Amortized Analysis

Dynamic Tables

Dynamic Table

T : table

Operations: Insert, Delete items of T

Dynamic Table

T : table

Operations: Insert, Delete items of T

Load Factor $\alpha(T)$ =

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(\text{T}) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T	**a**	**b**	**c**	

size(T) = 4

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T				
	a	**b**	**c**	

size(T) = 4

$\alpha(T) = 3/4$

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T

a	**b**	**c**	

size(T) = 4

Insert(**d**)

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T	**a**	**b**	**c**	**d**

size(T) = 4

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T	a	b	c	d

size(T) = 4

Table is full, i.e. $\alpha(T) = 1$

Dynamic Table

T : table

Operations: Insert, Delete items of T

Load Factor $\alpha(T) = \dfrac{\text{\# items currently stored in T}}{\text{size(T)}}$

T				
	a	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

Table is full, i.e. $\alpha(T) = 1$

Dynamic Table

T : table

Operations: Insert, Delete items of T

$$\text{Load Factor } \alpha(T) = \frac{\text{\# items currently stored in T}}{\text{size(T)}}$$

T				
	a	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

Table is full, i.e. $\alpha(T) = 1$

Problem: Insert element when T is full

Table Expansion

Table Expansion

(1) Allocate new table T larger than T

Table Expansion

(1) Allocate new table T larger than T

Typically, size(new T) = 2 size(T)

Table Expansion

(1) Allocate new table T larger than T

Typically, size(new T) = 2 size(T)

(2) Copy all items of T into the new T

Table Expansion

(1) Allocate new table T larger than T

Typically, size(new T) = 2 size(T)

(2) Copy all items of T into the new T

(3) Insert the new element into the new T

Table Expansion

(1) Allocate new table T larger than T

Typically, size(new T) = 2 size(T)

(2) Copy all items of T into the new T

(3) Insert the new element into the new T

With this scheme, α(T) remains $\geq$ 1/2

(i.e. no more than half the space of T is wasted)

Dynamic Table

T	**a**	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

Dynamic Table

T	a	b	c	d

Insert(**e**)

size(T) = 4

T								

size(T) = 8

Dynamic Table

T				
	a	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

T								
	a	**b**	**c**	**d**				

size(T) = 8

Dynamic Table

T

a	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

T

a	**b**	**c**	**d**	**e**			

size(T) = 8

Dynamic Table

T

a	**b**	**c**	**d**

Insert(**e**)

size(T) = 4

T

a	**b**	**c**	**d**	**e**			

Cost of Insert(**e**) = 4 + 1

size(T) = 8

Dynamic Table

T	**a**	**b**	**c**	**d**

size(T) = 4

Insert(**e**)

T	**a**	**b**	**c**	**d**	**e**			

size(T) = 8

Cost of Insert(**e**) = $\overbrace{4}^{\text{Cost of table expansion}} + \overbrace{1}^{\text{Cost of inserting e}}$

Amortized Analysis

Starting from empty table T of size 1,

What is the total cost of n successive Inserts into T?

Aggregate Analysis

Example: $n = 25$

Aggregate Analysis

Example: $n = 25$

Total cost =

Aggregate Analysis

Example: n = 25

Cost of inserting elements

Cost of table expansion

Total cost =

Aggregate Analysis

Example: n = 25

	Cost of inserting elements	Cost of table expansion
Total cost =	25	

Aggregate Analysis

Example: n = 25

Cost of inserting elements

Cost of table expansion

Total cost = 25

Aggregate Analysis

Example: n = 25

	Cost of inserting elements	Cost of table expansion
Total cost =	25	+ 1

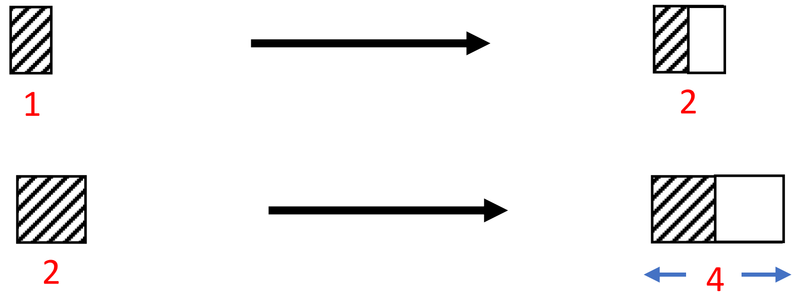
Aggregate Analysis

Example: n = 25

Cost of inserting elements

Cost of table expansion

Total cost = 25 + 1



Aggregate Analysis

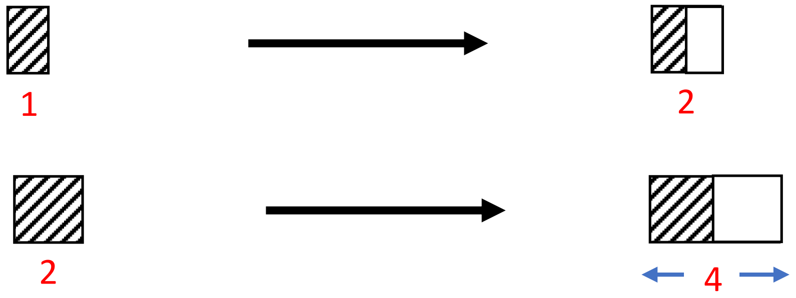
Example: n = 25

Cost of inserting elements

Cost of table expansion

Total cost = 25 + 1

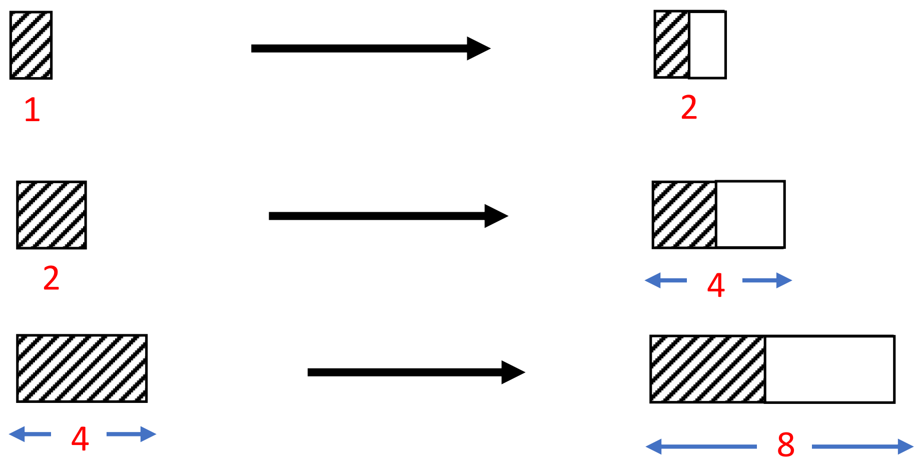
+ 2



Aggregate Analysis

Example: n = 25

	Cost of inserting elements	Cost of table expansion
Total cost =	25	+ 1
		+ 2



Aggregate Analysis

Example: $n = 25$

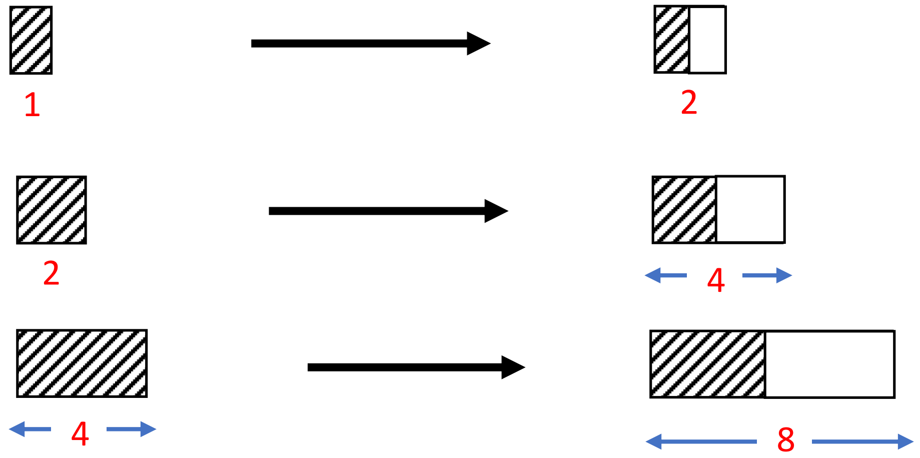
Cost of inserting elements

Cost of table expansion

Total cost = 25 + 1

+ 2

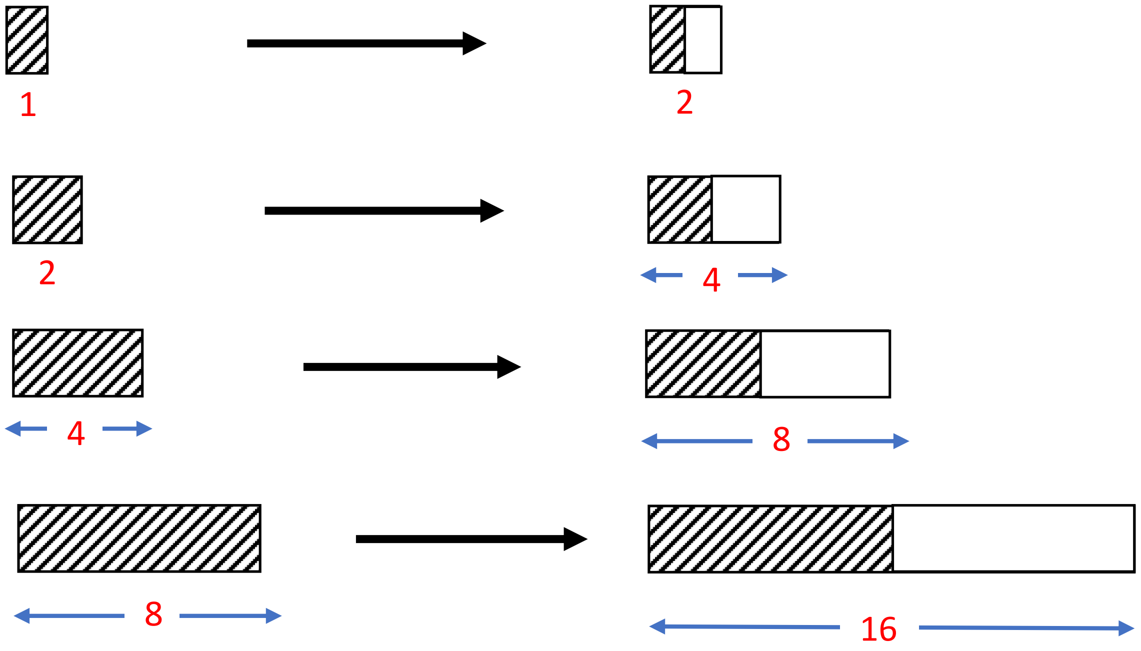
+ 4



Aggregate Analysis

Example: n = 25

	Cost of inserting elements	Cost of table expansion
Total cost =	25	+ 1
		+ 2
		+ 4



Aggregate Analysis

Example: n = 25

Cost of inserting elements

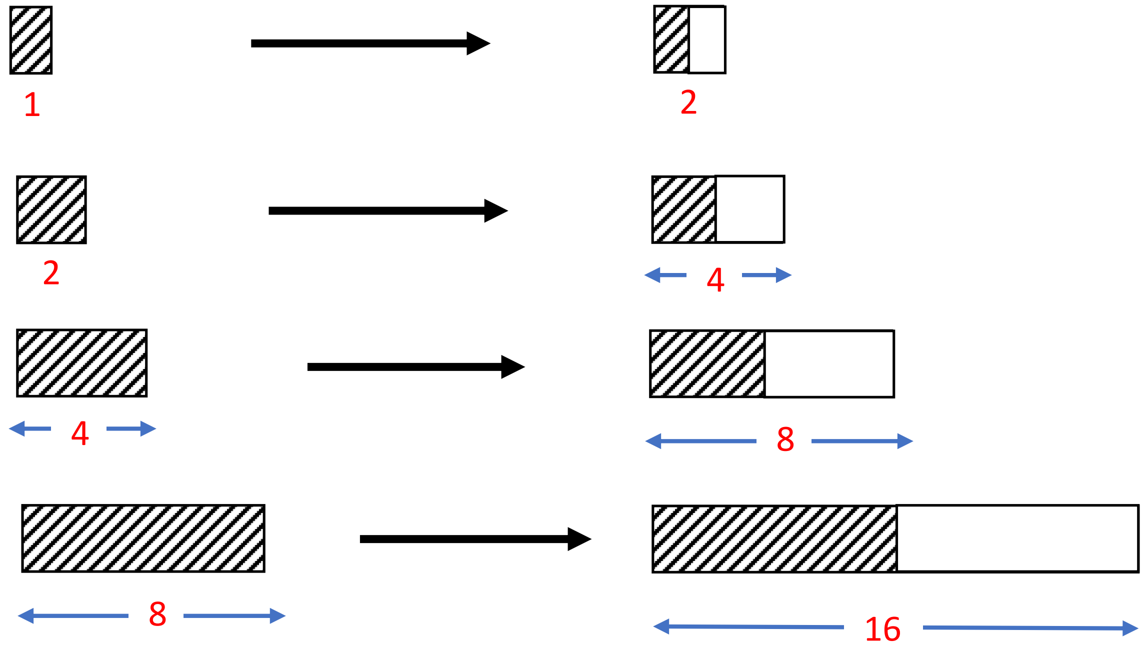
Cost of table expansion

Total cost = 25 + 1

+ 2

+ 4

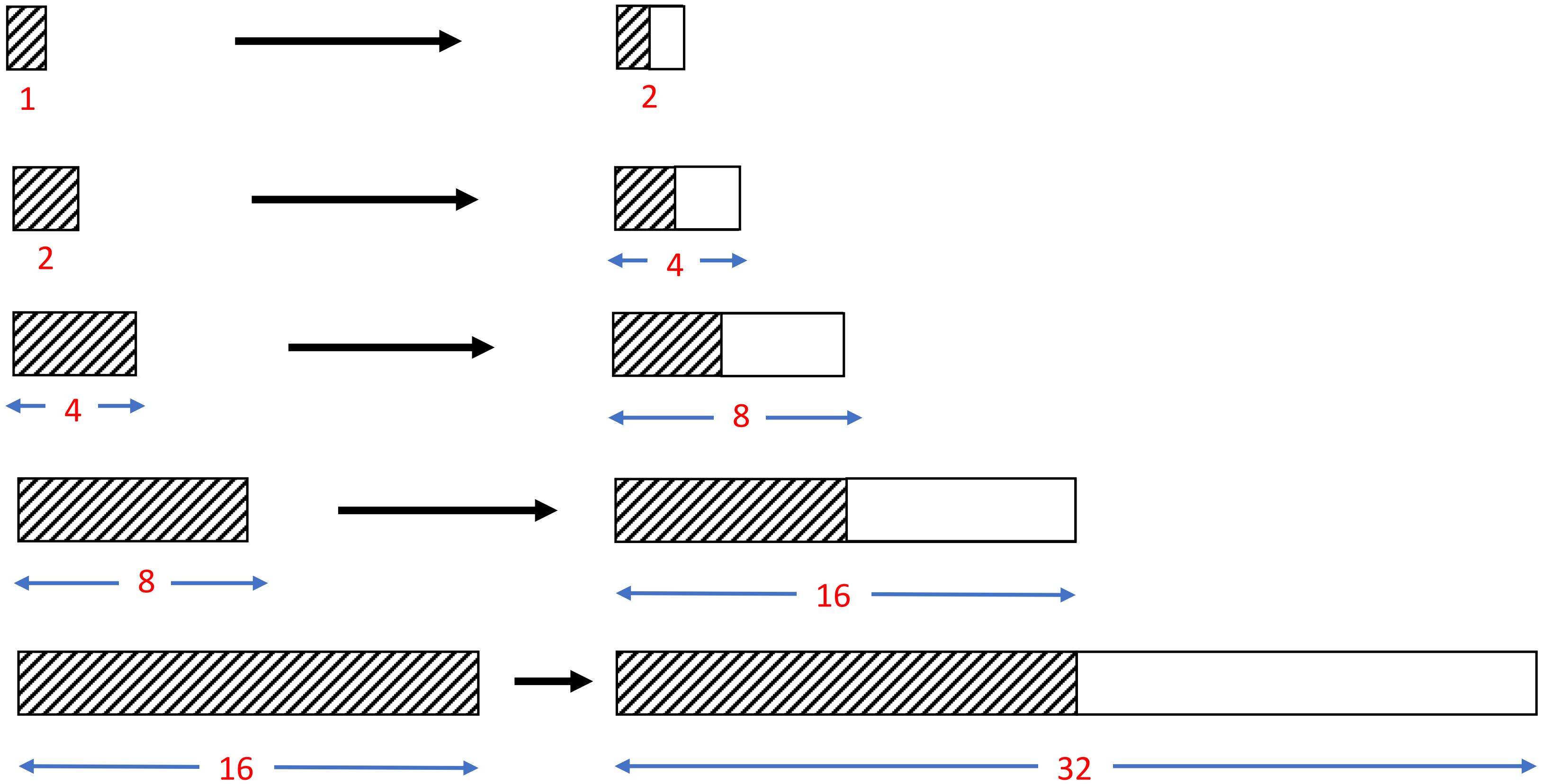
+ 8



Aggregate Analysis

Example: $n = 25$

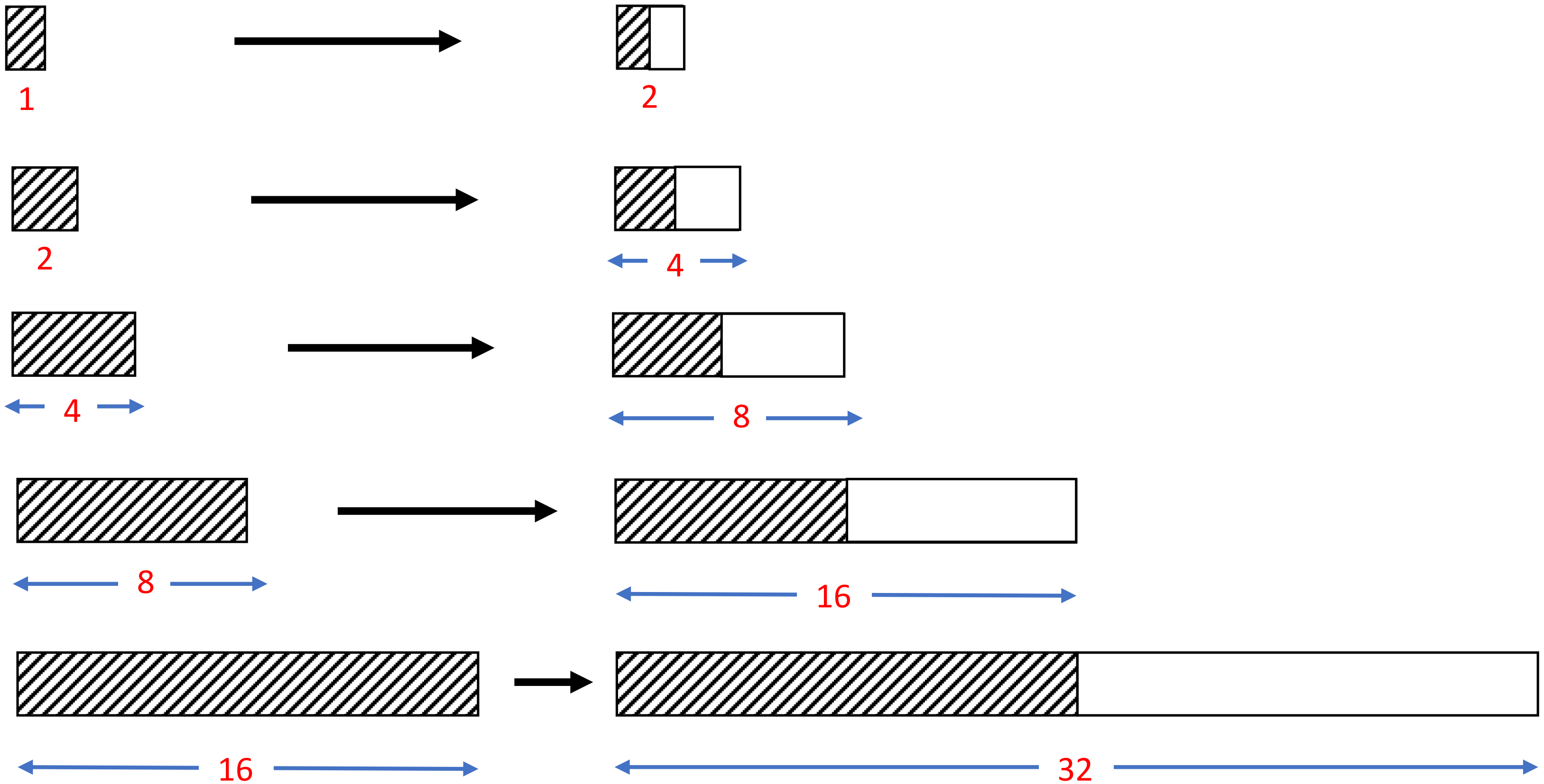
	Cost of inserting elements	Cost of table expansion
Total cost =	25	+ 1
		+ 2
		+ 4
		+ 8



Aggregate Analysis

Example: n = 25

	Cost of inserting elements	Cost of table expansion
Total cost =	25	+ 1
		+ 2
		+ 4
		+ 8
		+ 16



Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Total cost of n Inserts = n + all powers of 2 smaller than n

Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Total cost of n Inserts = n + all powers of 2 smaller than n

$$\text{Total cost of } n \text{ Inserts} \leq n + \sum_{k=0}^{k=\lfloor \log_2 n \rfloor} 2^k$$

Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Total cost of n Inserts = n + all powers of 2 smaller than n

$$\text{Total cost of } n \text{ Inserts} \leq n + \sum_{k=0}^{k=\lfloor \log_2 n \rfloor} 2^k$$

$$\leq n + 2n$$

Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Total cost of n Inserts = n + all powers of 2 smaller than n

$$\text{Total cost of } n \text{ Inserts} \leq n + \sum_{k=0}^{k=\lfloor \log_2 n \rfloor} 2^k$$

$$\leq n + 2n$$

$$= 3n$$

Aggregate Analysis

Total cost of 25 Inserts = 25 + all powers of 2 smaller than 25

Total cost of n Inserts = n + all powers of 2 smaller than n

$$\text{Total cost of } n \text{ Inserts} \leq n + \sum_{k=0}^{k=\lfloor \log_2 n \rfloor} 2^k$$

$$\leq n + 2n$$

$$= 3n$$

Amortized cost per Insert $\leq$ 3n/n

Aggregate Analysis

Total cost of 25 Inserts $=$ 25 + all powers of 2 smaller than 25

Total cost of n Inserts $=$ n + all powers of 2 smaller than n

$$\text{Total cost of } n \text{ Inserts} \leq n + \sum_{k=0}^{k=\lfloor \log_2 n \rfloor} 2^k$$

$$\leq n + 2n$$

$$= 3n$$

Amortized cost per Insert $\leq$ $3n/n$ $\Rightarrow$ Amortized cost per Insert is $O(1)$

Accounting method

Recall that if:

c_i : actual cost of i^{th} operation

$\hat{c}_i$: cost charged for the i^{th} operation [i.e. amortized cost of i^{th} operation]

Accounting method

Recall that if:

c_i : actual cost of i^{th} operation

$\hat{c}_i$: cost charged for the i^{th} operation [i.e. amortized cost of i^{th} operation]

We require $\sum_{i=1}^{n} \hat{c}_i \geq \sum_{i=1}^{n} c_i$ for all sequence of n operations

Accounting method

Recall that if:

c_i : actual cost of i^{th} operation

$\hat{c}_i$: cost charged for the i^{th} operation [i.e. amortized cost of i^{th} operation]

We require $\sum_{i=1}^{n} \hat{c}_i \geq \sum_{i=1}^{n} c_i$ for all sequence of n operations

Equivalently, $\sum_{i=1}^{n} \hat{c}_i - \sum_{i=1}^{n} c_i \geq 0$

Accounting method

Recall that if:

c_i : actual cost of i^{th} operation

$\hat{c}_i$: cost charged for the i^{th} operation   [i.e. amortized cost of i^{th} operation]

We require $\sum_{i=1}^{n} \hat{c}_i \geq \sum_{i=1}^{n} c_i$ for all sequence of n operations

Equivalently, $\underbrace{\sum_{i=1}^{n} \hat{c}_i - \sum_{i=1}^{n} c_i}_{\text{Total credit in the data structure}} \geq 0$

Accounting method

Insert(x)

	0	0	0	0	0	0	0	0
T	**a**	**b**	**c**	**d**				

Accounting method

Insert(**x**)

Insert(**x**) is charged

	0	0	0	0	0	0	0	0
T	**a**	**b**	**c**	**d**				

Accounting method

Insert(**x**)

	0	0	0	0	0	0	0	0
T	**a**	**b**	**c**	**d**	**x**			

Insert(**x**) is charged

$1 for inserting **x** (actual cost)

Accounting method

Insert(**x**)

	0	0	0	0	$1	0	0	0
T	**a**	**b**	**c**	**d**	**x**			

Insert(**x**) is charged

$1 for inserting **x** (actual cost)
+ $1 credit on **x** (for copying **x** over)

Accounting method

Insert(**x**)

	$1	0	0	0	$1	0	0	0
T	**a**	**b**	**c**	**d**	**x**			

Insert(**x**) is charged

$1 for inserting **x** (actual cost)
+ $1 credit on **x** (for copying **x** over)
+ $1 credit on **a** (for copying **a** over)

Accounting method

Insert(**x**)

	$1	0	0	0	$1	0	0	0
T	**a**	**b**	**c**	**d**	**x**			

Insert(**x**) is charged $3

$1 for inserting **x** (actual cost)
+ $1 credit on **x** (for copying **x** over)
+ $1 credit on **a** (for copying **a** over)

Accounting method

Insert(**y**)

Insert(**y**) is charged $3

	$1	$1	0	0	$1	$1	0	0
T	**a**	**b**	**c**	**d**	**x**	**y**		

Accounting method

Insert(**z**)

Insert(**z**) is charged $3

	$1	$1	$1	0	$1	$1	$1	0
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	

Accounting method

Insert(w)

Insert(w) is charged $3

	$1	$1	$1	$1	$1	$1	$1	$1
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**

Accounting method

When table full,

Total Credit = # elements in the Table

	\$1	\$1	\$1	\$1	\$1	\$1	\$1	\$1
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**

Accounting method

	$1	$1	$1	$1	$1	$1	$1	$1
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**

When table full,

Total Credit = # elements in the Table

On next Insert, use credits to move elements into new table

Accounting method

Insert(**v**)

	$1	$1	$1	$1	$1	$1	$1	$1
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**

When table full,

Total Credit = # elements in the Table

On next Insert, use credits to move elements into new table

Accounting method

Insert(v)

	$1	$1	$1	$1	$1	$1	$1	$1
T	a	b	c	d	x	y	z	w

	0	0	0	0	0	0	0	0								
T	a	b	c	d	x	y	z	w								

Accounting method

Insert(**v**)

Insert(**v**) is charged $3

	$1	$1	$1	$1	$1	$1	$1	$1
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**

	$1	0	0	0	0	0	0	0	$1							
T	**a**	**b**	**c**	**d**	**x**	**y**	**z**	**w**	**v**							

Accounting method

Insert(v)

Insert(v) is charged $3

and so on…

	$1	$1	$1	$1	$1	$1	$1	$1
T	a	b	c	d	x	y	z	w

	$1	0	0	0	0	0	0	0	$1							
T	a	b	c	d	x	y	z	w	v							

Accounting method

σ : Sequence of n Inserts, starting from empty table T of size 1

Accounting method

σ : Sequence of n Inserts, starting from empty table T of size 1

Charging \$3 per Insert in σ
ensures total credit is always ≥ 0

Accounting method

σ : Sequence of n Inserts, starting from empty table T of size 1

Charging \$3 per Insert in σ ensures total credit is always ≥ 0 $\Rightarrow$ Amortized cost per Insert is O(1)

Insert and Delete

Insert and Delete

Problem: After deleting items, $\alpha(T)$ decreases

Insert and Delete

Problem: After deleting items, $\alpha(T)$ decreases $\Rightarrow$ Memory waste increases

Insert and Delete

Problem: After deleting items, $\alpha(\mathrm{T})$ decreases $\Rightarrow$ Memory waste increases

If $\alpha(\mathrm{T})$ becomes too small, reduce memory waste by

Insert and Delete

Problem: After deleting items, $\alpha(\mathrm{T})$ decreases $\Rightarrow$ Memory waste increases

If $\alpha(\mathrm{T})$ becomes too small, reduce memory waste by

(1) Allocating new smaller table,

(2) Copying all the items to the new table

Insert and Delete

Problem: After deleting items, $\alpha(\mathrm{T})$ decreases $\Rightarrow$ Memory waste increases

If $\alpha(\mathrm{T})$ becomes too small, reduce memory waste by

(1) Allocating new smaller table,

(2) Copying all the items to the new table

Insert and Delete

Problem: After deleting items, $\alpha(T)$ decreases $\Rightarrow$ Memory waste increases

How small ?

If $\alpha(T)$ becomes too small, reduce memory waste by

How small ?

(1) Allocating new smaller table,

(2) Copying all the items to the new table

Insert and Delete

Problem: After deleting items, $\alpha(\mathrm{T})$ decreases $\Rightarrow$ Memory waste increases

How small ?

If $\alpha(\mathrm{T})$ becomes too small, reduce memory waste by

How small ?

(1) Allocating new smaller table,

(2) Copying all the items to the new table

We want:

Insert and Delete

Problem: After deleting items, $\alpha(\mathrm{T})$ decreases $\Rightarrow$ Memory waste increases

How small ?

If $\alpha(\mathrm{T})$ becomes too small, reduce memory waste by

How small ?

(1) Allocating new smaller table,

(2) Copying all the items to the new table

We want:

(1) $\alpha(\mathrm{T}) \geq$ constant c (to reduce memory waste)

Insert and Delete

Problem: After deleting items, $\alpha(T)$ decreases $\Rightarrow$ Memory waste increases

How small ?

If $\alpha(T)$ becomes too small, reduce memory waste by

How small ?

(1) Allocating new smaller table,

(2) Copying all the items to the new table

We want:

(1) $\alpha(T) \geq$ constant c (to reduce memory waste)

(2) Amortized cost per operation (insert/delete) is $O(1)$

Insert and Delete: Naïve approach

Insert and Delete: Naïve approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Insert and Delete: Naïve approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Delete : If α(T) = 1/2, and Delete occurs, size(new T) = 1/2 size(T)

Insert and Delete: Naïve approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Delete : If α(T) = 1/2, and Delete occurs, size(new T) = 1/2 size(T)

Approach ensures $\alpha(T) \geq 1/2$

Insert and Delete: Naïve approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Delete : If α(T) = 1/2, and Delete occurs, size(new T) = 1/2 size(T)

Approach ensures α(T) $\geq$ 1/2

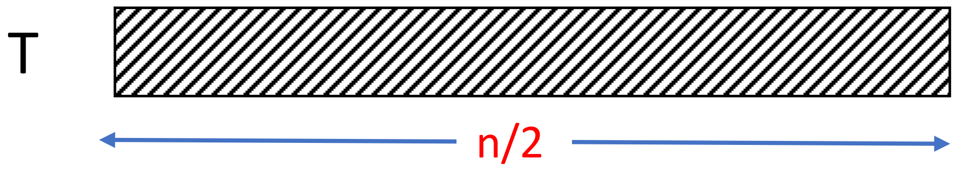
σ : Arbitrary sequence of n Inserts and Deletes, starting from empty table T of size 1

What is the amortized cost per operation in σ ?

Naïve approach : Bad sequence σ of $n = 2^k$ operations

Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $n/2$ Inserts,

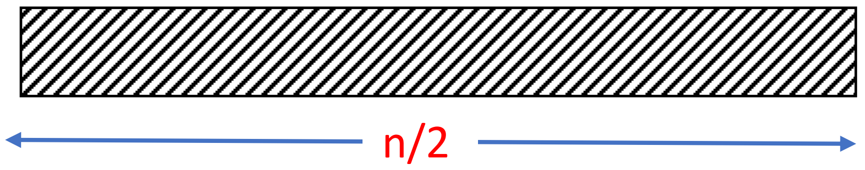


Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}$

Cost : $\geq n/2$

T

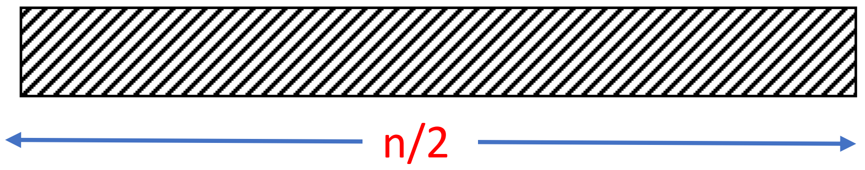


Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : n/2 Inserts, Insert,

Cost : $\geq n/2$

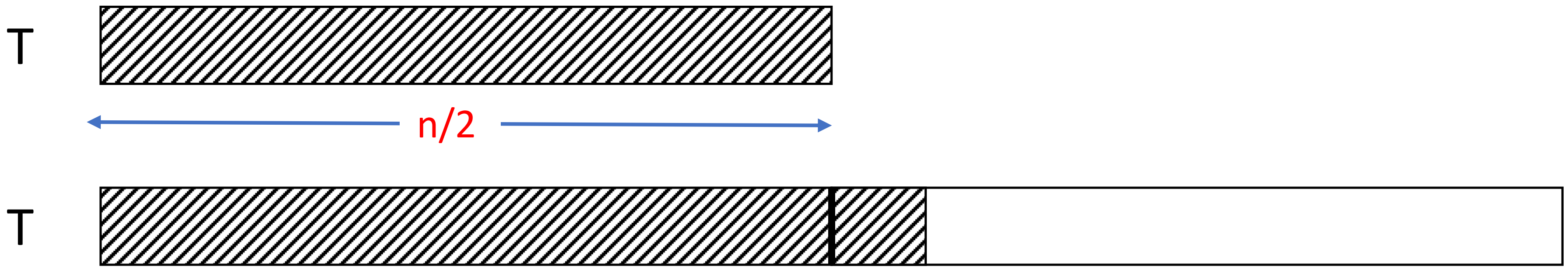
T



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts}}_{\geq n/2}$, Insert,

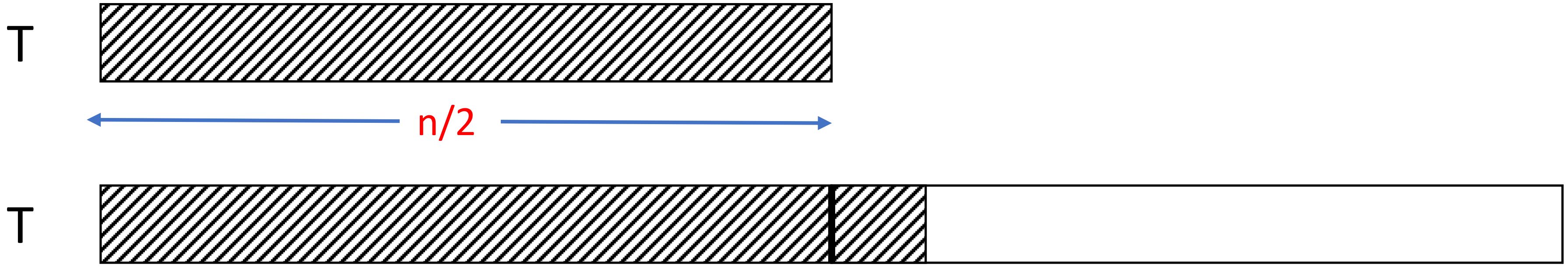
Cost : $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts,}}_{\geq n/2} \ \underbrace{\text{Insert,}}_{\geq n/2}$$

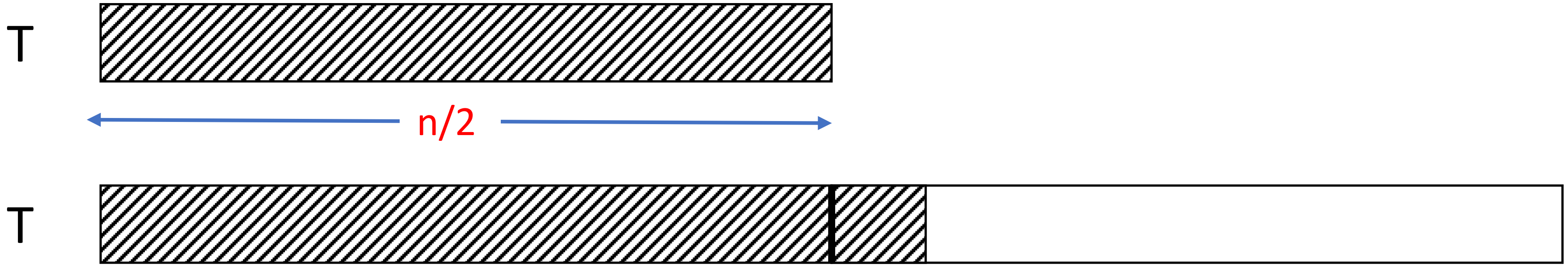
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts},}_{\geq n/2}$ $\underbrace{\text{Insert},}_{\geq n/2}$ Delete,

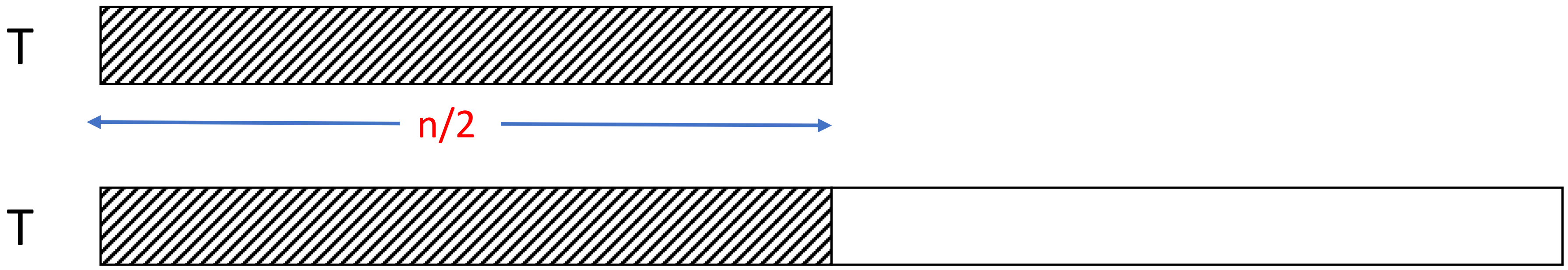
Cost : $\geq n/2$ $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts}}_{\geq n/2}$, $\underbrace{\text{Insert}}_{\geq n/2}$, Delete,

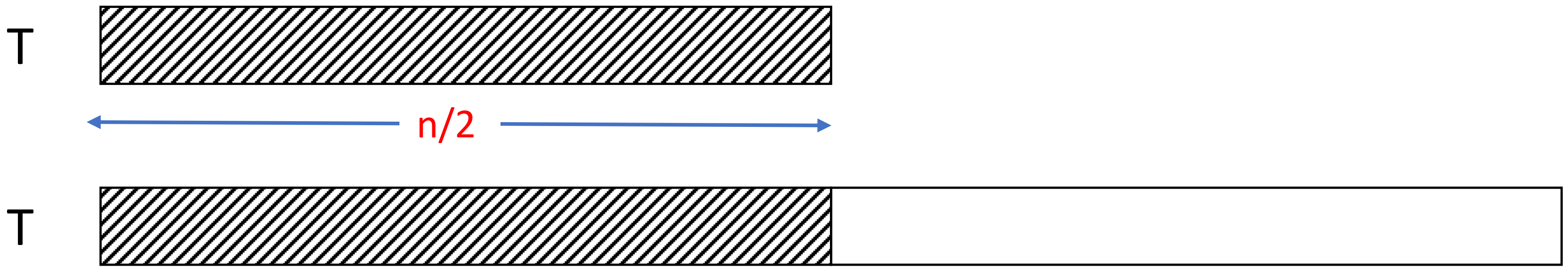
Cost : $\geq n/2$ $\quad$ $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts},}_{\geq n/2}$ $\underbrace{\text{Insert},}_{\geq n/2}$ Delete, Delete,

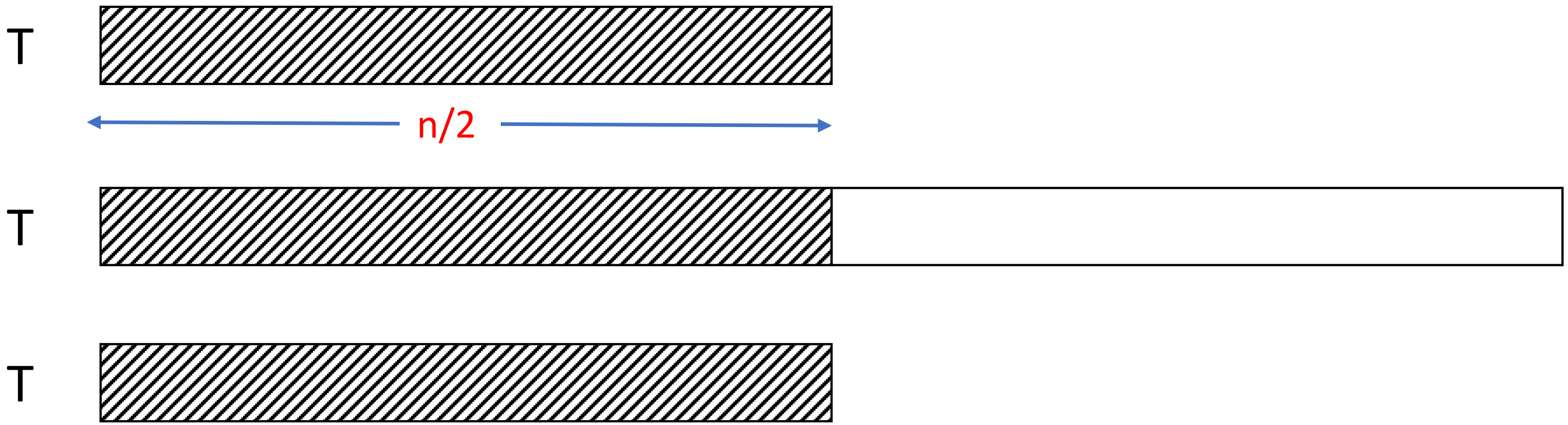
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}_{\geq n/2}$ $\underbrace{\text{Insert}}_{\geq n/2}$, Delete, Delete,

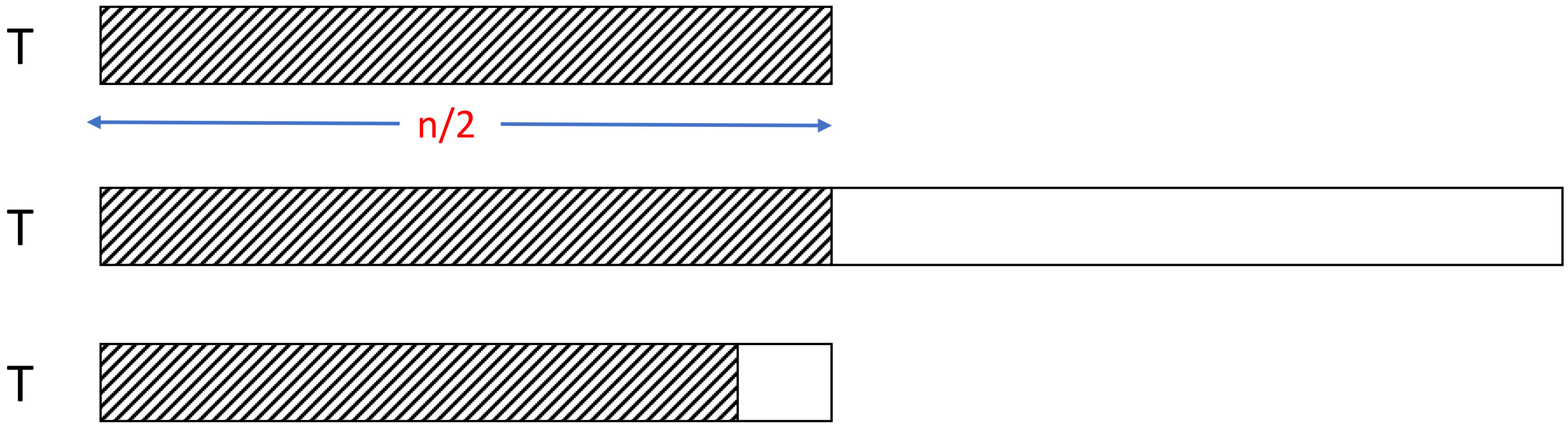
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}_{\geq n/2}$ $\underbrace{\text{Insert}}_{\geq n/2}$, Delete, Delete,

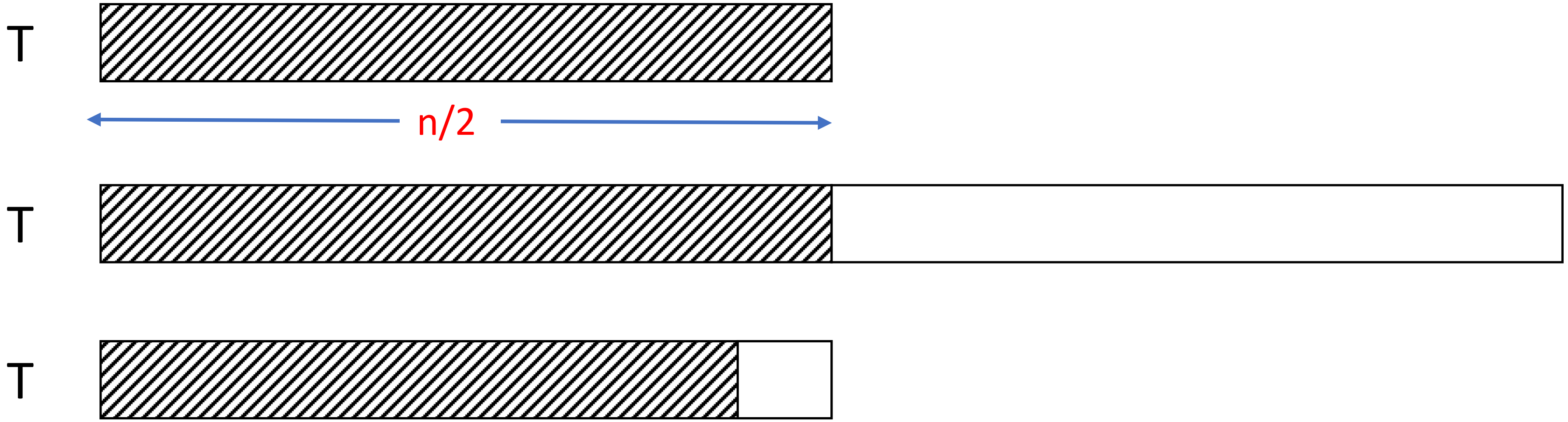
Cost : $\geq n/2$ $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}_{\geq n/2}$ $\underbrace{\text{Insert,}}_{\geq n/2}$ $\underbrace{\text{Delete, Delete,}}_{\geq n/2}$

Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

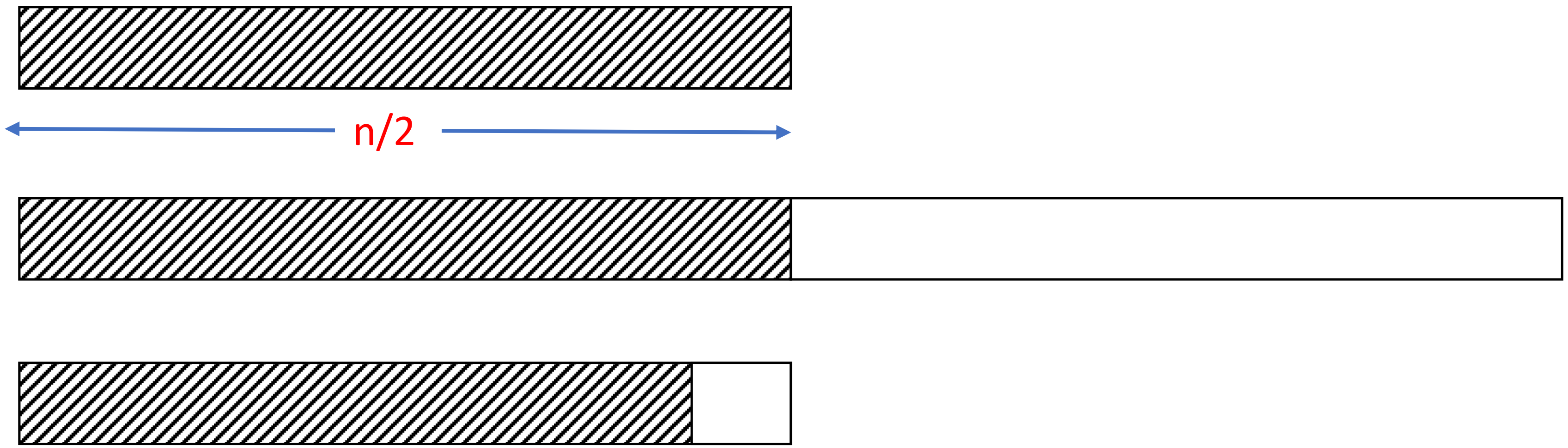
σ : $\underbrace{n/2 \text{ Inserts},}_{\geq n/2}$ $\underbrace{\text{Insert},}_{\geq n/2}$ $\underbrace{\text{Delete, Delete},}_{\geq n/2}$ Insert, Insert,

Cost : $\geq n/2$ $\geq n/2$ $\geq n/2$

T

T

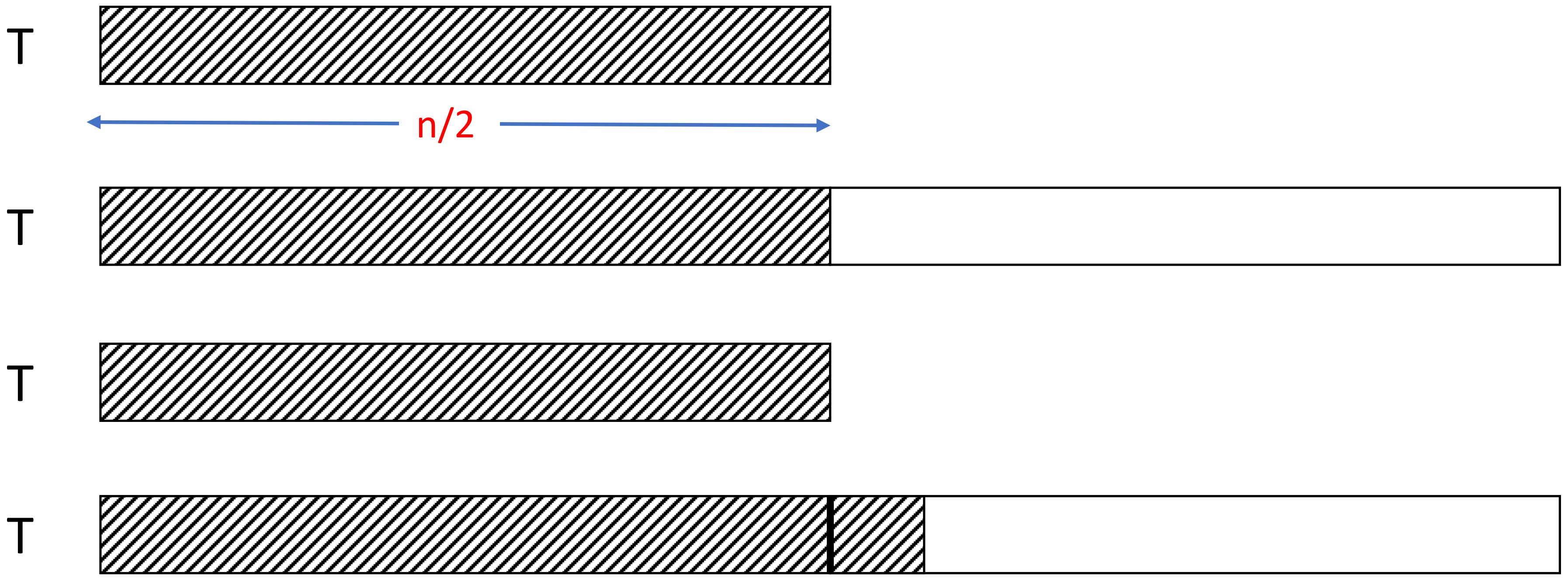
T



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}_{\geq n/2}$ $\underbrace{\text{Insert,}}_{\geq n/2}$ $\underbrace{\text{Delete, Delete,}}_{\geq n/2}$ Insert, Insert,

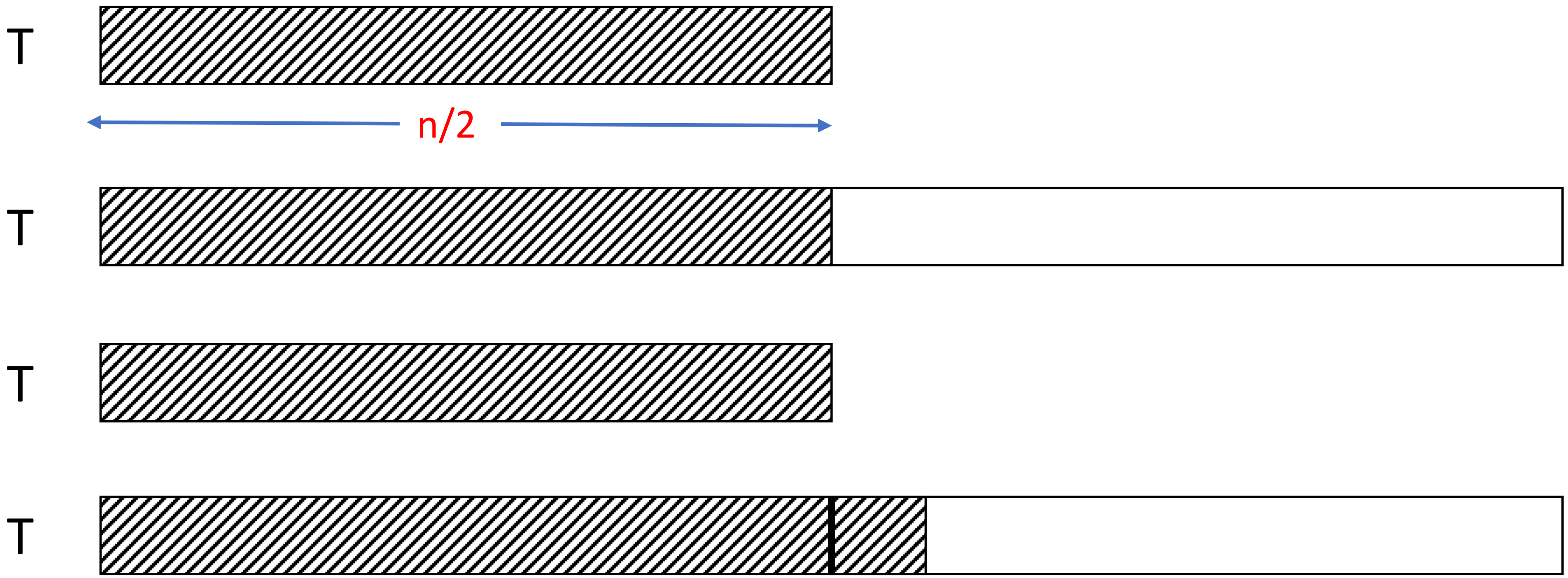
Cost : $\geq n/2$ $\geq n/2$ $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts},}_{\geq n/2} \; \underbrace{\text{Insert},}_{\geq n/2} \; \underbrace{\text{Delete, Delete},}_{\geq n/2} \; \underbrace{\text{Insert, Insert},}_{\geq n/2}$$

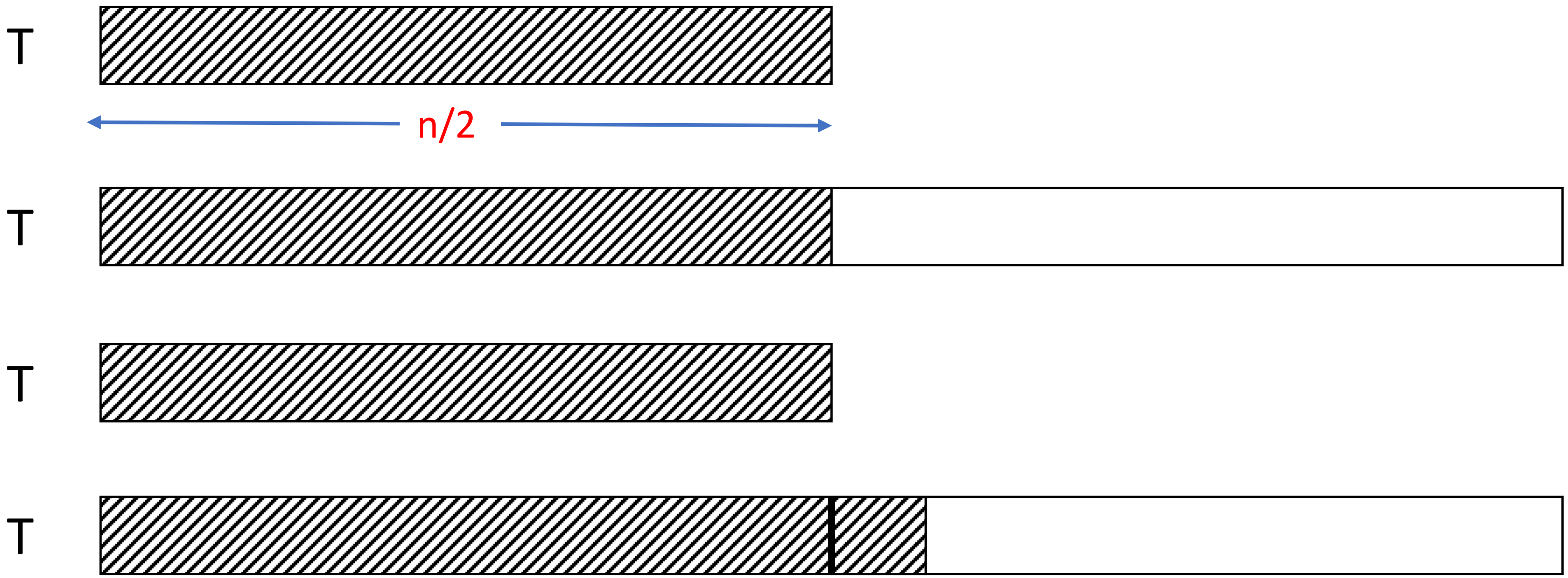
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts,}}_{\geq n/2} \; \underbrace{\text{Insert,}}_{\geq n/2} \; \underbrace{\text{Delete, Delete,}}_{\geq n/2} \; \underbrace{\text{Insert, Insert,}}_{\geq n/2} \; \text{Delete,}$$

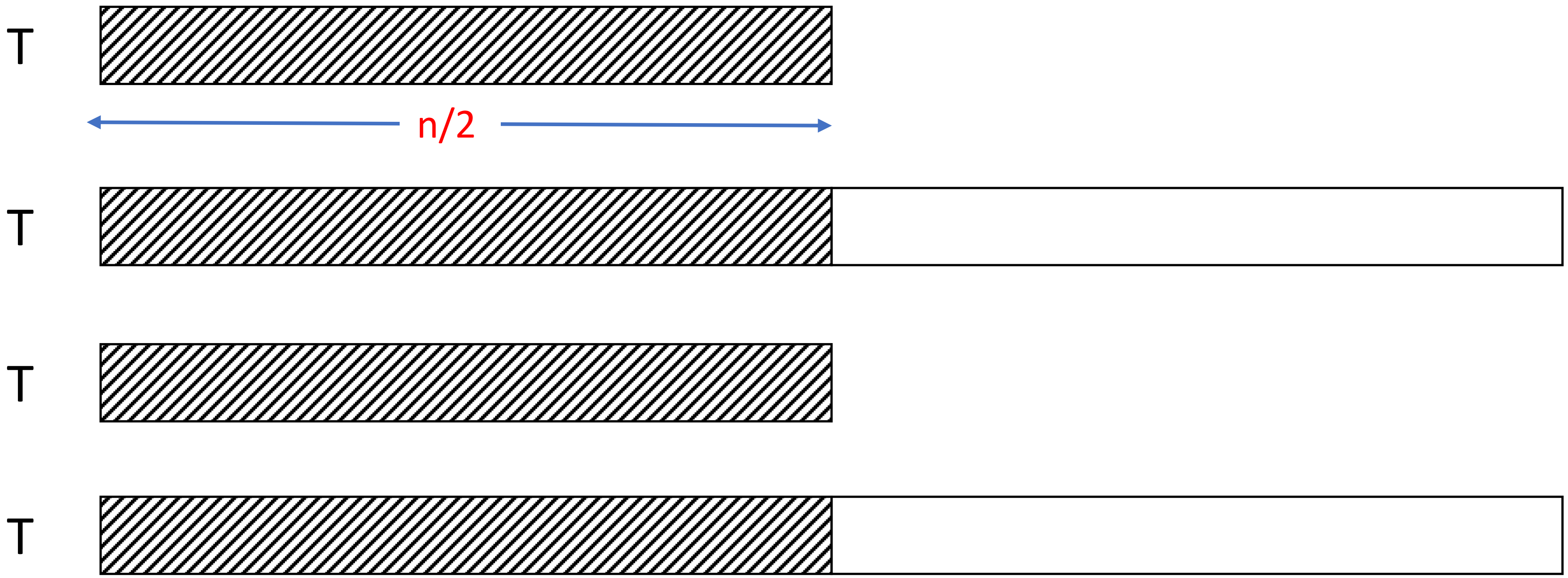
Cost : $\geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}$ $\underbrace{\text{Insert,}}$ $\underbrace{\text{Delete, Delete,}}$ $\underbrace{\text{Insert, Insert,}}$ Delete,

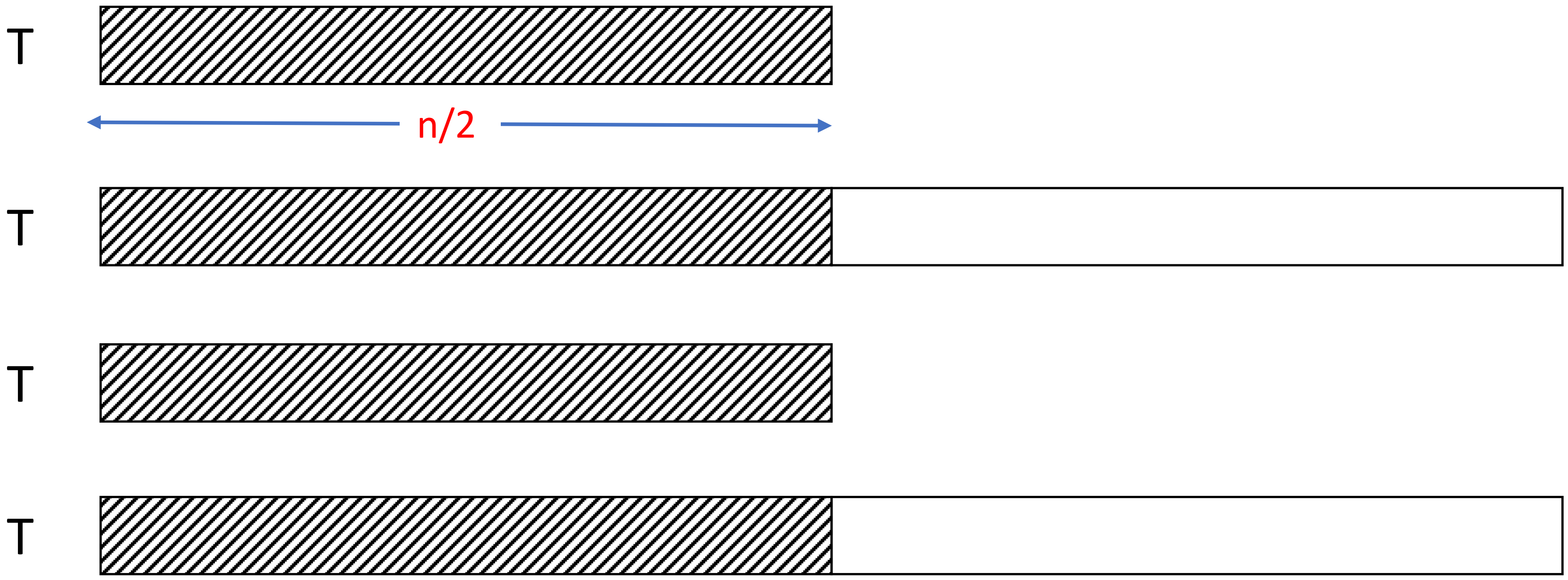
Cost : $\geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $n/2$ Inserts, Insert, Delete, Delete, Insert, Insert, Delete, Delete,

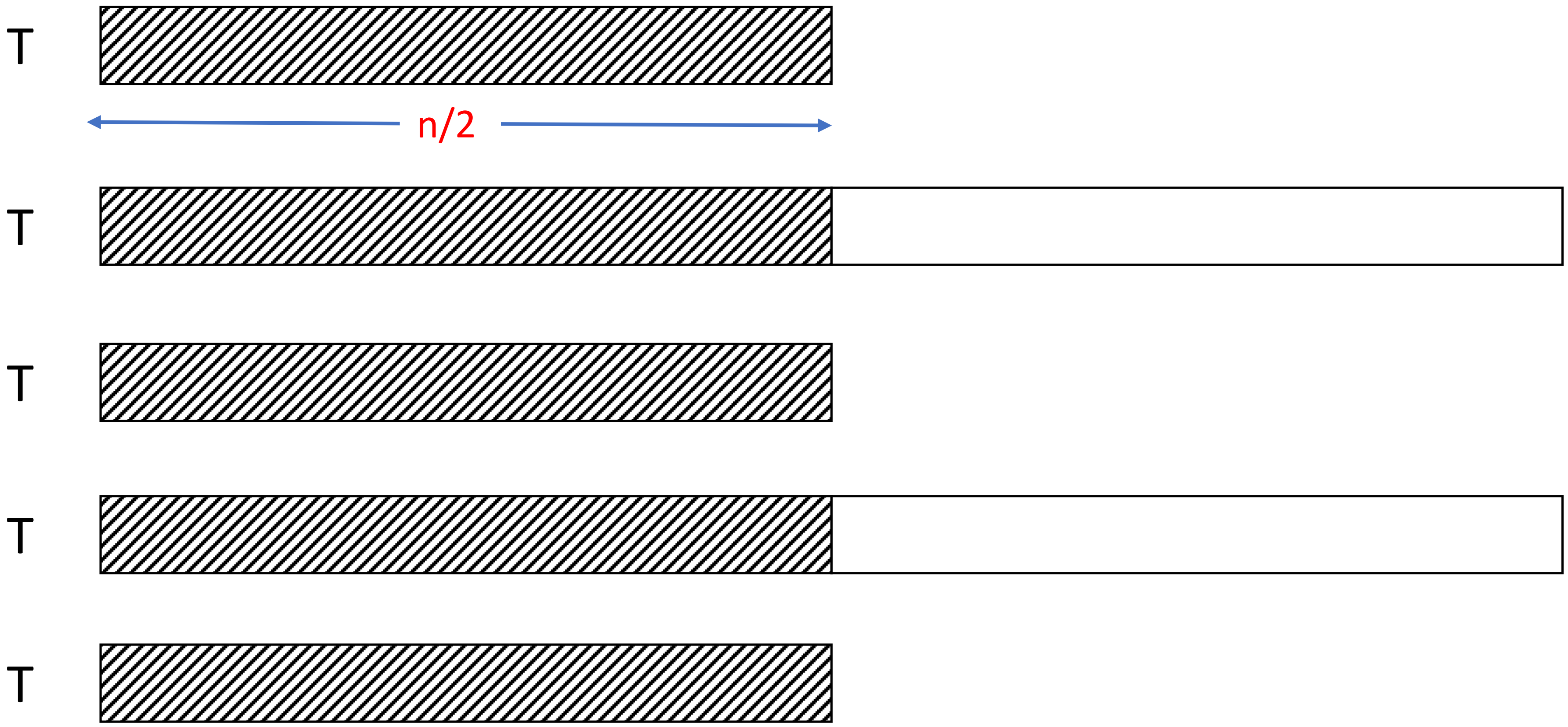
Cost : $\geq n/2$ $\geq n/2$ $\geq n/2$ $\geq n/2$



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts,}}_{\geq n/2} \ \underbrace{\text{Insert,}}_{\geq n/2} \ \underbrace{\text{Delete, Delete,}}_{\geq n/2} \ \underbrace{\text{Insert, Insert,}}_{\geq n/2} \ \text{Delete, Delete,}$$

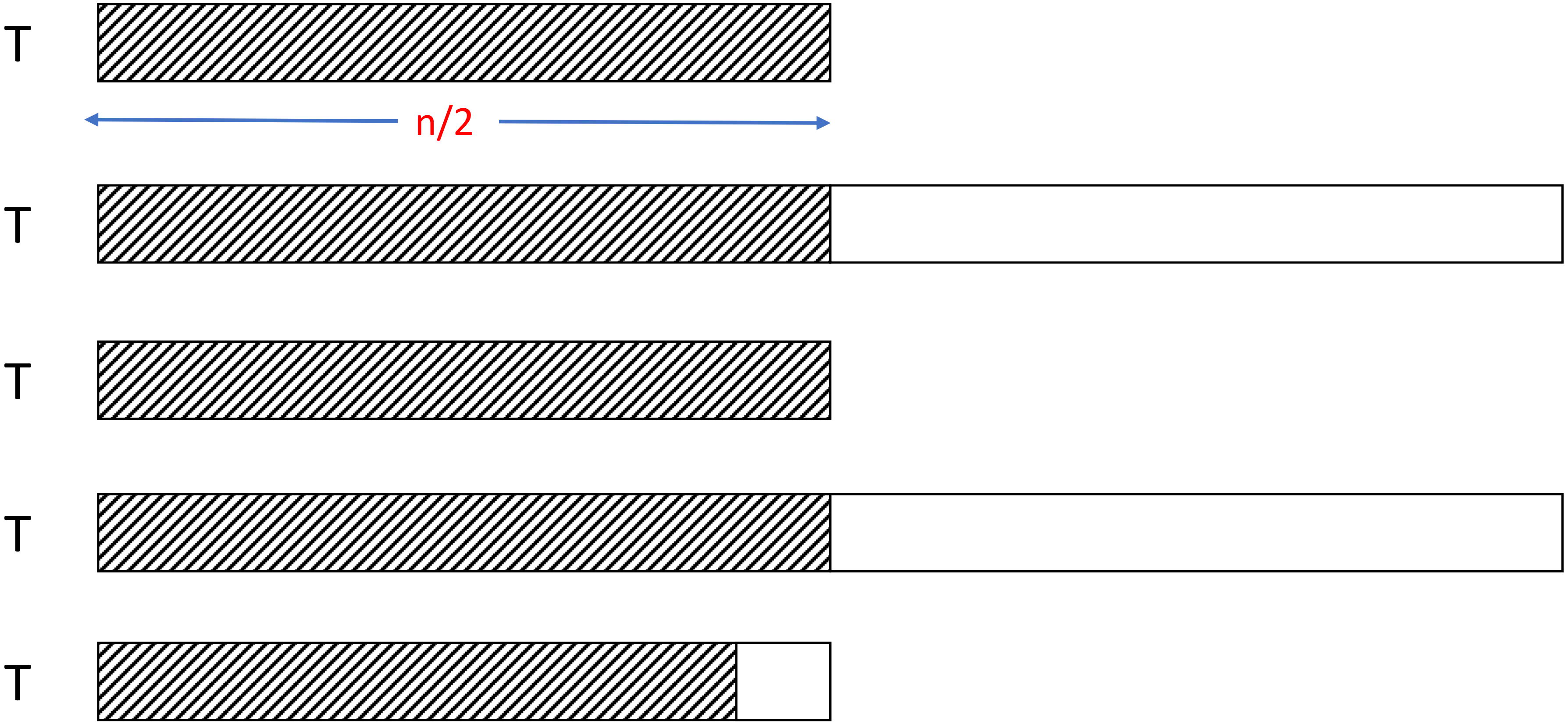
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts,}}_{\geq n/2}$ $\underbrace{\text{Insert,}}_{\geq n/2}$ $\underbrace{\text{Delete, Delete,}}_{\geq n/2}$ $\underbrace{\text{Insert, Insert,}}_{\geq n/2}$ Delete, Delete,

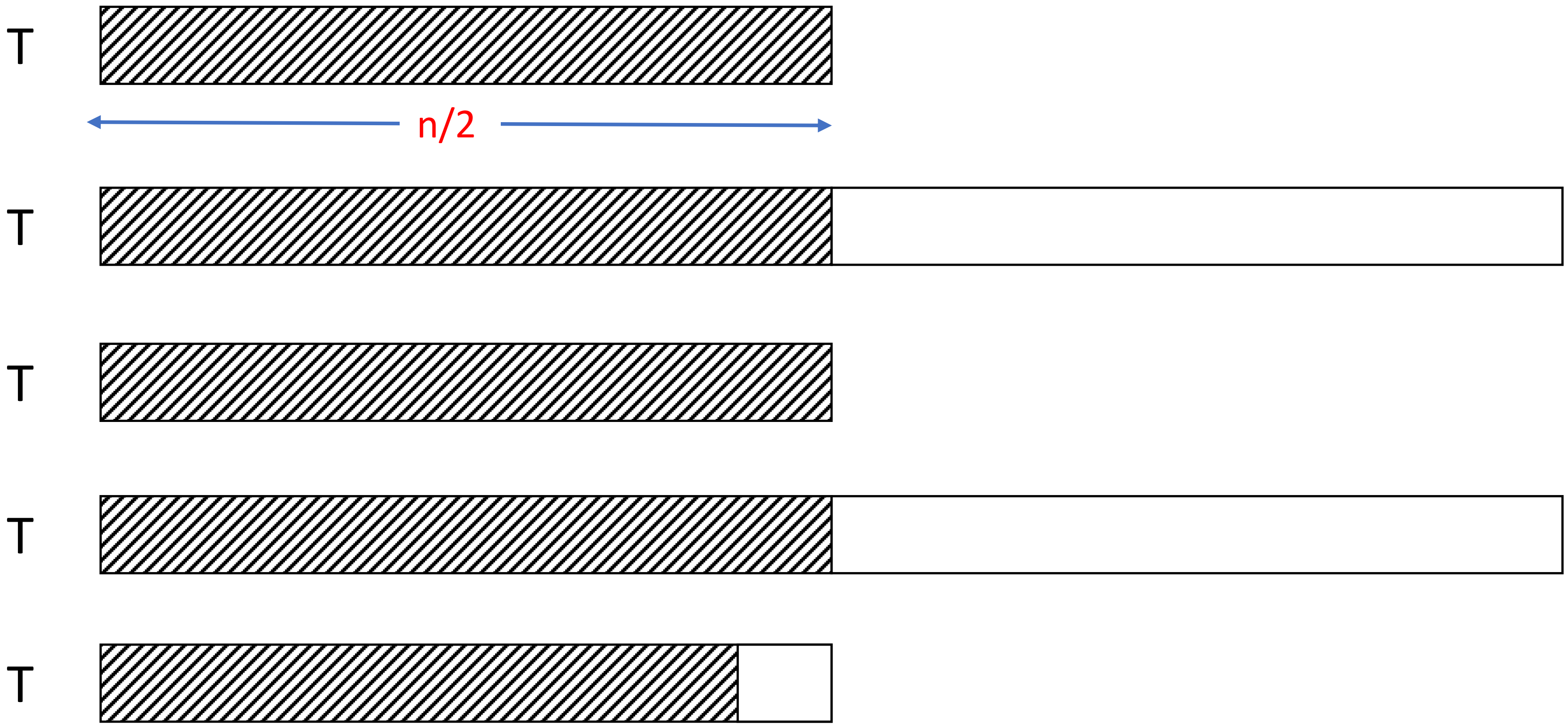
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts,}}_{\geq n/2}\ \underbrace{\text{Insert,}}_{\geq n/2}\ \underbrace{\text{Delete, Delete,}}_{\geq n/2}\ \underbrace{\text{Insert, Insert,}}_{\geq n/2}\ \underbrace{\text{Delete, Delete,}}_{\geq n/2}$$

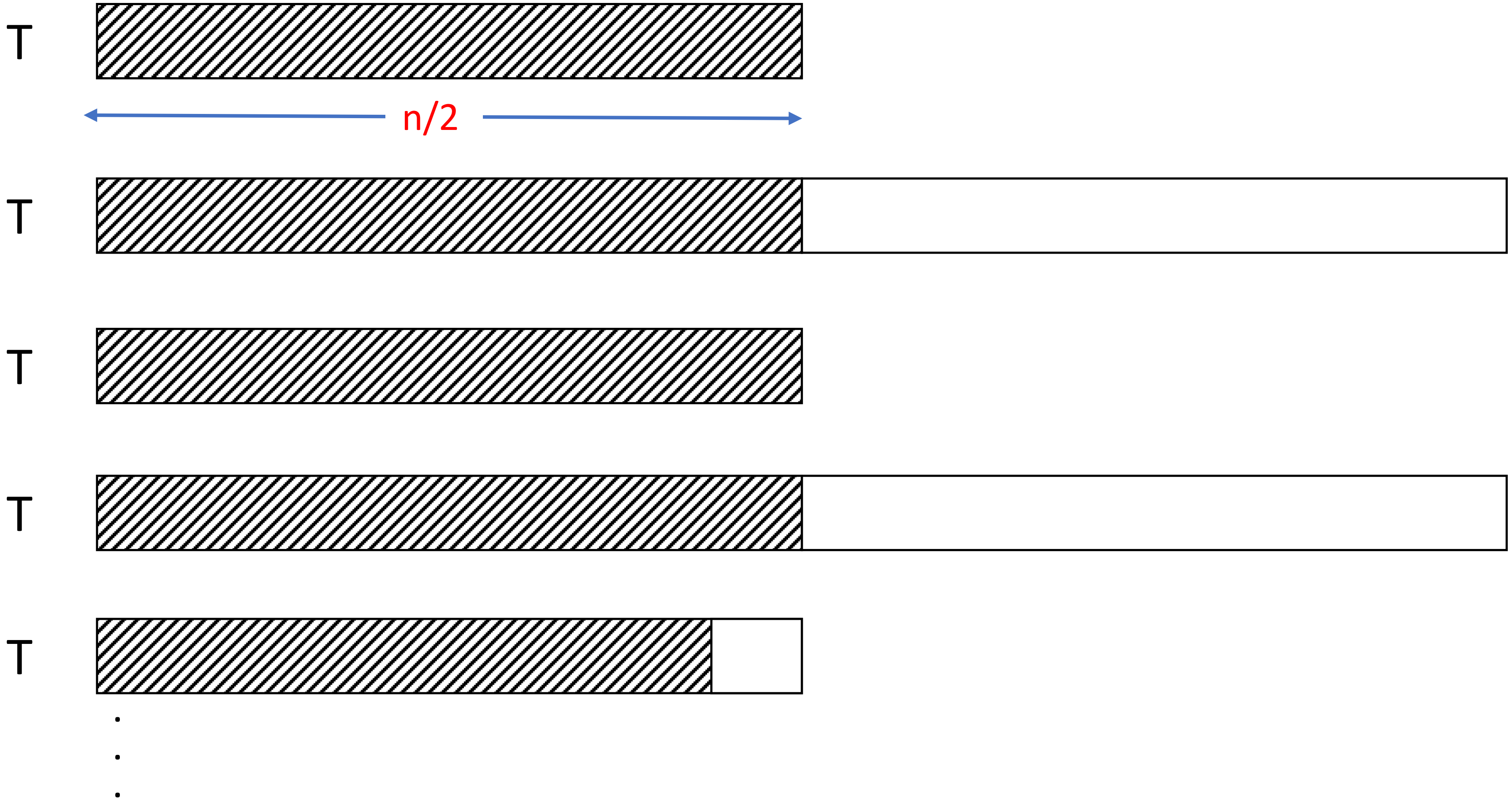
Cost :



Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $\underbrace{n/2 \text{ Inserts},}_{\geq n/2}\ \underbrace{\text{Insert},}_{\geq n/2}\ \underbrace{\text{Delete, Delete},}_{\geq n/2}\ \underbrace{\text{Insert, Insert},}_{\geq n/2}\ \underbrace{\text{Delete, Delete},}_{\geq n/2} \ldots$

Cost : ≥ n/2 ≥ n/2 ≥ n/2 ≥ n/2 ≥ n/2



Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts}}_{\geq n/2}, \underbrace{\text{Insert}}_{\geq n/2}, \underbrace{\text{Delete, Delete}}_{\geq n/2}, \underbrace{\text{Insert, Insert}}_{\geq n/2}, \underbrace{\text{Delete, Delete}}_{\geq n/2}, \ldots$$

Cost : $\geq n/2$ $\geq n/2$ $\geq n/2$ $\geq n/2$ $\geq n/2$

Total cost $\geq$ (n/4)(n/2)

Naïve approach : Bad sequence σ of $n = 2^k$ operations

σ : $n/2$ Inserts, Insert, Delete, Delete, Insert, Insert, Delete, Delete, …

Cost : $\geq n/2$ $\quad\geq n/2$ $\quad\geq n/2$ $\quad\geq n/2$ $\quad\geq n/2$

Total cost $\geq (n/4)(n/2)$

Total cost is $\Omega(n^2)$

Naïve approach : Bad sequence σ of $n = 2^k$ operations

$$\sigma : \underbrace{n/2 \text{ Inserts,}}_{\geq\, n/2}\ \underbrace{\text{Insert,}}_{\geq\, n/2}\ \underbrace{\text{Delete, Delete,}}_{\geq\, n/2}\ \underbrace{\text{Insert, Insert,}}_{\geq\, n/2}\ \underbrace{\text{Delete, Delete,}}_{\geq\, n/2}\ \ldots$$

Cost : $\geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$ $\quad \geq n/2$

Total cost $\geq (n/4)(n/2)$

Total cost is $\Omega(n^2)$

Amortized cost per operation is $\Omega(n)$

Insert and Delete: Good Approach

Insert and Delete: Good Approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Insert and Delete: Good Approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Delete : If α(T) = <u>1/4</u>, and Delete occurs, size(new T) = 1/2 size(T)

Insert and Delete: Good Approach

Insert : If $\alpha(T) = 1$, and Insert occurs, size(new T) = 2 size(T)

Delete : If $\alpha(T) = \underline{1/4}$, and Delete occurs, size(new T) = $1/2$ size(T)

Approach ensures $\alpha(T) \geq 1/4$

Insert and Delete: Good Approach

Insert : If α(T) = 1, and Insert occurs, size(new T) = 2 size(T)

Delete : If α(T) = 1/4, and Delete occurs, size(new T) = 1/2 size(T)

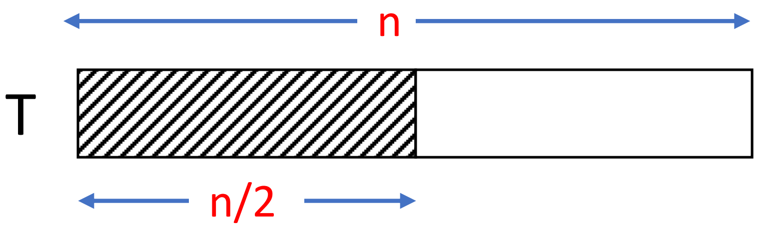
Approach ensures $\alpha(T) \geq 1/4$

σ : Arbitrary sequence of n Inserts and Deletes, starting from empty table T of size 1

What is the amortized cost per operation in σ ?

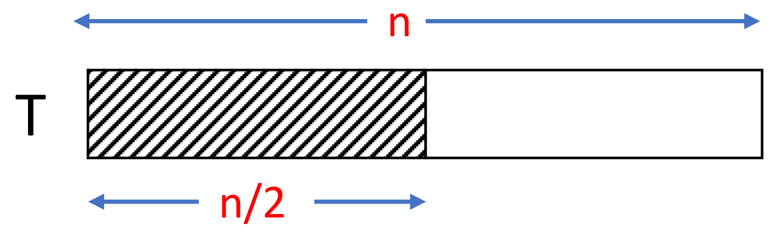
Amortized Analysis

Say a new T is just created,
(Total credit : $0)



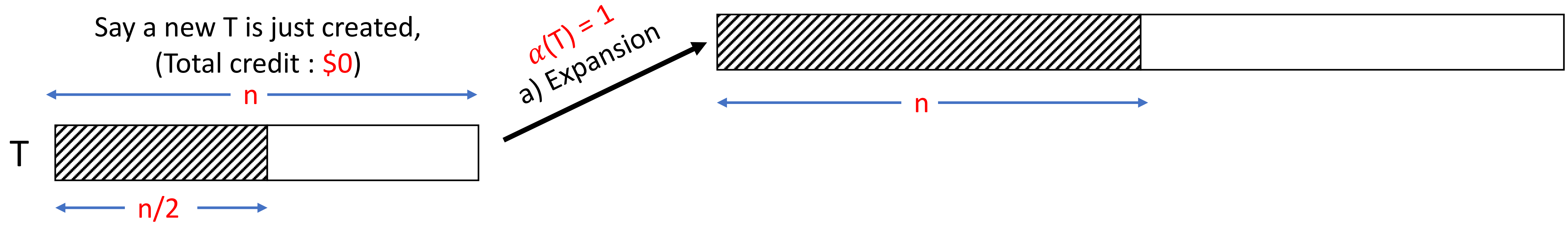
Amortized Analysis

Say a new T is just created,
(Total credit : $0)



A sequence of Inserts, Deletes applied to T may cause:

Amortized Analysis

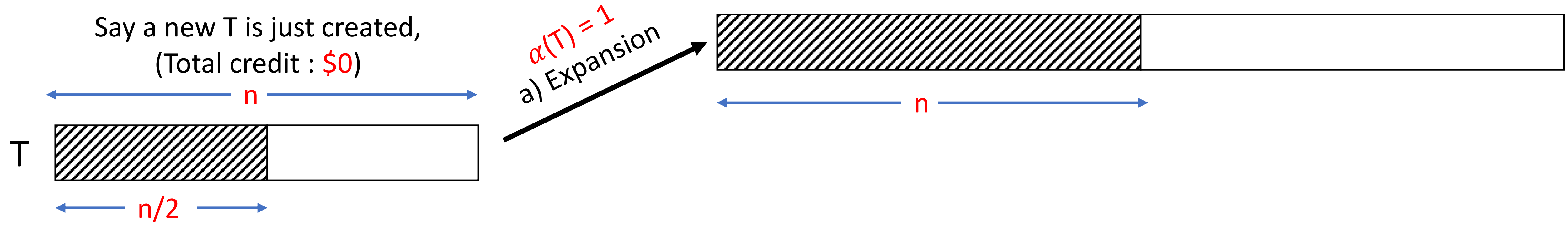


A sequence of Inserts, Deletes applied to T may cause:

a) Expansion.

Amortized Analysis

Say a new T is just created,
(Total credit : $0)

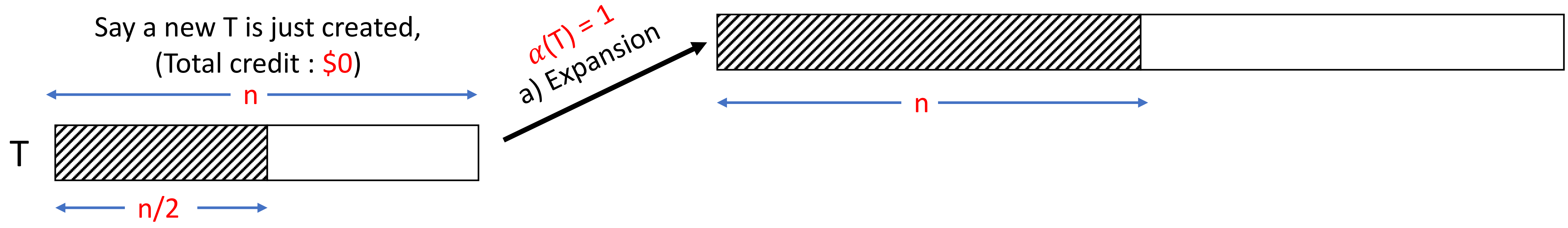


A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq n/2$ Inserts.

Amortized Analysis

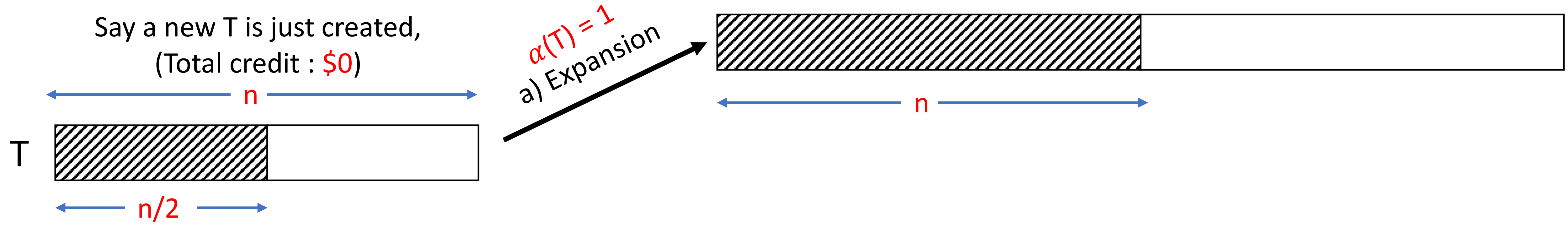


A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

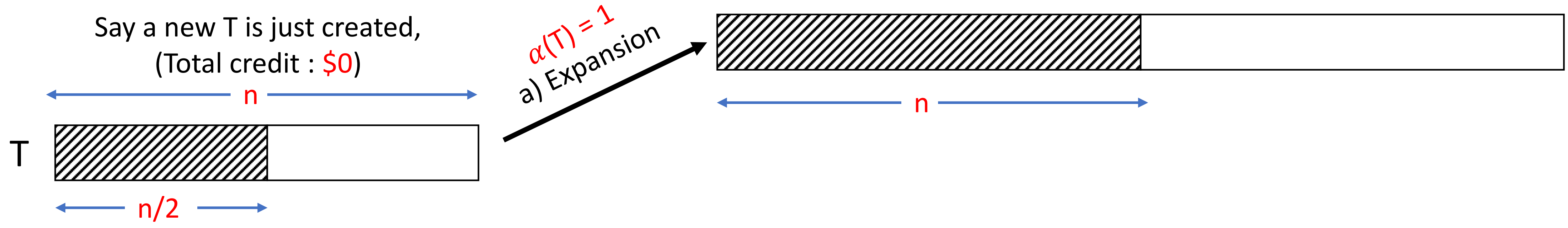
- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

Charging Scheme:

Charge each Insert $3

$1 actual cost + $2 credit for future expansion

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

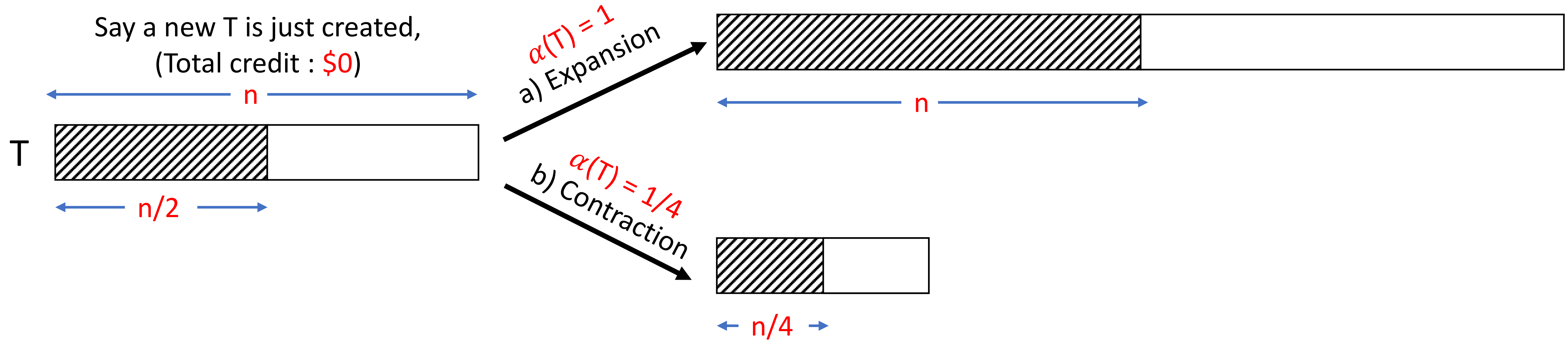
Charging Scheme:

Charge each Insert $3

$1 actual cost + $2 credit for future expansion

n/2 Inserts ⇒ (n/2) ($2) = $n credit,
which covers cost of table expansion

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains ≥ n/2 Inserts.
- Cost of copying items into new T : n

a) Contraction.

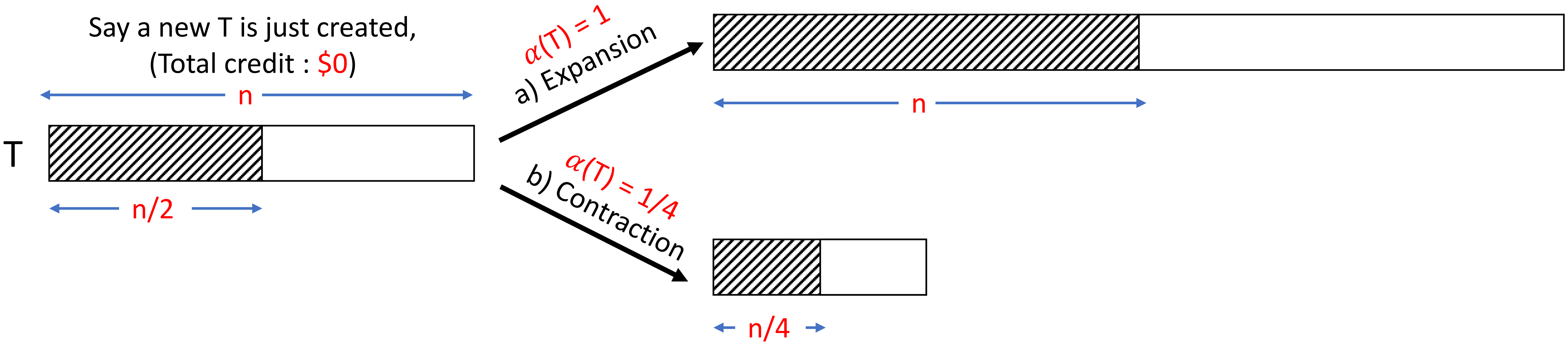
Charging Scheme:

Charge each Insert \$3

\$1 actual cost + \$2 credit for future expansion

n/2 Inserts ⇒ (n/2) (\$2) = \$n credit,
which covers cost of table expansion

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

a) Contraction. In this case:

- The seq contains $\geq$ n/4 Deletes.

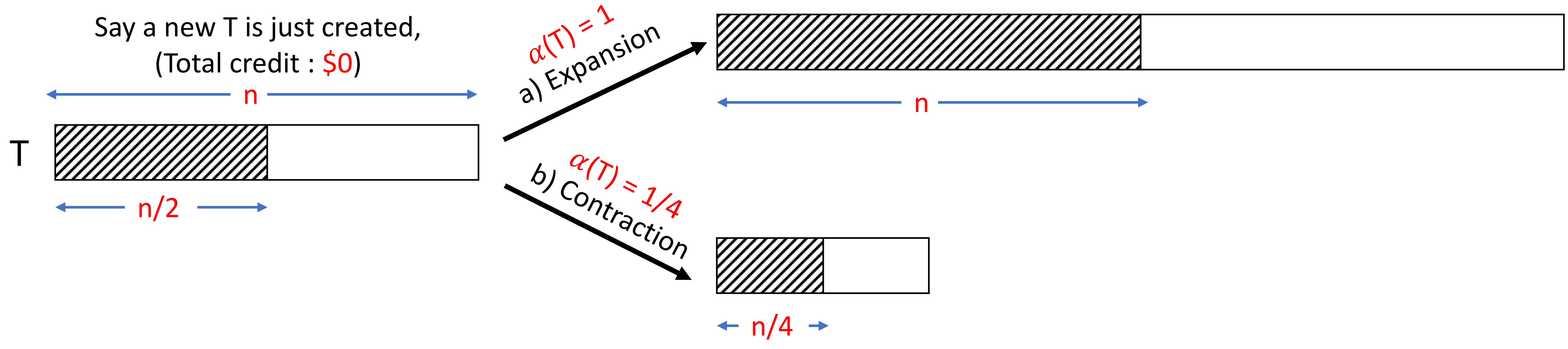
Charging Scheme:

Charge each Insert $3

$1 actual cost + $2 credit for future expansion

n/2 Inserts $\Rightarrow$ (n/2) ($2) = $n credit,
which covers cost of table expansion

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

a) Contraction. In this case:

- The seq contains $\geq$ n/4 Deletes.
- Cost of copying items into new T : n/4

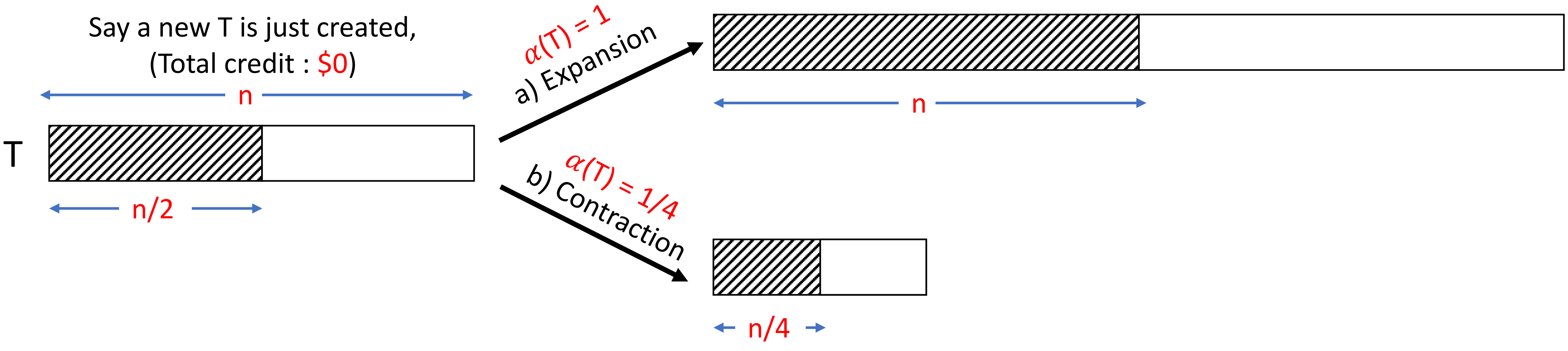
Charging Scheme:

Charge each Insert $3

$1 actual cost + $2 credit for future expansion

n/2 Inserts $\Rightarrow$ (n/2) ($2) = $n credit,
which covers cost of table expansion

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains ≥ n/2 Inserts.
- Cost of copying items into new T : n

a) Contraction. In this case:

- The seq contains ≥ n/4 Deletes.
- Cost of copying items into new T : n/4

Charging Scheme:

Charge each Insert $3

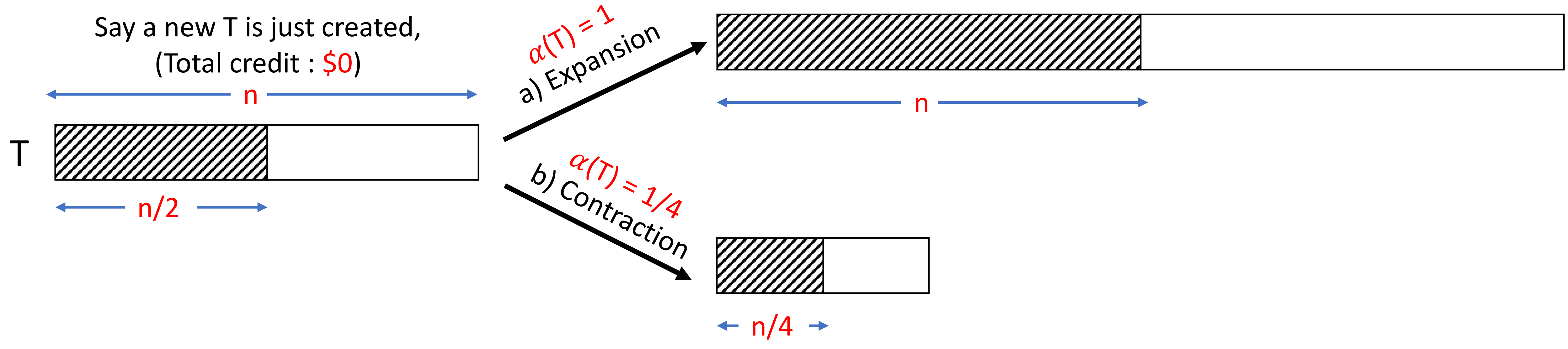
$1 actual cost + $2 credit for future expansion

n/2 Inserts ⇒ (n/2) ($2) = $n credit,
which covers cost of table expansion

Charge each Delete $2

$1 actual cost + $1 credit for future contraction

Amortized Analysis



A sequence of Inserts, Deletes applied to T may cause:

a) Expansion. In this case:

- The seq contains $\geq$ n/2 Inserts.
- Cost of copying items into new T : n

a) Contraction. In this case:

- The seq contains $\geq$ n/4 Deletes.
- Cost of copying items into new T : n/4

Charging Scheme:

Charge each Insert $3

$1 actual cost + $2 credit for future expansion

n/2 Inserts $\Rightarrow$ (n/2) ($2) = $n credit,
which covers cost of table expansion

Charge each Delete $2

$1 actual cost + $1 credit for future contraction

n/4 Deletes $\Rightarrow$ (n/4) ($1) = $n/4 credit,
which covers cost of table contraction