LONGEST COMMON EXTENSIONS IN SUBLINEAR SPACE

PHILIP BILLE
INGE LI GØRTZ
MATHIAS BÆK TEJS KNUDSEN
MOSHE LEWENSTEIN
HJALTE WEDEL VILDHØJ

CPM 2015
June 29, 2015

THE LONGEST COMMON EXTENSION PROBLEM

Prepreprocess T of length n to support the query:

LCE(i,j): return the length of the longest common prefix of T[i…n] and T[j…n]

THE LONGEST COMMON EXTENSION PROBLEM

PREFIX

Prepreprocess T of length n to support the query:

LCE(i,j): return the length of the longest common prefix of T[i…n] and T[j…n]

THE LONGEST COMMON EXTENSION PROBLEM

PREFIX

Preprepreprocess T of length n to support the query:

LCE(i,j): return the length of the longest common prefix of T[i…n] and T[j…n]

Example

T =

1	2	3	4	5	6	7	8	9	10	11
A	C	A	C	B	A	C	B	A	C	C

LCE(3,6)=5

THE LONGEST COMMON EXTENSION PROBLEM

PREFIX

Prepreprocess T of length n to support the query:

LCE(i,j): return the length of the longest common prefix of T[i…n] and T[j…n]

Example

	1	2	3	4	5	6	7	8	9	10	11
T =	A	C	A	C	B	A	C	B	A	C	C
Suffix 3			A	C	B	A	C	B	A	C	C
Suffix 6						A	C	B	A	C	C

LCE(3,6)=5

THE LONGEST COMMON EXTENSION PROBLEM

PREFIX

Prepreprocess T of length n to support the query:

LCE(i,j): return the length of the longest common prefix of T[i…n] and T[j…n]

Example

LCE(3,6)=5

TWO SIMPLE SOLUTIONS

ℓ = LCE(i,j)

TWO SIMPLE SOLUTIONS

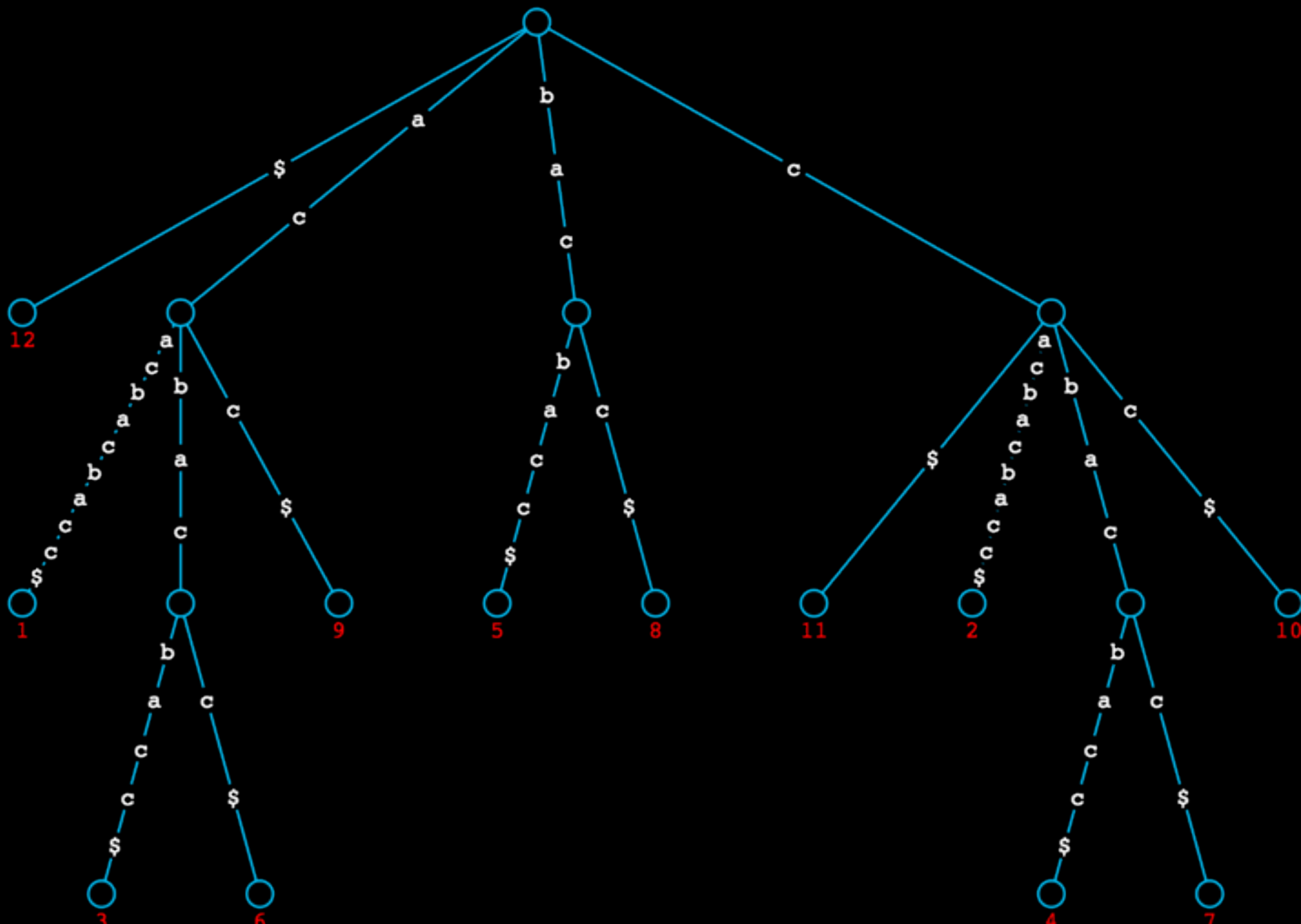
ℓ=LCE(i,j)

		Space	Time
1	Store nothing	O(1)	$O(\ell)=O(n)$

TWO SIMPLE SOLUTIONS

ℓ=LCE(i,j)

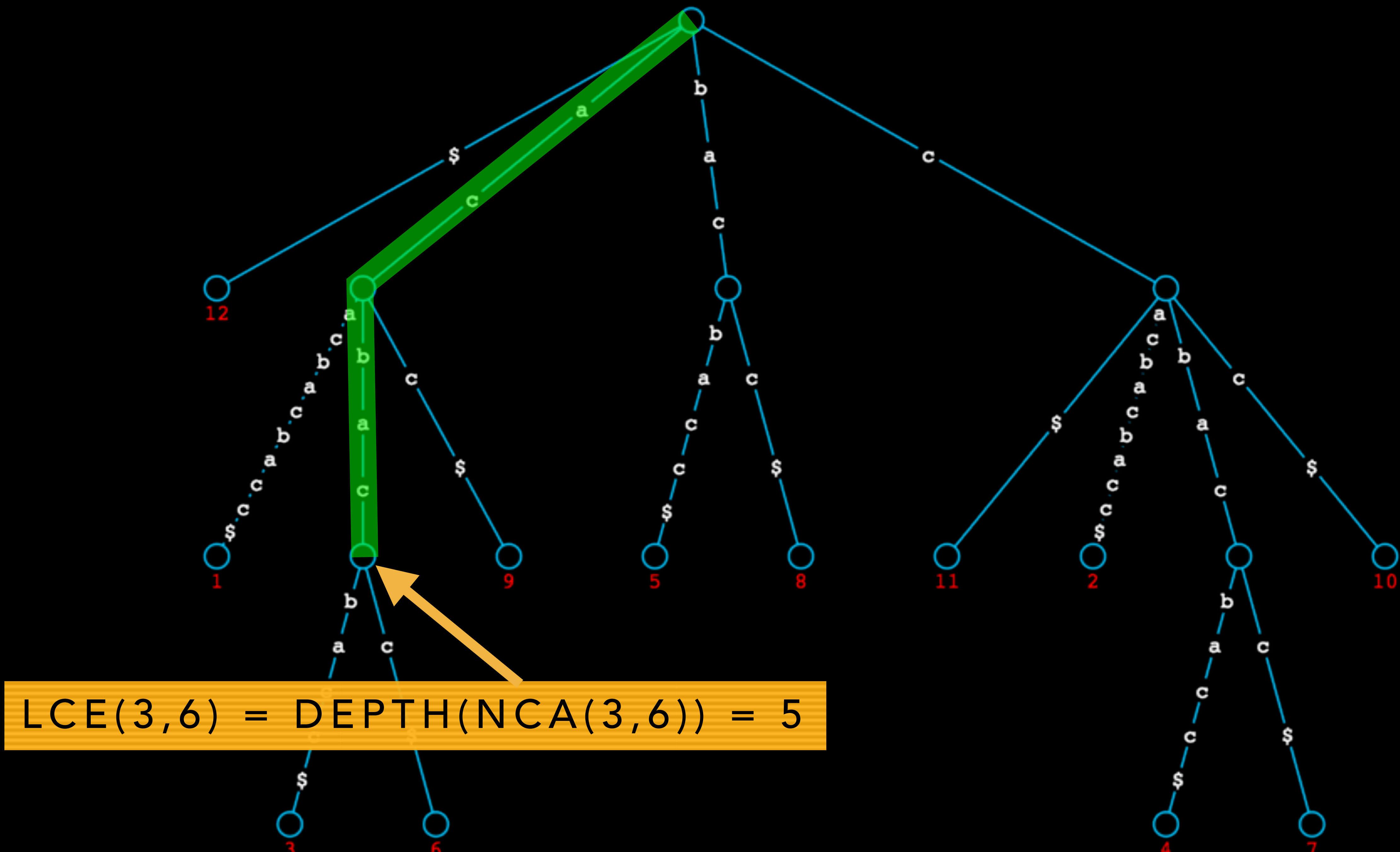
		Space	Time
1	Store nothing	O(1)	O(ℓ)=O(n)
2	Store the suffix tree of T	O(n)	O(1)



TWO SIMPLE SOLUTIONS

ℓ=LCE(i,j)

		Space	Time
1	Store nothing	O(1)	O(ℓ)=O(n)
2	Store the suffix tree of T	O(n)	O(1)



OUR RESULTS

ℓ = LCE(i,j)

SIMPLE SOLUTIONS

		Space	Time
1	Store nothing	O(1)	$O(\ell)=O(n)$
2	Store the suffix tree of T	O(n)	O(1)

OUR RESULTS

ℓ=LCE(i,j)

SIMPLE SOLUTIONS

		Space	Time
1	Store nothing	O(1)	$O(\ell)=O(n)$
2	Store the suffix tree of T	O(n)	O(1)

Can we obtain $O(n/\tau)$ space and $O(\tau)$ time for all $1 \leq \tau \leq n$?

OUR RESULTS

ℓ=LCE(i,j)

SIMPLE SOLUTIONS

		Space	Time
1	Store nothing	$O(1)$	$O(\ell)=O(n)$
2	Store the suffix tree of T	$O(n)$	$O(1)$

CPM 2012 RESULTS*

		Space	Time	Trade-off range
3	Deterministic trade-off	$O(n/\tau)$	$O(\tau^2)$	$1\leq\tau\leq\sqrt{n}$
4	Randomized trade-off	$O(n/\tau)$	$O(\tau\log(\ell/\tau))$	$1\leq\tau\leq n$

Can we obtain $O(n/\tau)$ space and $O(\tau)$ time for all $1\leq\tau\leq n$?

*Philip Bille, Inge Li Gørtz, Benjamin Sach, Hjalte Wedel Vildhøj, Time-Space Trade-Offs for Longest Common Extensions, CPM 2012

OUR RESULTS

ℓ=LCE(i,j)

SIMPLE SOLUTIONS

		Space	Time
1	Store nothing	$O(1)$	$O(\ell)=O(n)$
2	Store the suffix tree of T	$O(n)$	$O(1)$

CPM 2012 RESULTS*

		Space	Time	Trade-off range
3	Deterministic trade-off	$O(n/\tau)$	$O(\tau^2)$	$1\leq\tau\leq\sqrt{n}$
4	Randomized trade-off	$O(n/\tau)$	$O(\tau\log(\ell/\tau))$	$1\leq\tau\leq n$

CPM 2015 RESULTS

		Space	Time	Trade-off range
5	NEW deterministic trade-off	$O(n/\tau)$	$O(\tau\log^2(n/\tau))$	$1/\log n\leq\tau\leq n$
6	NEW randomized trade-off	$O(n/\tau)$	$O(\tau)$	$1\leq\tau\leq n$

*Philip Bille, Inge Li Gørtz, Benjamin Sach, Hjalte Wedel Vildhøj, Time-Space Trade-Offs for Longest Common Extensions, CPM 2012

THE NEW DETERMINISTIC TRADE-0FF

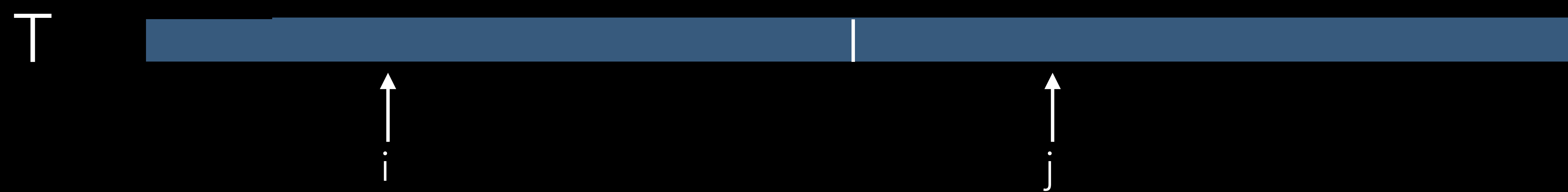
TWO STRUCTURES

Data Structure 1: $O(n/\tau)$ space and $O(\tau)$ time, but works only if $|i-j|<\tau$

Data Structure 2: $O(n/\tau)$ space and $O(\tau \log^2(n/\tau))$ time:
Reduces an LCE(i,j) query to another query LCE(i',j') s.t. $|i'-j'|<\tau$

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$

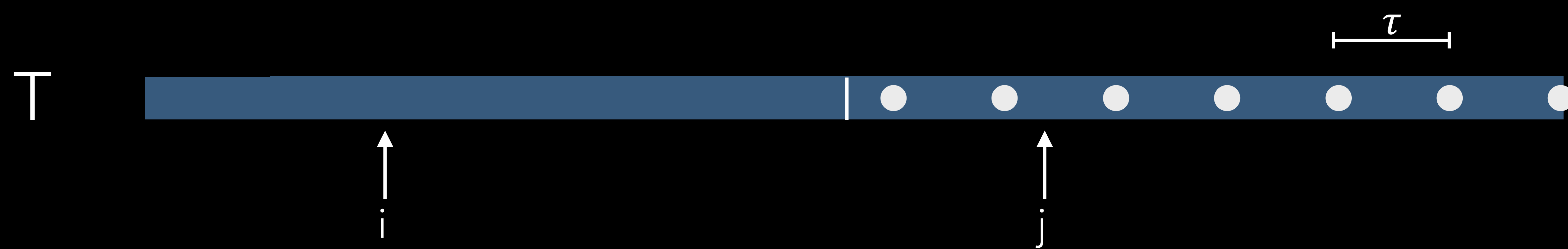


Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$

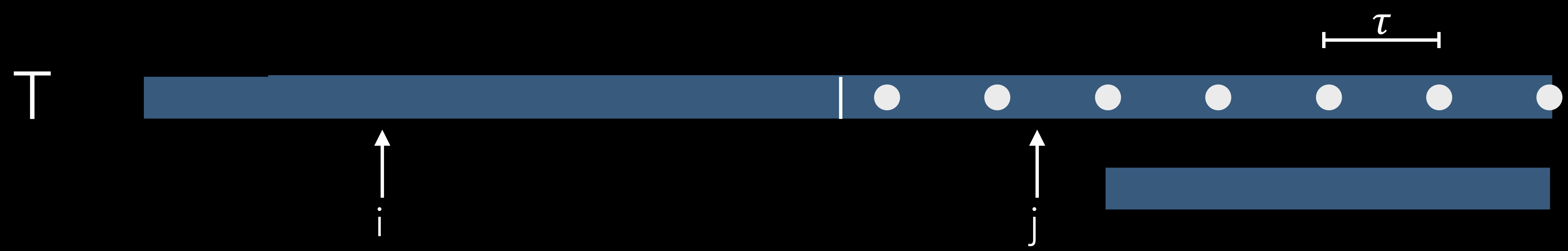


Lemma
An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i′,j′) query such that i′ and j′ are in the same half of T

Proof

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$



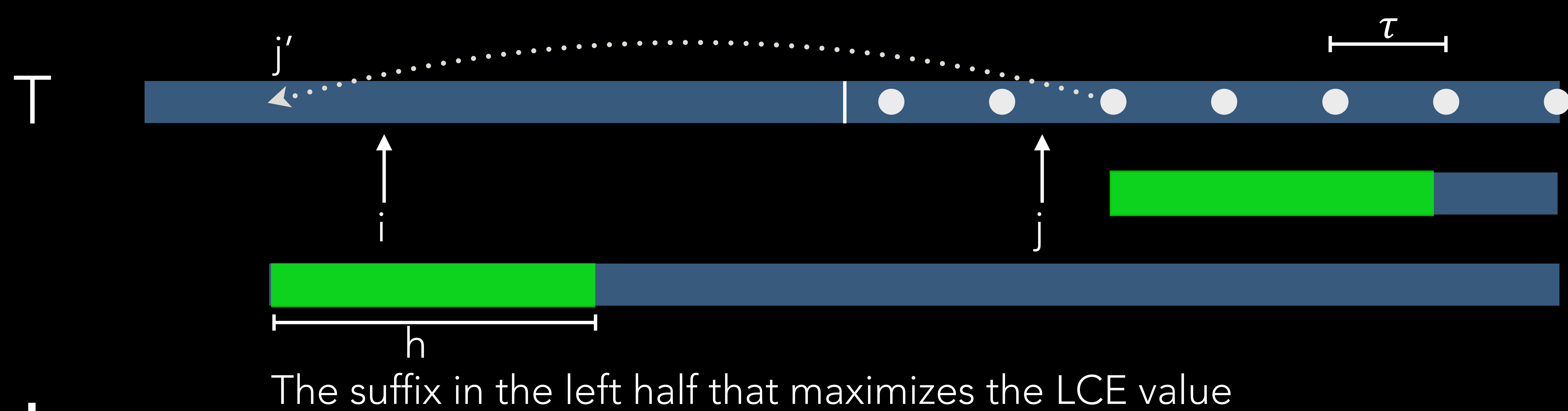
Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$

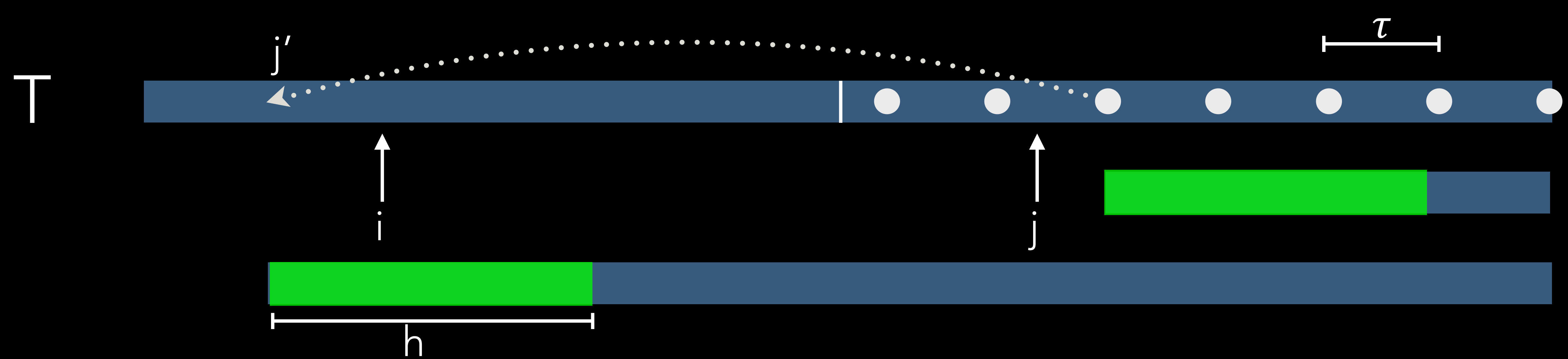


Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

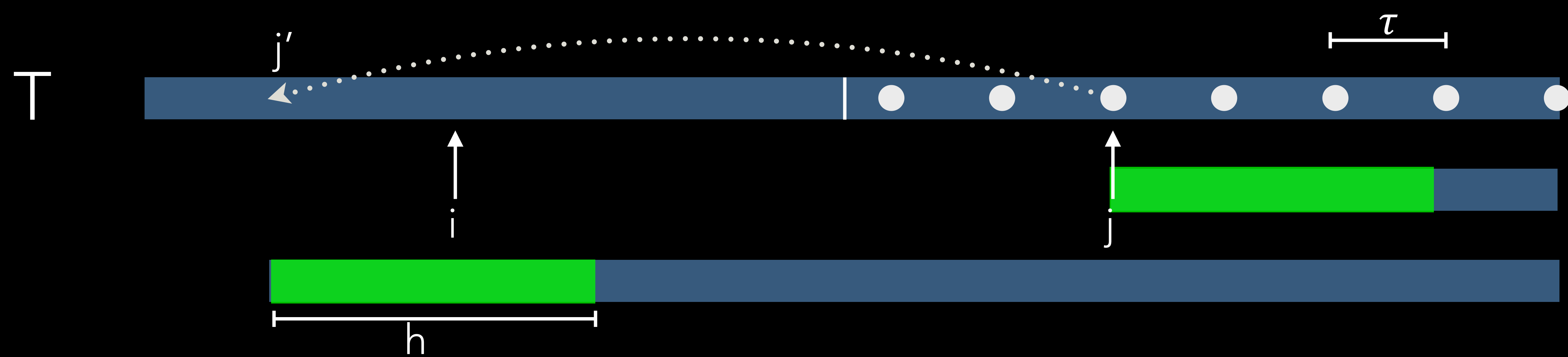
Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

- Assume that j is a sampled position

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

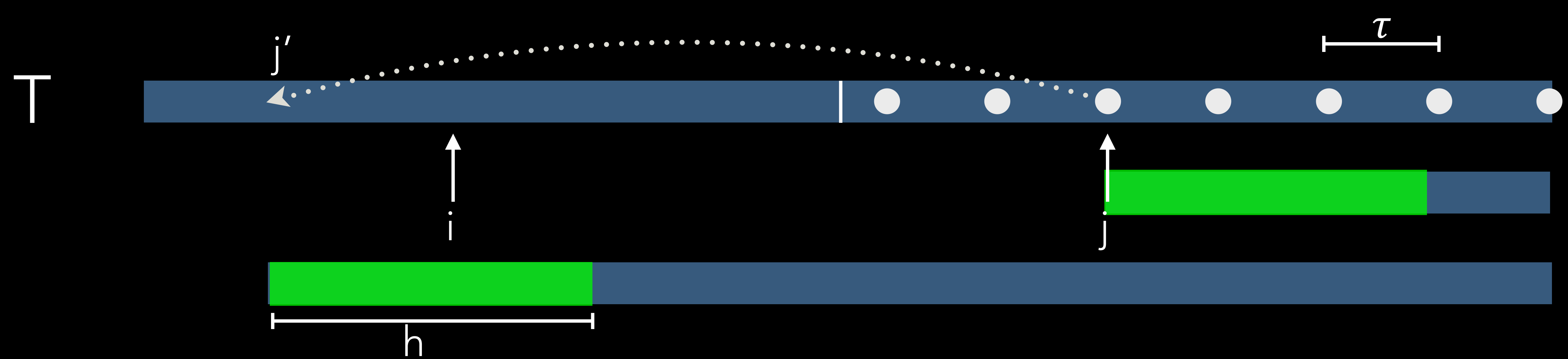
Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

- Assume that j is a sampled position

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

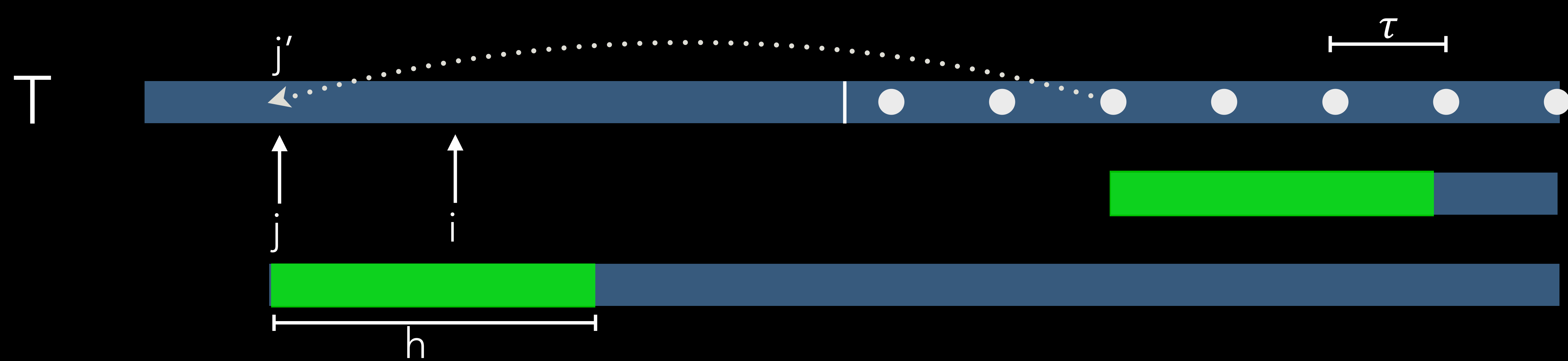
Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

- Assume that j is a sampled position
- Then LCE(i,j) ≤ h, so we can compute LCE(i,j) as LCE(i,j')

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

Lemma

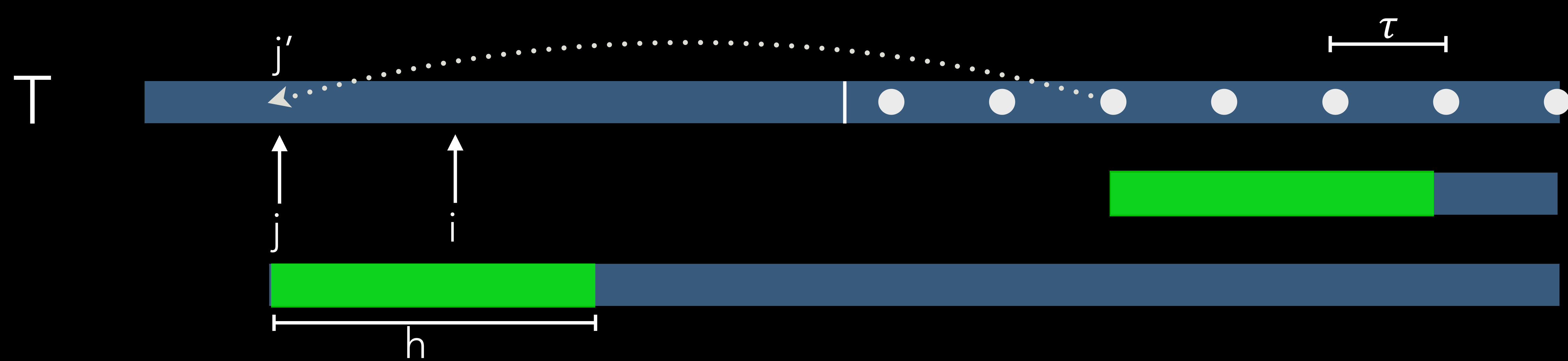
An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

Proof

- Assume that j is a sampled position
- Then LCE(i,j) ≤ h, so we can compute LCE(i,j) as LCE(i,j')

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i',j') query such that i' and j' are in the same half of T

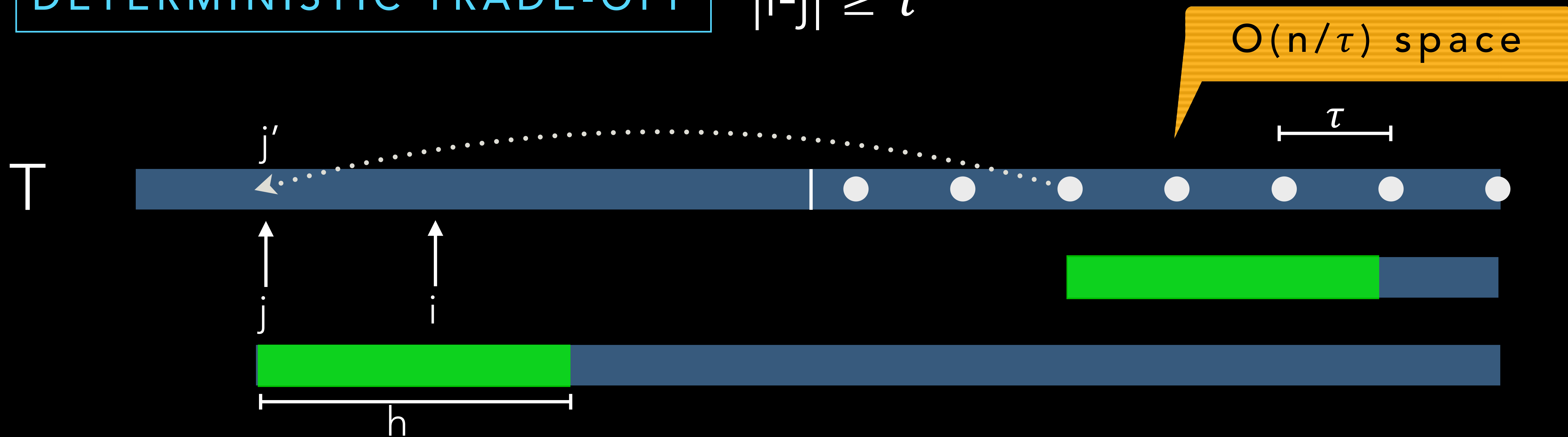
Proof

- Assume that j is a sampled position
- Then LCE(i,j) ≤ h, so we can compute LCE(i,j) as LCE(i,j')

$$LCE(i,j) = \min(\, LCE(i,j') \,,\, h \,)$$

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$



The suffix in the left half that maximizes the LCE value

Lemma

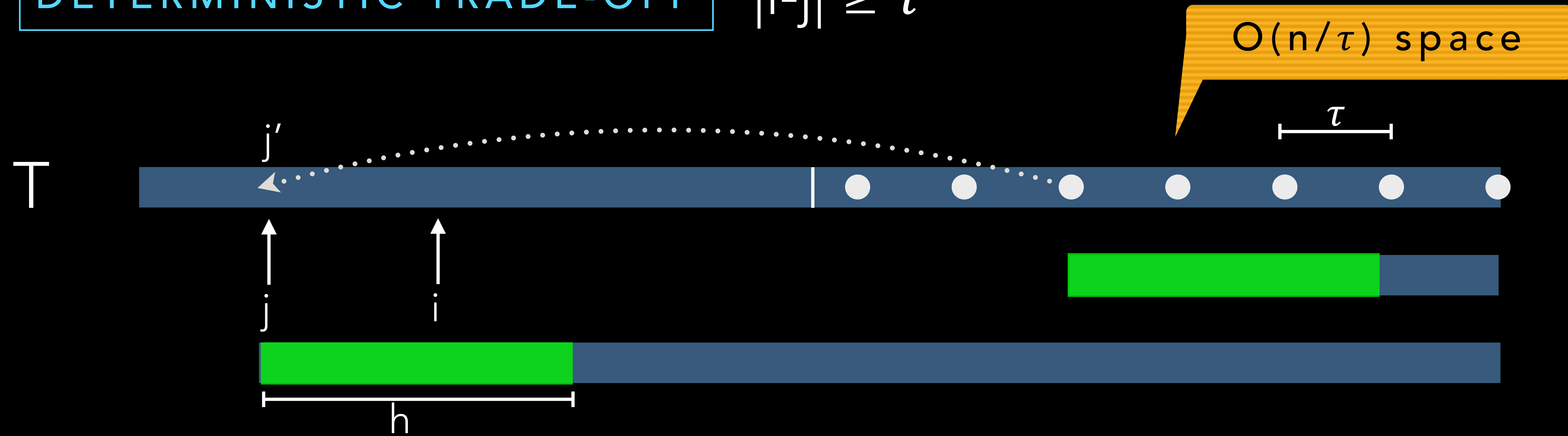
An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i’,j’) query such that i’ and j’ are in the same half of T

Proof

- Assume that j is a sampled position
- Then LCE(i,j) ≤ h, so we can compute LCE(i,j) as LCE(i,j’)

$$\mathrm{LCE}(i,j) = \min(\mathrm{LCE}(i,j'), h)$$

DETERMINISTIC TRADE-OFF $|i\text{-}j| \geq \tau$



The suffix in the left half that maximizes the LCE value

Lemma

An LCE(i,j) query where i and j are in separate halves of T can be reduced to another LCE(i’,j’) query such that i’ and j’ are in the same half of T

Proof

- Assume that j is a sampled position
- Then LCE(i,j) ≤ h, so we can compute LCE(i,j) as LCE(i,j’)

O(τ) time

$$\mathrm{LCE}(i,j) = \min(\,\mathrm{LCE}(i,j'),\, h\,)$$

DETERMINISTIC TRADE-OFF

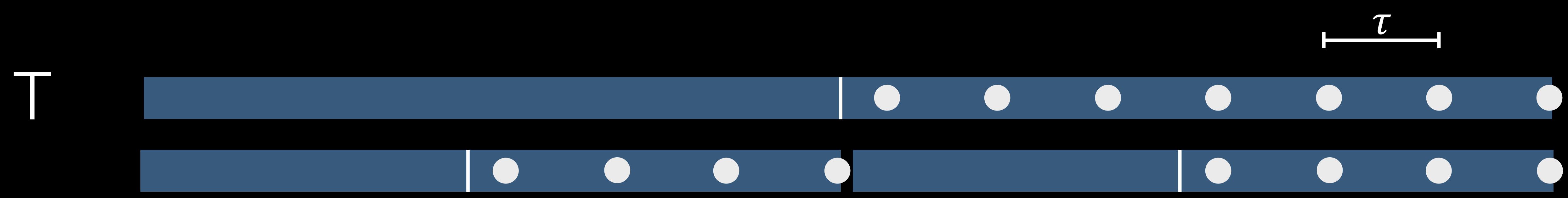
$|i-j| \geq \tau$

DETERMINISTIC TRADE-OFF

$|i\text{-}j| \geq \tau$

- Build data structure recursively for left and right half of T

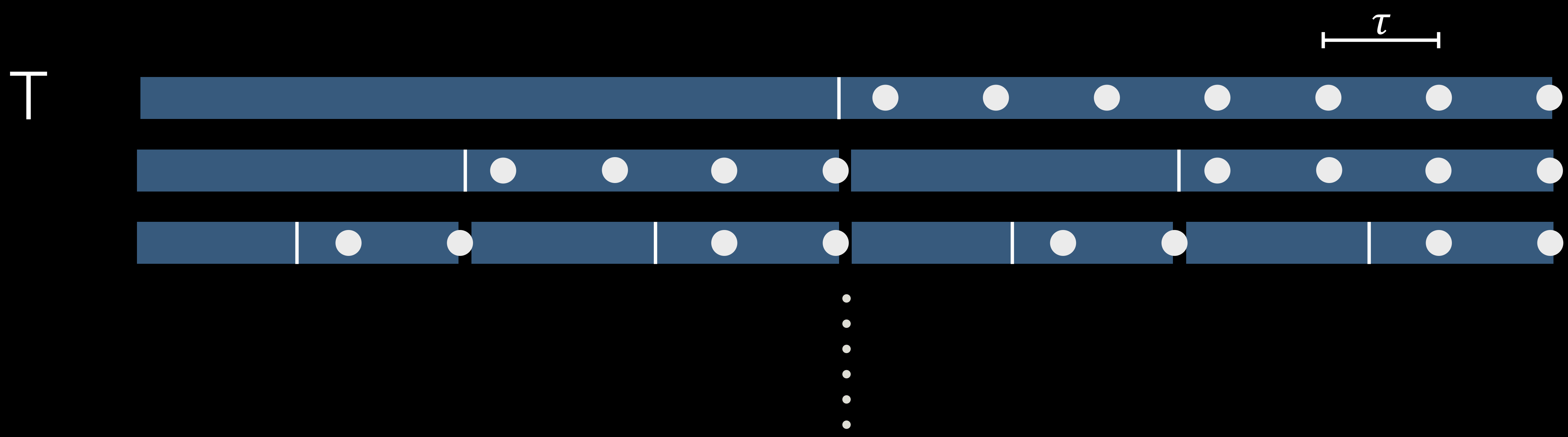
DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



- Build data structure recursively for left and right half of T

DETERMINISTIC TRADE-OFF

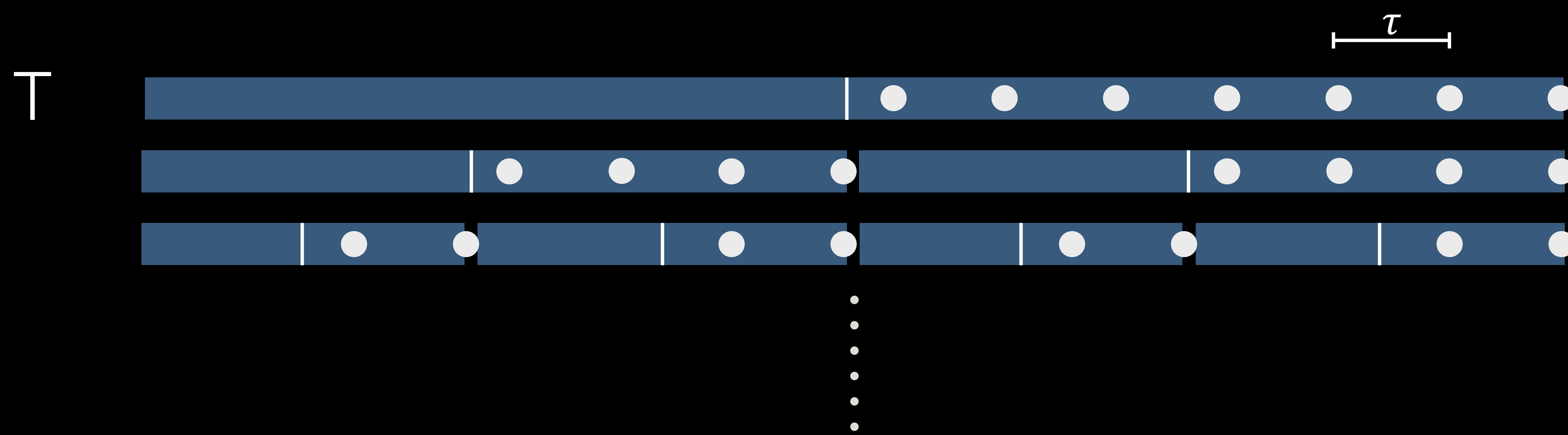
$|i-j| \geq \tau$



- Build data structure recursively for left and right half of T

DETERMINISTIC TRADE-OFF

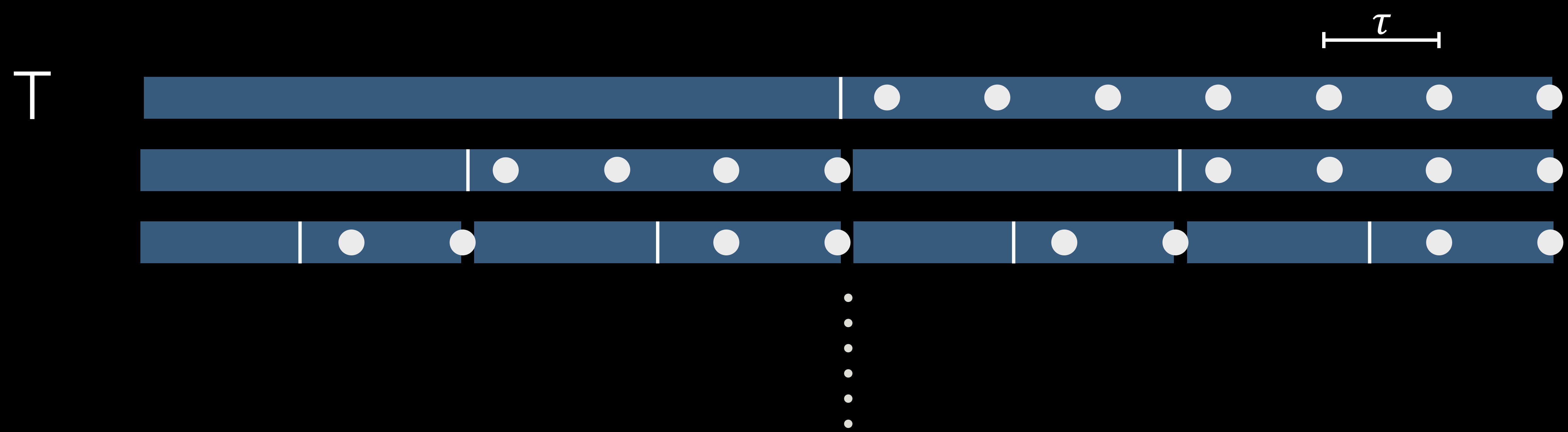
$|i\text{-}j| \geq \tau$



- Build data structure recursively for left and right half of T
- Stop when strings are $<2\tau$

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$



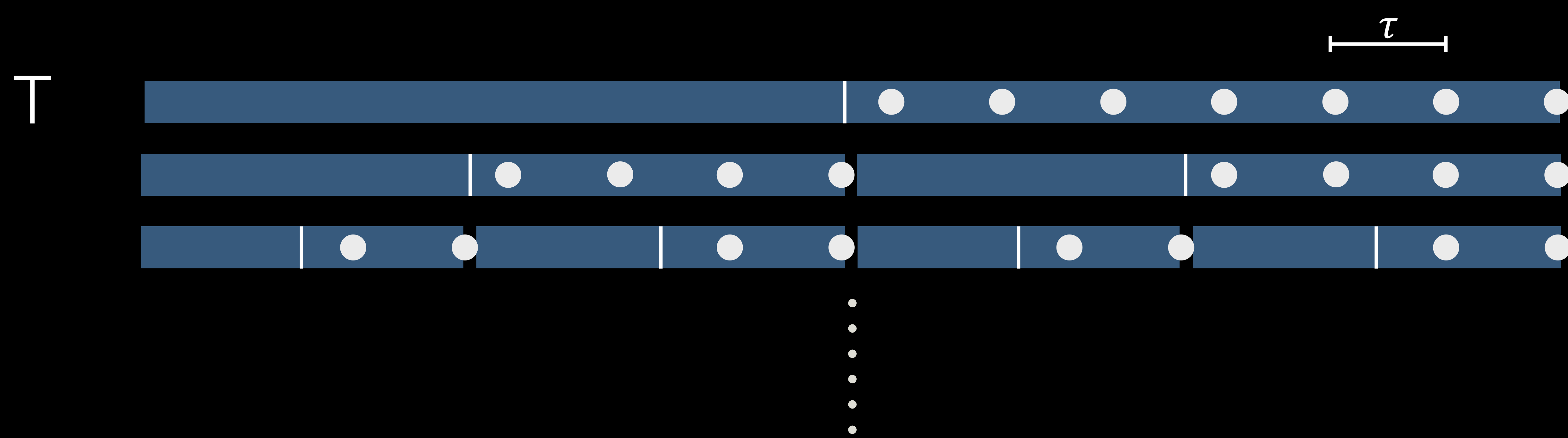
- Build data structure recursively for left and right half of T
- Stop when strings are $<2\tau$

Analysis

- $n/(2\tau)$ sampled positions on each level
- $\log(n/\tau)$ levels
- $O(\tau)$ time on each level

DETERMINISTIC TRADE-OFF

$|i-j| \geq \tau$



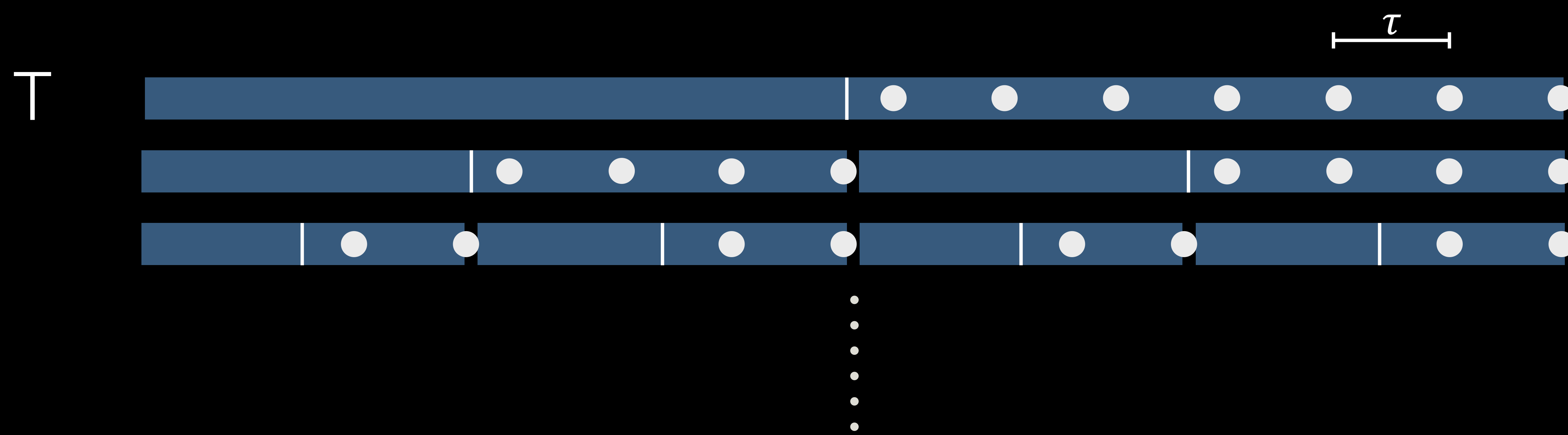
- Build data structure recursively for left and right half of T
- Stop when strings are $<2\tau$

Analysis

- $n/(2\tau)$ sampled positions on each level
- $\log(n/\tau)$ levels
- $O(\tau)$ time on each level

$O((n/\tau)\log(n/\tau))$ space
$O(\tau\log(n/\tau))$ time

DETERMINISTIC TRADE-OFF $|i-j| \geq \tau$



- Build data structure recursively for left and right half of T
- Stop when strings are $<2\tau$

Analysis

- $n/(2\tau)$ sampled positions on each level
- $\log(n/\tau)$ levels
- $O(\tau)$ time on each level

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time

SHAVING TWO LOGS

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau)$ time

SHAVING TWO LOGS

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

$O(n/\tau)$ space
$O(\tau)$ time

SHAVING TWO LOGS

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau)$ time

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

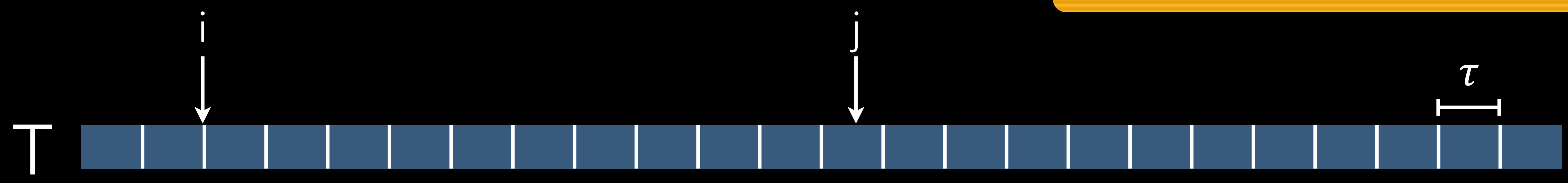
T

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

RANDOMIZED TRADE-OFF

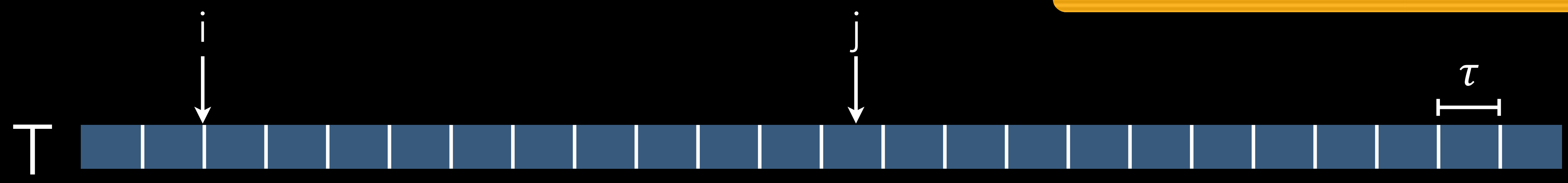
$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time



Answering an LCE query

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

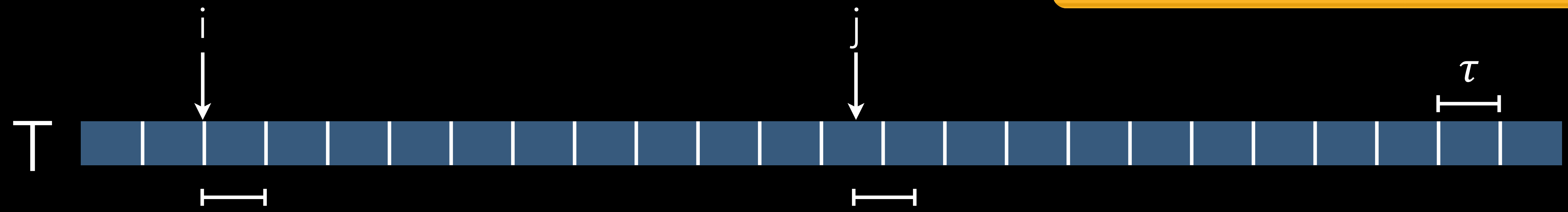


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

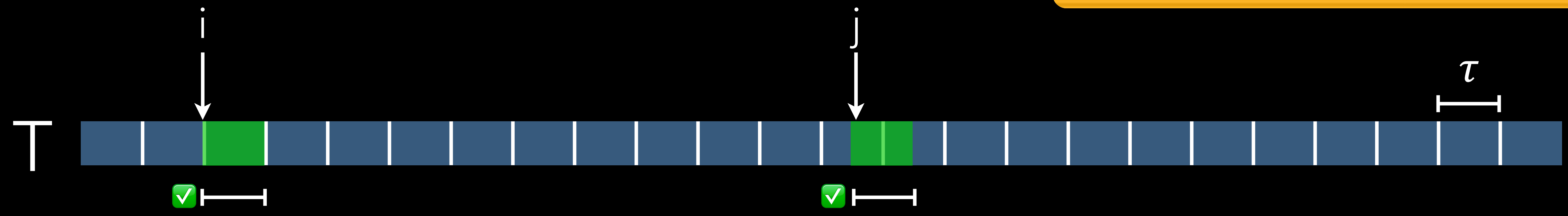


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

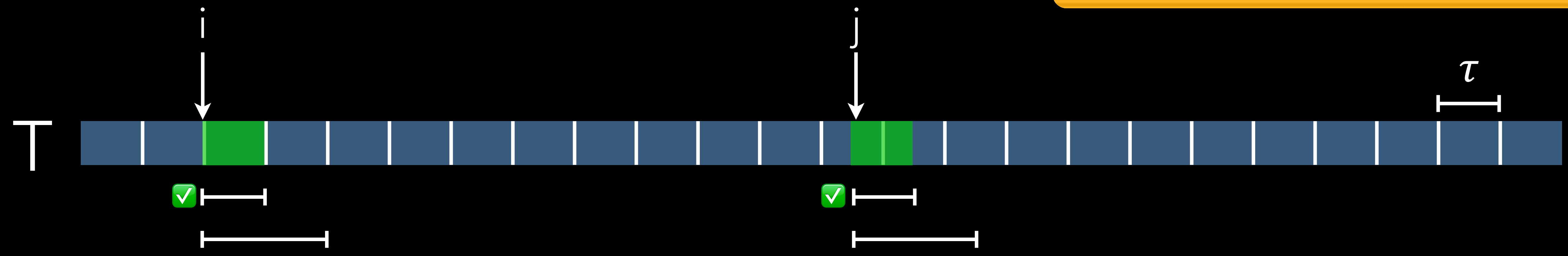


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

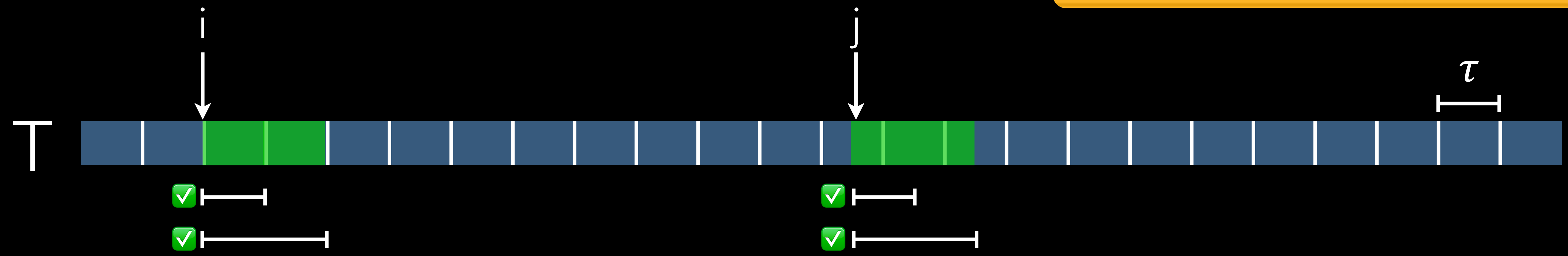


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

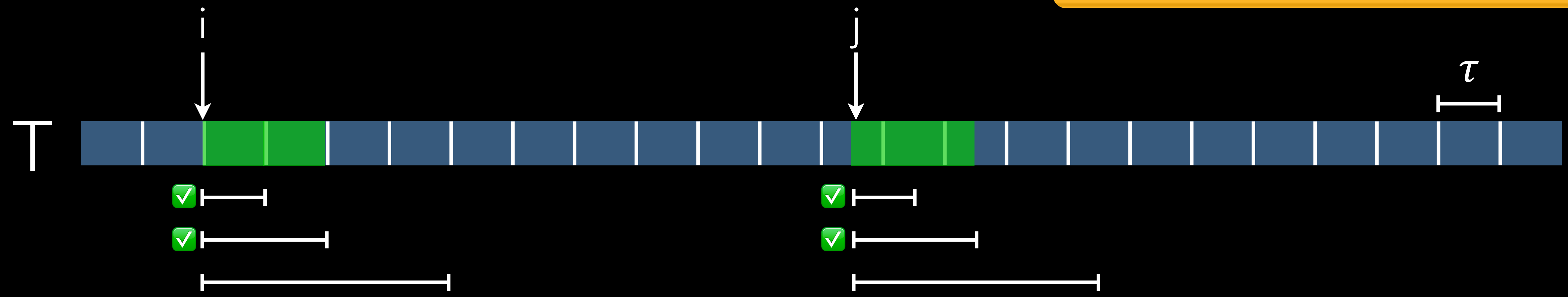


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

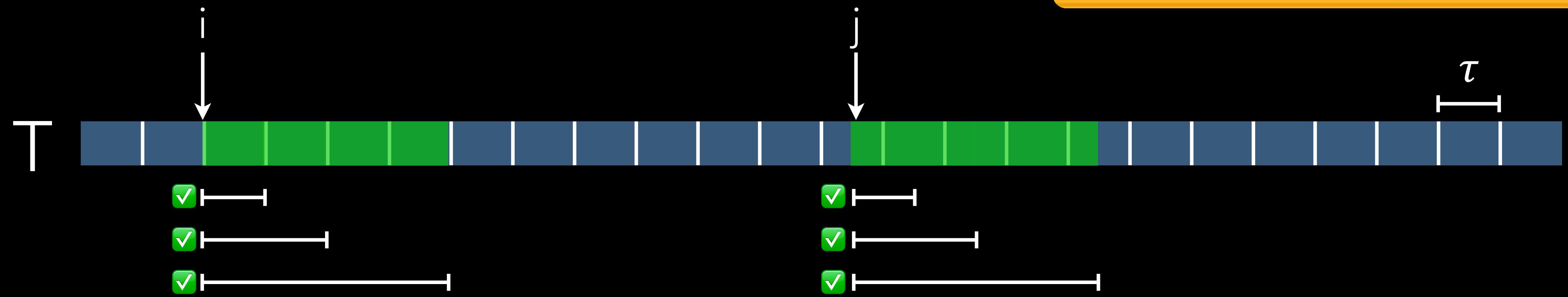


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

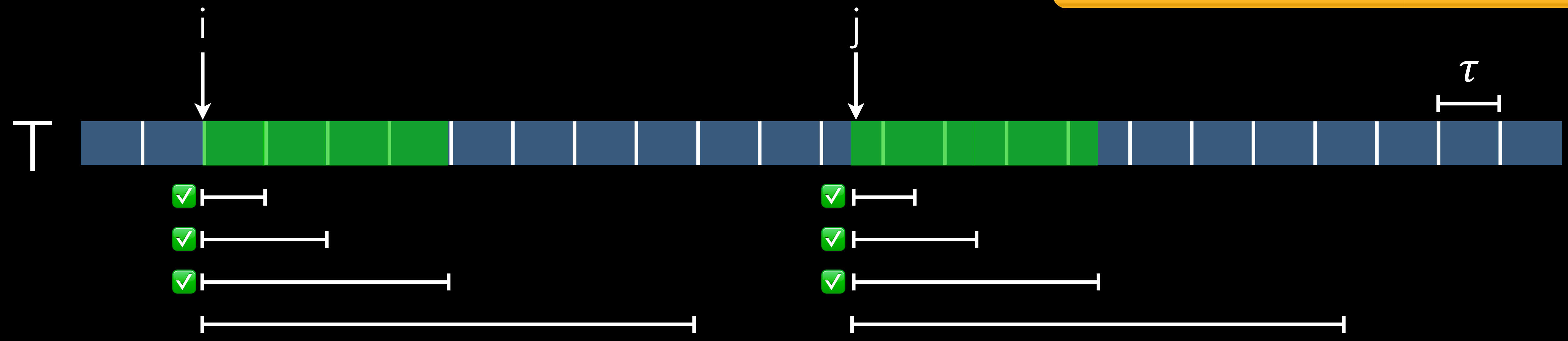


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

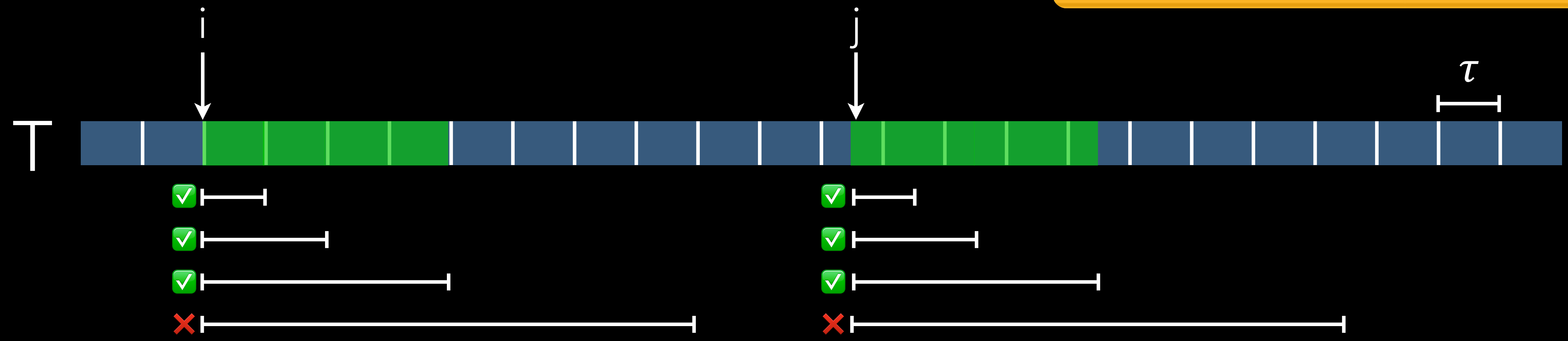


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

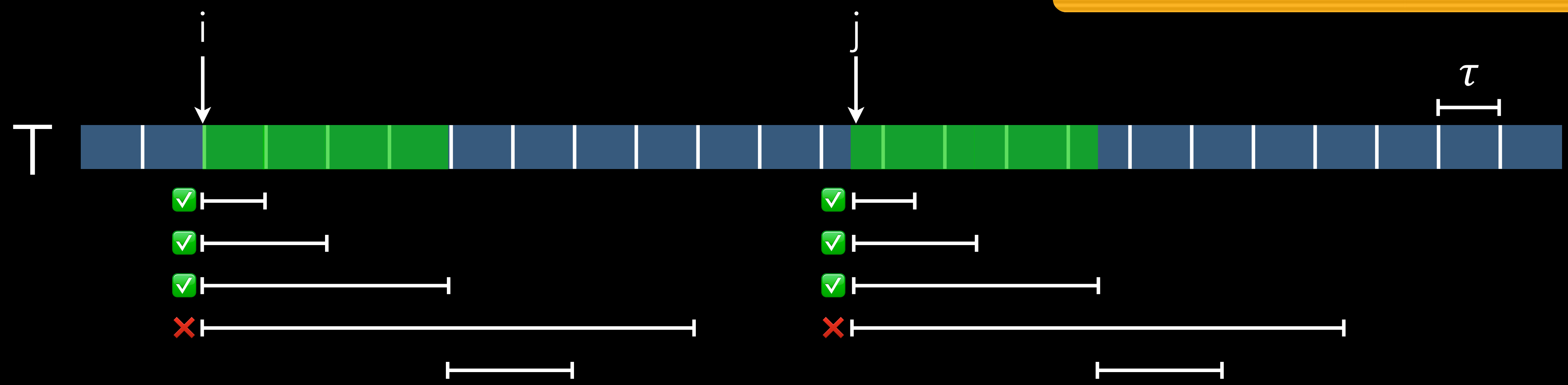


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

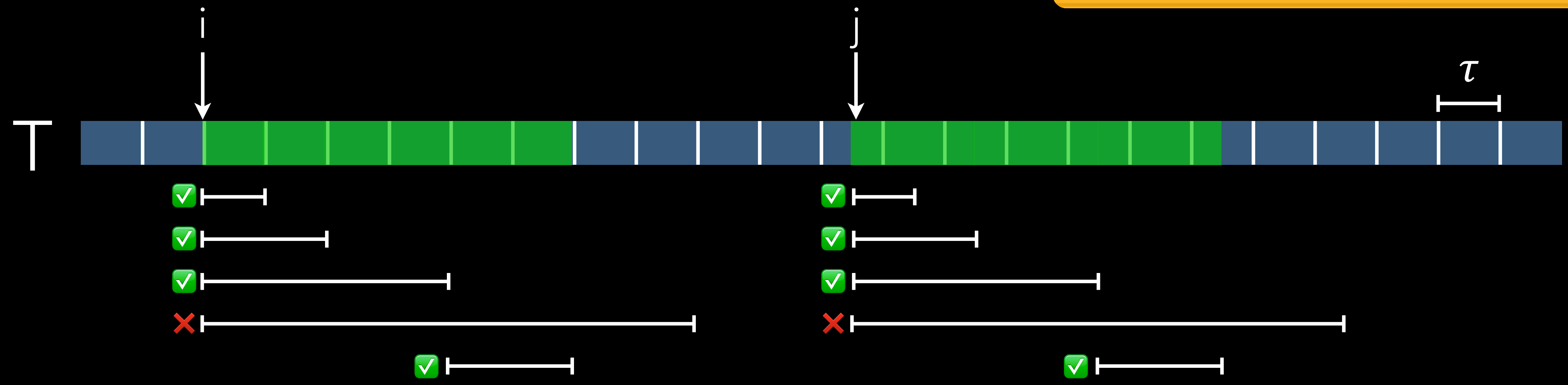


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

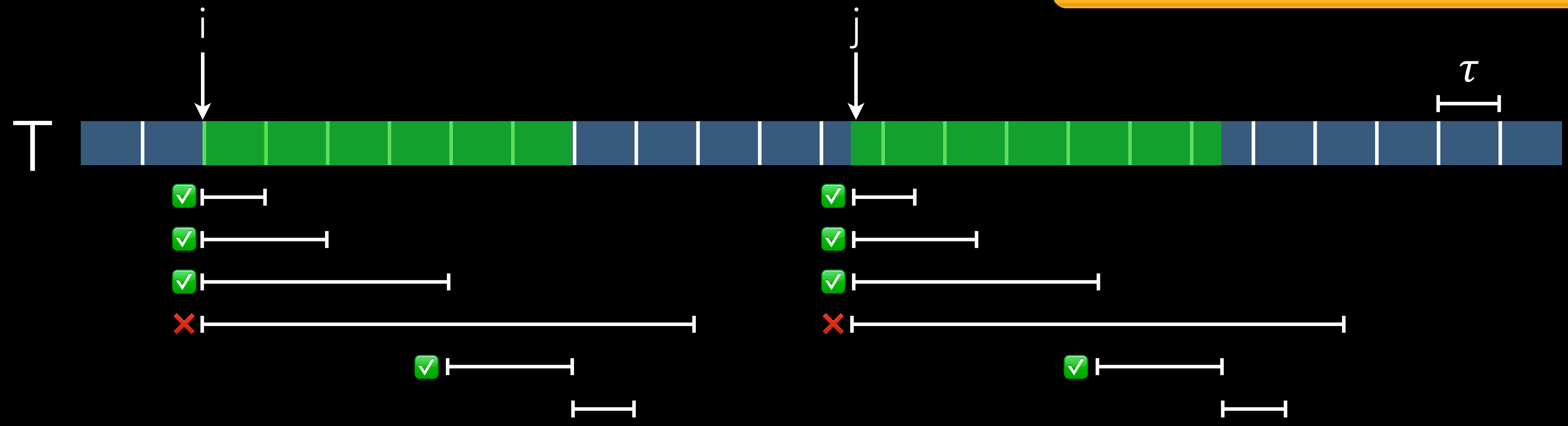


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

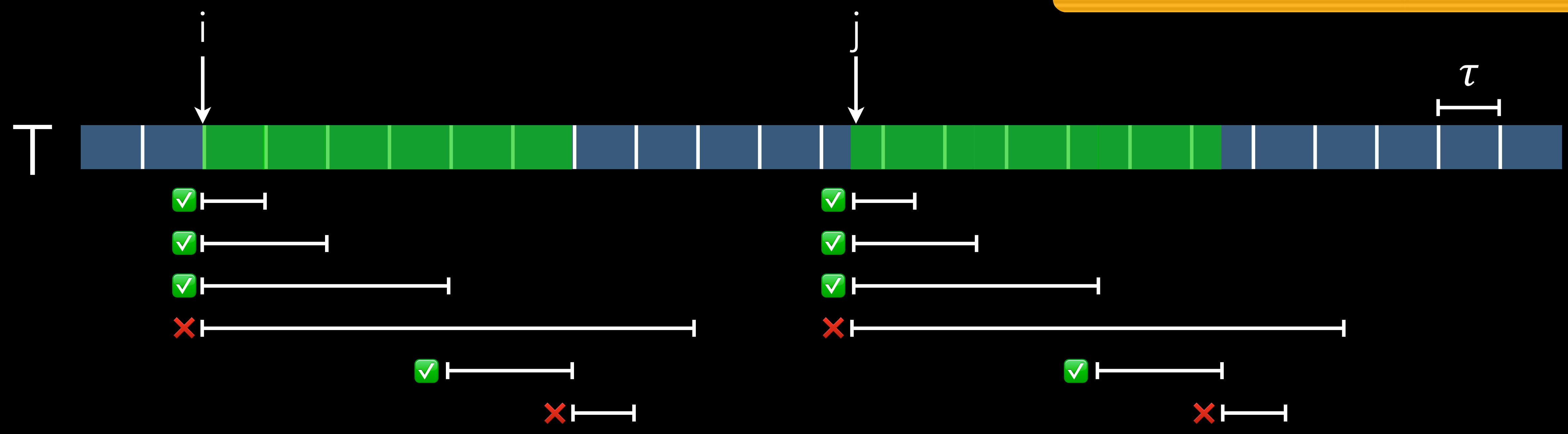


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

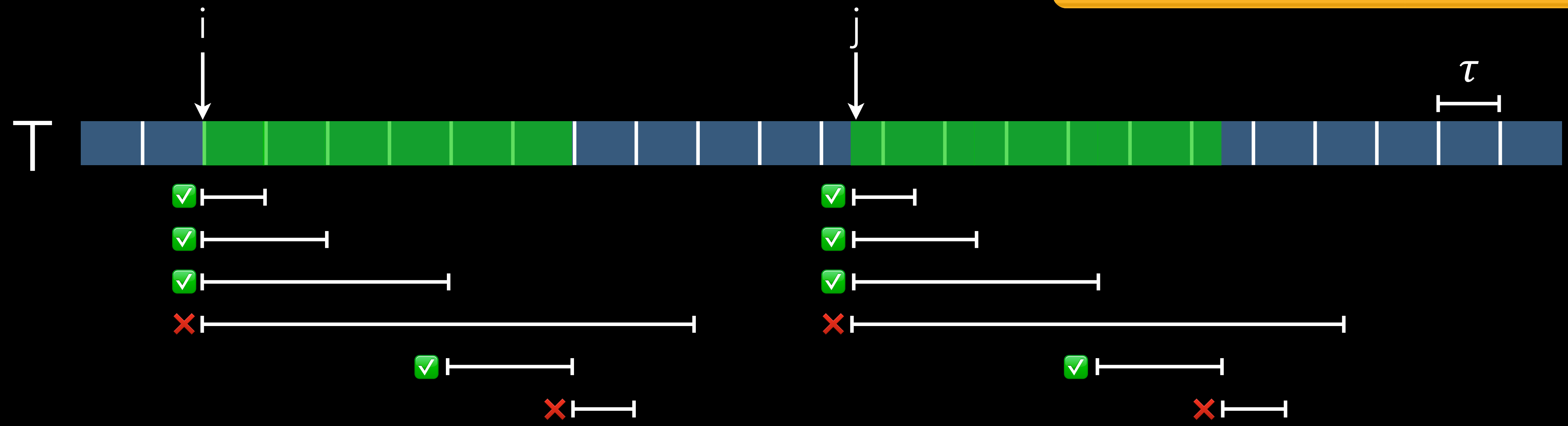


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch
   (Compare the substrings by their Karp-Rabin fingerprints)

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

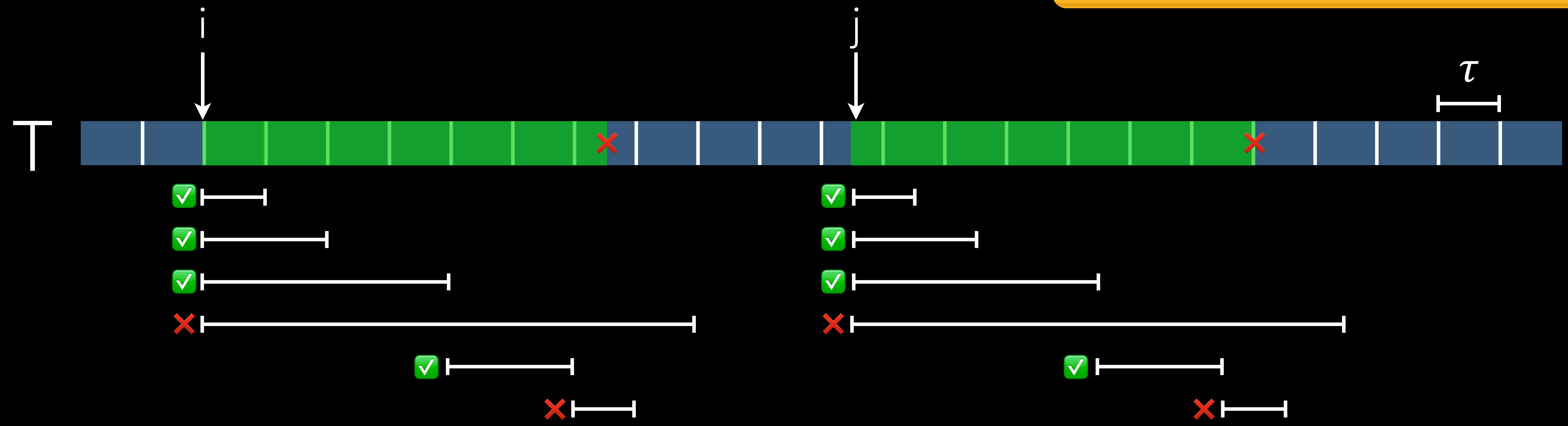


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)
2. Scan the interval directly to find the mismatch

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

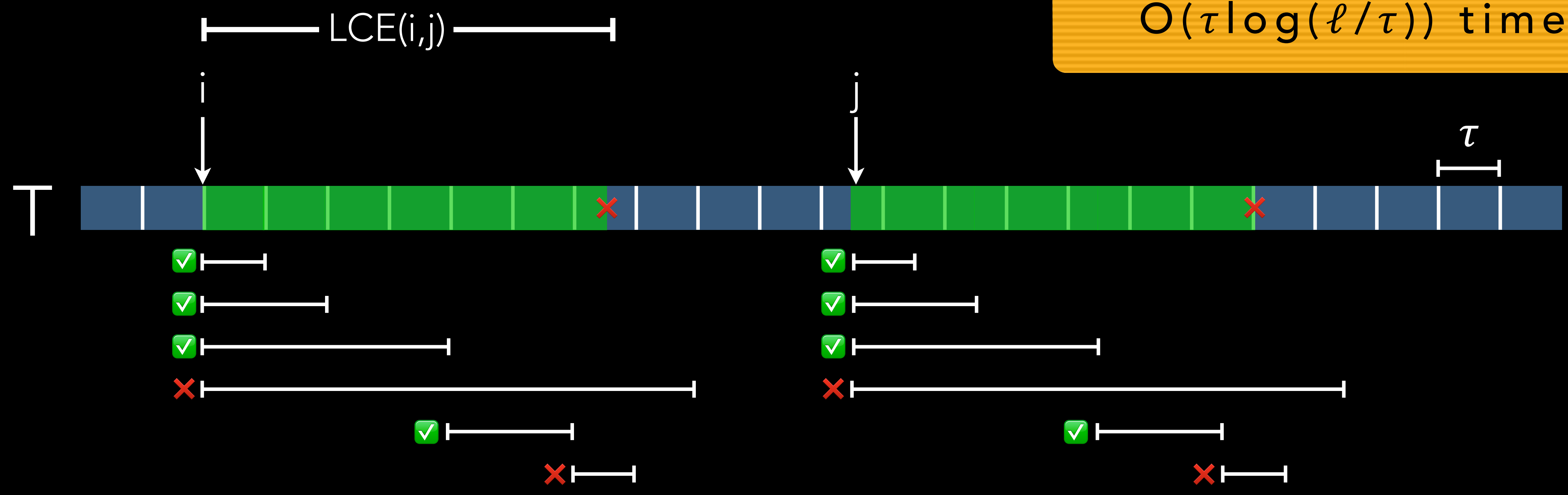


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)
2. Scan the interval directly to find the mismatch

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

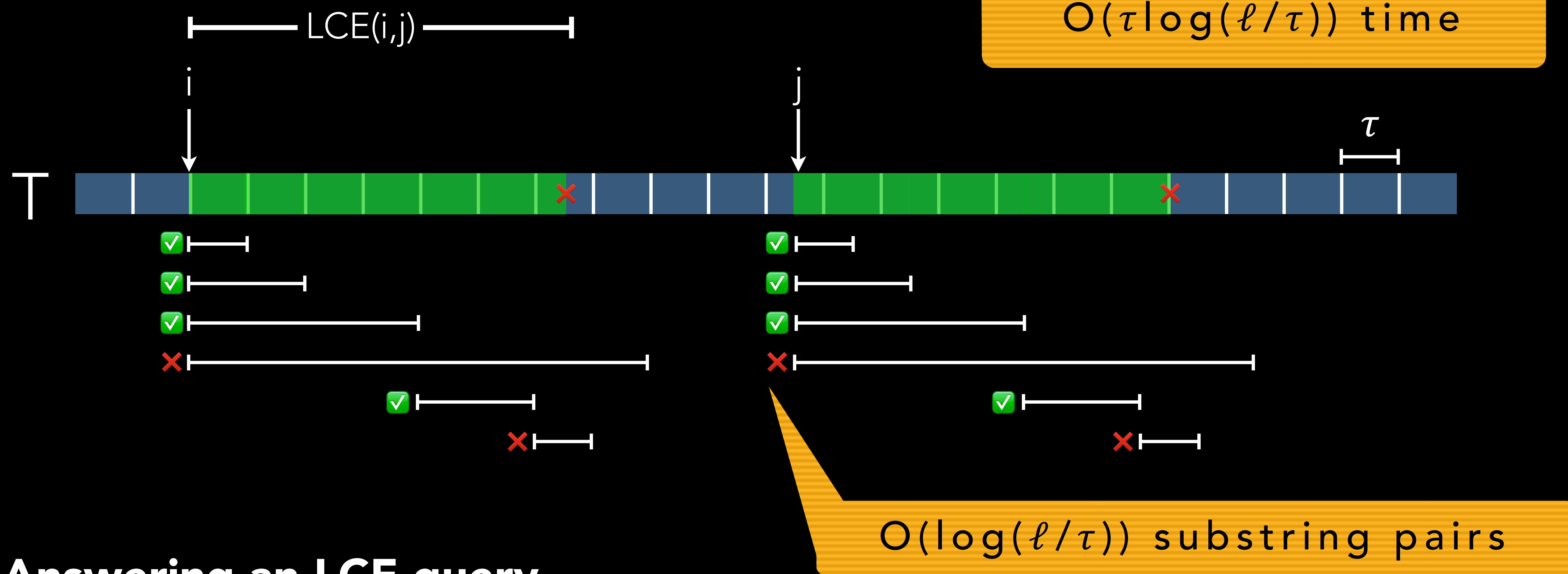


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)
2. Scan the interval directly to find the mismatch

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

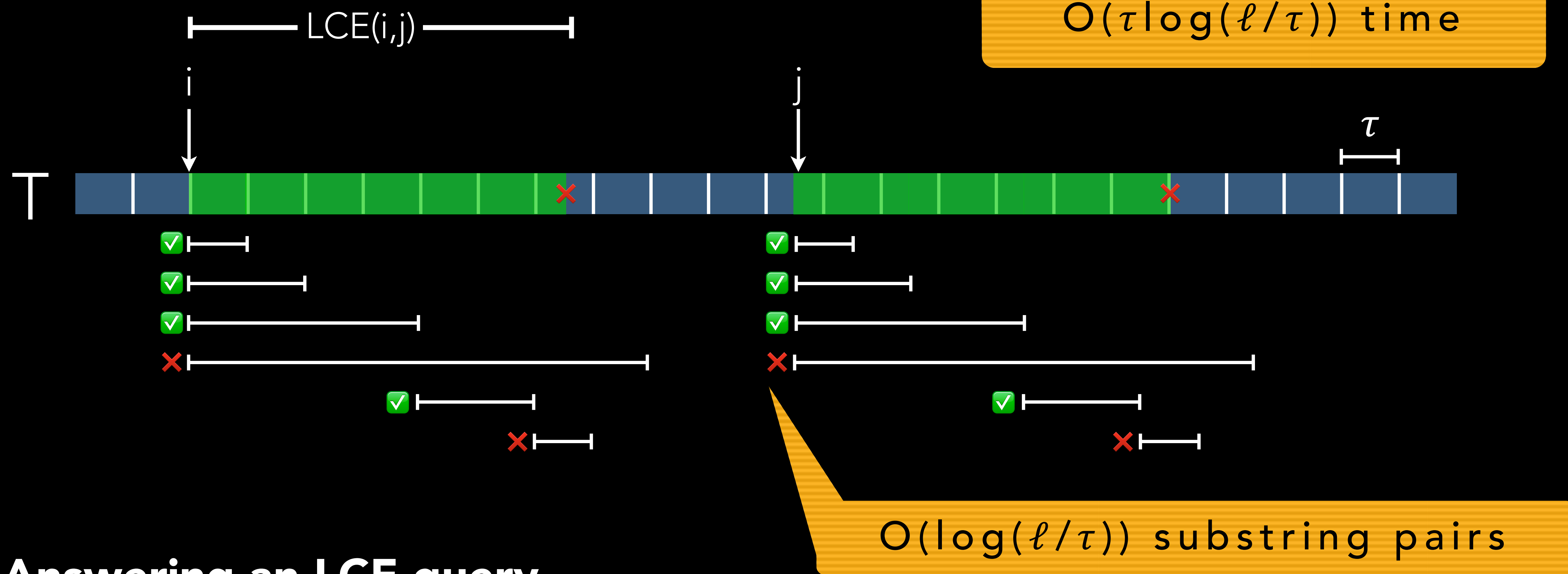


Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)
2. Scan the interval directly to find the mismatch

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time



Answering an LCE query

1. Perform exponential search to find an interval containing the first mismatch (Compare the substrings by their Karp-Rabin fingerprints)
2. Scan the interval directly to find the mismatch

Data structure

Stores fingerprint of every block aligned suffix

⇒ the fingerprint of any substring can be retrieved in $O(\tau)$ time

NEXT STEP

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time ✅

↓

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time ✅

↓

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau)$ time

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions

T

RANDOMIZED TRADE-OFF

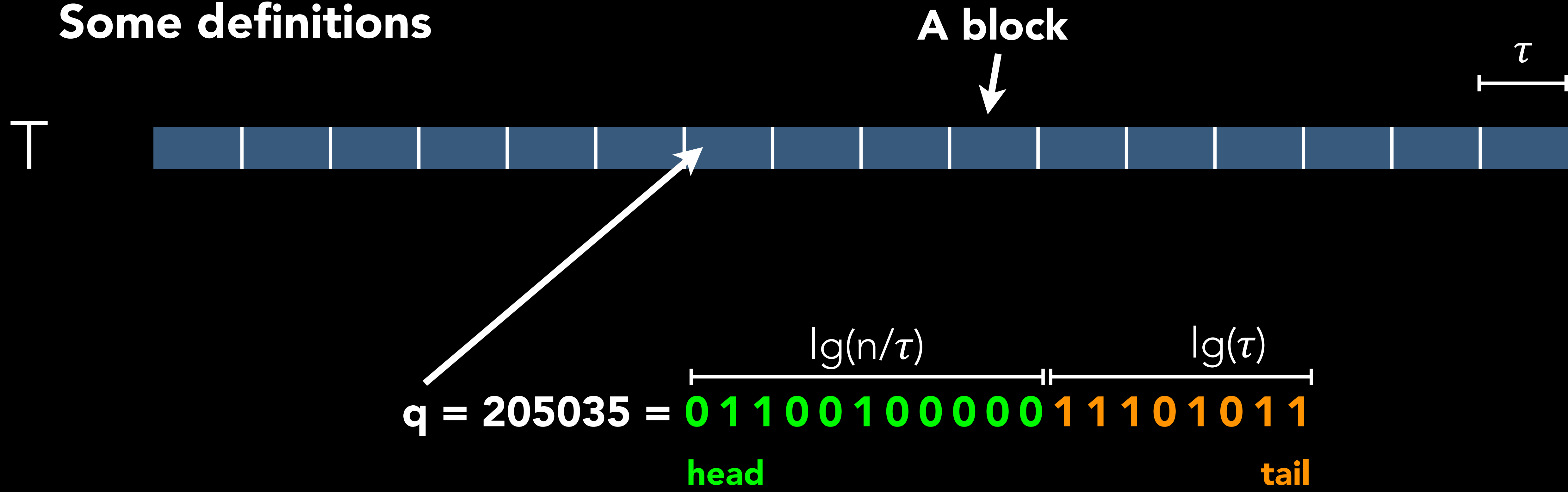
$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

Some definitions

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

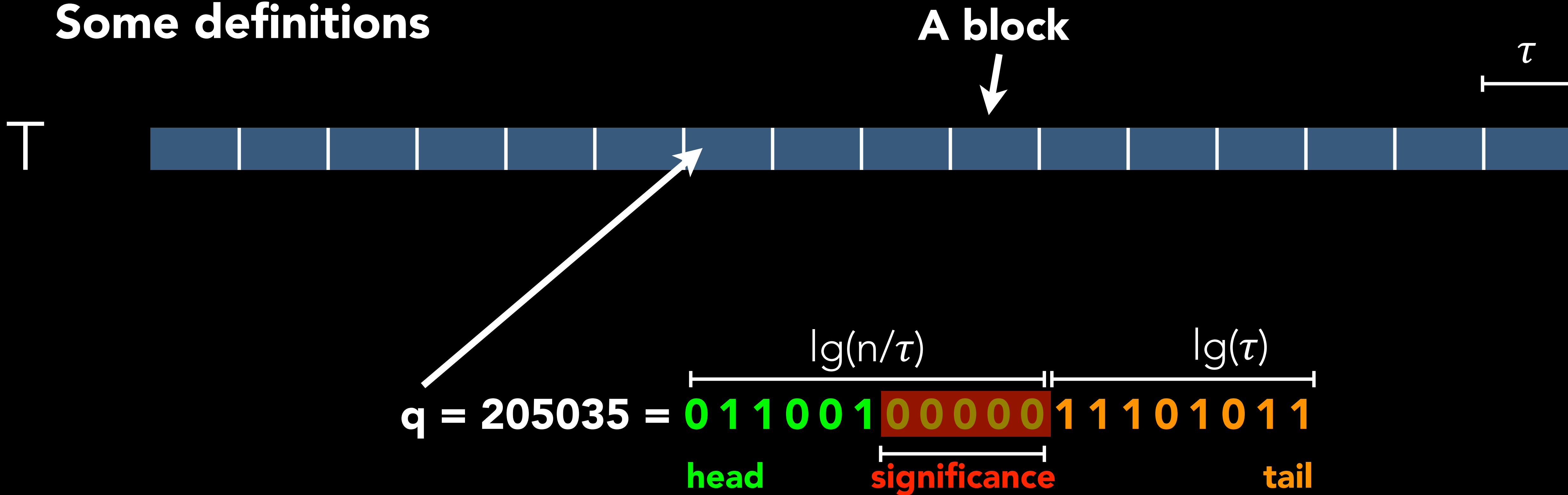
Some definitions



RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

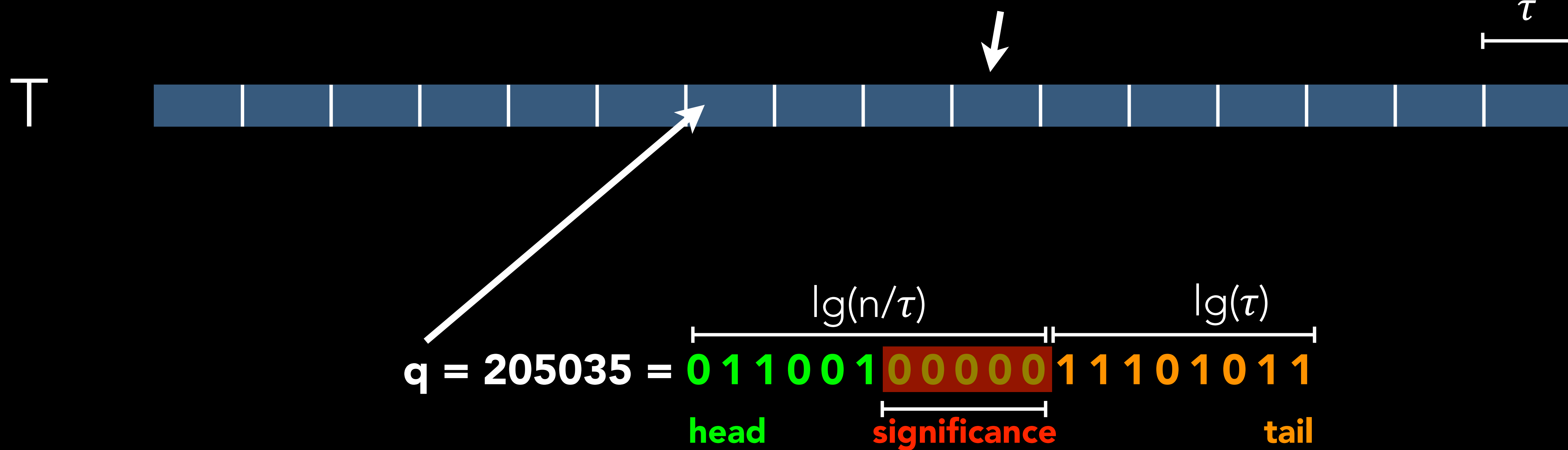
Some definitions



RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions

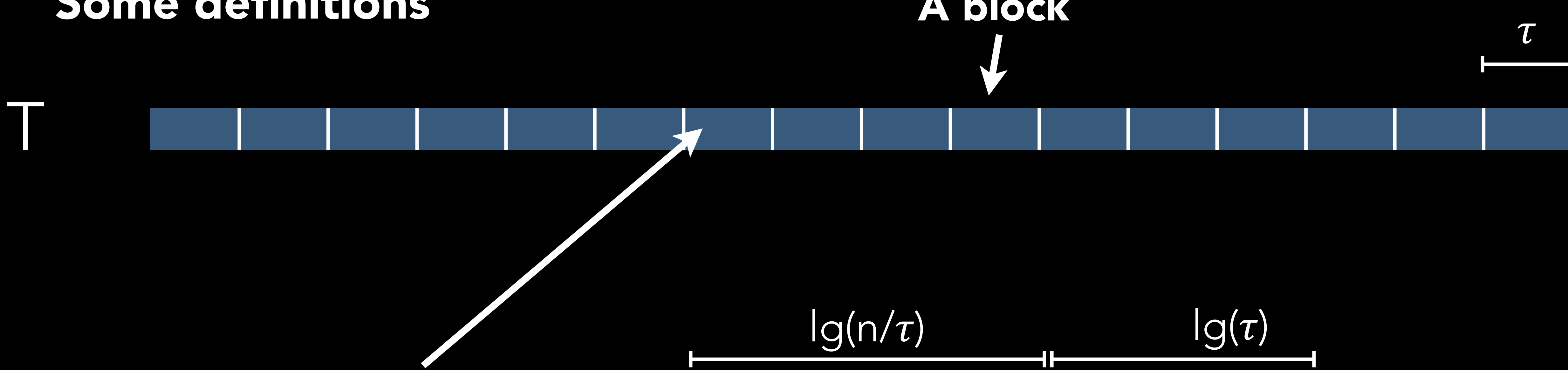


In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions



$\lg(n/\tau)$ $\lg(\tau)$

q = 205035 = 0 1 1 0 0 1 0 0 0 0 0 1 1 1 0 1 0 1 1

head **significance** **tail**

In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions

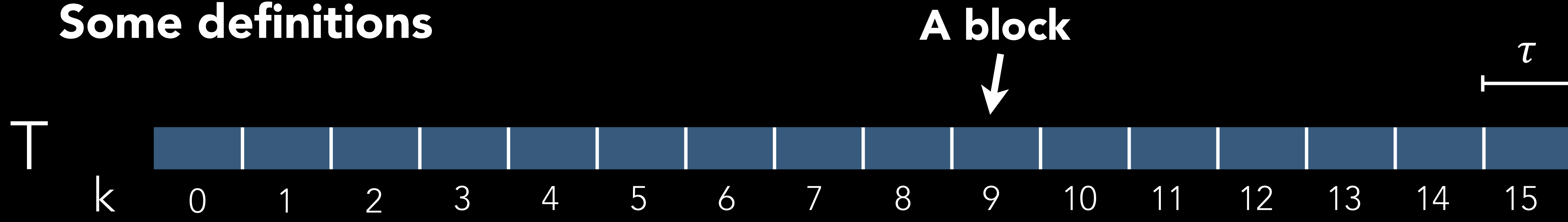
In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions



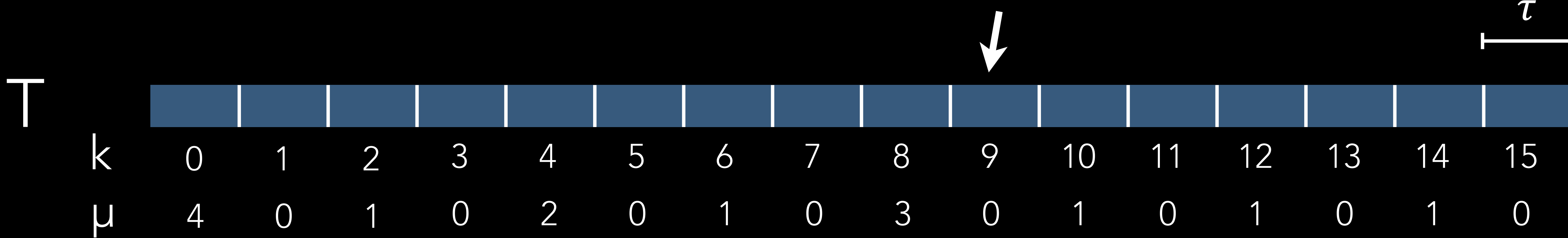
In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions



In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2} , \tau)$

Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

Some definitions

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
μ	4	0	1	0	2	0	1	0	3	0	1	0	1	0	1	0
b_k	4	1	1	1	2	1	1	1	2	1	1	1	2	1	1	1

In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

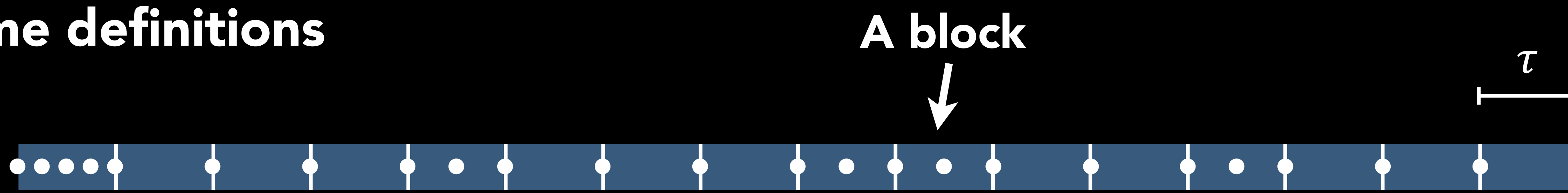
Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

Some definitions

T



k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
μ	4	0	1	0	2	0	1	0	3	0	1	0	1	0	1	0
b_k	4	1	1	1	2	1	1	1	2	1	1	1	2	1	1	1

In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

Significance of block k

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

Some definitions

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
μ	4	0	1	0	2	0	1	0	3	0	1	0	1	0	1	0
b_k	4	1	1	1	2	1	1	1	2	1	1	1	2	1	1	1

In a block k we *sample* b_k evenly spaced positions, where $b_k = \min(2^{\mu/2}, \tau)$

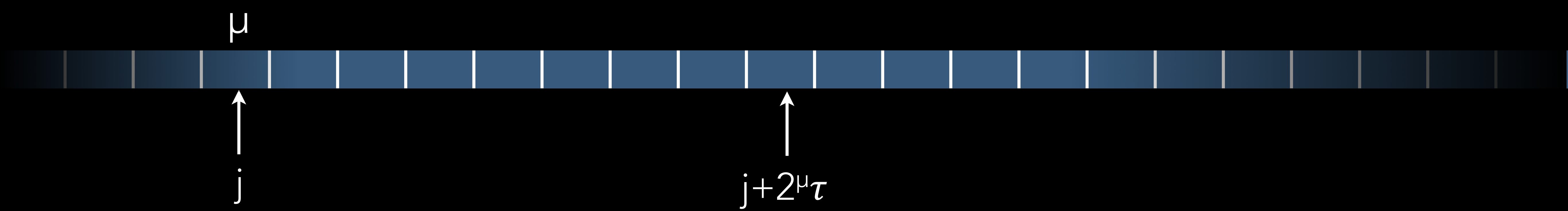
Significance of block k

Bounding the number of sampled positions

$$|\mathcal{S}| = \sum_{k=0}^{n/\tau-1} b_k \leq \sum_{\mu=0}^{\lg(n/\tau)} 2^{\lg(n/\tau)-\mu} 2^{\lfloor \mu/2 \rfloor} \leq \frac{n}{\tau} \sum_{\mu=0}^{\infty} 2^{-\mu/2} = \left(2+\sqrt{2}\right)\frac{n}{\tau} = O\left(\frac{n}{\tau}\right)$$

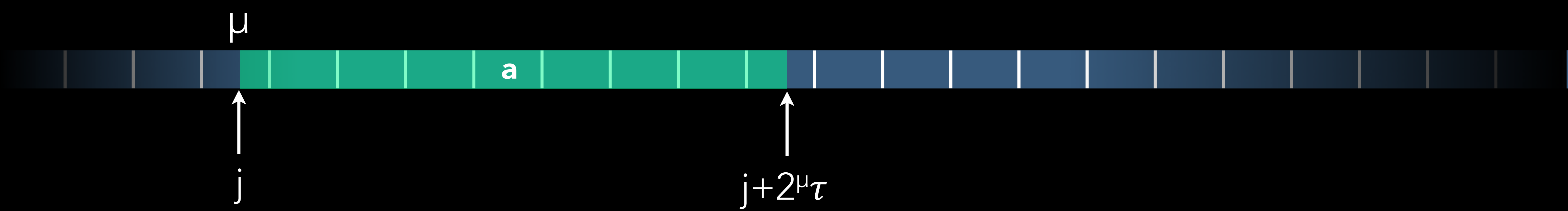
RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time



RANDOMIZED TRADE-OFF

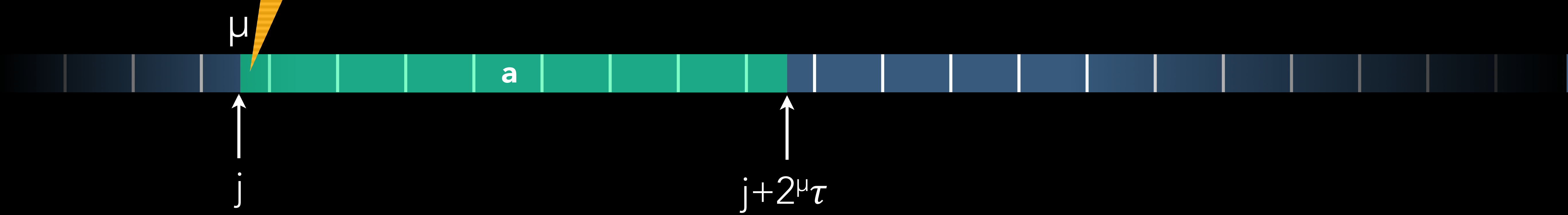
$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time



RANDOMIZED TRADE-OFF

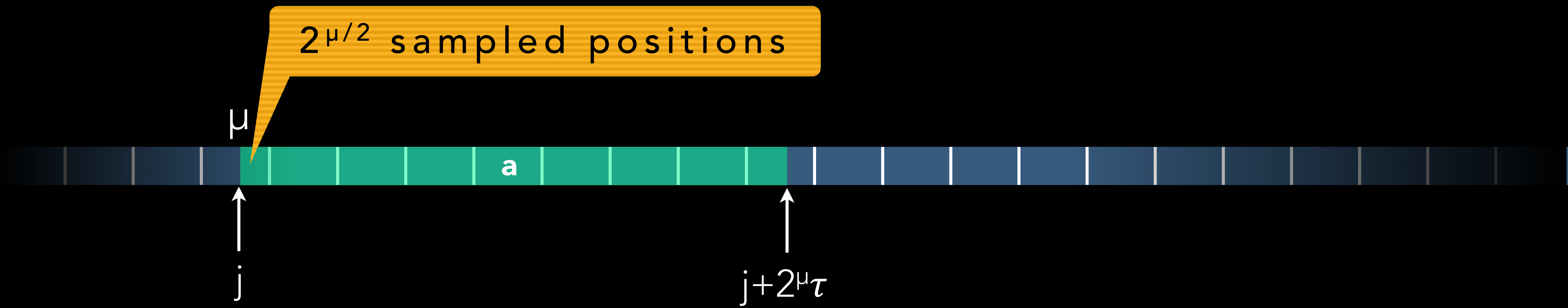
$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

$2^{\mu/2}$ sampled positions



RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

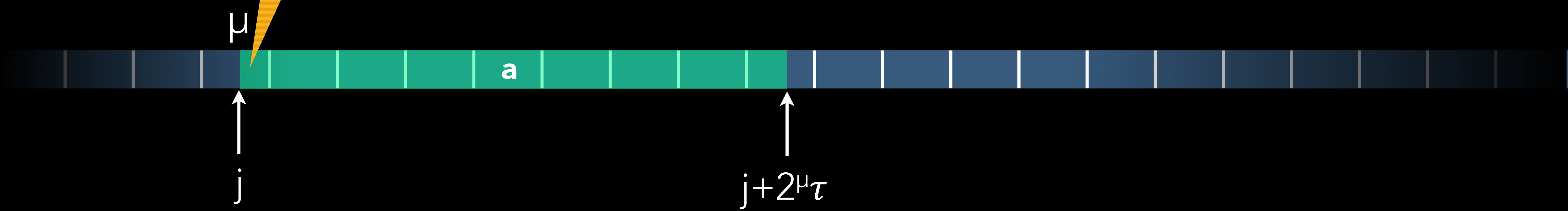


$$j = \ldots.1\overbrace{0\ldots\ldots0}^{\mu}[\text{tail}]$$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

$2^{\mu/2}$ sampled positions



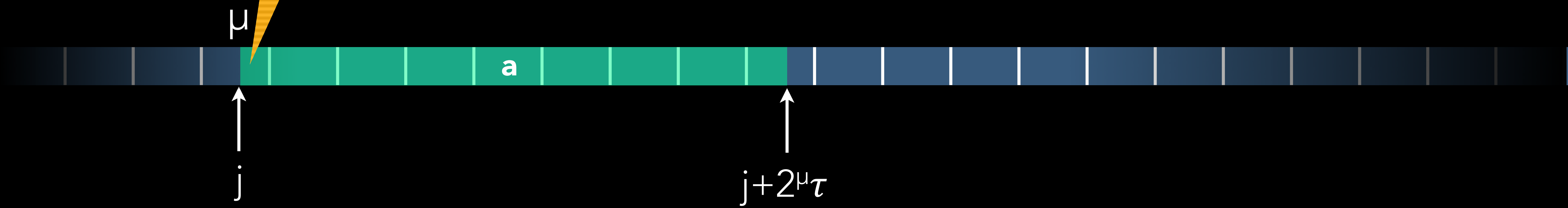
$$j = \ldots.1\overbrace{0\ldots..0}^{\mu}[\text{tail}]$$

$$2^{\mu}\tau = \ldots.10\ldots..0000..$$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

$2^{\mu/2}$ sampled positions



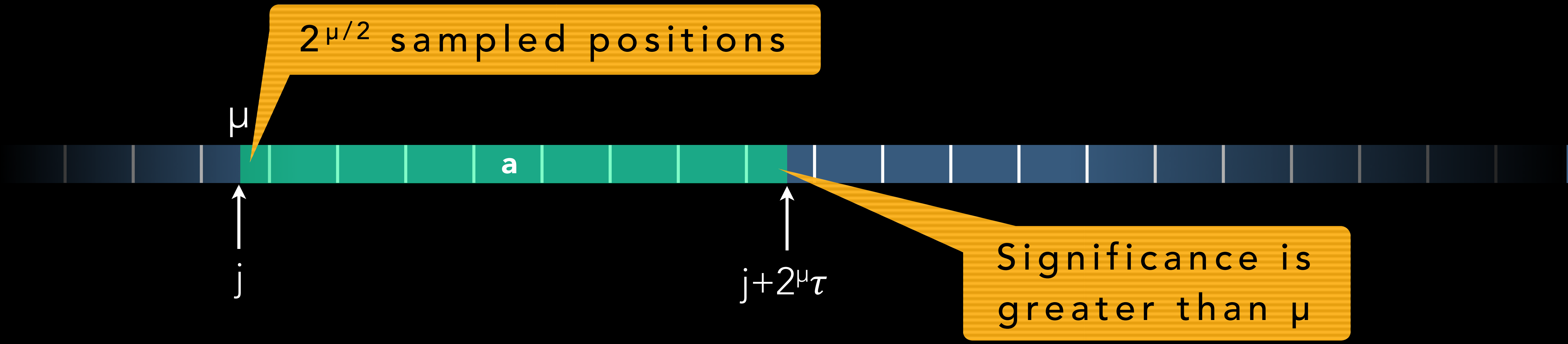
$$j = \ldots.1\overbrace{0\ldots\ldots0}^{\mu}[\text{tail}]$$

$$2^{\mu}\tau = \ldots.10\ldots\ldots0000..$$

$$j+2^{\mu}\tau = \ldots.00\ldots\ldots0[\text{tail}]$$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time



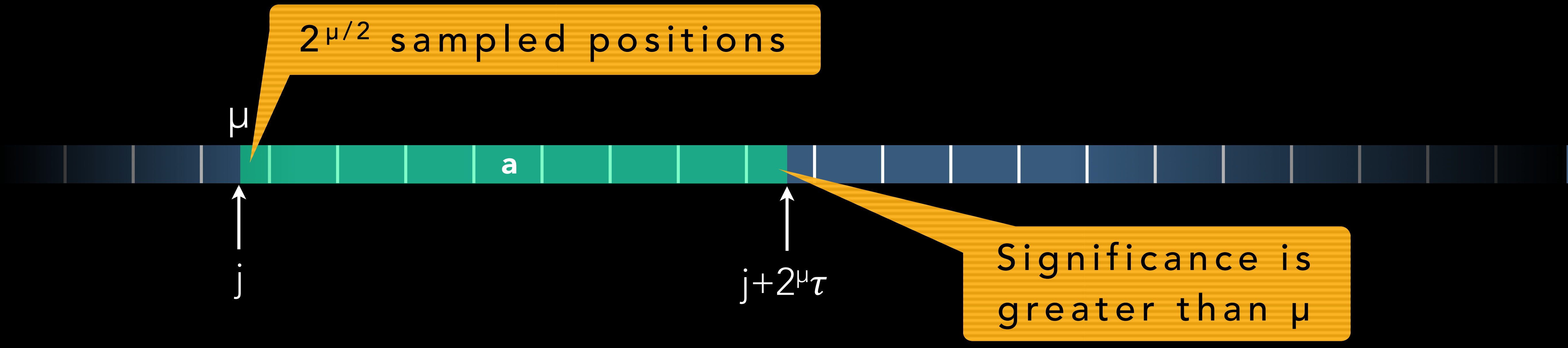
$$j = \ldots.1\overbrace{0\ldots..0}^{\mu}[\text{tail}]$$

$$2^{\mu}\tau = \ldots.10\ldots..0000..$$

$$j+2^{\mu}\tau = \ldots.00\ldots..0[\text{tail}]$$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time



$$\overbrace{}^{\mu}$$

$$j = \ldots.10\ldots..0[\text{tail}]$$

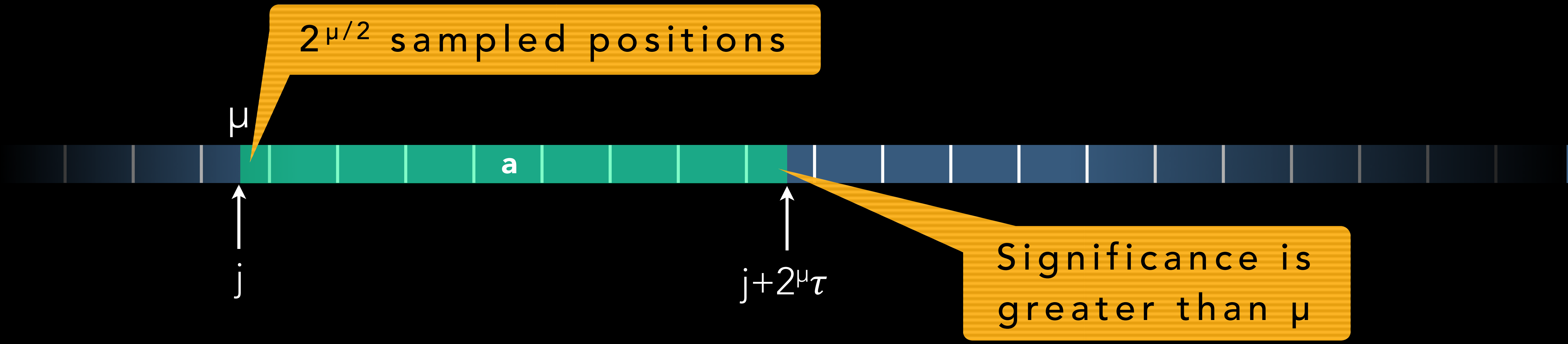
$$2^{\mu}\tau = \ldots.10\ldots..0000..$$

$$j+2^{\mu}\tau = \ldots.00\ldots..0[\text{tail}]$$

Distance to a sampled position is at most $\tau/2^{\mu/2}$

RANDOMIZED TRADE-OFF

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time



$$j = \ldots 1\,\overbrace{0 \ldots\ldots 0}^{\mu}[\text{tail}]$$

$$2^{\mu}\tau = \ldots 1\,0 \ldots\ldots 0\,000\ldots$$

$$j+2^{\mu}\tau = \ldots 0\,0 \ldots\ldots 0[\text{tail}]$$

Distance to a sampled position is at most $\tau/2^{\mu/2}$

$\Longrightarrow$ Time to compute $\varphi(a)$ is $O(1+\tau/2^{\mu/2})$

RANDOMIZED TRADE-OFF

Analysis

$O(n/\tau)$ space
$O(\tau+\log(\ell/\tau))$ time

Query time

Cost of computing a fingerprint is $O(1+\tau/2^{\mu/2})$, and μ iterates from 0 to $\log(\ell/\tau)$ and back to 0, thus the query time becomes

$$O\left(\sum_{\mu=0}^{\lg(\ell/\tau)} 1+\tau/2^{\lfloor\mu/2\rfloor}\right) = O(\tau+\log(\ell/\tau))$$

Space

Cost is the total number of sampled positions/fingerprints

$$|\mathcal{S}| = \sum_{k=0}^{n/\tau-1} b_k \leq \sum_{\mu=0}^{\lg(n/\tau)} 2^{\lg(n/\tau)-\mu}2^{\lfloor\mu/2\rfloor} \leq \frac{n}{\tau}\sum_{\mu=0}^{\infty} 2^{-\mu/2} = \left(2+\sqrt{2}\right)\frac{n}{\tau} = O\left(\frac{n}{\tau}\right)$$

NEXT STEP

$O(n/\tau)$ space
$O(\tau \log^2(n/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau \log(\ell/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau + \log(\ell/\tau))$ time

↓

$O(n/\tau)$ space
$O(\tau)$ time

SOLUTION FOR LONG LCES

$O(n/\tau)$ space
$O(\tau)$ time

Theorem

There is an $O(n/\tau)$ space data structure that in $O(1)$ time either

A. computes the answer to an LCE(i,j) query, or

B. returns a certificate that $\ell < \tau^2$

Observation

In case B the query time of our previous algorithm becomes $O(\tau + \log(\ell/\tau)) = O(\tau)$

Technique

Difference covers

SUMMARY & OPEN PROBLEMS

MAIN THEOREM

The LCE problem can be solved in $O(n/\tau)$ space and $O(\tau)$ time for all $1 \leq \tau \leq n$

Lower bound from RMQ implies a time-space product of $\Omega(n/\log n)$
Can we close this gap?

Can we obtain optimal preprocessing times?