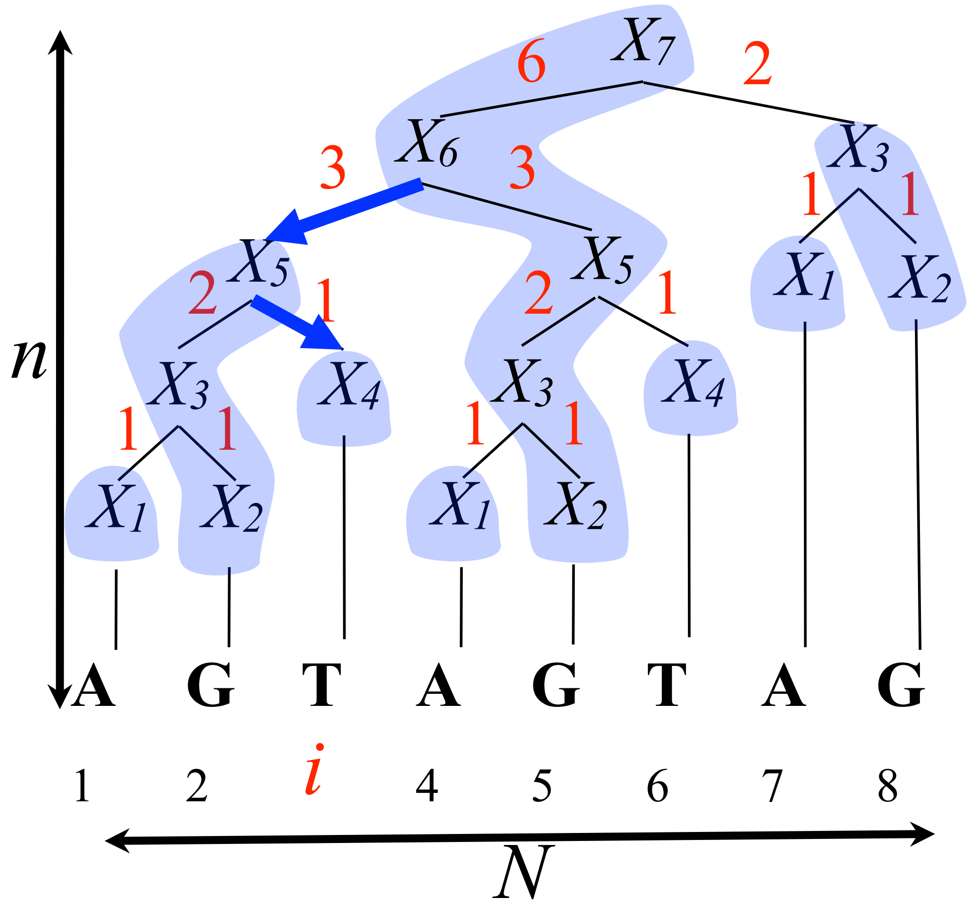


- The way to $i$ goes through O(log $N$) paths
- Query: binary-search all paths on the way
- Space: can't actually store all paths

# O($n$) Representation of Heavy Paths

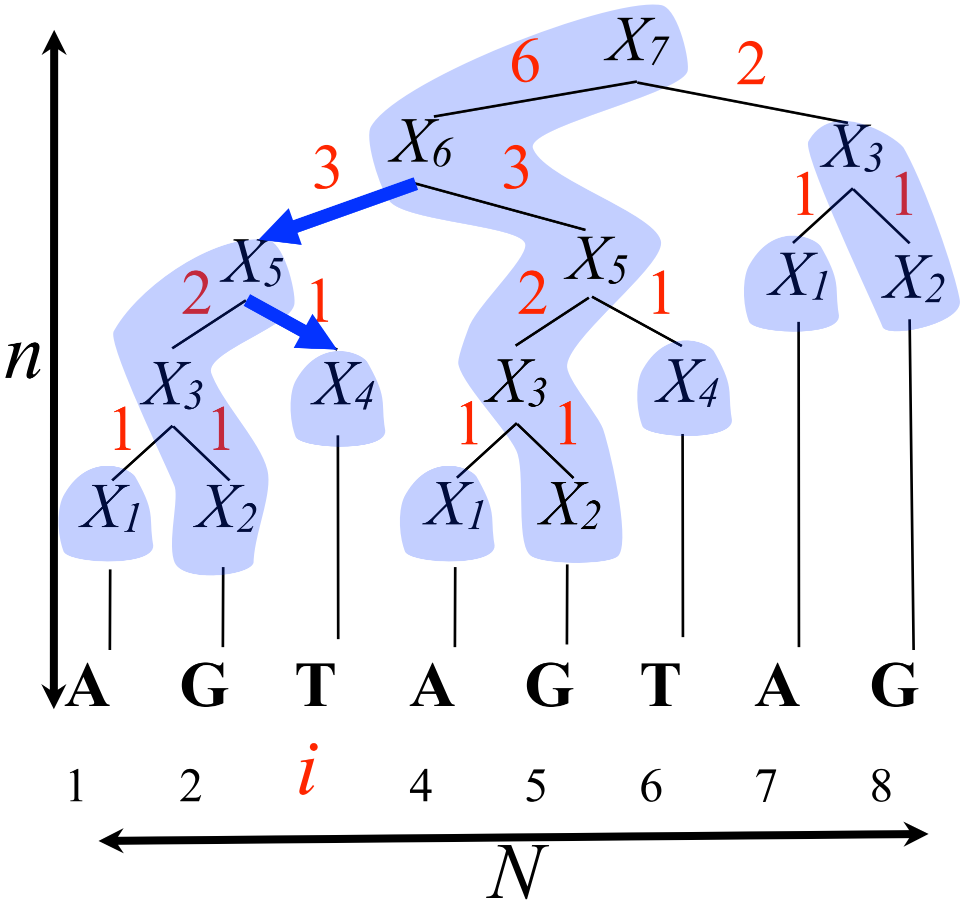


# O($n$) Representation of Heavy Paths



$X_7$

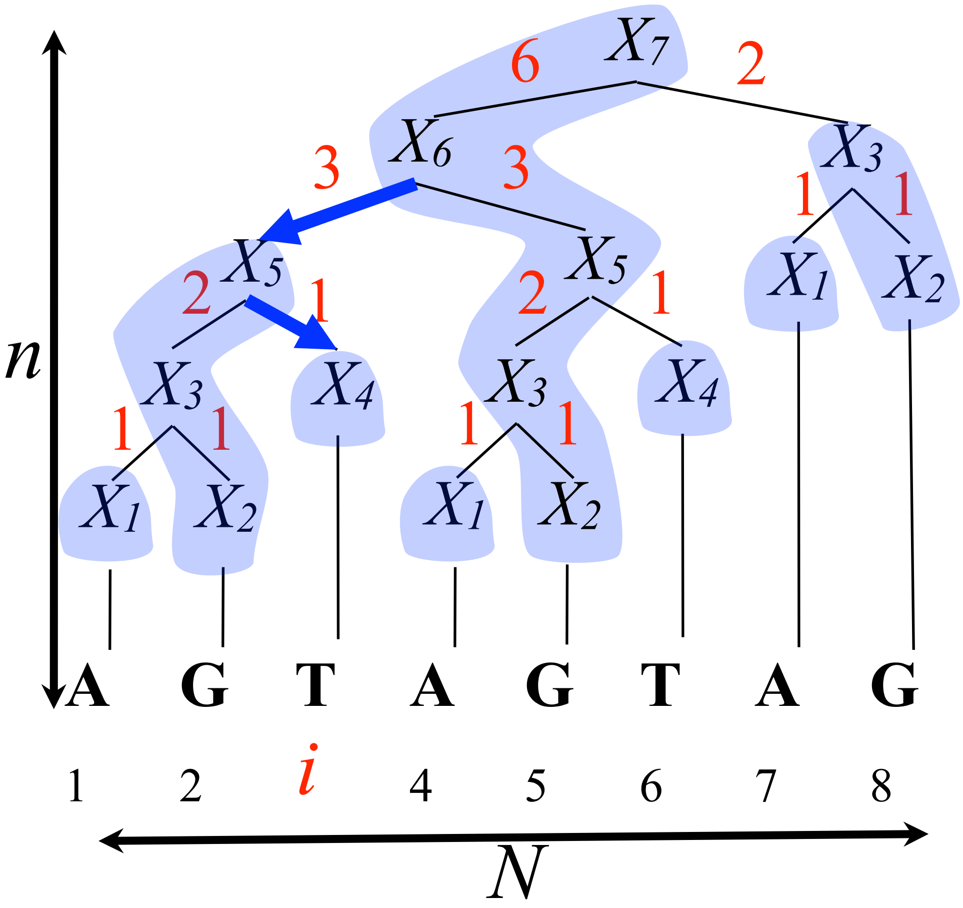
# O($n$) Representation of Heavy Paths

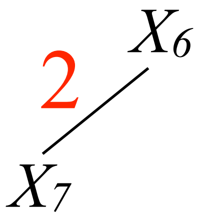


$X_6$

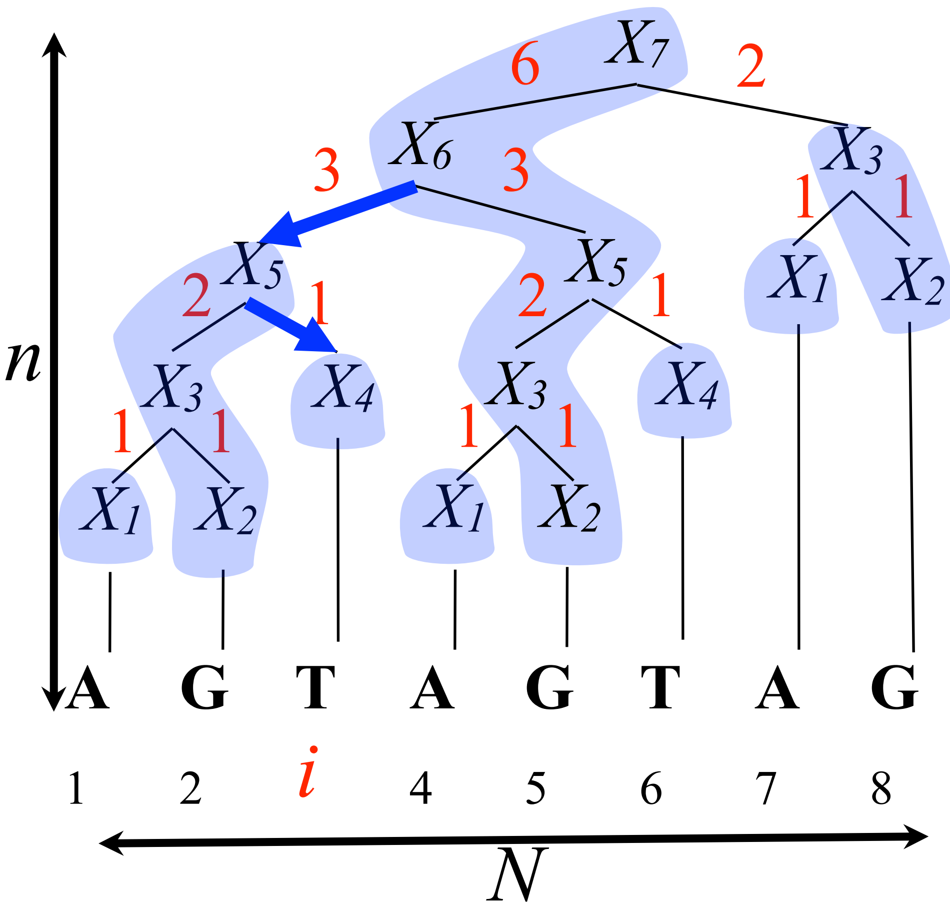
$X_7$

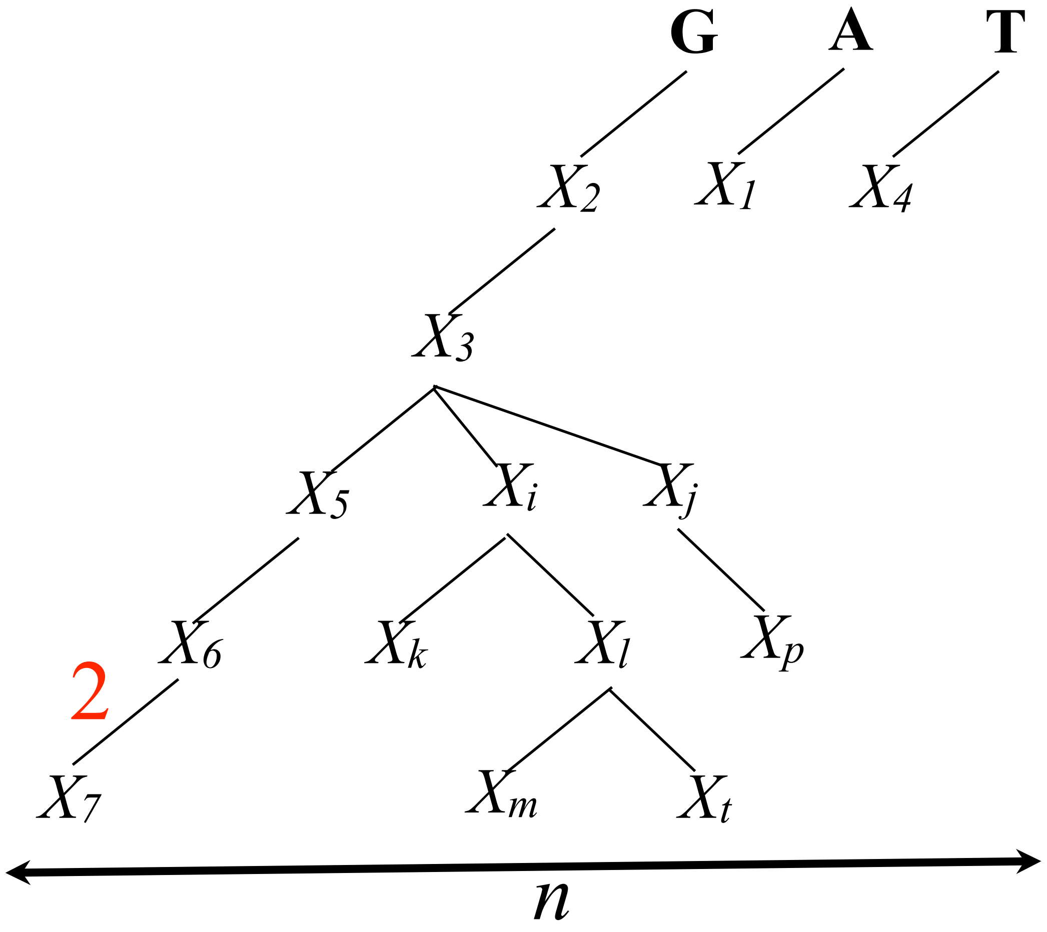
# O($n$) Representation of Heavy Paths



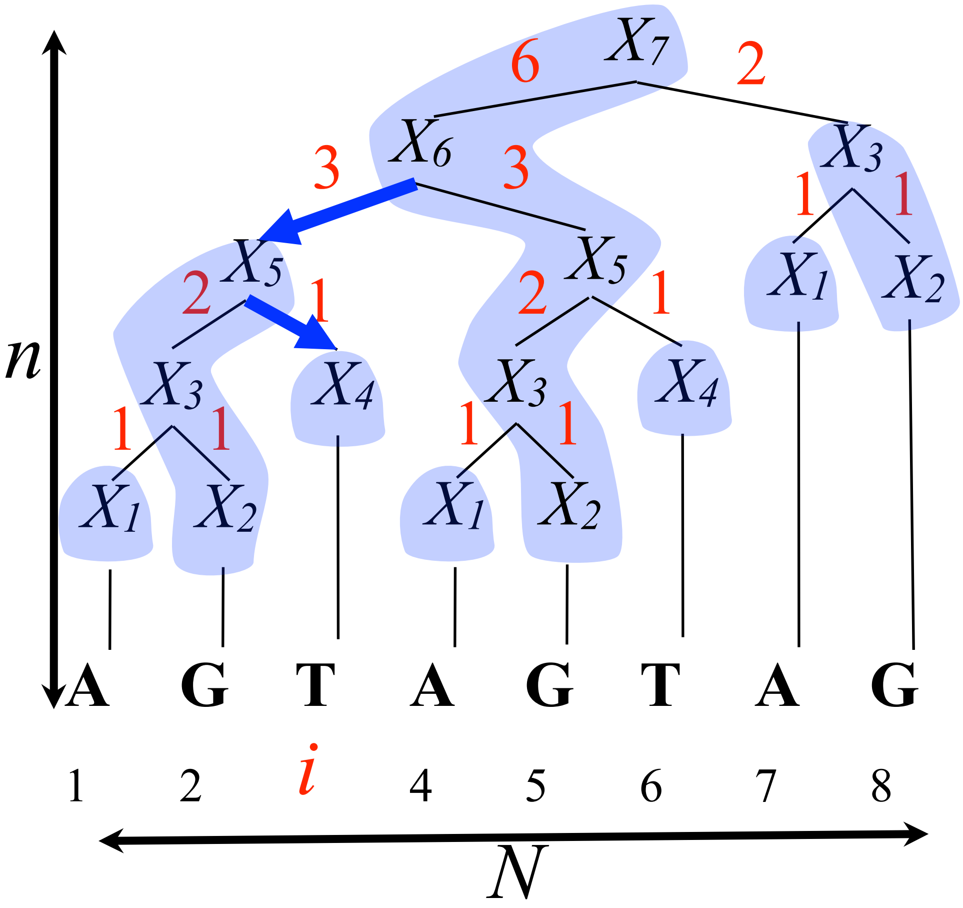


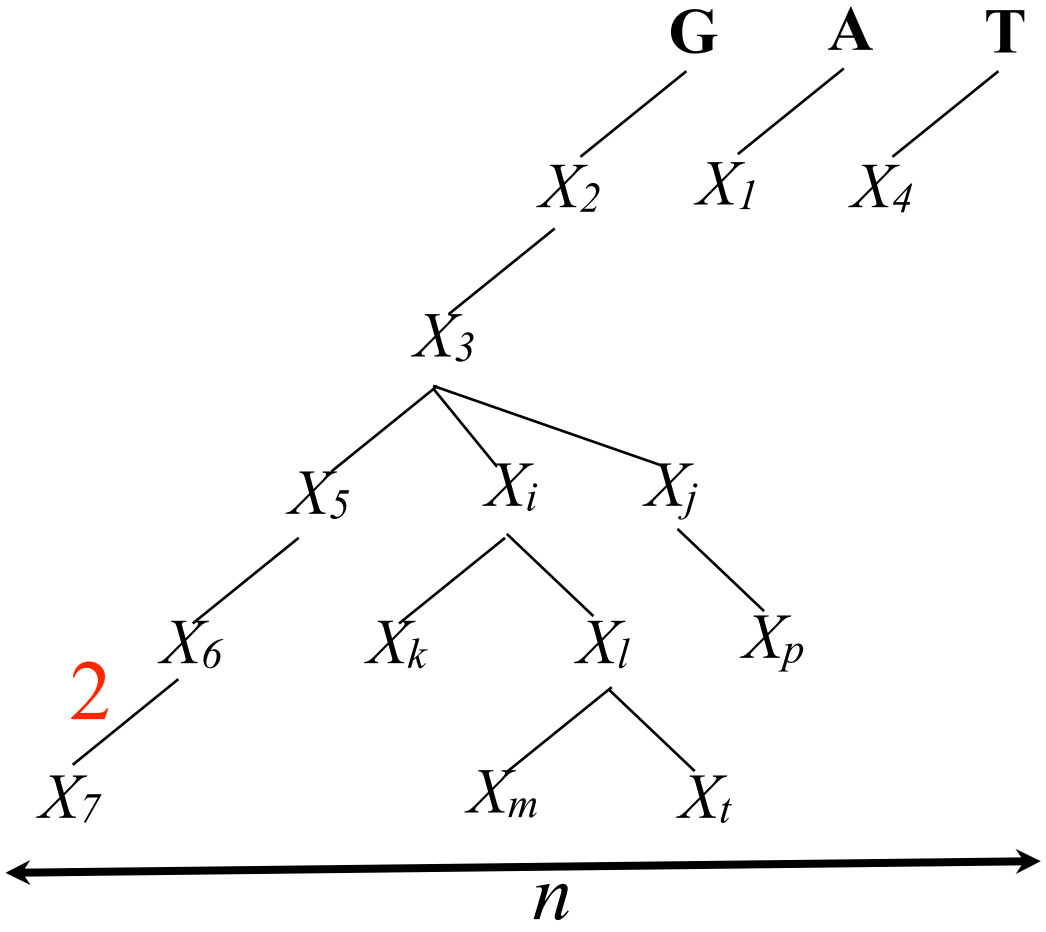
# O($n$) Representation of Heavy Paths





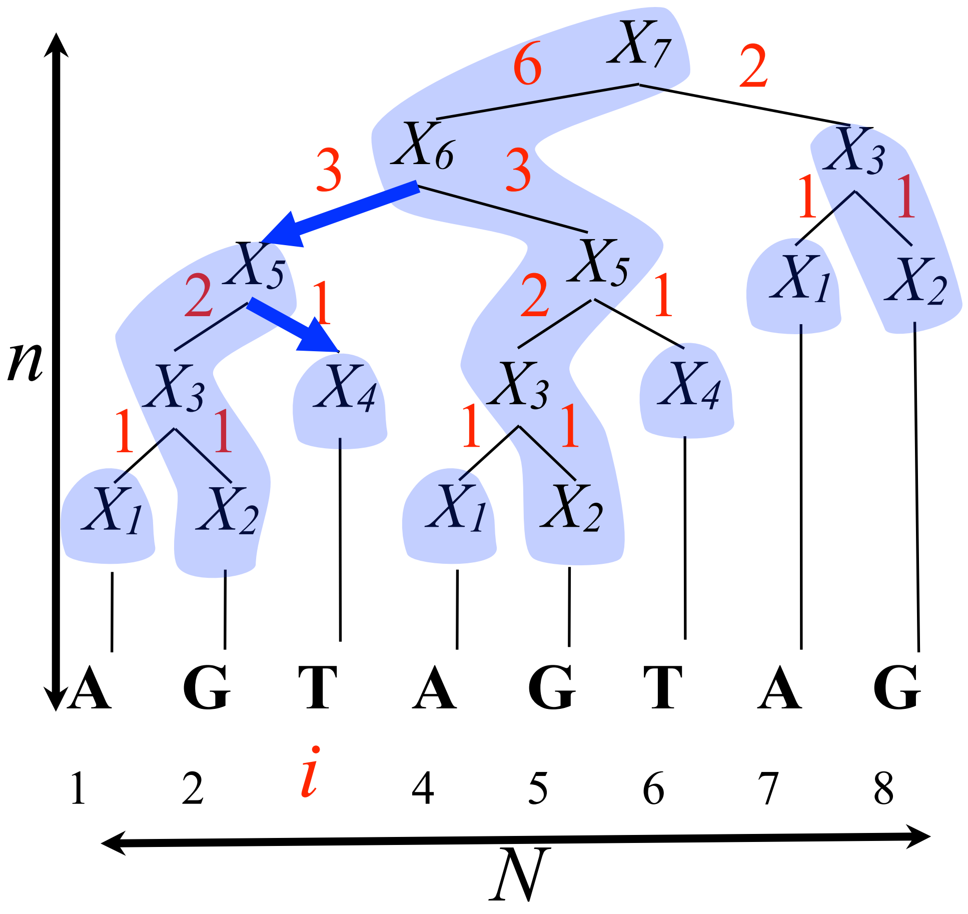
# O($n$) Representation of Heavy Paths

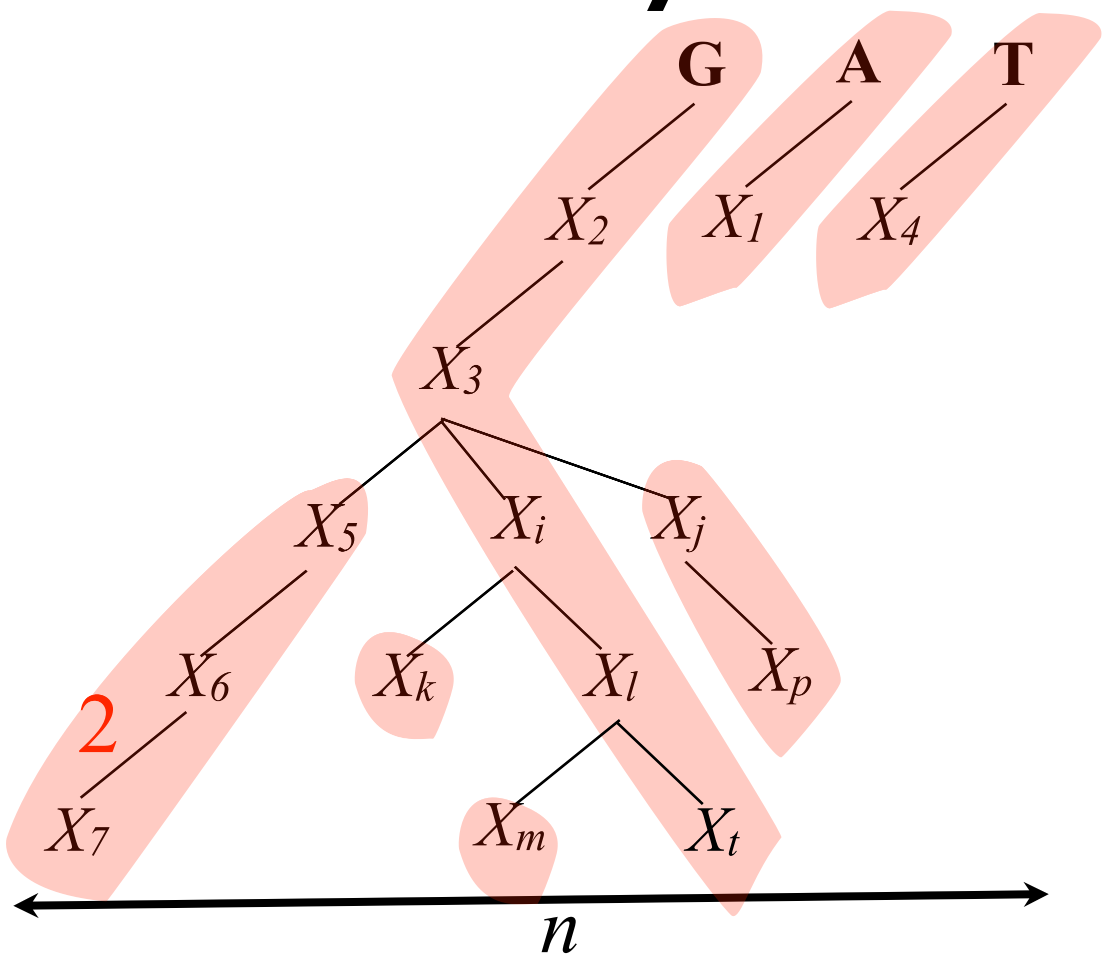




- Binary-search for $i$ = Lowest ancestor of distance $i$.

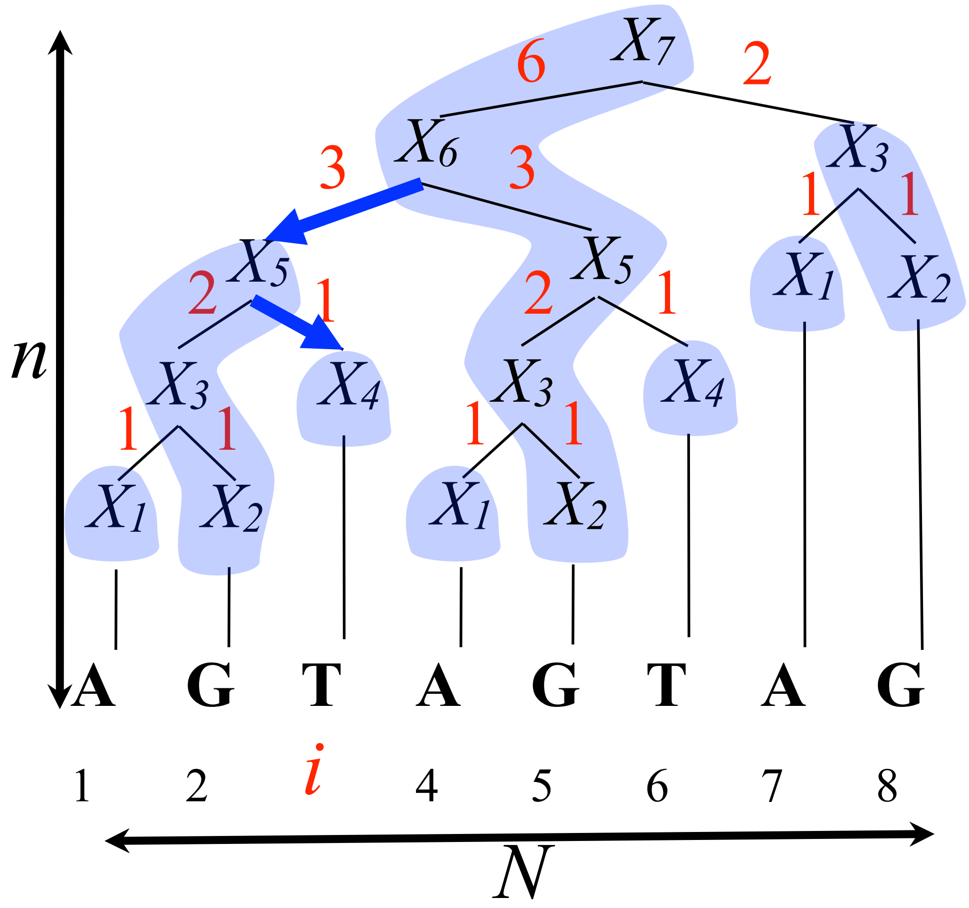
# O($n$) Representation of Heavy Paths

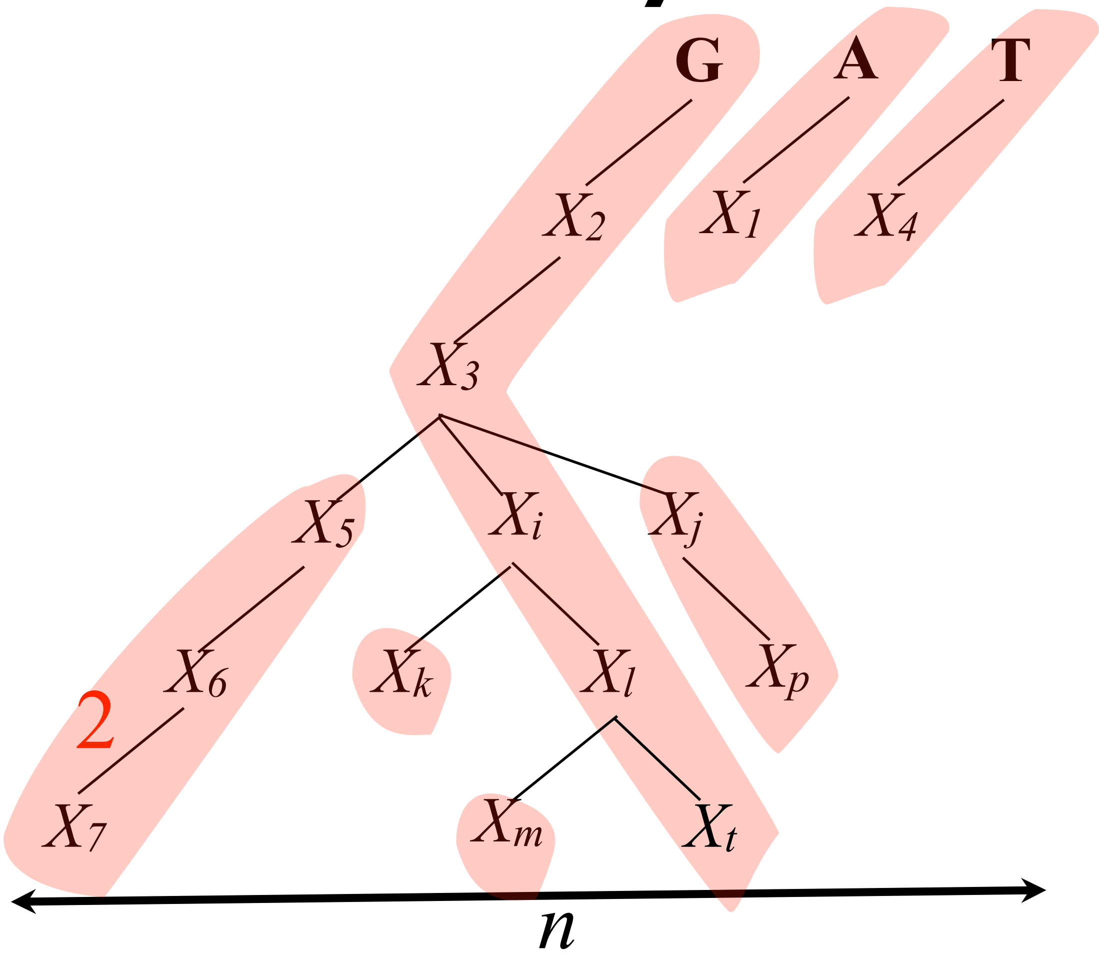




- Binary-search for *i* = Lowest ancestor of distance *i*.
- A heavy path decomp. of the heavy path representation

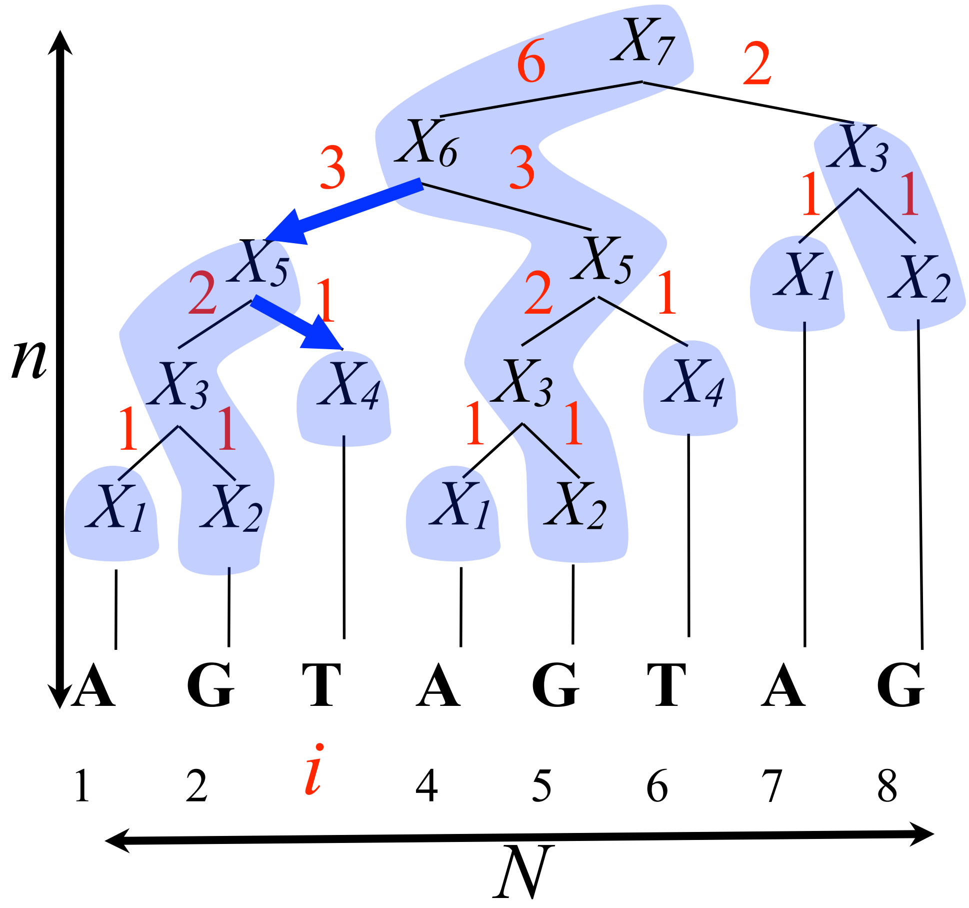
# O($n$) Representation of Heavy Paths

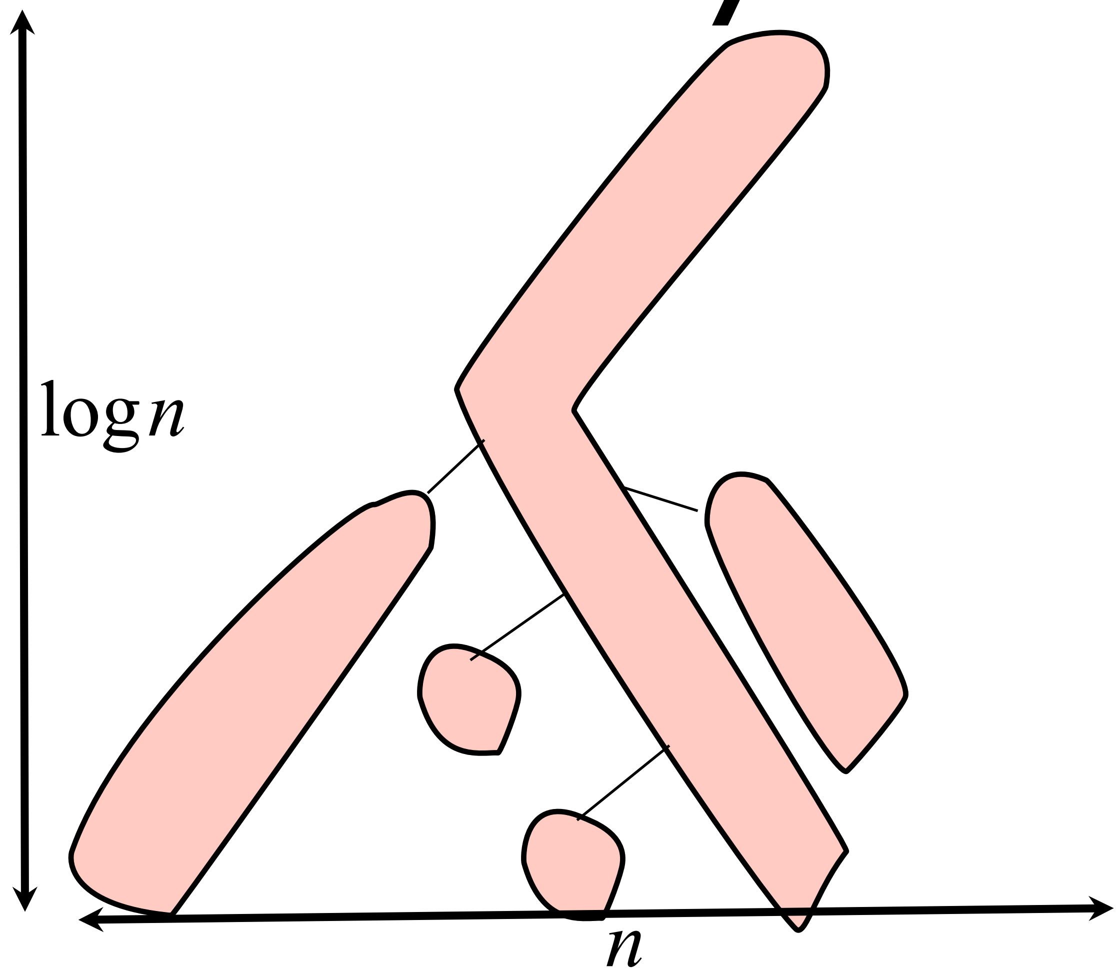




- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: O(log $N/x$) time, total O($n$) space

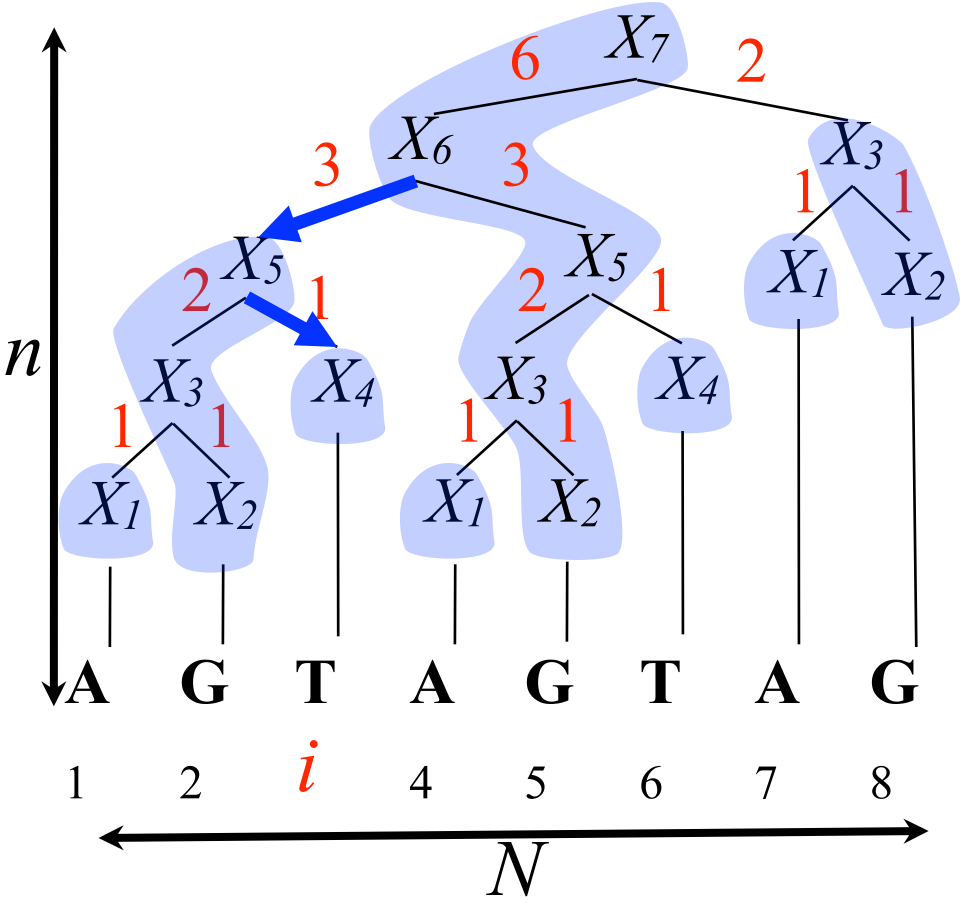
# O($n$) Representation of Heavy Paths

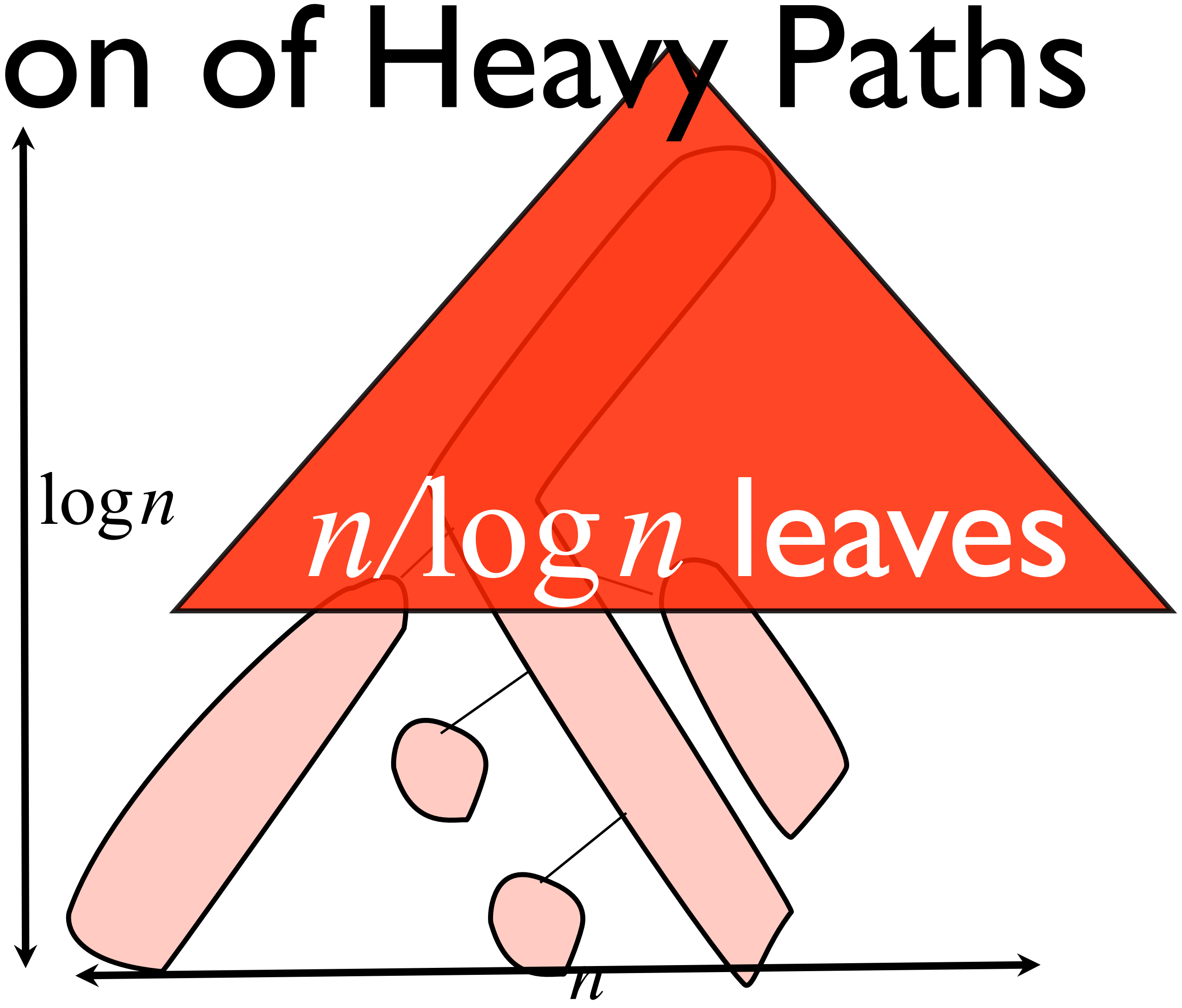




- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: $O(\log N/x)$ time, total $O(n)$ space
- between-paths: $O(\log N/x)$ time, total $O(n \log n)$ space

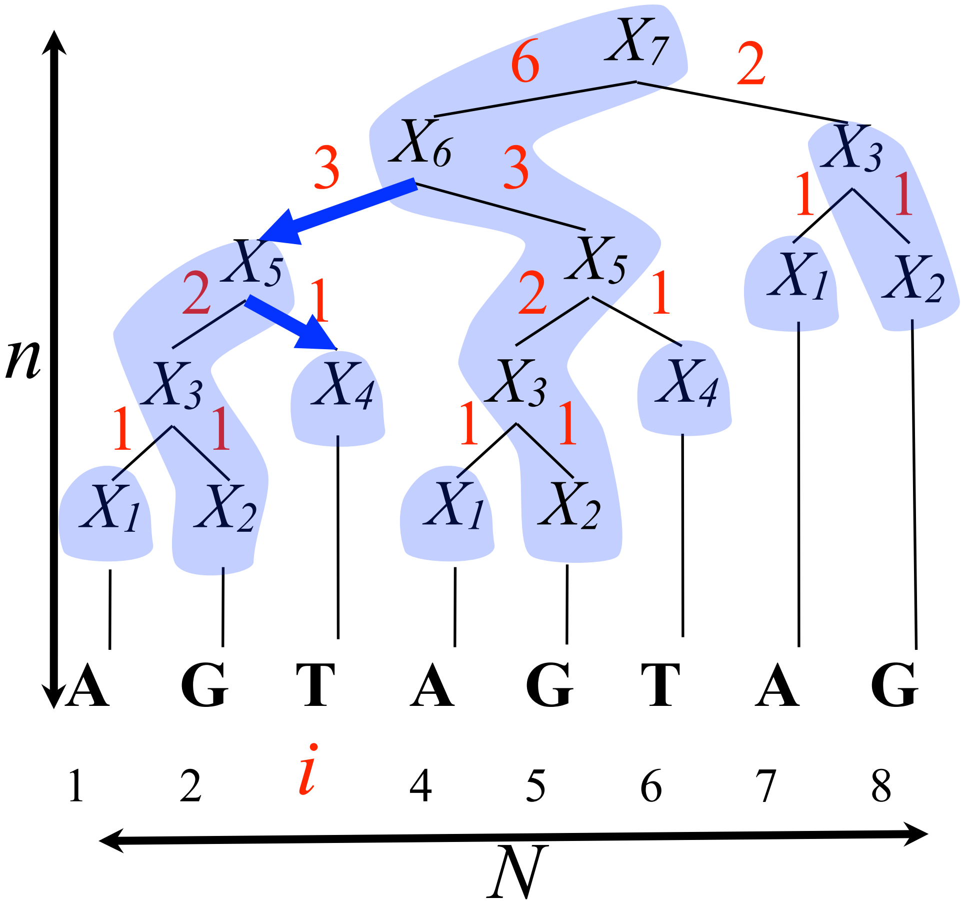
# O($n$) Representation of Heavy Paths

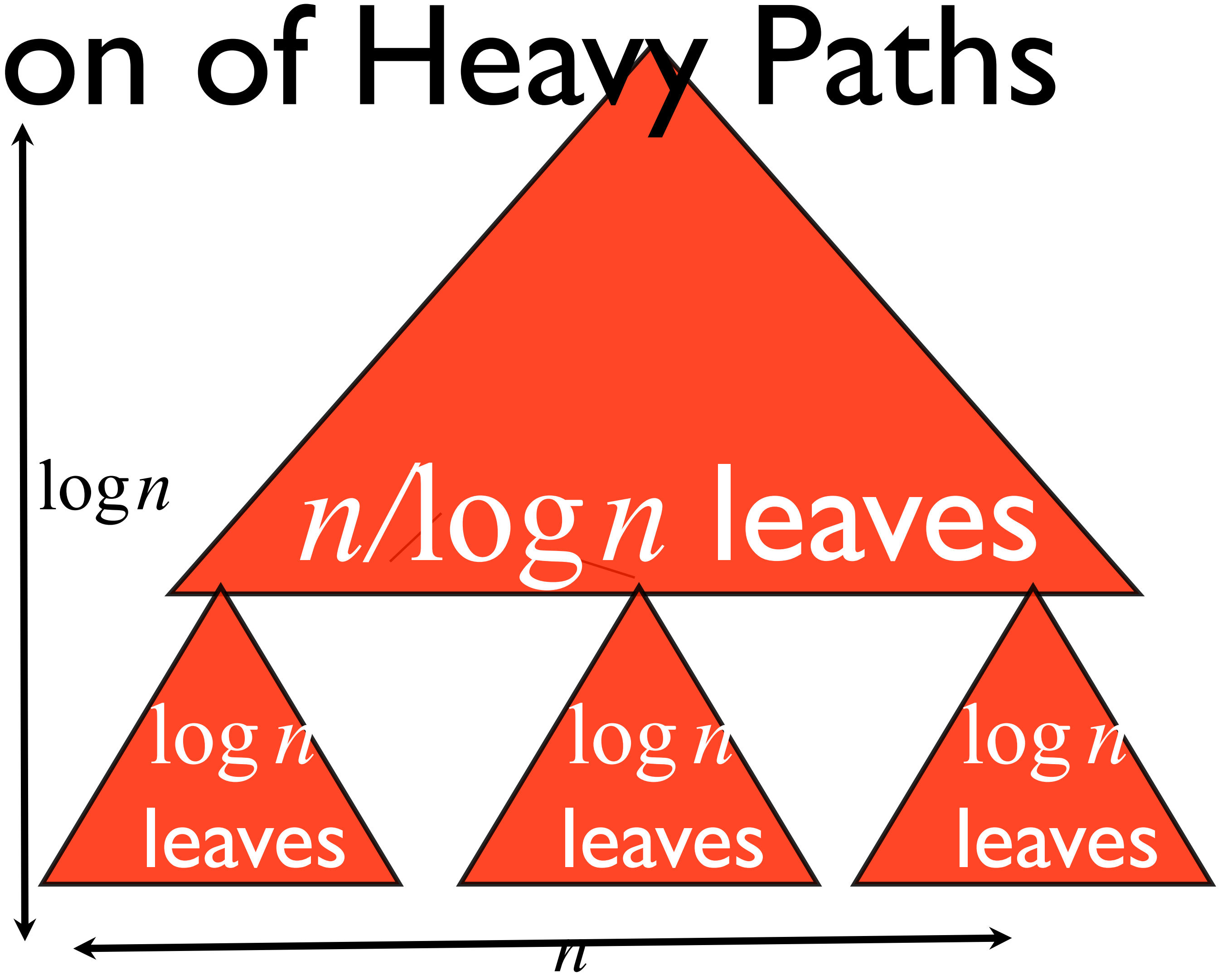




- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: O(log $N/x$) time, total O($n$) space
- between-paths: O(log $N/x$) time, total O($n$ log $n$) space

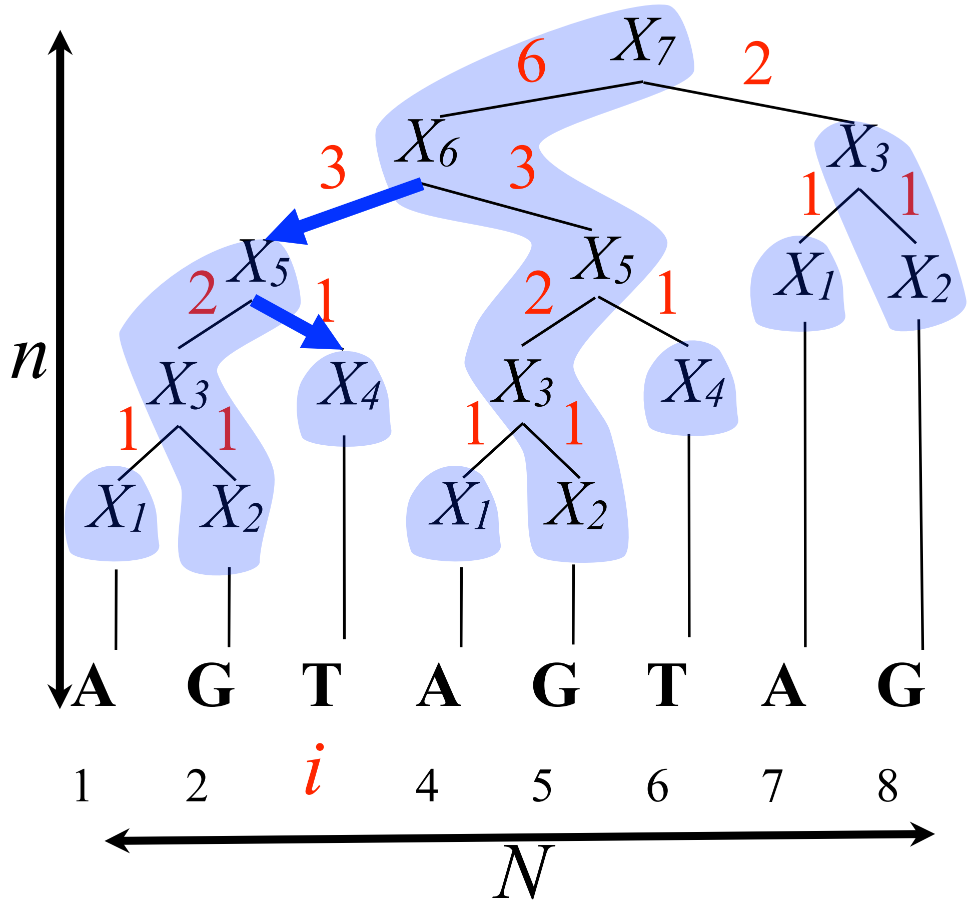
# O($n$) Representation of Heavy Paths

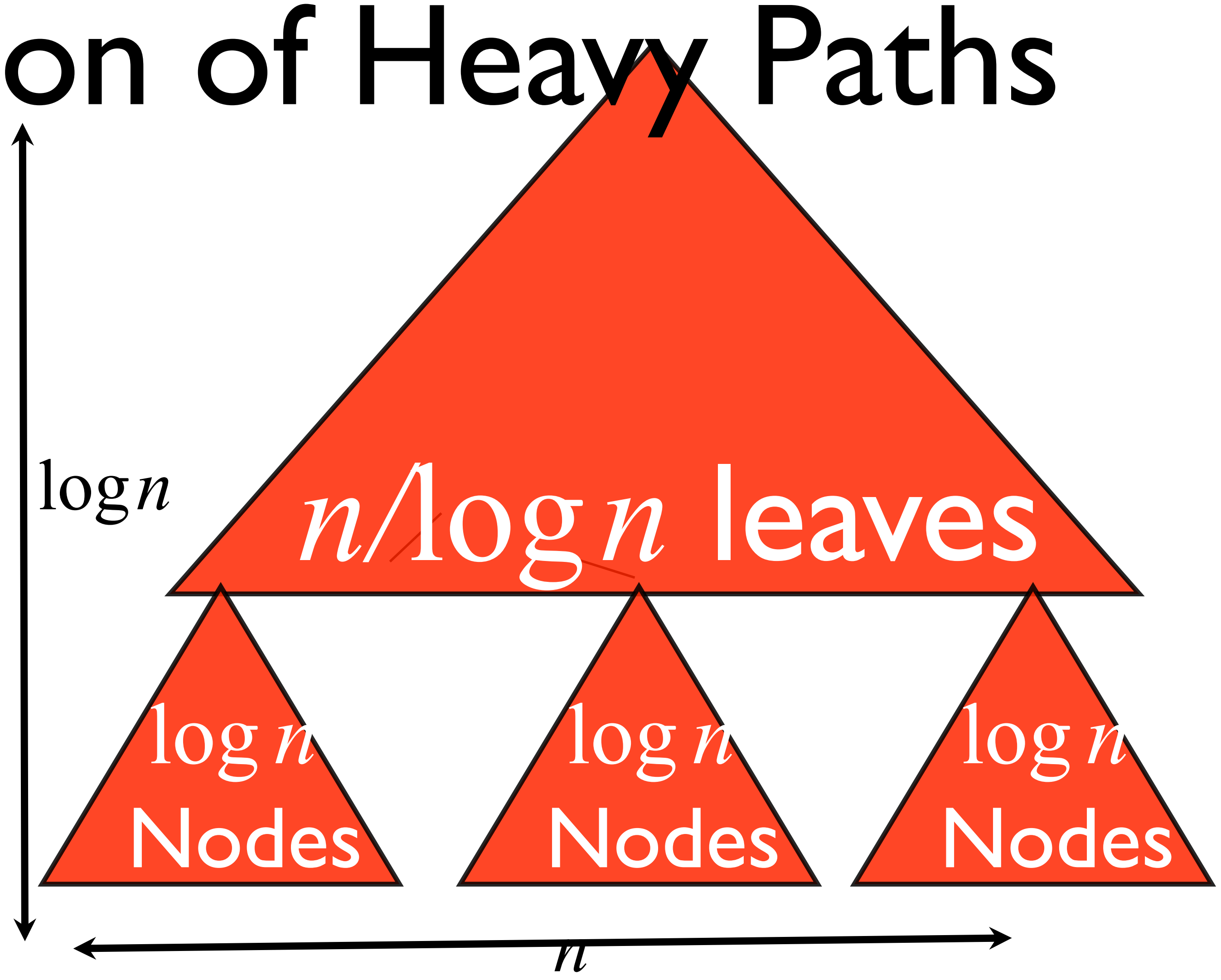




- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: O($\log N/x$) time, total O($n$) space
- between-paths: O($\log N/x$) time, total O($n \log n$) space

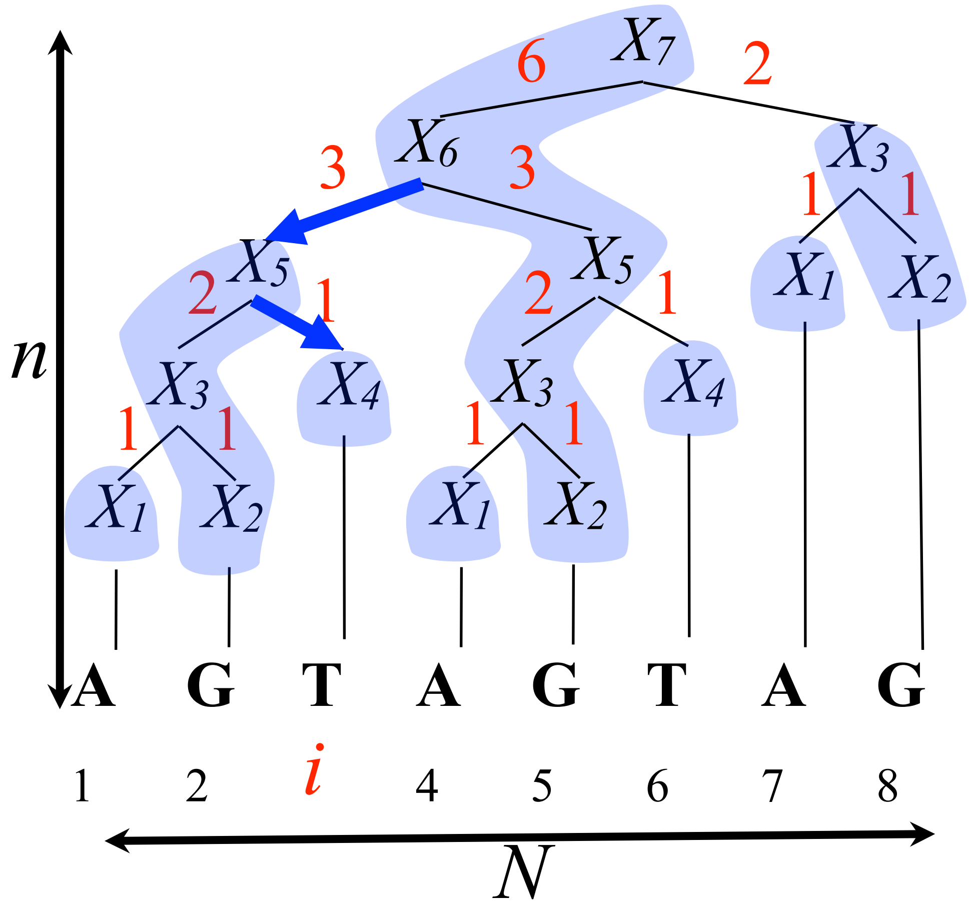
# O($n$) Representation of Heavy Paths

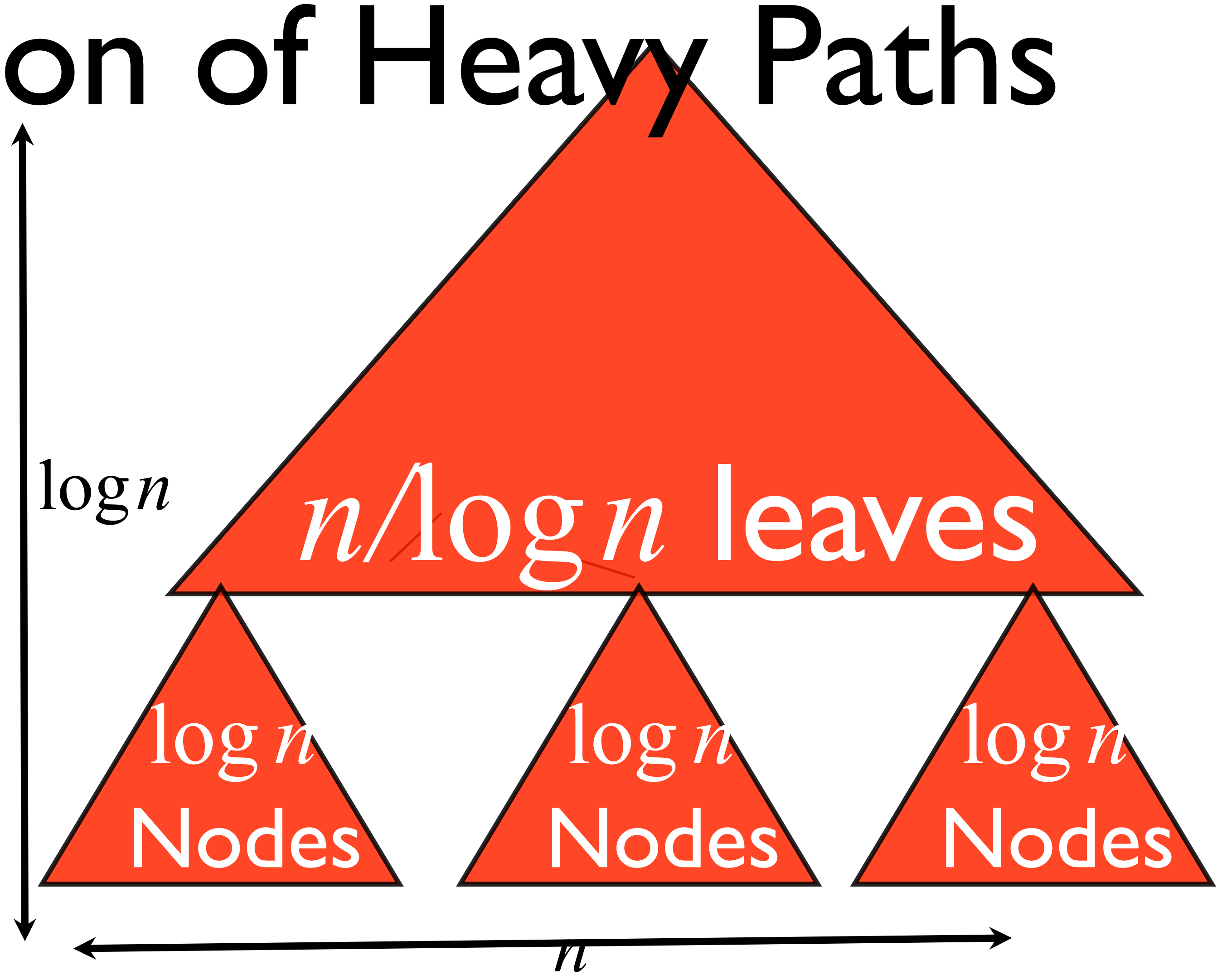




- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: O(log $N/x$) time, total O($n$) space
- between-paths: O(log $N/x$) time, total O($n \log n$) space

# O($n$) Representation of Heavy Paths





- Binary-search for $i$ = Lowest ancestor of distance $i$.
- A heavy path decomp. of the heavy path representation
- in-path: O($\log N/x$) time, total O($n$) space
- between-paths: O($\log N/x$) time, total ~~O($n \log n$)~~ space

O($n\,\alpha(n)$)

# Thank You!