Advancing Environmental Sustainability in Data Centers via Carbon Depreciation Models

The 16th International Green and Sustainable Computing Conference

Shixin Ji*, Zhuoping Yang*, Xingzhen Chen*, Alex K. Jones§, Peipei Zhou*

Brown University*; Syracuse University§

shixin_ji@brown.edu
peipei_zhou@brown.edu
https://peipeizhou-eecs.github.io/

To Begin with: a Second-hand Vehicle

Consider owning a car:

https://www.iseecars.com/car/bmw-7-series/resale-value

To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

https://www.iseecars.com/car/bmw-7-series/resale-value

To Begin with: a Second-hand Vehicle

Consider owning a car:

+

On-time investment

Recurring Cost

https://www.iseecars.com/car/bmw-7-series/resale-value

To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

\+

Recurring Cost

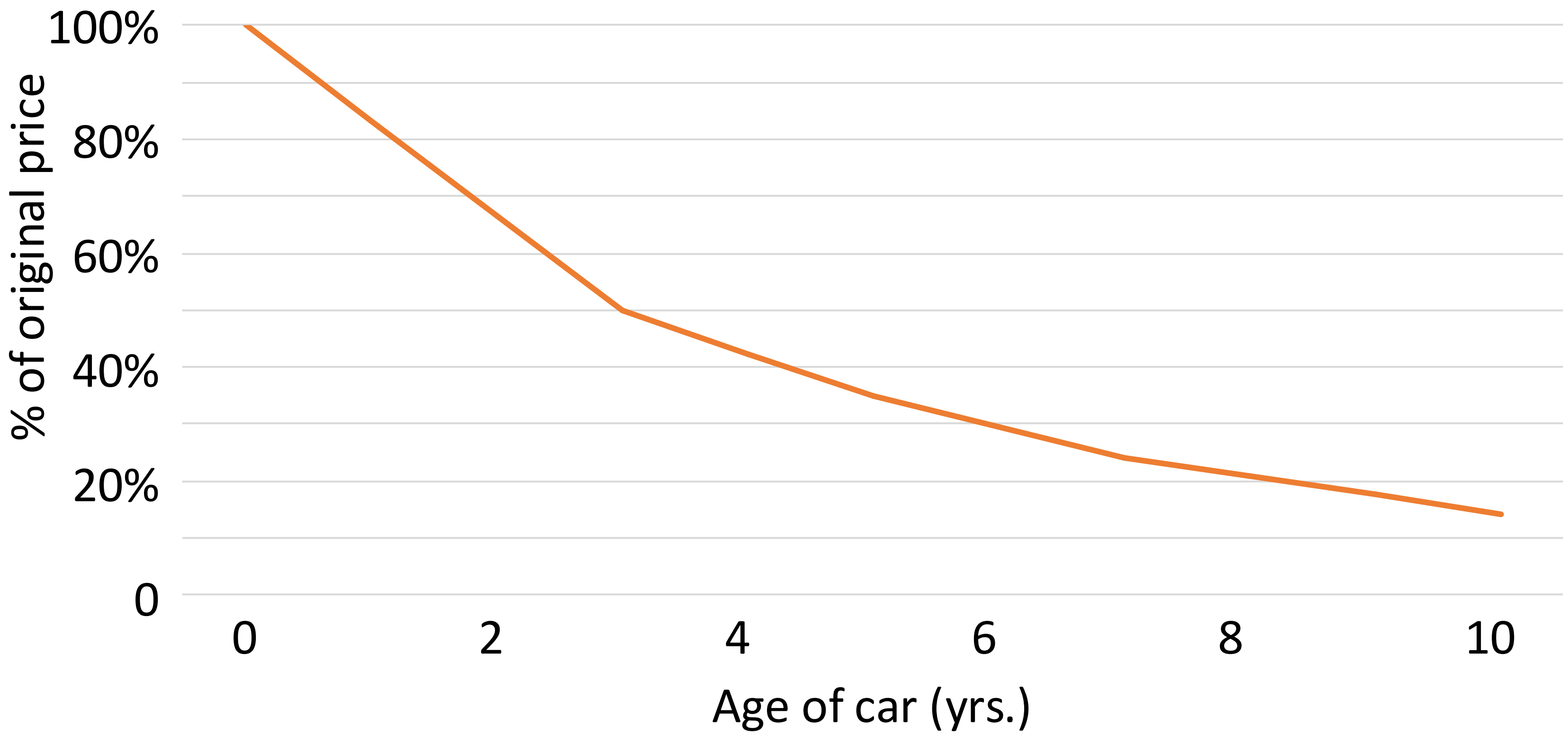
To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

+

Recurring Cost



https://www.iseecars.com/car/bmw-7-series/resale-value

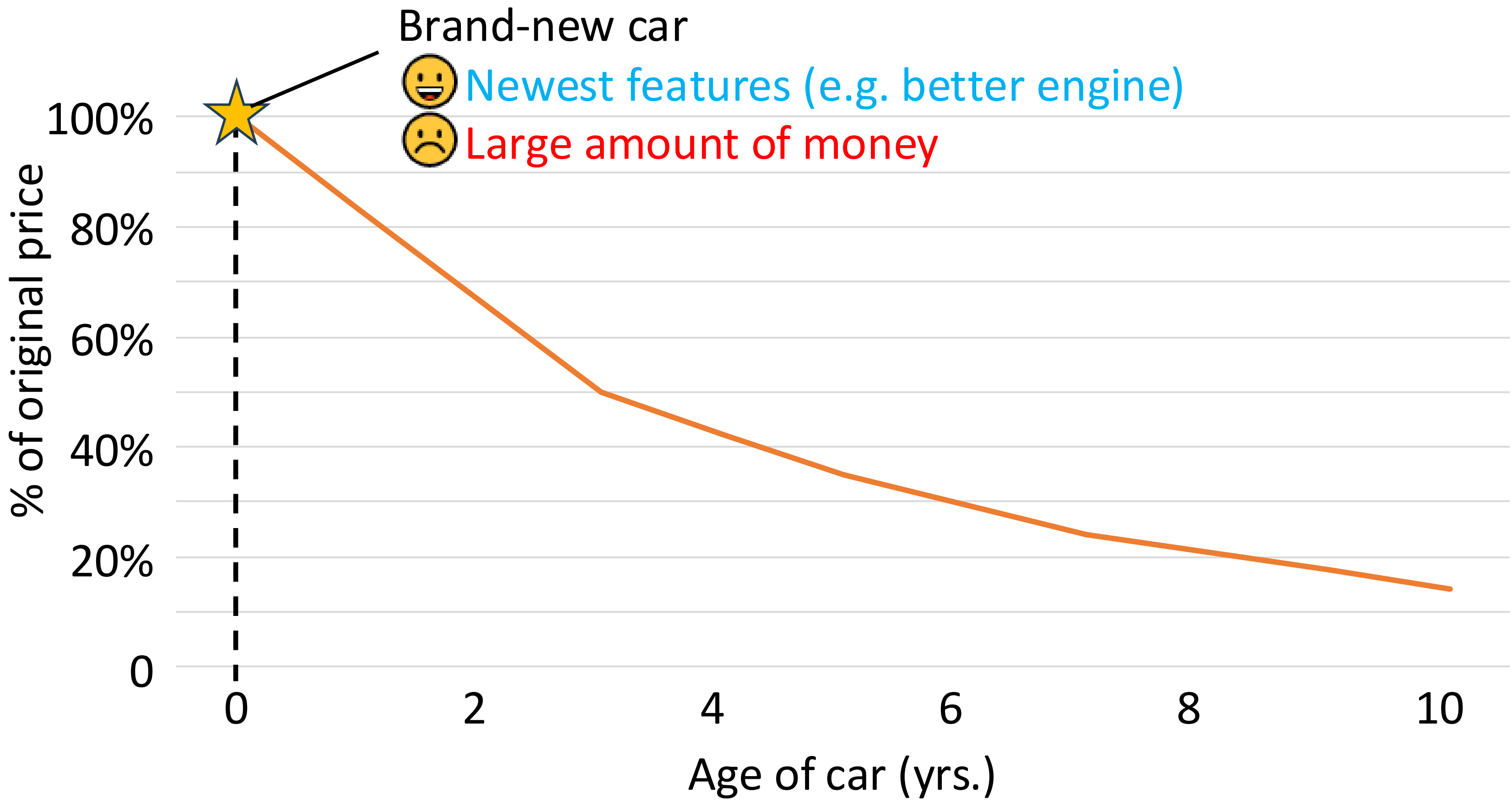
To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

+

Recurring Cost



https://www.iseecars.com/car/bmw-7-series/resale-value

To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

+

Recurring Cost

Brand-new car

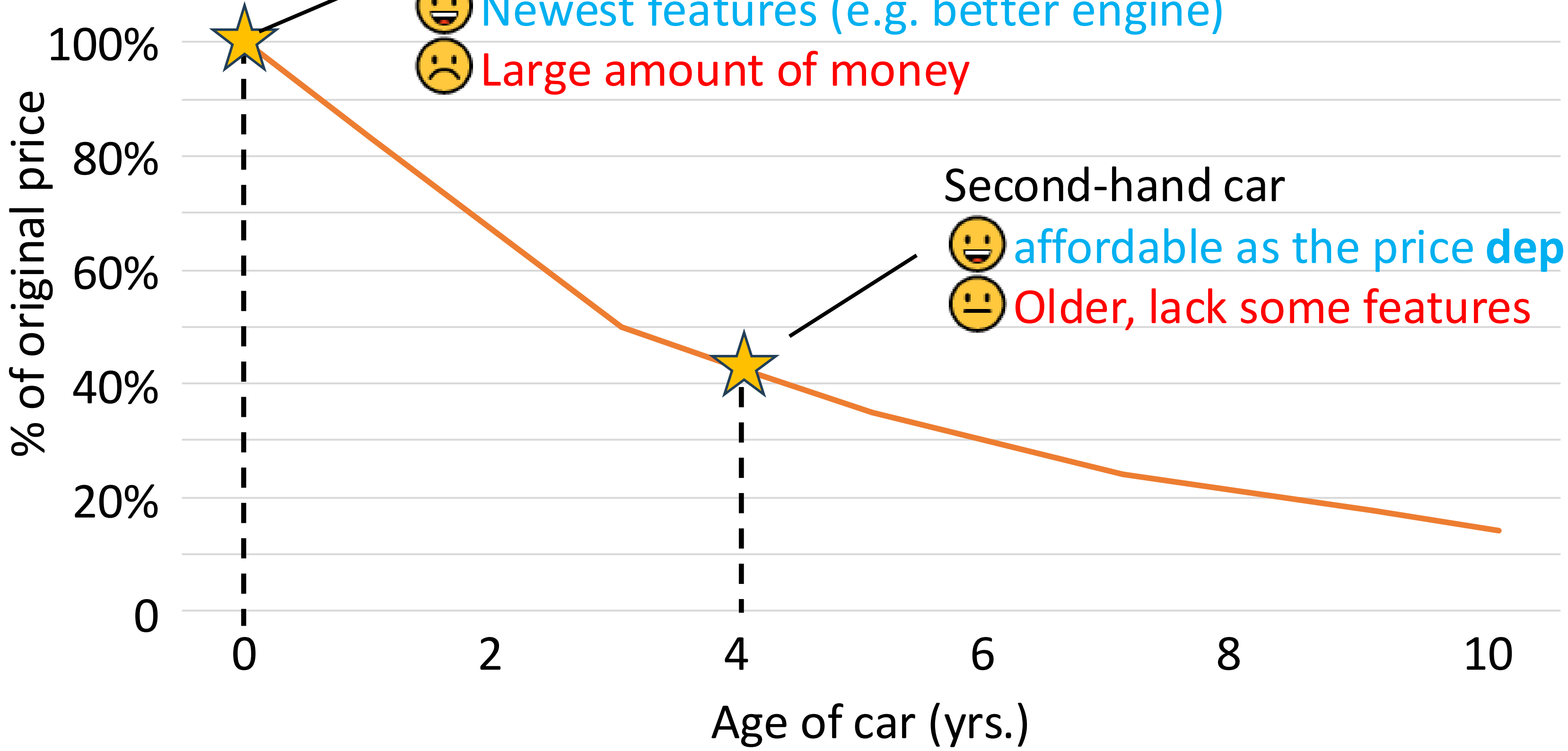
😀 Newest features (e.g. better engine)

☹️ Large amount of money

Second-hand car

😀 affordable as the price **depreciates**

😐 Older, lack some features



https://www.iseecars.com/car/bmw-7-series/resale-value

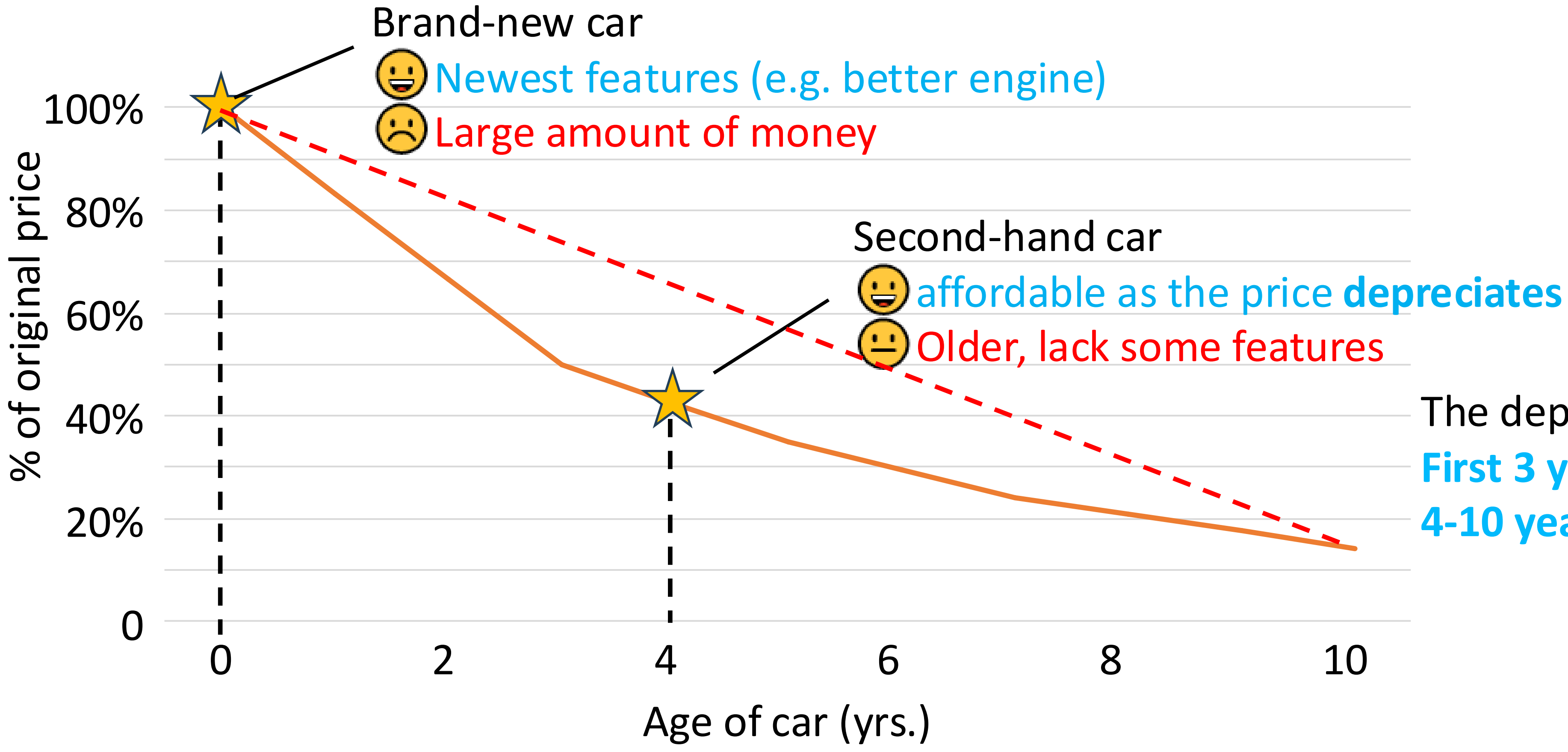
To Begin with: a Second-hand Vehicle

Consider owning a car:

On-time investment

+

Recurring Cost



The depreciation curve is non-linear

First 3 years: 100% → 43%

4-10 years: 43% → 15%

https://www.iseecars.com/car/bmw-7-series/resale-value

To Begin with: a Second-hand Vehicle

Owning a car vs. using Uber: Calculate per-use cost

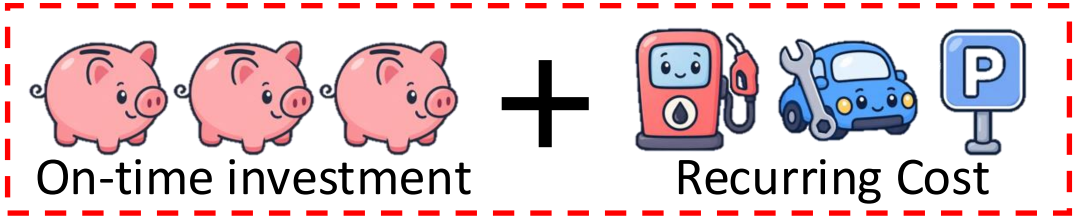
On-time investment

+

Recurring Cost

To Begin with: a Second-hand Vehicle

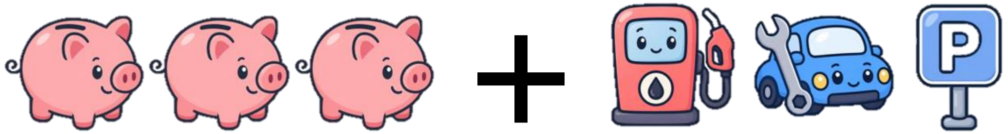
Owning a car vs. using Uber: Calculate per-use cost



Case 1: using the car every day

To Begin with: a Second-hand Vehicle

Owning a car vs. using Uber: Calculate per-use cost

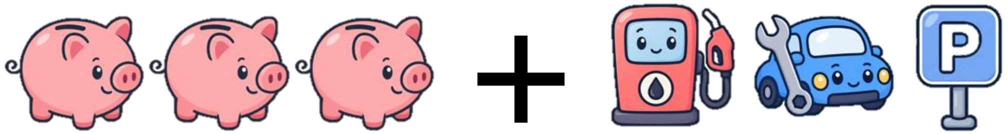


On-time investment + Recurring Cost

Case 1: using the car every day

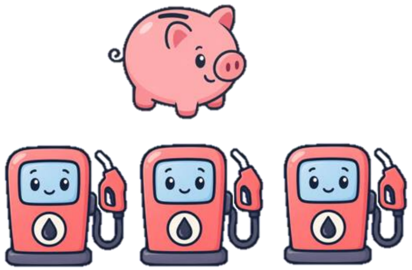
To Begin with: a Second-hand Vehicle

Owning a car vs. using Uber: Calculate per-use cost



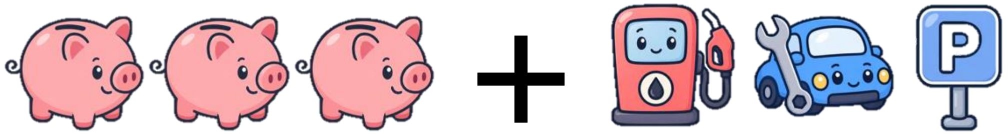
On-time investment + Recurring Cost

Case 1: using the car every day



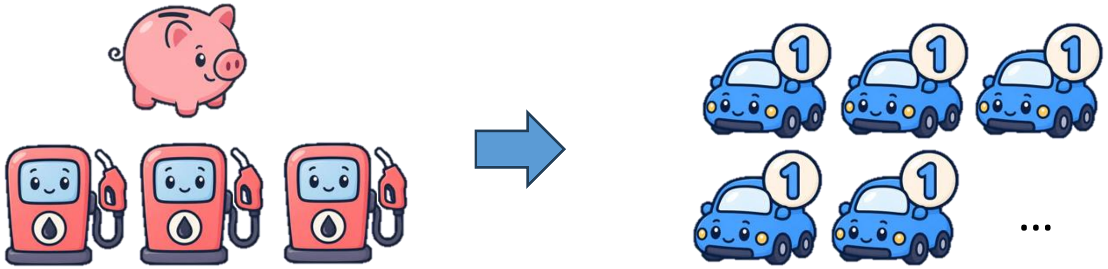
To Begin with: a Second-hand Vehicle

Owning a car vs. using Uber: Calculate per-use cost



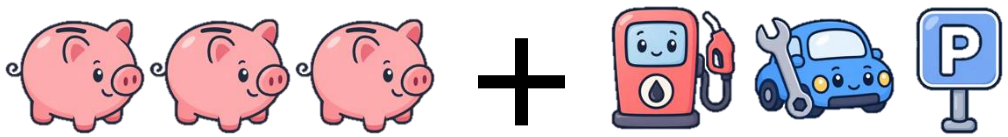
On-time investment + Recurring Cost

Case 1: using the car every day



To Begin with: a Second-hand Vehicle

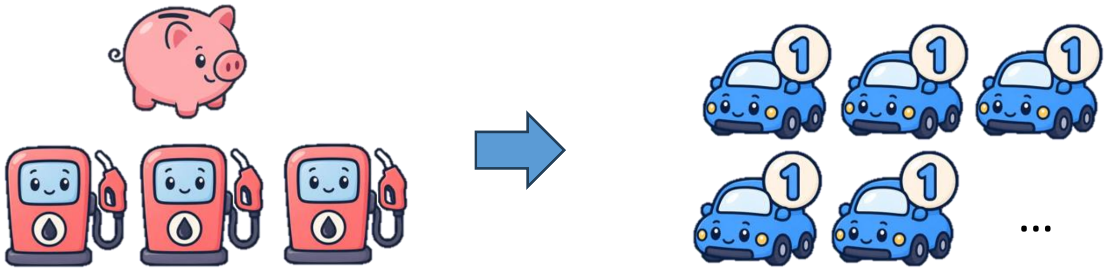
Owning a car vs. using Uber: Calculate per-use cost

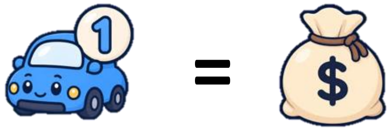


On-time investment

Recurring Cost

Case 1: using the car every day





Lower cost per ride

To Begin with: a Second-hand Vehicle

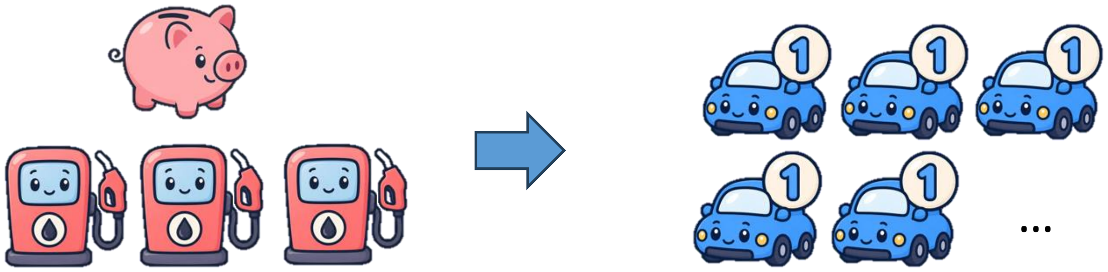
Owning a car vs. using Uber: Calculate per-use cost

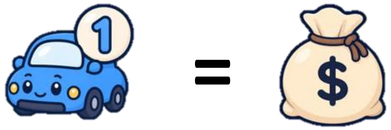
On-time investment

+

Recurring Cost

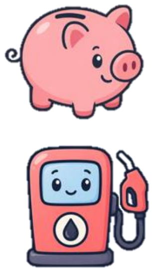
Case 1: using the car every day





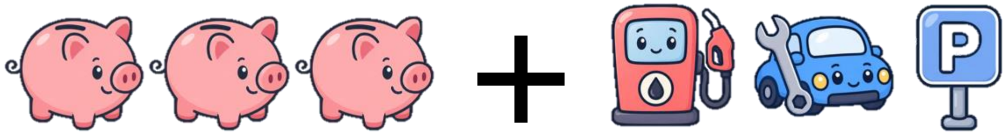
Lower cost per ride

Case 2: using a car once per month



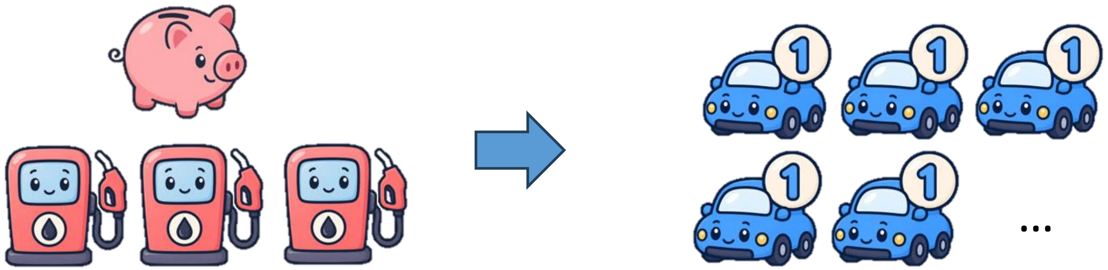
To Begin with: a Second-hand Vehicle

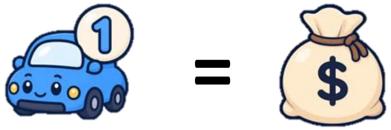
Owning a car vs. using Uber: Calculate per-use cost



On-time investment + Recurring Cost

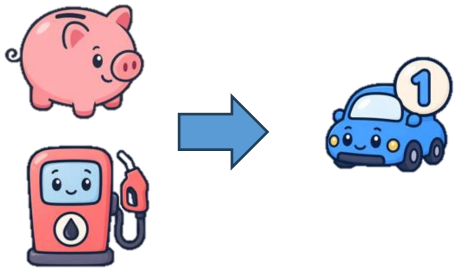
Case 1: using the car every day





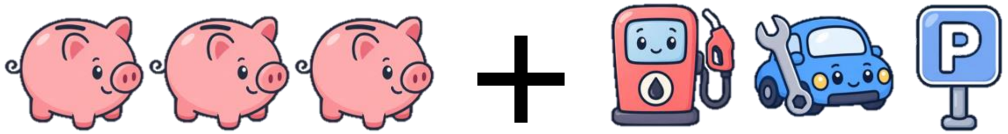
Lower cost per ride

Case 2: using a car once per month



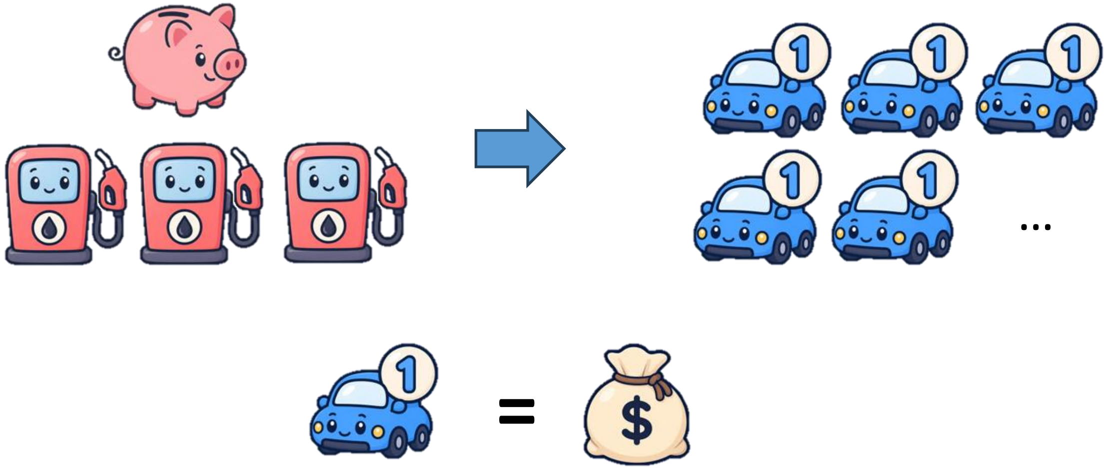
To Begin with: a Second-hand Vehicle

Owning a car vs. using Uber: Calculate per-use cost



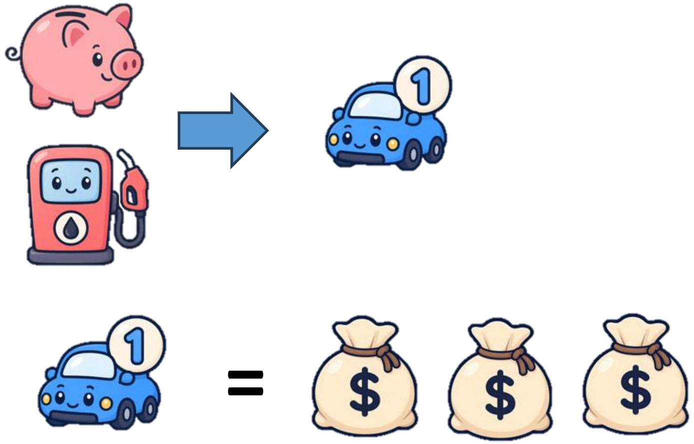
On-time investment + Recurring Cost

Case 1: using the car every day



Lower cost per ride

Case 2: using a car once per month



Higher cost per ride

Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle

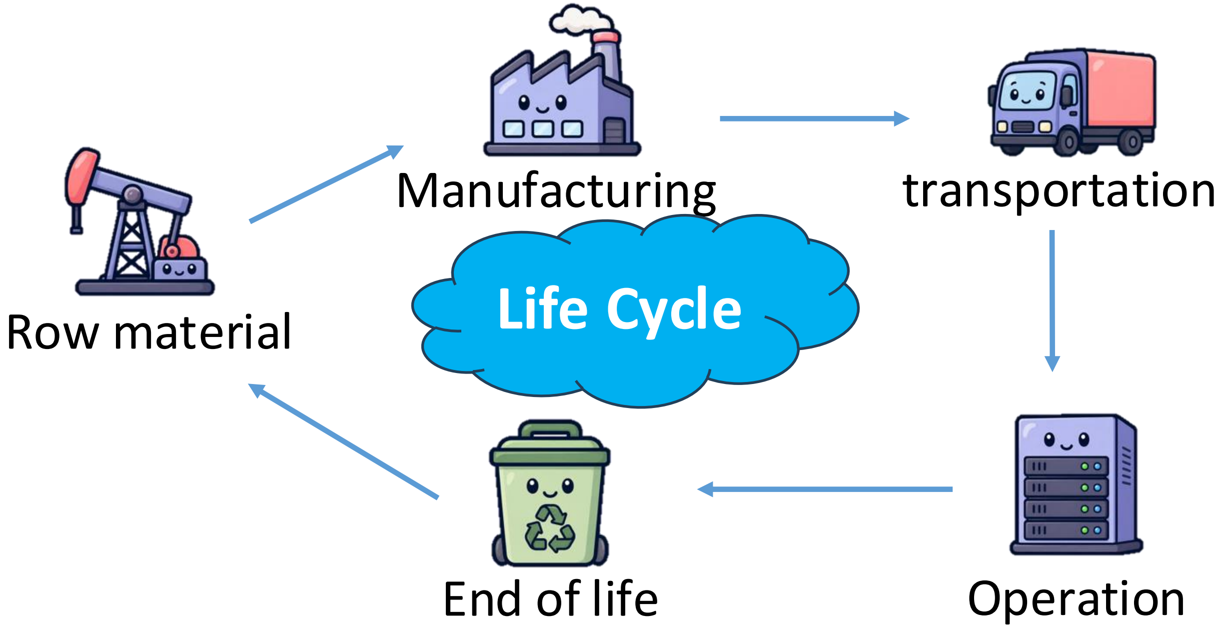
https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf

Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle



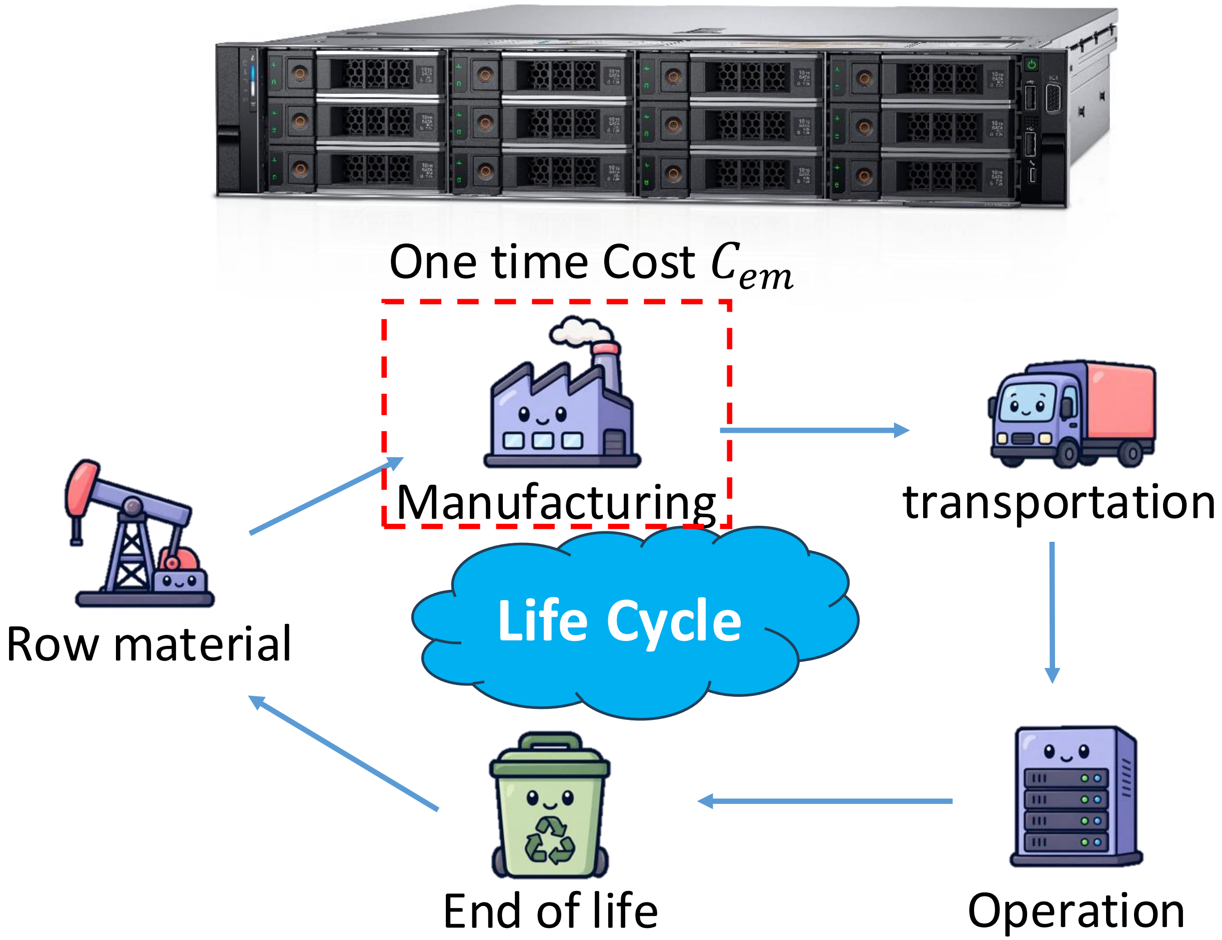
https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf

Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle



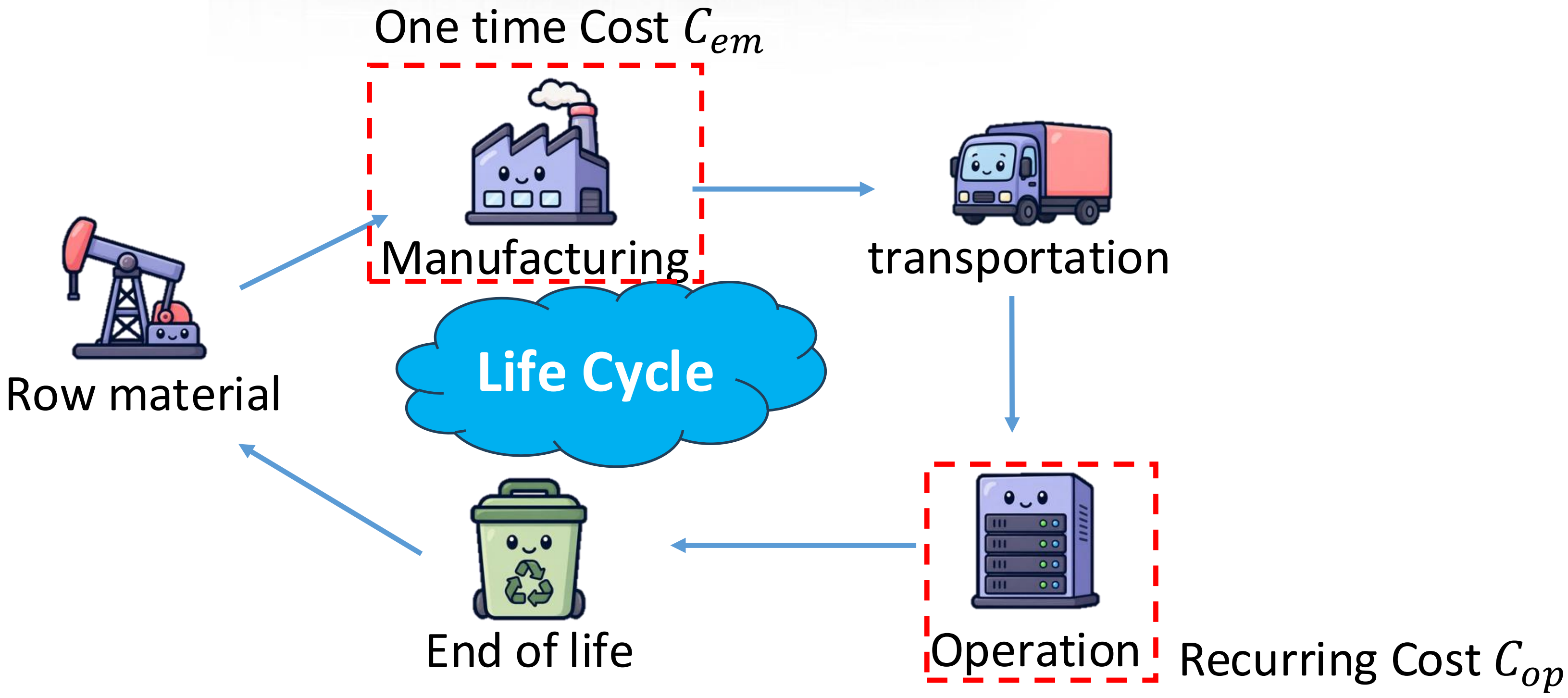
https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf

Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle



https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf

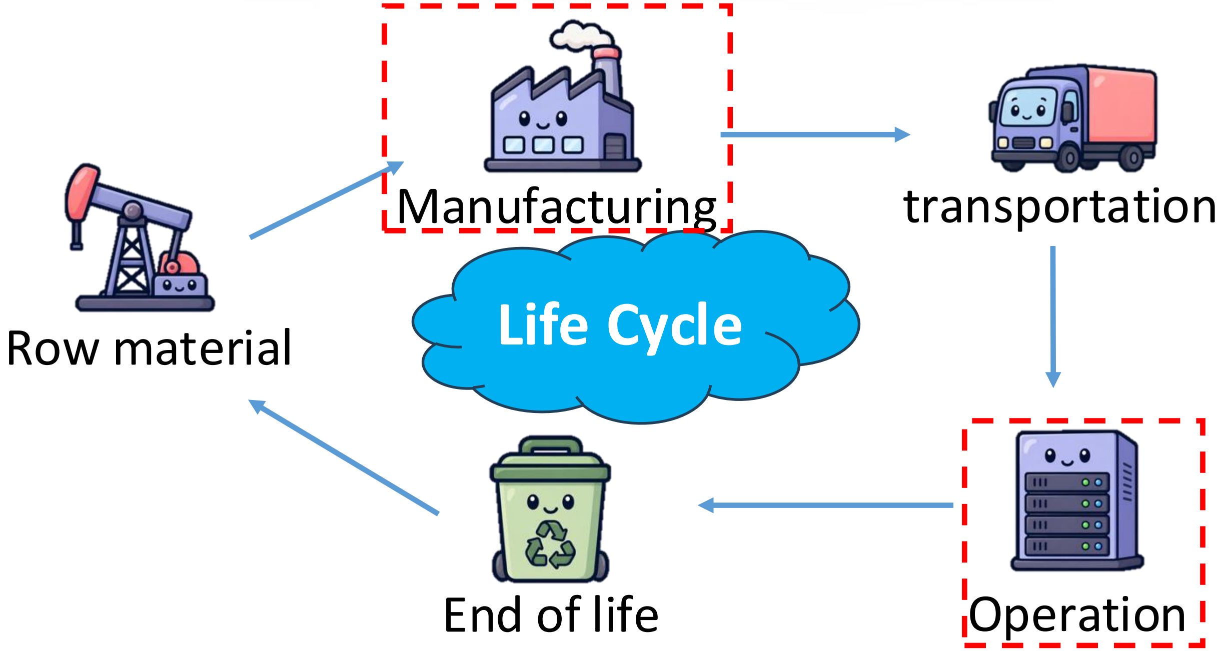
Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle

One time Cost C_{em}



Recurring Cost C_{op}

Embodied carbon +
Operational carbon:
> 95% of overall carbon cost

https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf

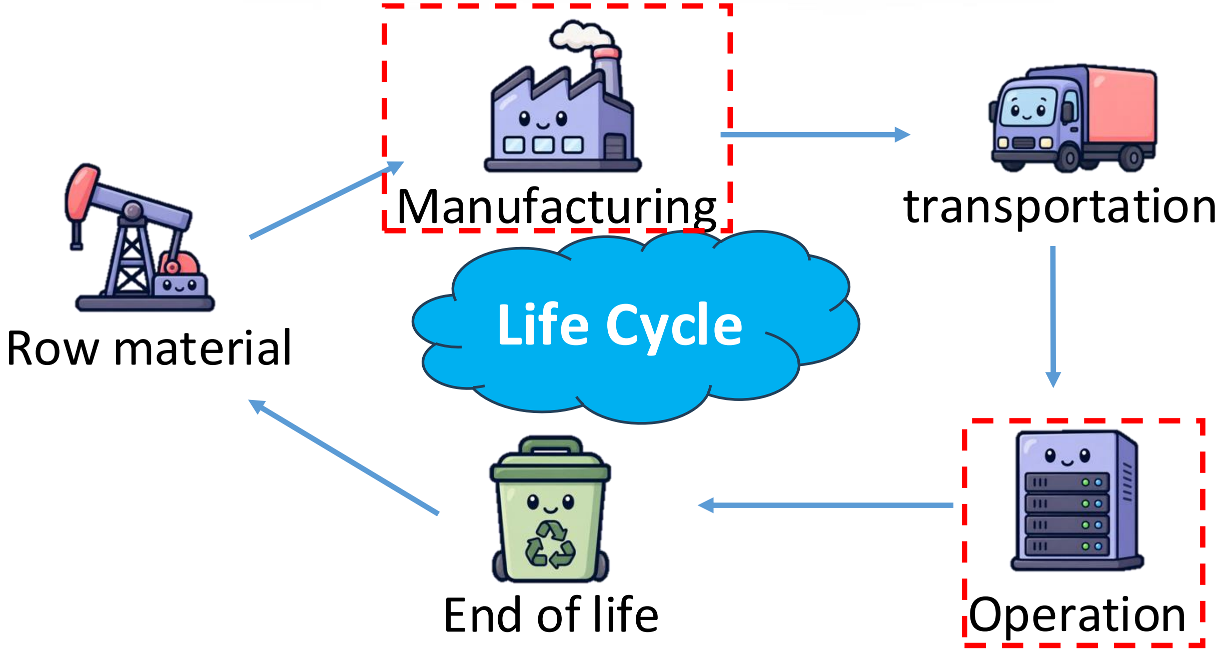
Carbon Cost for Datacenter? Dell Server R840

Consider owning a datacenter…

1 Dell Server R840

Carbon emission in whole life cycle

One time Cost C_{em}



Recurring Cost C_{op}

15600

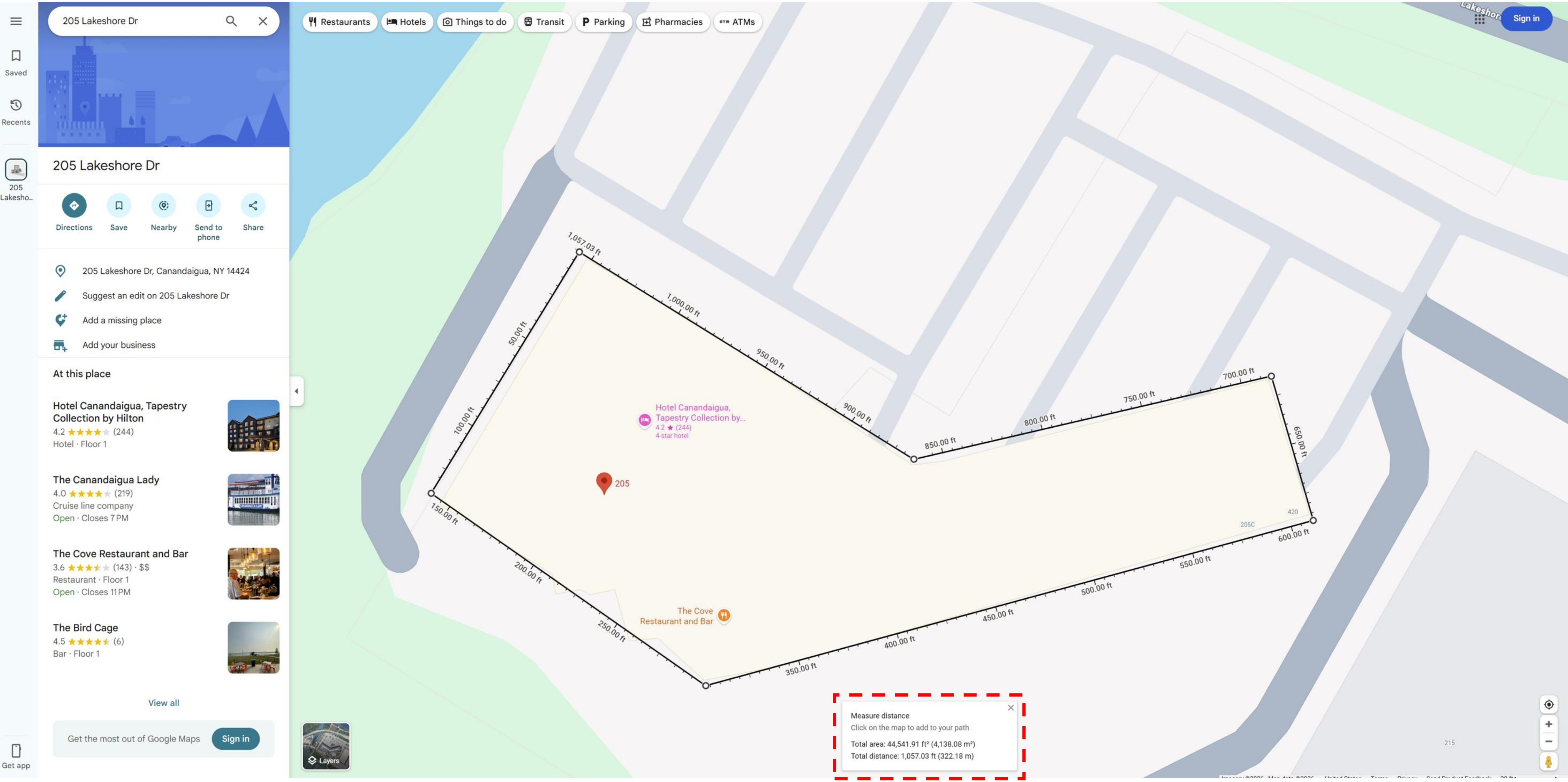
kgCO2e

According to vendor report

18 acres of US forests can absorb in a year

Embodied carbon + Operational carbon: > 95% of overall carbon cost

https://i.dell.com/sites/csdocuments/corpcomm_docs/en/carbon-footprint-poweredge-r840.pdf


205 Lakeshore Dr
Saved
Recents
205 Lakesho..
205 Lakeshore Dr
Directions
Save
Nearby
Send to phone
Share
205 Lakeshore Dr, Canandaigua, NY 14424
Suggest an edit on 205 Lakeshore Dr
Add a missing place
Add your business
At this place
Hotel Canandaigua, Tapestry Collection by Hilton
4.2 (244)
Hotel · Floor 1
The Canandaigua Lady
4.0 (219)
Cruise line company
Open · Closes 7 PM
The Cove Restaurant and Bar
3.6 (143) · $$
Restaurant · Floor 1
Open · Closes 11 PM
The Bird Cage
4.5 (6)
Bar · Floor 1
View all
Get the most out of Google Maps
Sign in
Get app
Restaurants
Hotels
Things to do
Transit
Parking
Pharmacies
ATMs
Sign in
Layers
1,057.03 ft
1,000.00 ft
950.00 ft
900.00 ft
850.00 ft
800.00 ft
750.00 ft
700.00 ft
650.00 ft
600.00 ft
550.00 ft
500.00 ft
450.00 ft
400.00 ft
350.00 ft
250.00 ft
200.00 ft
150.00 ft
100.00 ft
50.00 ft
Hotel Canandaigua, Tapestry Collection by...
4.2 (244)
4-star hotel
205
The Cove Restaurant and Bar
Measure distance
Click on the map to add to your path
Total area: 44,541.91 ft² (4,138.08 m²)
Total distance: 1,057.03 ft (322.18 m)

Hotel we are in: ~40,000 ft^2 = 0.9 acres

One sever ←→ 20x hotel area of forest

Design Choice for Datacenter Provisioner with Accelerators

There are newer and older GPUs…

For a datacenter carbon-aware provisioner…

…

Design Choice for Datacenter Provisioner with Accelerators

There are newer and older GPUs…

For a datacenter carbon-aware provisioner…

2017

V100
Volta Generation
14.13 TFLOPS

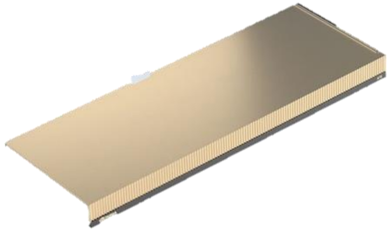
…

Design Choice for Datacenter Provisioner with Accelerators

There are newer and older GPUs…

For a datacenter carbon-aware provisioner…

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

V100
Volta Generation
14.13 TFLOPS

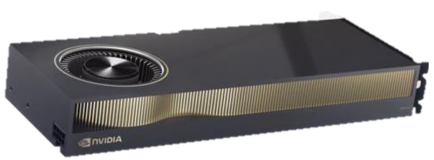
…

Design Choice for Datacenter Provisioner with Accelerators

There are newer and older GPUs…

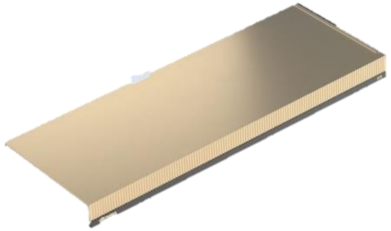
For a datacenter carbon-aware provisioner…

2023



RTX5000
Ada Generation
65.28 TFLOPS

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

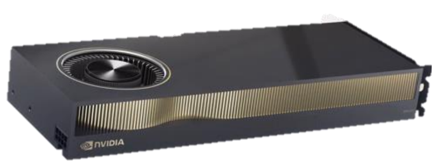
V100
Volta Generation
14.13 TFLOPS

…

Design Choice for Datacenter Provisioner with Accelerators

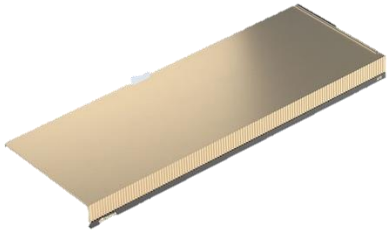
There are newer and older GPUs…

2023



RTX5000
Ada Generation
65.28 TFLOPS

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

V100
Volta Generation
14.13 TFLOPS

…

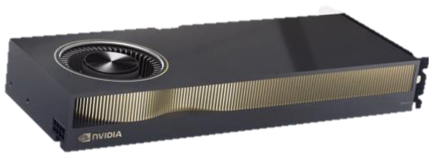
For a datacenter carbon-aware provisioner…

Upgrade to newer generation

Design Choice for Datacenter Provisioner with Accelerators

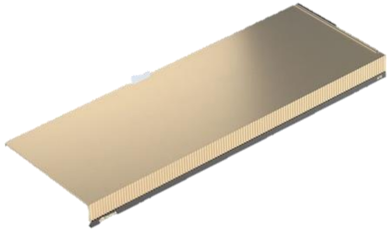
There are newer and older GPUs…

2023



RTX5000
Ada Generation
65.28 TFLOPS

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

V100
Volta Generation
14.13 TFLOPS

…

For a datacenter carbon-aware provisioner…

Upgrade to newer generation

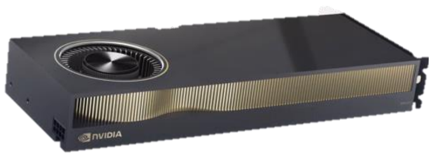
😀 New arch., better perf., better energy efficiency → reduce operational carbon

☹️ Immediate embodied carbon investment

Design Choice for Datacenter Provisioner with Accelerators

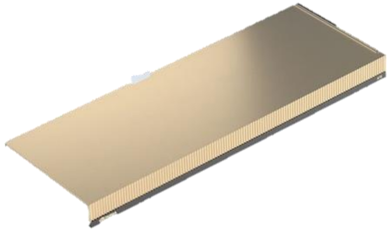
There are newer and older GPUs…

2023



RTX5000
Ada Generation
65.28 TFLOPS

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

V100
Volta Generation
14.13 TFLOPS

…

For a datacenter carbon-aware provisioner…

Upgrade to newer generation

😀 New arch., better perf., better energy efficiency → reduce operational carbon

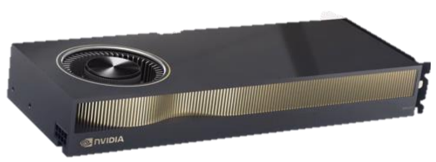
☹️ Immediate embodied carbon investment

Extend the lifetime of existing ones

Design Choice for Datacenter Provisioner with Accelerators

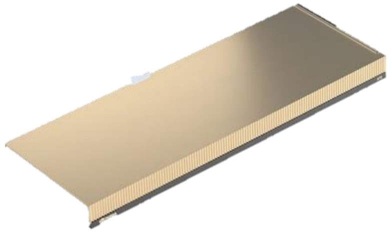
There are newer and older GPUs…

2023



RTX5000
Ada Generation
65.28 TFLOPS

2020



A10G
Ampere Generation
35.02 TFLOPS

2017

V100
Volta Generation
14.13 TFLOPS

…

For a datacenter carbon-aware provisioner…

Upgrade to newer generation

😀 New arch., better perf., better energy efficiency → reduce operational carbon

☹️ Immediate embodied carbon investment

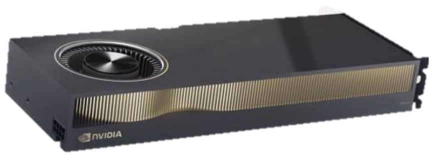
Extend the lifetime of existing ones

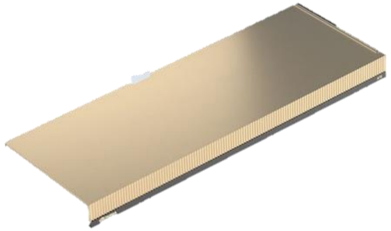
😀 Save embodied carbon cost

☹️ Lower energy efficiency, higher operational carbon

Design Choice for Datacenter Provisioner with Accelerators

There are newer and older GPUs…

2023

RTX5000
Ada Generation
65.28 TFLOPS

2020

A10G
Ampere Generation
35.02 TFLOPS

2017


V100
Volta Generation
14.13 TFLOPS

…

For a datacenter carbon-aware provisioner…

Upgrade to newer generation

😀 New arch., better perf., better energy efficiency → reduce operational carbon

☹ Immediate embodied carbon investment

Extend the lifetime of existing ones

😀 Save embodied carbon cost

☹ Lower energy efficiency, higher operational carbon

? **Quantify these design choices**

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Challenge 1: Simply averaging out does not follow the reuse principle

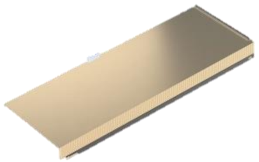
in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

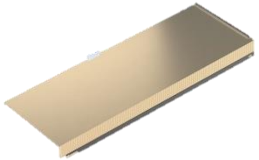


A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s



A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

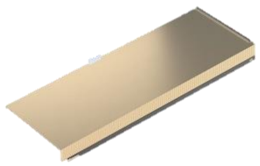
Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = 650 \cdot \frac{t}{6\ yrs.} + t \cdot 420 \cdot CI$

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s



A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + t \cdot 420 \cdot CI$

- $C_{em} * \frac{runtime}{lifetime}$

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

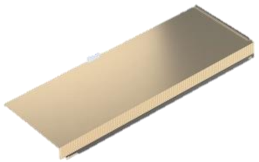
Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s



A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Per 1M inference
$C_{em} = 0.003\ KgCO_2e$
$C_{op} = 0.019\ KgCO_2e$

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Per 1M inference
$C_{em} = 0.003\ KgCO_2e$
$C_{op} = 0.019\ KgCO_2e$

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s

A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Per 1M inference
$C_{em} = \boxed{0.003}\ KgCO_2e$
$C_{op} = \boxed{0.019}\ KgCO_2e$

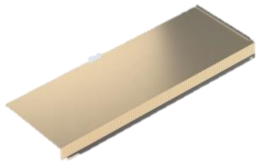
The newer GPU is better than the older one in both C_{em} and C_{op}

Per 1M inference
$C_{em} = \boxed{0.002}\ KgCO_2e$
$C_{op} = \boxed{0.016}\ KgCO_2e$

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s



A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Per 1M inference
$C_{em} = \boxed{0.003}\ KgCO_2e$
$C_{op} = \boxed{0.019}\ KgCO_2e$

The newer GPU is better than the older one in both C_{em} and C_{op}

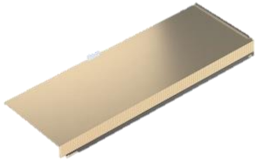
Per 1M inference
$C_{em} = \boxed{0.002}\ KgCO_2e$
$C_{op} = \boxed{0.016}\ KgCO_2e$

Reusing hardware shows no gain!

Challenge 1: Simply averaging out does not follow the reuse principle

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

V100 server
Embodied carbon: 650 $KgCO_2e$
Power: 420 W
Per 1M inference: 860 s



A10G server
Embodied carbon: 600 $KgCO_2e$
Power: 530 W
Per 1M inference: 580 s

Other assumption: 6-year lifetime, carbon intensity(CI) set to 0.188(active) and 0.018(idle)

Baseline solution: simply average out

Exec time $t = 860 \cdot 10^{-6}$

Carbon cost $C = \boxed{650 \cdot \frac{t}{6\ yrs.}} + \boxed{t \cdot 420 \cdot CI}$

- $C_{em} * \frac{runtime}{lifetime}$
- C_{op} during runtime

Per 1M inference
$C_{em} = \boxed{0.003}\ KgCO_2e$
$C_{op} = \boxed{0.019}\ KgCO_2e$

The newer GPU is better than the older one in both C_{em} and C_{op}

Per 1M inference
$C_{em} = \boxed{0.002}\ KgCO_2e$
$C_{op} = \boxed{0.016}\ KgCO_2e$

Reusing hardware shows no gain!

For the same tier, different generations of chips:

- C_{em}: similar → similar after average
- The baseline method thus relies more on C_{op}

Challenge 2: Secondary carbon at idle time is not accounted for

Challenge 2: Secondary carbon at idle time is not accounted for

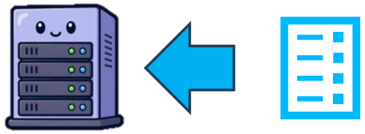
in 2020, consider how many A10G servers should we provision for the same target throughput amount

Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case

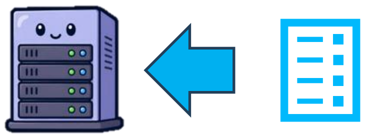
1x A10G server, 90% utilization



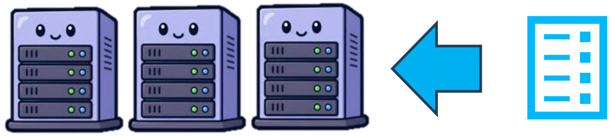
Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case
1x A10G server, 90% utilization



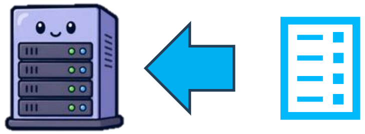
Over-provisioning
3x A10G servers, 30% utilization each



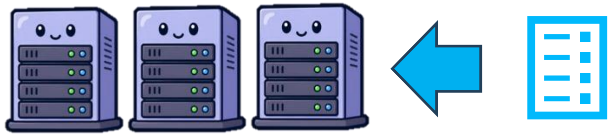
Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case
1x A10G server, 90% utilization



Over-provisioning
3x A10G servers, 30% utilization each



Baseline solution: simply average out

Exec time $t = 580 \cdot 10^{-6}$

Carbon cost $C = 600 \cdot \frac{t}{6\,yrs.} + t \cdot 530 \cdot CI$

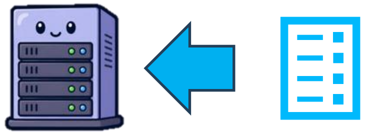
Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

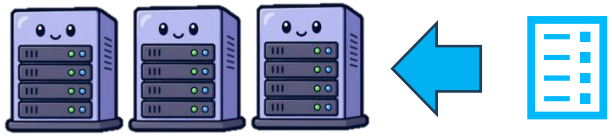
Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case
1x A10G server, 90% utilization



Over-provisioning
3x A10G servers, 30% utilization each



Baseline solution: simply average out

Exec time $t = 580 \cdot 10^{-6}$

Carbon cost $C = \boxed{600 \cdot \frac{t}{6\ yrs.} + t \cdot 530 \cdot CI}$

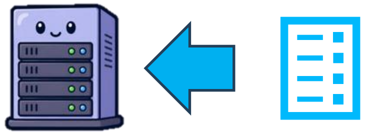
Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

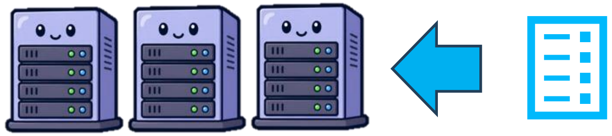
Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case
1x A10G server, 90% utilization



Over-provisioning
3x A10G servers, 30% utilization each



Baseline solution: simply average out

Exec time $t = 580 \cdot 10^{-6}$

Carbon cost $C = \boxed{600 \cdot \frac{t}{6\,yrs.} + t \cdot 530 \cdot CI}$

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

No differences between low and high utilization!

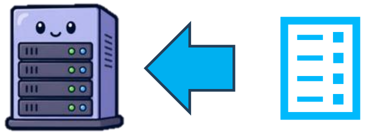
Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

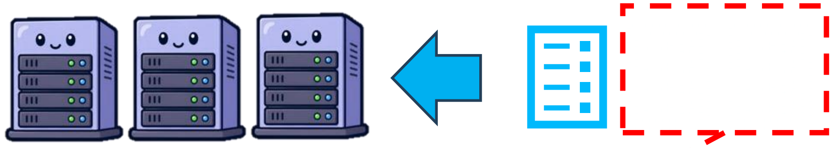
Regular Case

1x A10G server, 90% utilization



Over-provisioning

3x A10G servers, 30% utilization each



Pays C_{em} and C_{op} in idle time

Baseline solution: simply average out

Exec time $t = 580 \cdot 10^{-6}$

Carbon cost $C = \boxed{600 \cdot \frac{t}{6\ yrs.} + t \cdot 530 \cdot CI}$

Per 1M inference

$C_{em} = 0.002\ KgCO_2e$

$C_{op} = 0.016\ KgCO_2e$

No differences between low and high utilization!

Per 1M inference

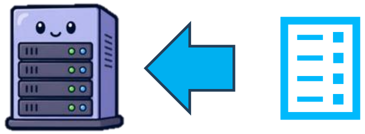
$C_{em} = 0.002\ KgCO_2e$

$C_{op} = 0.016\ KgCO_2e$

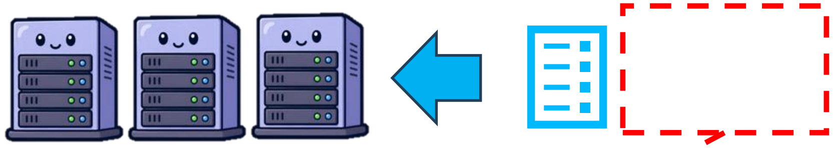
Challenge 2: Secondary carbon at idle time is not accounted for

in 2020, consider how many A10G servers should we provision for the same target throughput amount

Regular Case
1x A10G server, 90% utilization



Over-provisioning
3x A10G servers, 30% utilization each



Pays C_{em} and C_{op} in idle time

Baseline solution: simply average out

Exec time $t = 580 \cdot 10^{-6}$

Carbon cost $C = \boxed{600 \cdot \frac{t}{6\ yrs.} + t \cdot 530 \cdot CI}$

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

No differences between low and high utilization!

The secondary carbon (carbon in idle time) is not accounted for

Per 1M inference
$C_{em} = 0.002\ KgCO_2e$
$C_{op} = 0.016\ KgCO_2e$

Overview

Baseline method

Overview

Baseline method

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

Overview

Baseline method

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

Overview

Baseline method

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

- explore different non-linear depreciation models for C_{em}

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

Overview

Baseline method → Linear Depreciation models for C_{em}

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

→

- explore different non-linear depreciation models for C_{em}

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

Overview

Baseline method

Linear Depreciation models for C_{em}

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

- explore different non-linear depreciation models for C_{em}
- encourage the use of older hardware

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

Overview

Baseline method → Linear Depreciation models for C_{em}

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

→

- explore different non-linear depreciation models for C_{em}
- encourage the use of older hardware

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

→

- explore secondary carbon accounting
- promotes high server utilization

Overview

Baseline method - - - - → Linear Depreciation models for C_{em}

- (Almost) always prefer new devices,
- Inconsistent with the reusing principle

→

- explore different non-linear depreciation models for C_{em}
- encourage the use of older hardware

- Unaware of server utilization,
- Not able to distinguish good and bad(over) provision

→

- explore secondary carbon accounting
- promotes high server utilization

- study under different depreciation and secondary carbon accounting strategies

Non-linear Depreciation Models: What are them?

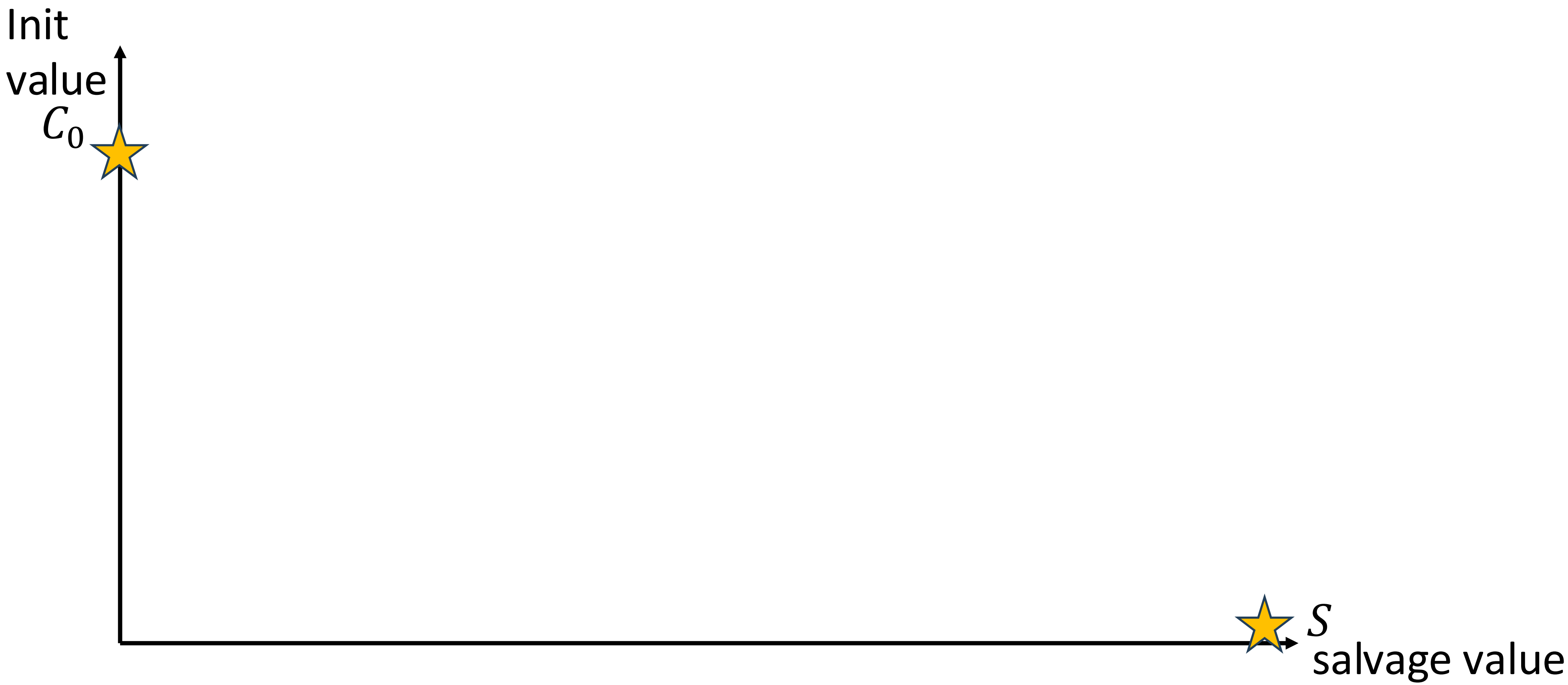
A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

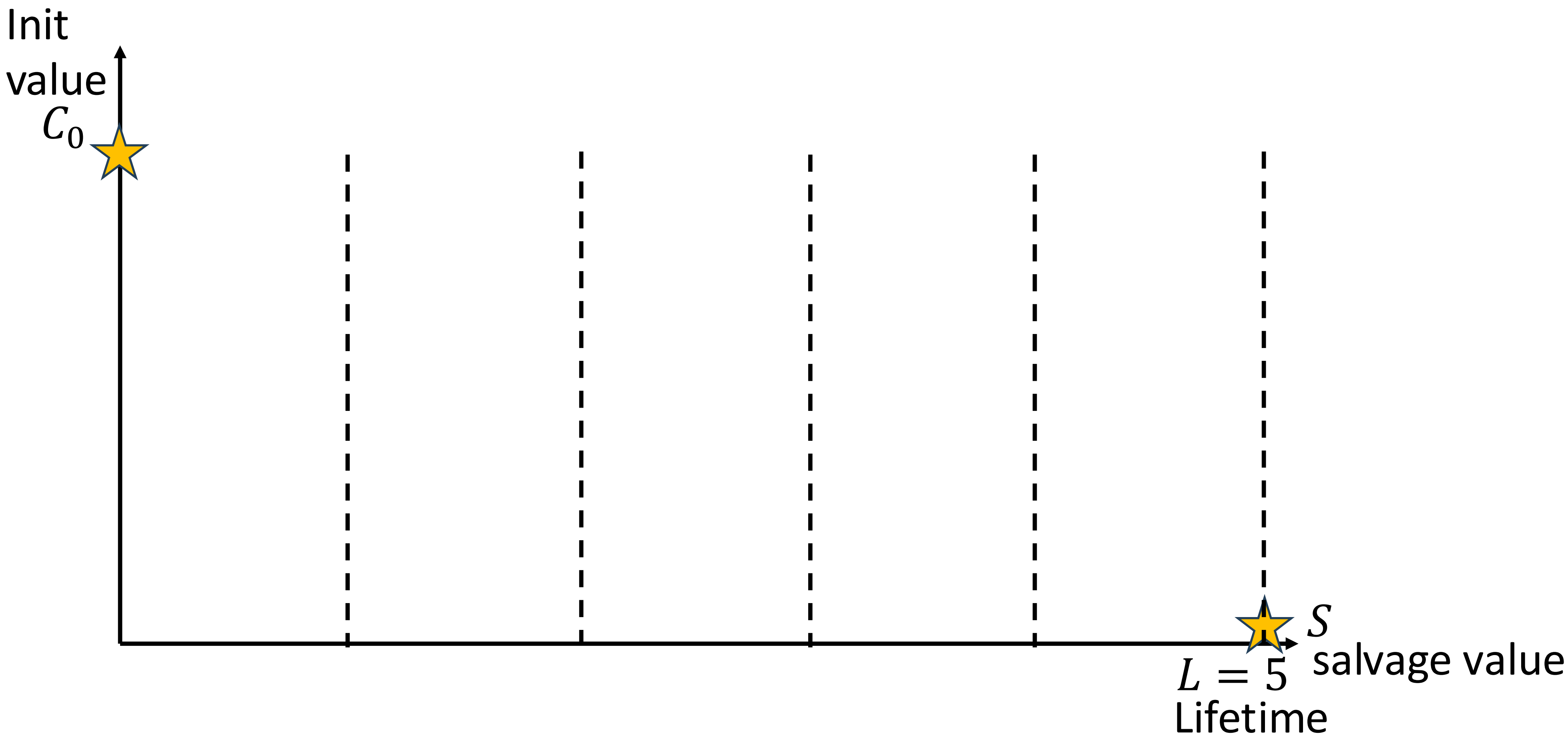
Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



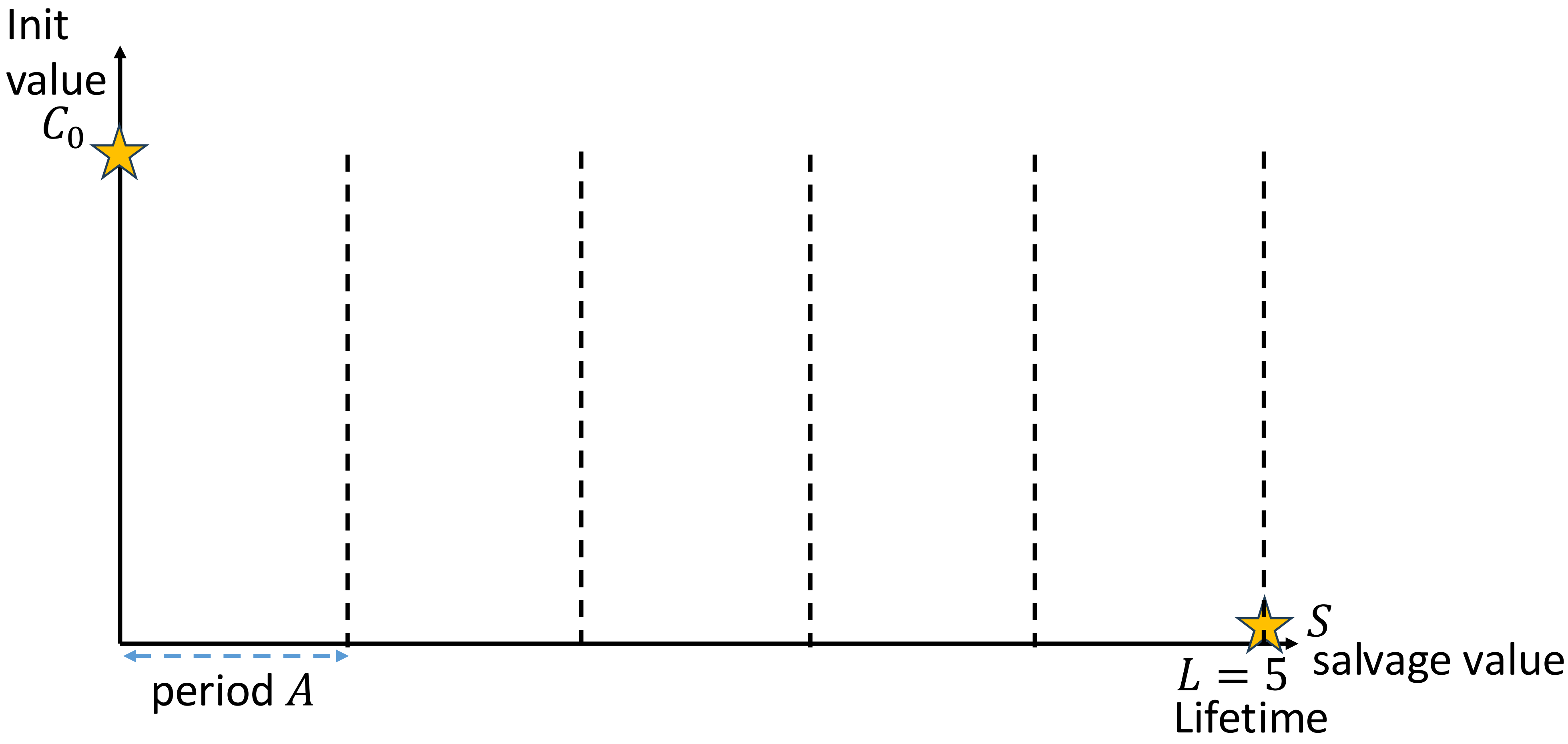
Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



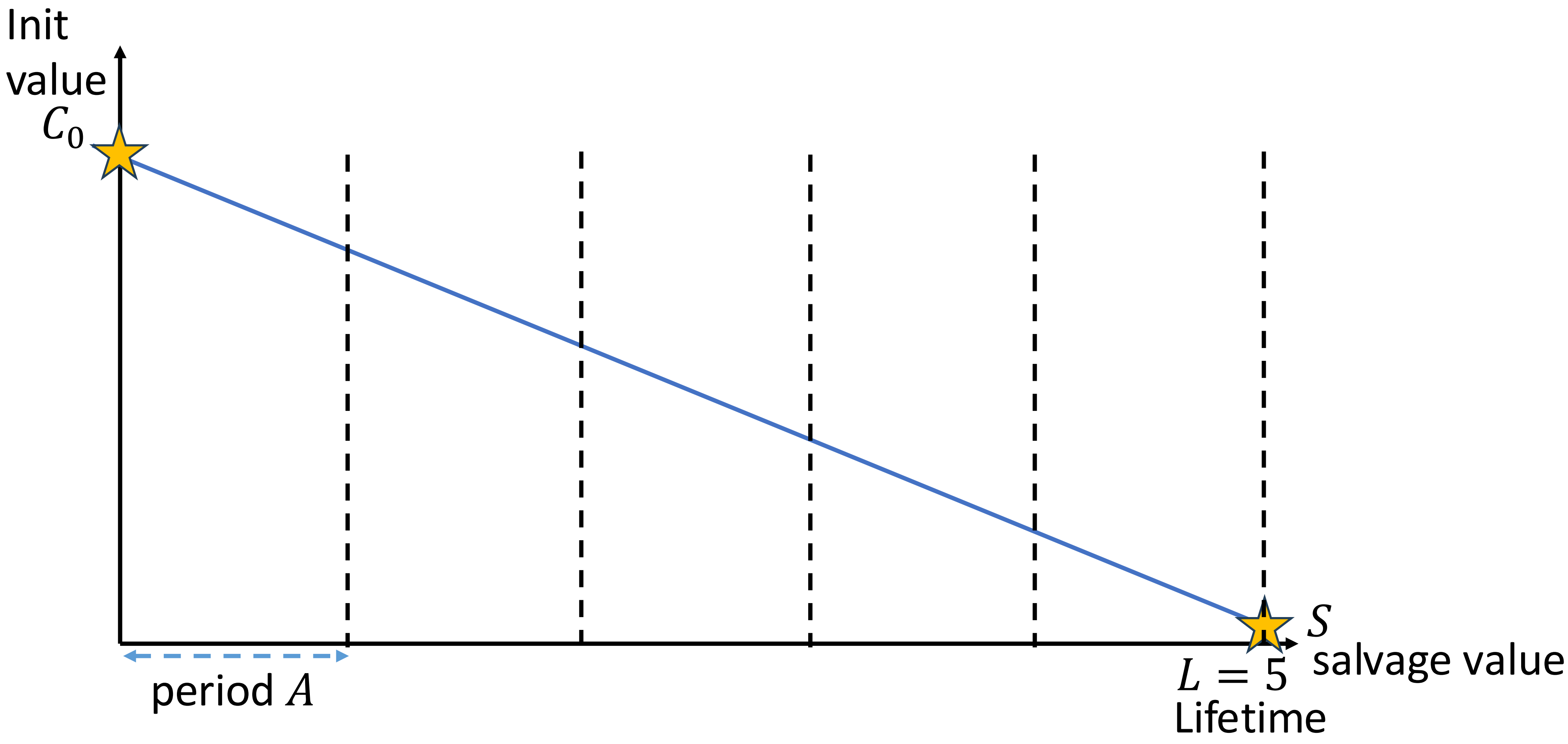
Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



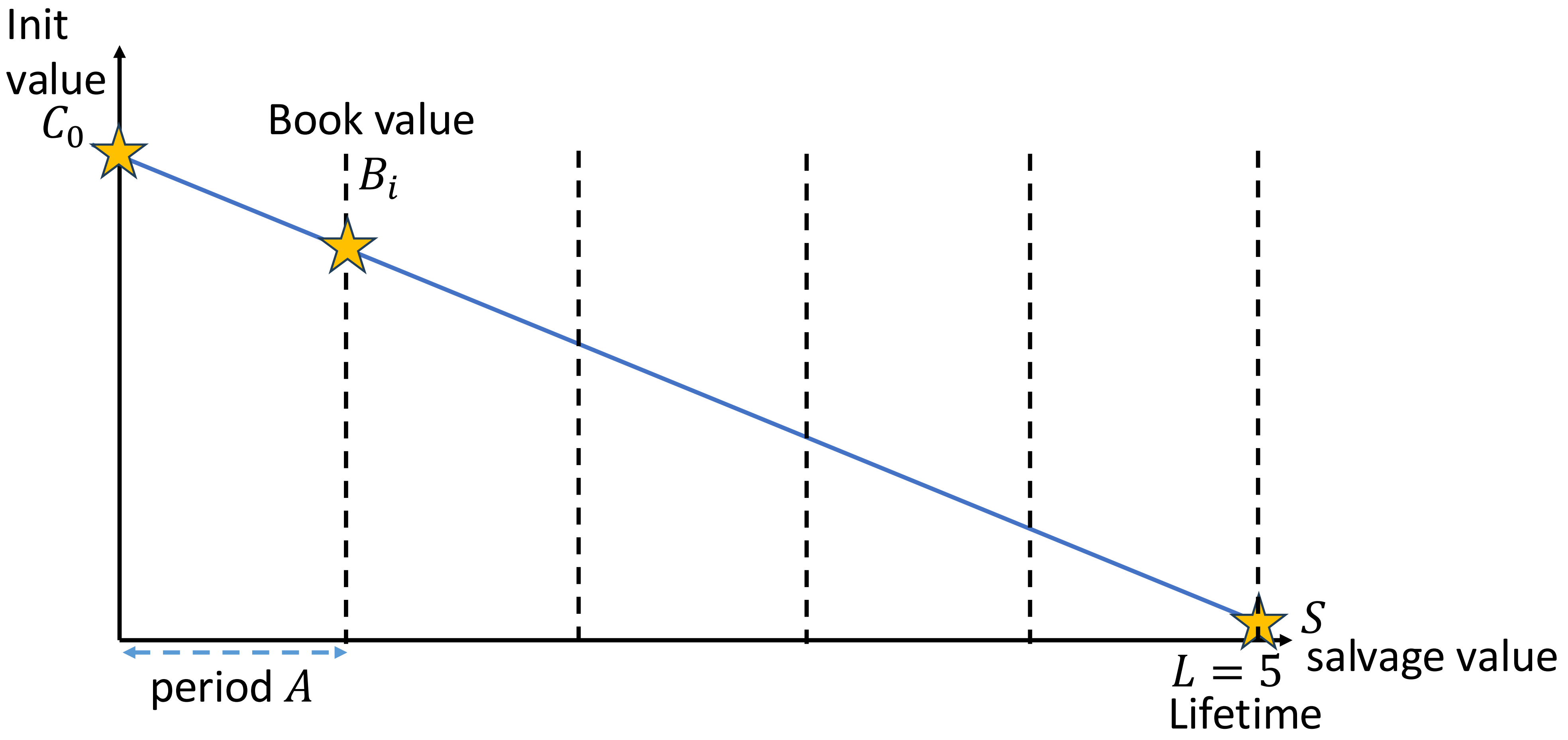
Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



Non-linear Depreciation Models: What are them?

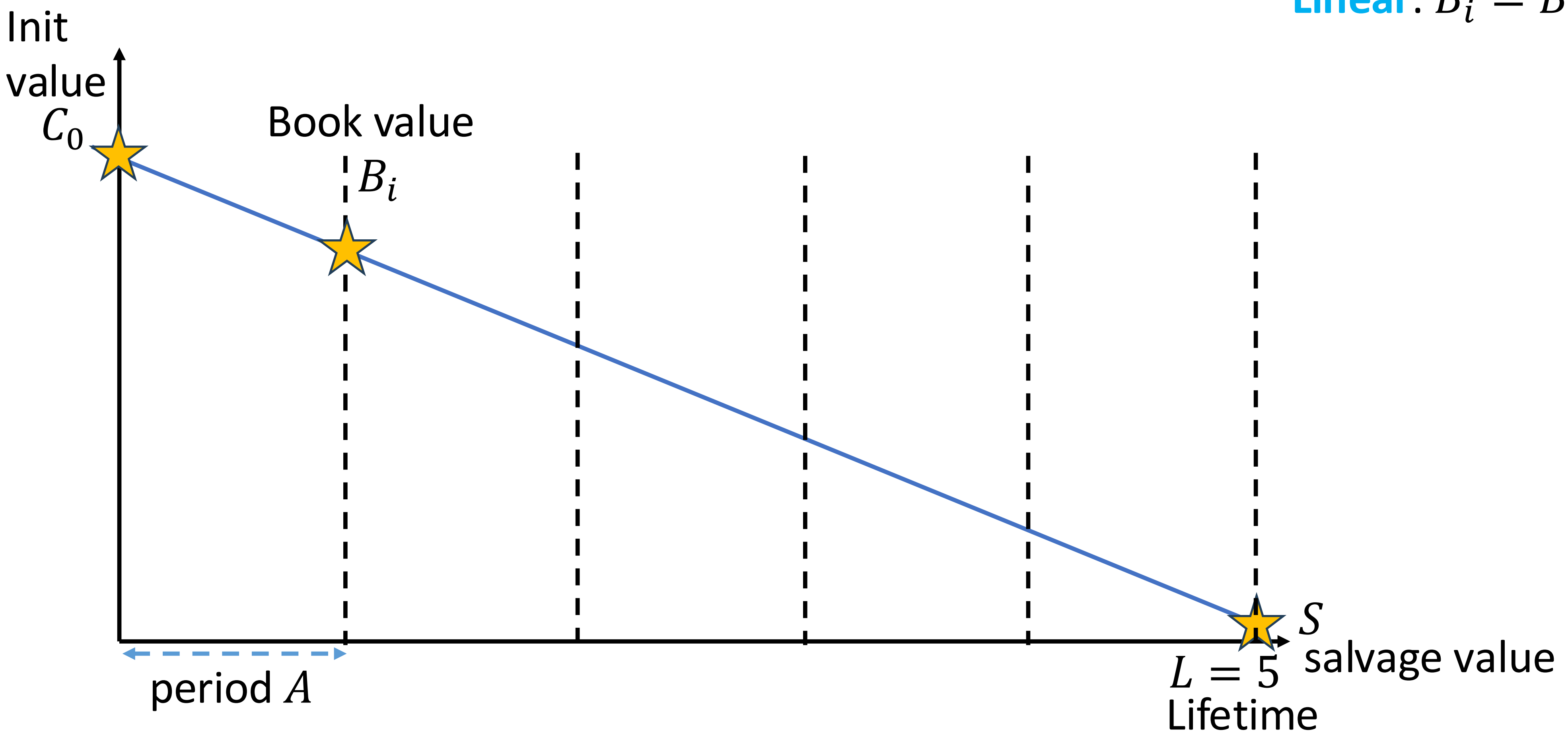
A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



Non-linear Depreciation Models: What are them?

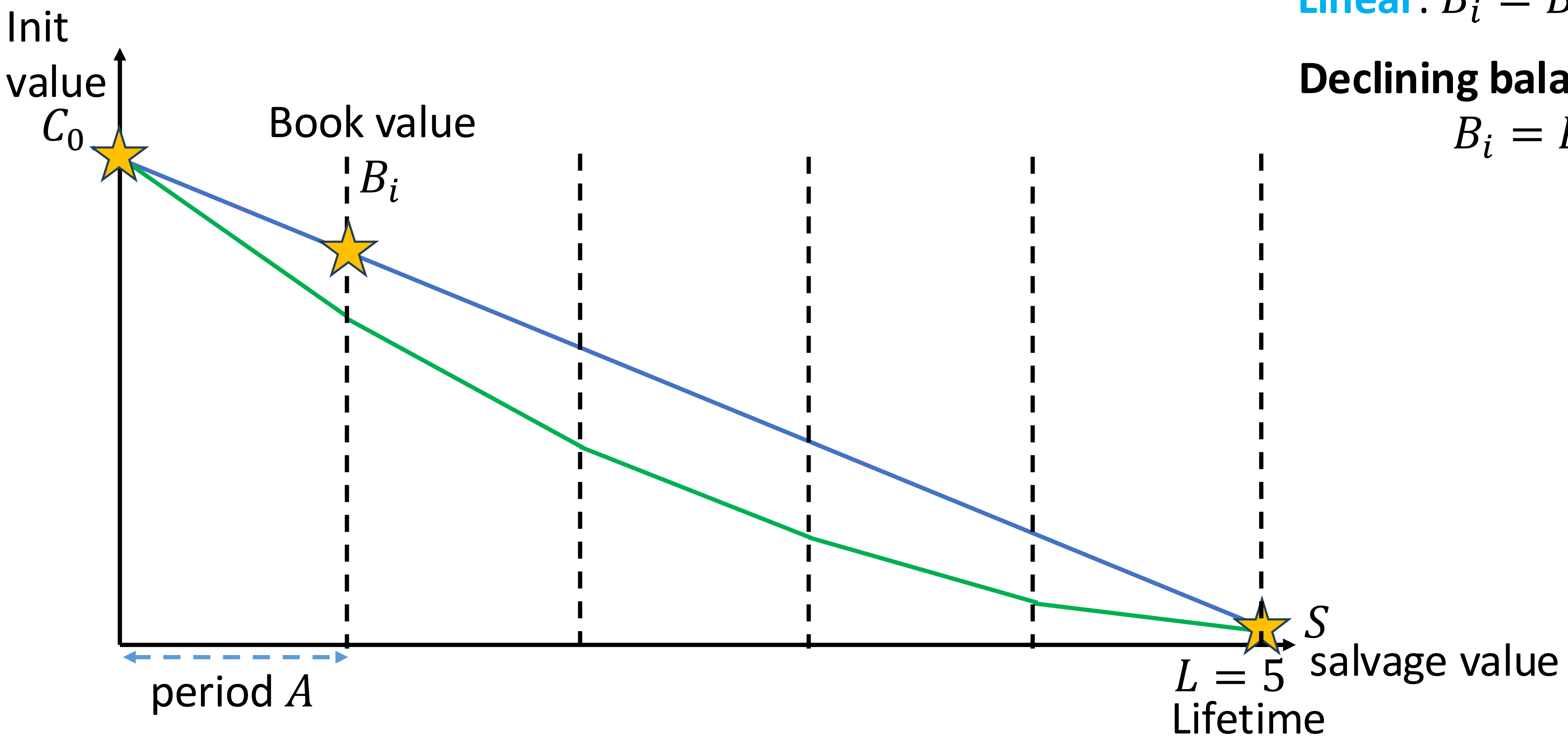
A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$



Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



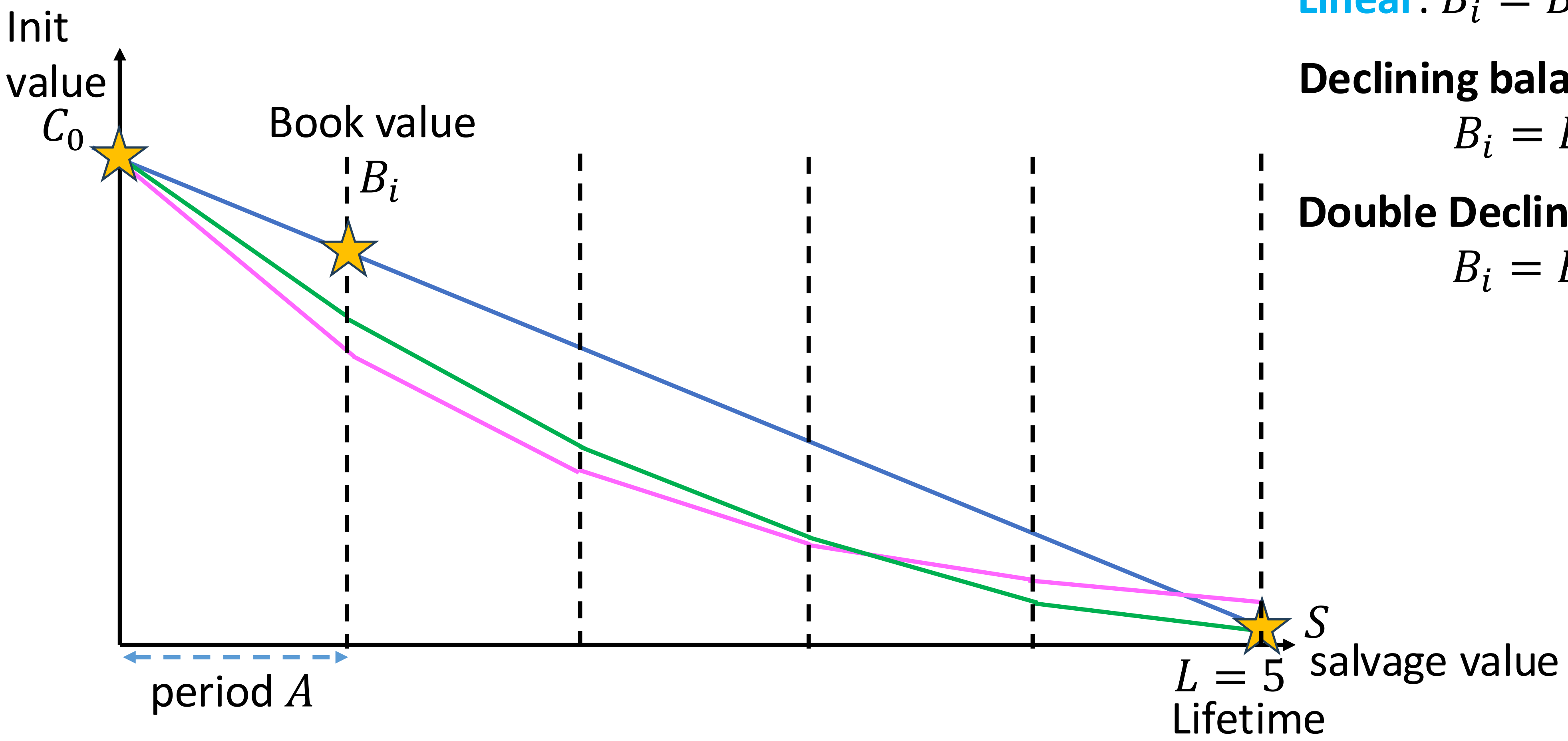
Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$

Declining balance (DB):

$$B_i = B_{i-1}(1 - r)$$

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$

Declining balance (DB):

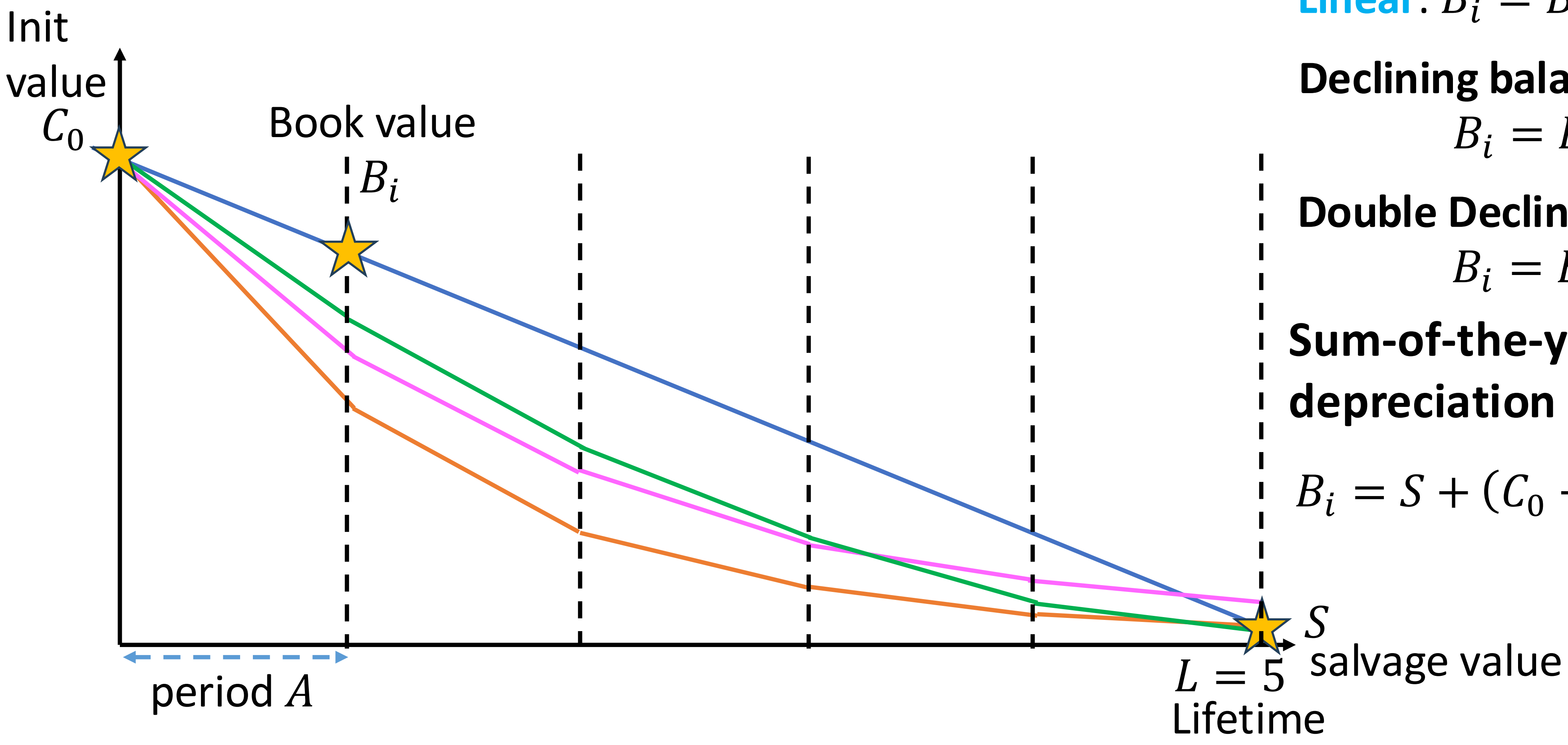
$$B_i = B_{i-1}(1-r)$$

Double Declining balance (DDB):

$$B_i = B_{i-1}(1-r)$$

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.



Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$

Declining balance (DB):

$$B_i = B_{i-1}(1 - r)$$

Double Declining balance (DDB):

$$B_i = B_{i-1}(1 - r)$$

Sum-of-the-years'-digits depreciation (SYD):

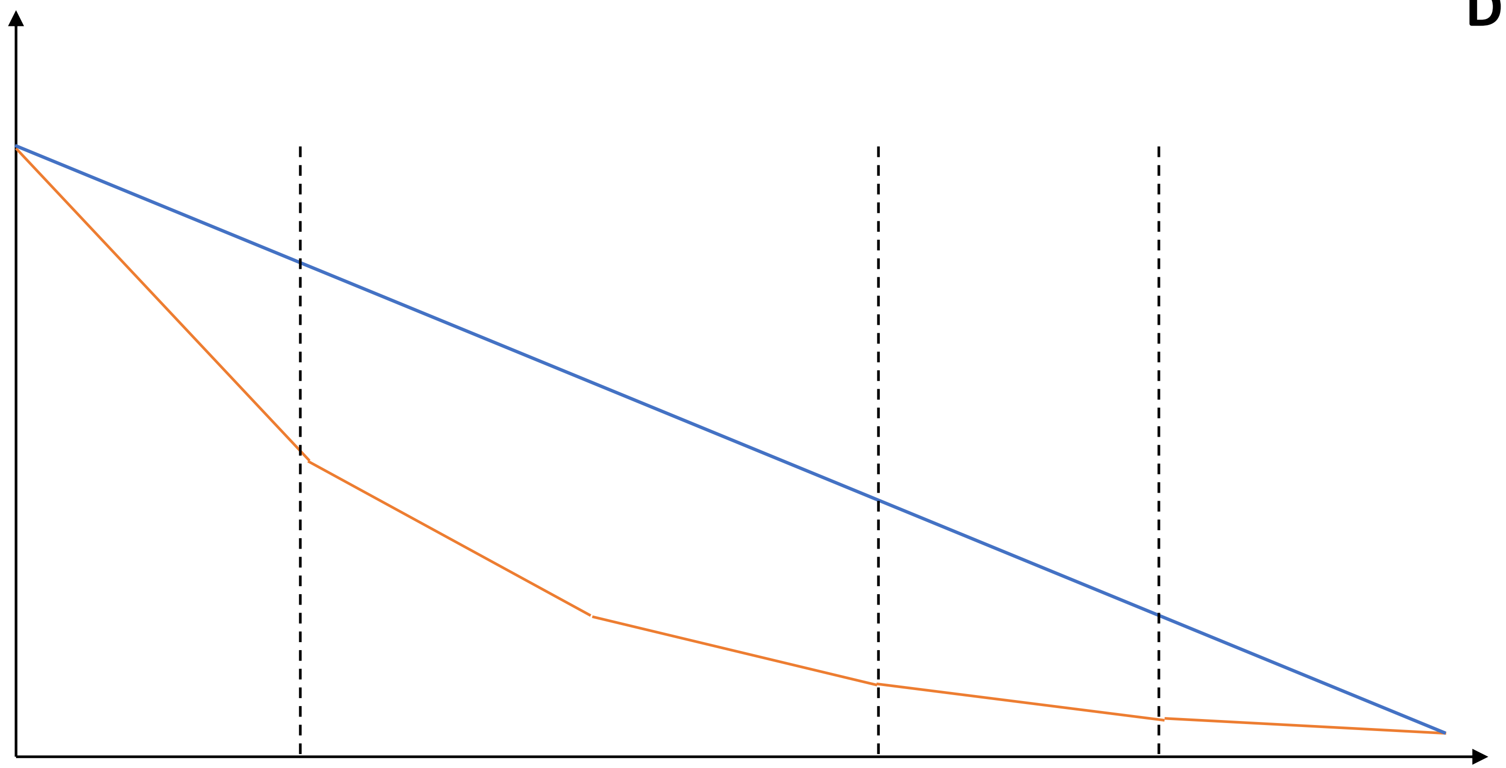
$$B_i = S + (C_0 - S)\frac{(L - i)(L - i + 1)}{(L)(L + 1)}$$

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1 - r)$

DB vs. linear depreciation

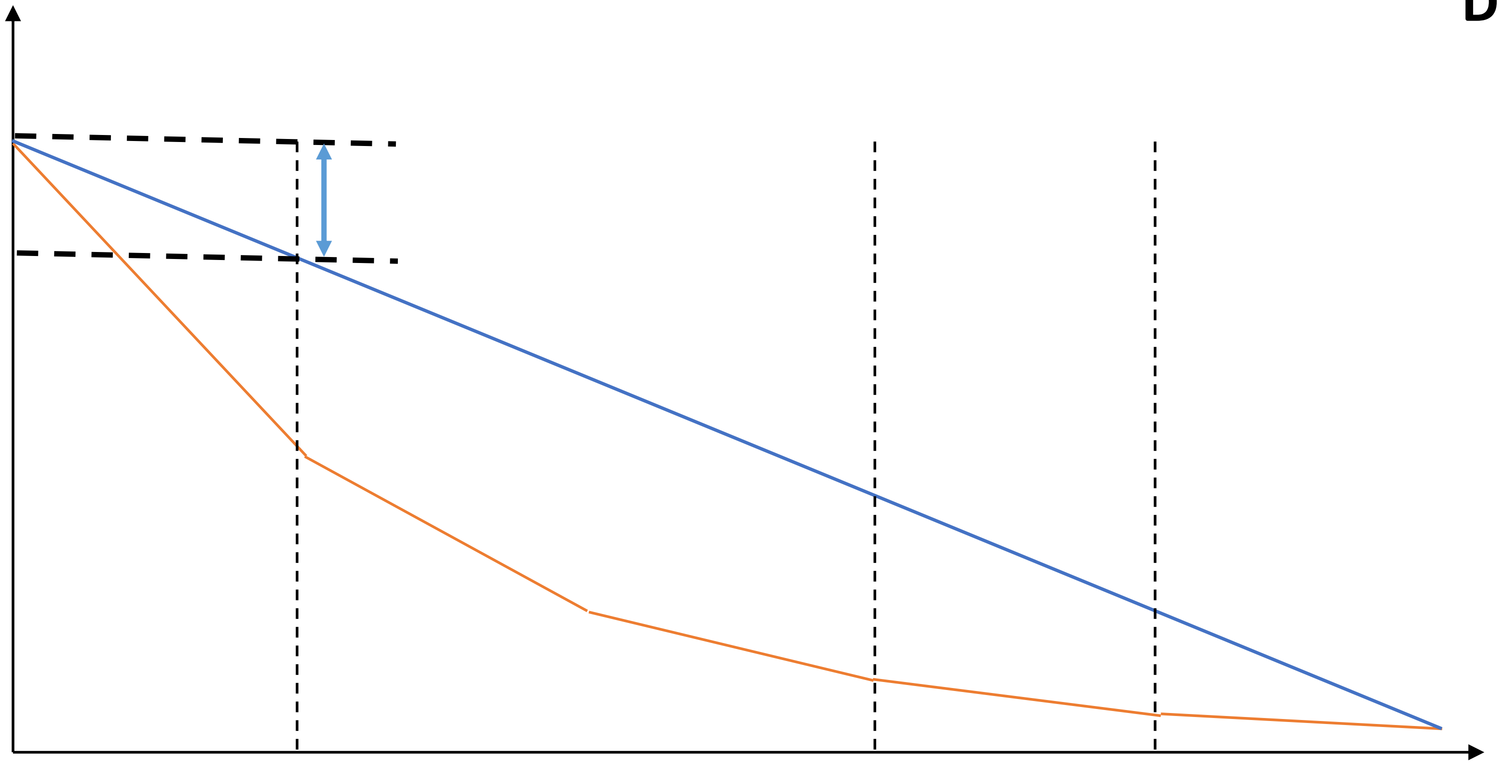


Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1-r)$

DB vs. linear depreciation

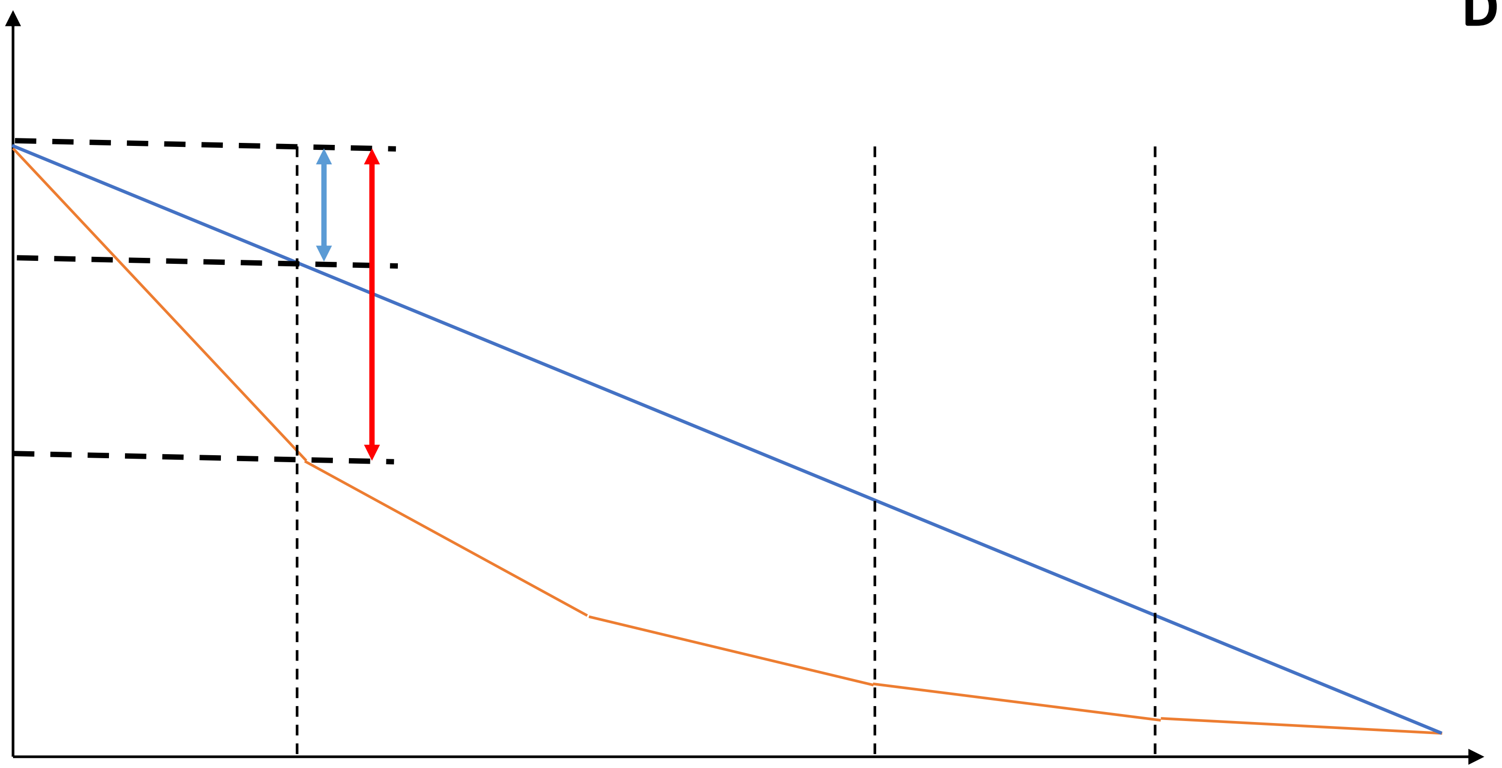


Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1-r)$

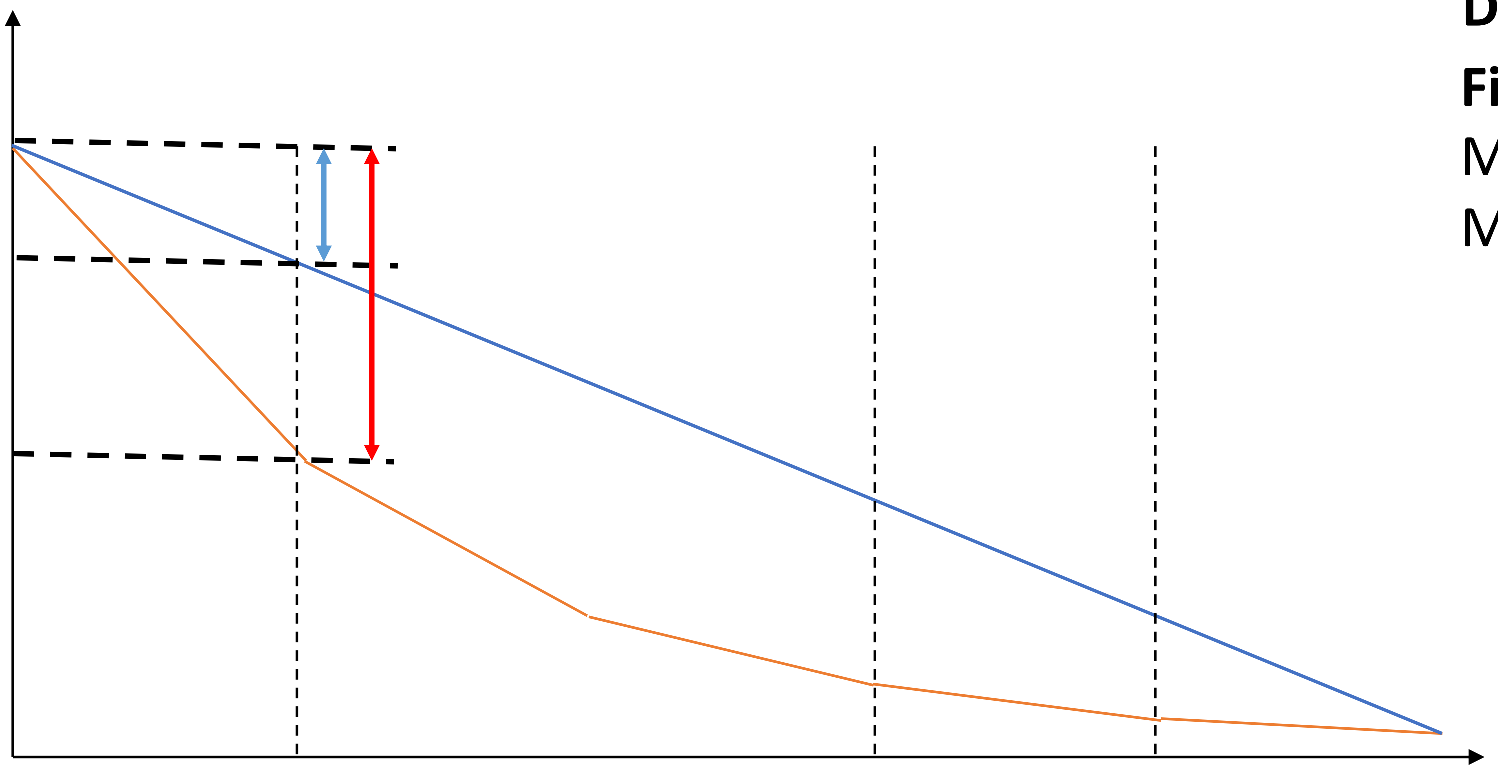
DB vs. linear depreciation



Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1 - r)$



DB vs. linear depreciation

First years:

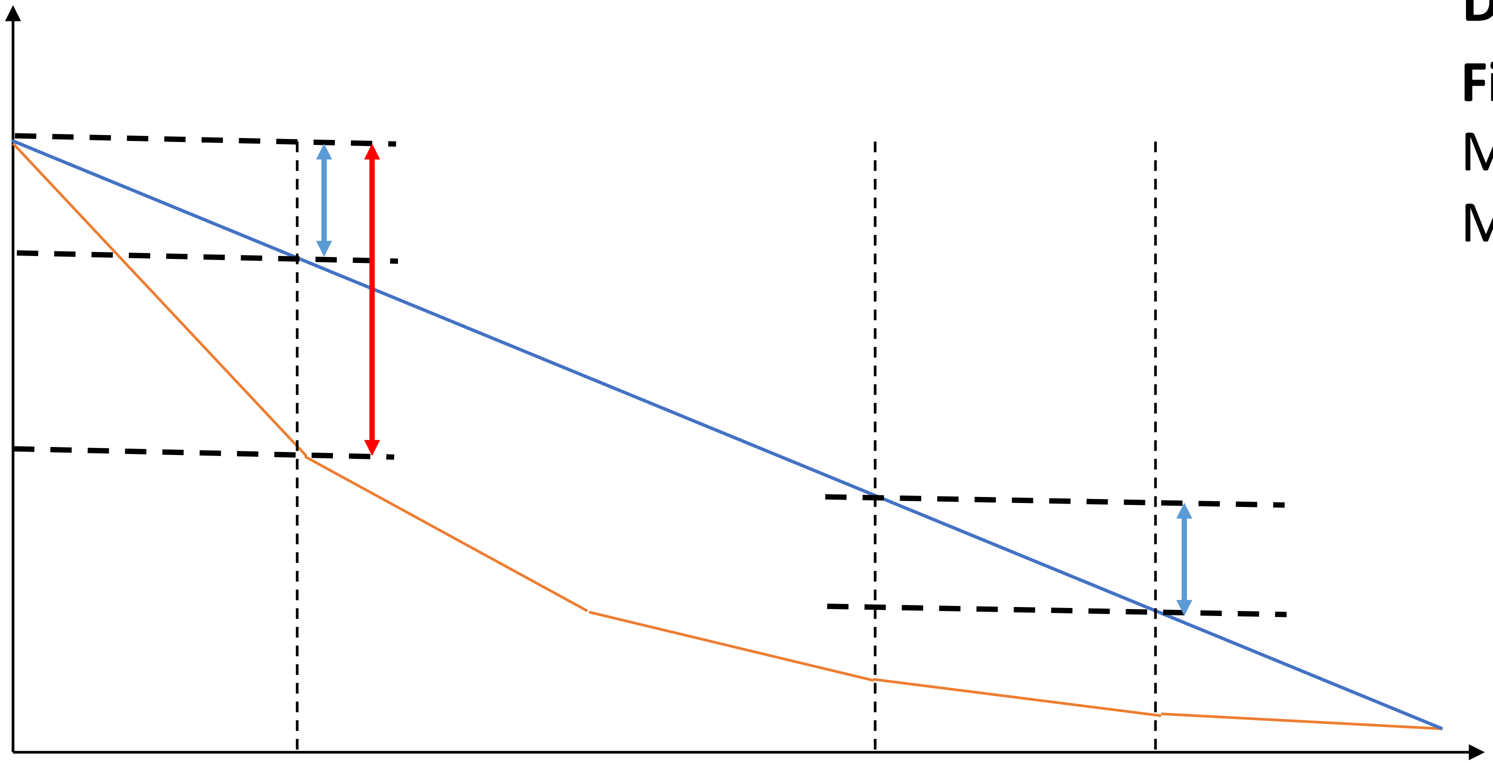
More value depreciated

More embodied carbon cost

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1 - r)$



DB vs. linear depreciation

First years:

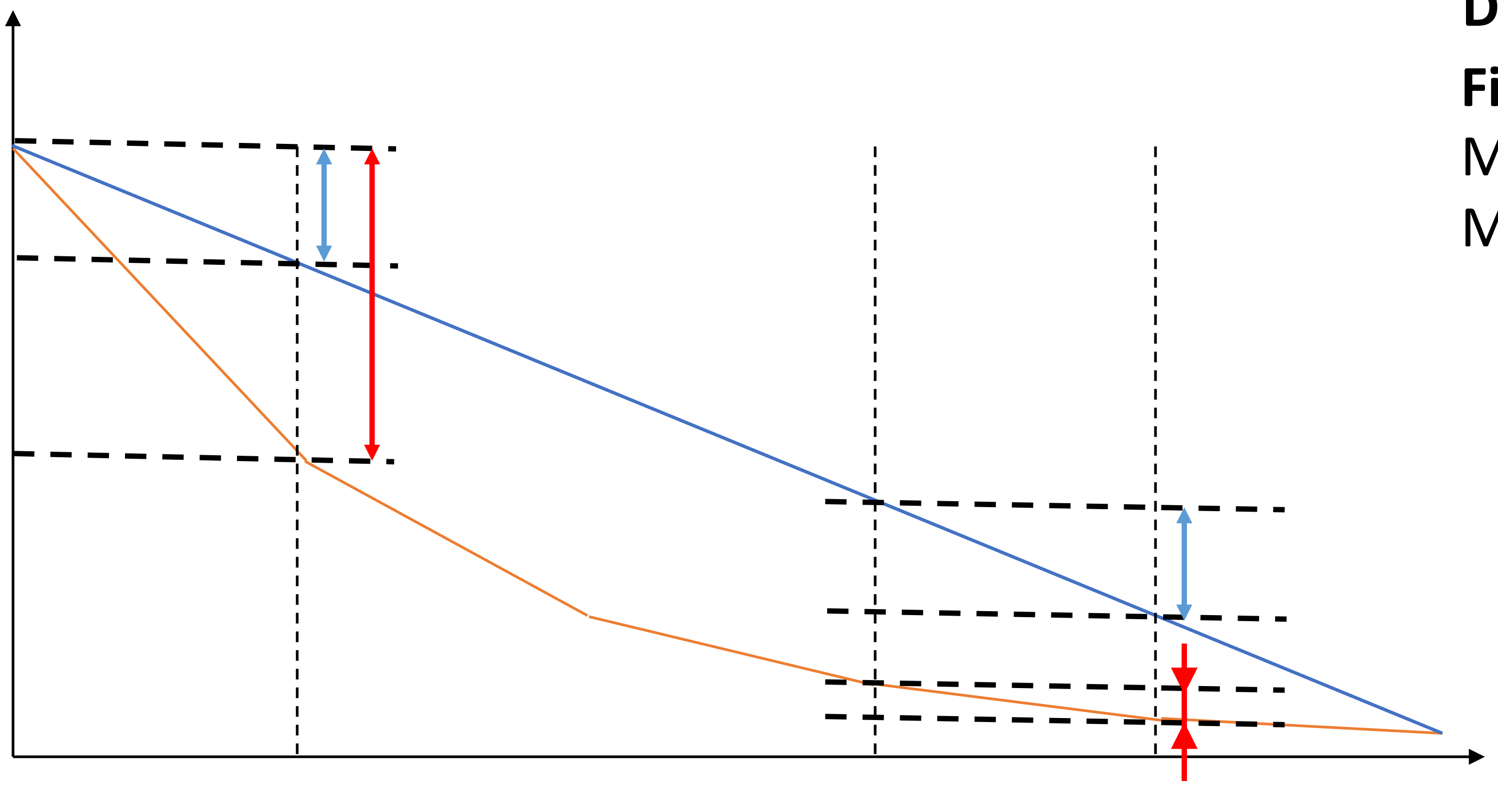
More value depreciated

More embodied carbon cost

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance** (**DB**):$B_i = B_{i-1}(1-r)$



DB vs. linear depreciation

First years:

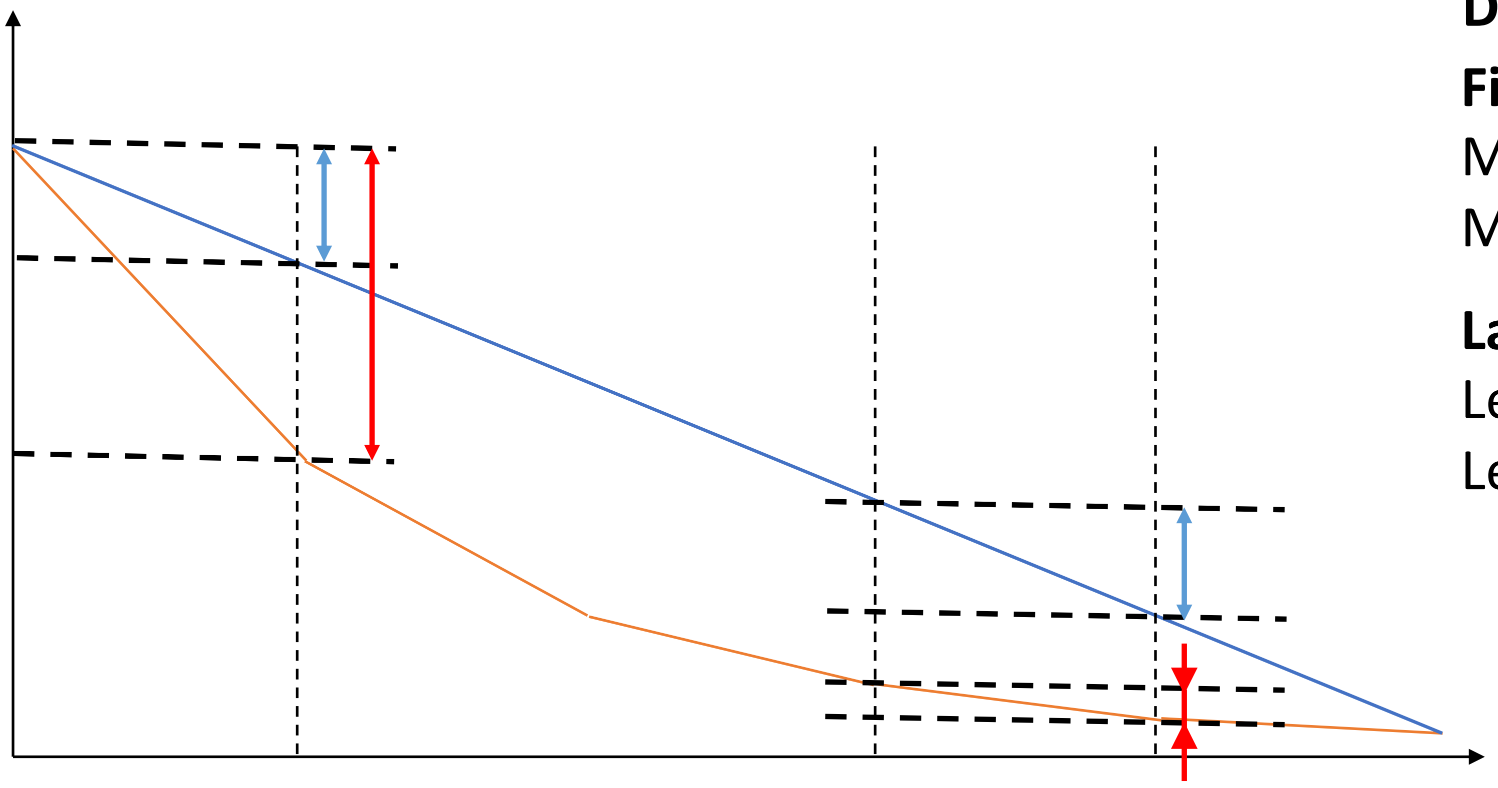
More value depreciated

More embodied carbon cost

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**: $B_i = B_{i-1}(1-r)$



DB vs. linear depreciation

First years:

More value depreciated

More embodied carbon cost

Last years:

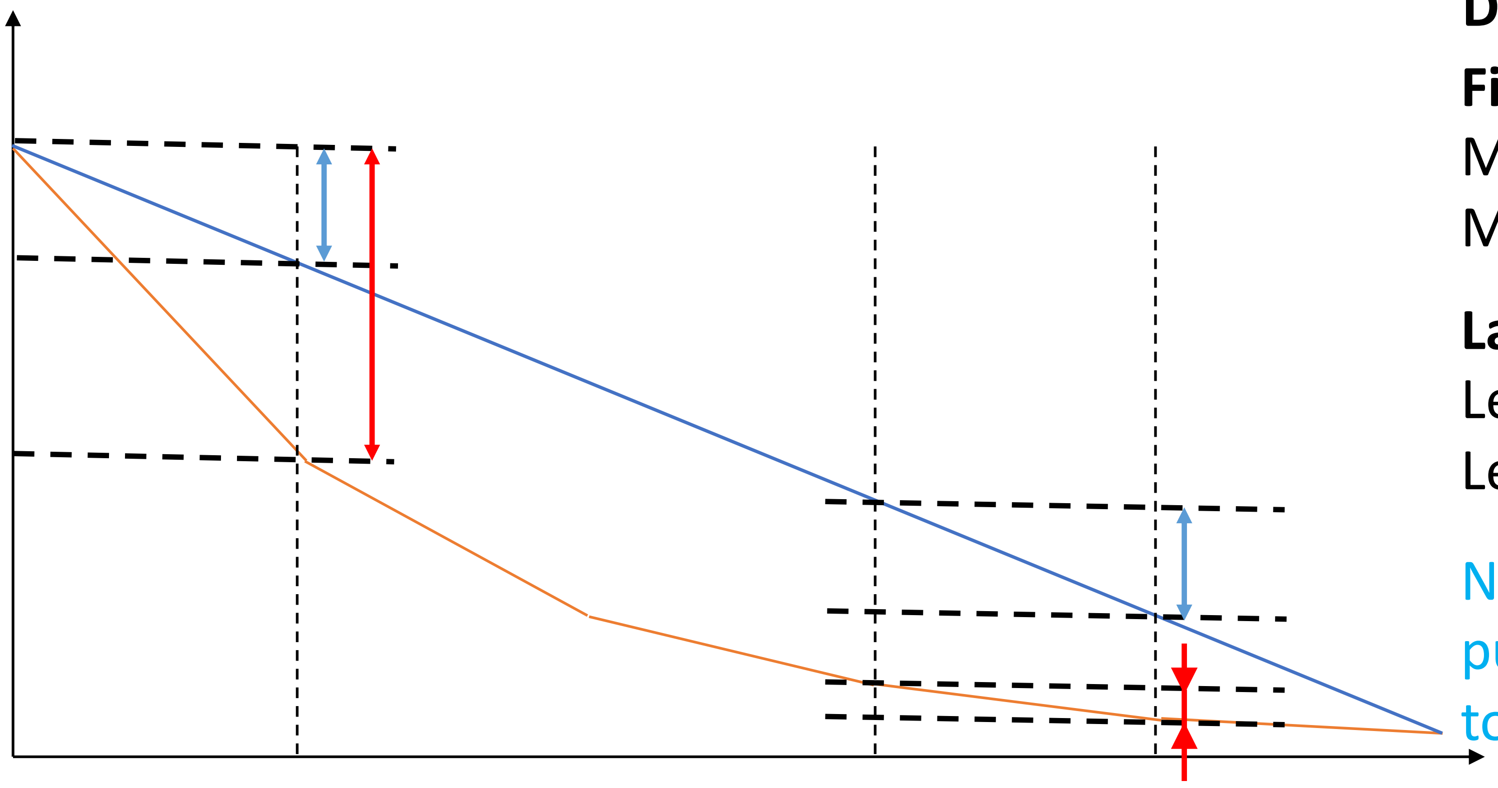
Less value depreciated

Less embodied carbon cost attributed

Non-linear Depreciation Models: What are them?

A **depreciation model** is a mathematical rule that allocates the depreciable value of an asset over its useful lifetime.

Linear: $B_i = B_{i-1} - \frac{C_0 - S}{T}$ **Declining balance (DB)**:$B_i = B_{i-1}(1-r)$



DB vs. linear depreciation

First years:

More value depreciated

More embodied carbon cost

Last years:

Less value depreciated

Less embodied carbon cost attributed

Non-linear depreciation model pushes more embodied carbon cost to the first layers.

Apply Depreciation Model to Cost Quantification

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

→

SCARIF
Framework

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

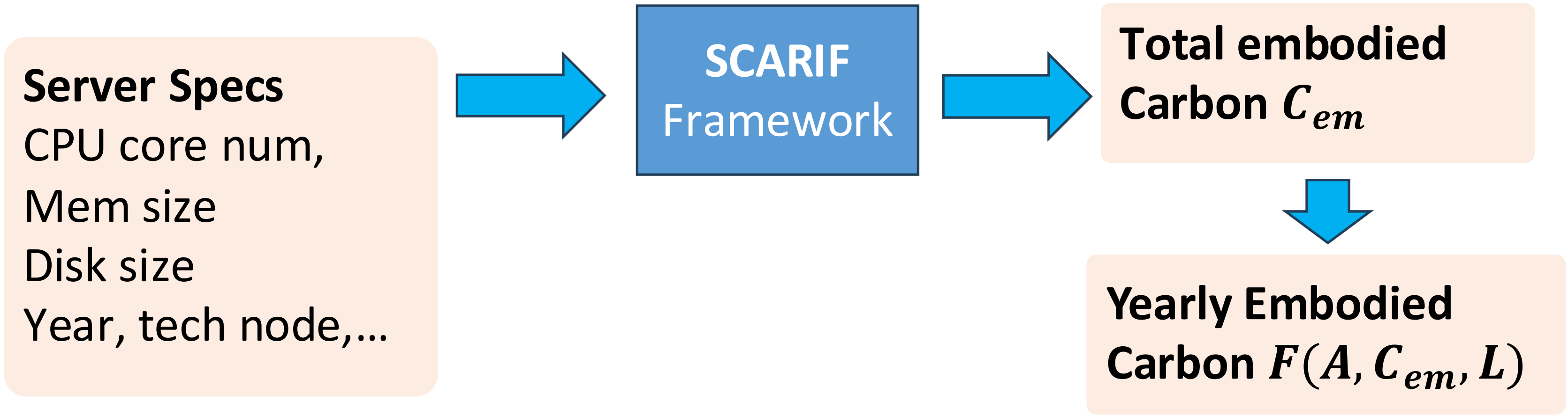
Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification



https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

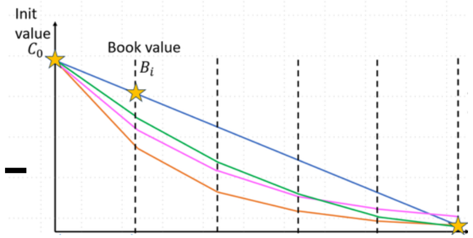
Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}

Yearly Embodied Carbon $F(A, C_{em}, L)$



Depreciation Models

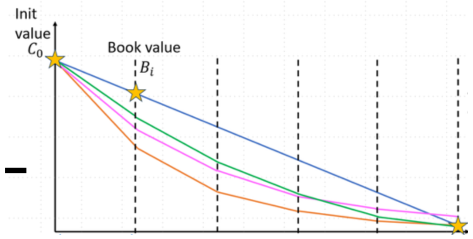
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

Per task embodied Carbon
$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

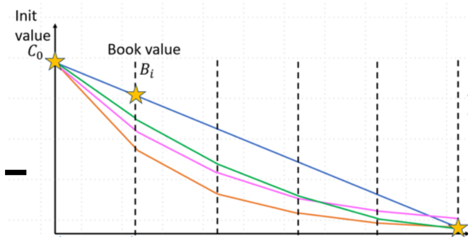
Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

App. Specs
Application
Target throughput

Grid Specs
CI_a, CI_i

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

Per task embodied Carbon
$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

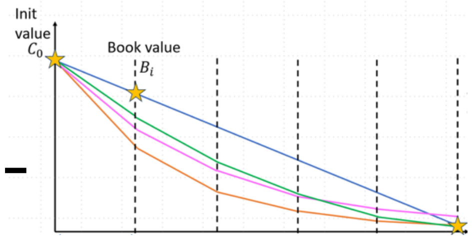
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

Per task embodied Carbon
$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

App. Specs
Application
Target throughput

Nvidia Triton

Grid Specs
CI_a, CI_i

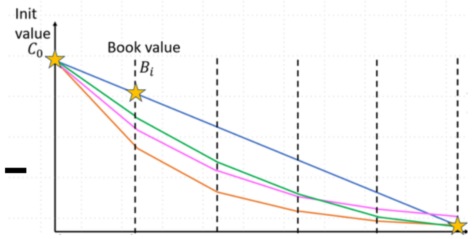
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

Per task embodied Carbon

$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

App. Specs
Application
Target throughput

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Grid Specs
CI_a, CI_i

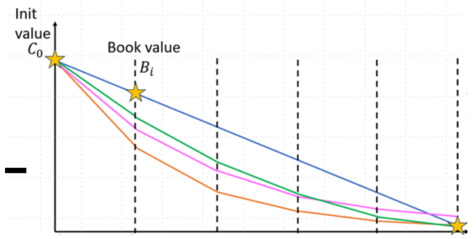
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

Per task embodied Carbon

$$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$$

App. Specs
Application
Target throughput

Server Utilization
U

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Grid Specs
CI_a, CI_i

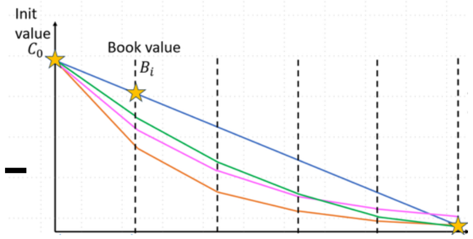
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Nvidia Triton

Server Utilization
U

t_{task}
Max throughput
P_a, P_i

Per task embodied Carbon
$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

Grid Specs
CI_a, CI_i

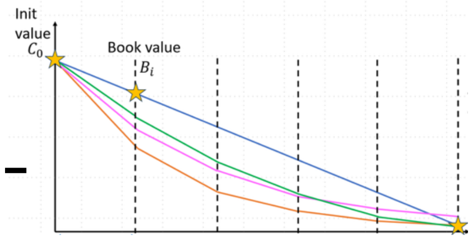
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Server Utilization
U

Per task embodied Carbon

$$= \boxed{\frac{t_{task}}{t_{year} \cdot U} \cdot F}$$

Secondary Carbon included
Utilization aware

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Grid Specs
CI_a, CI_i

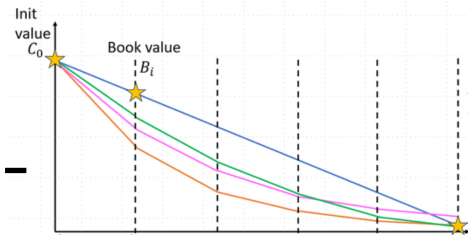
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Server Utilization
U

Per task embodied Carbon
$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$

Secondary Carbon included
Utilization aware

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Grid Specs
CI_a, CI_i

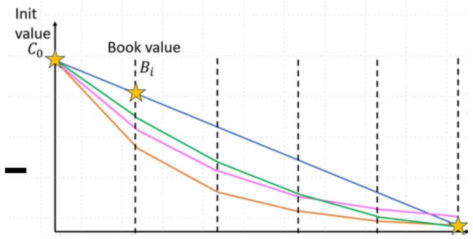
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Server Utilization
U

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Per task embodied Carbon
$$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$$

Secondary Carbon included
Utilization aware

Per task Operational Carbon
$$= P_a \cdot t_{task} \cdot CI_a + P_i \cdot t_{task} \cdot \frac{1-U}{U} \cdot CI_i$$

Grid Specs
CI_a, CI_i

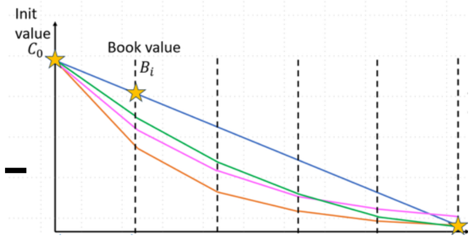
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Server Utilization
U

Nvidia Triton

t_{task}
Max throughput
P_a, P_i

Per task embodied Carbon

$= \boxed{\frac{t_{task}}{t_{year} \cdot U}} \cdot F$

Secondary Carbon included
Utilization aware

Per task Operational Carbon

$= \boxed{P_a \cdot t_{task} \cdot CI_a} + P_i \cdot t_{task} \cdot \frac{1-U}{U} \cdot CI_i$

Carbon in active state

Grid Specs
CI_a, CI_i

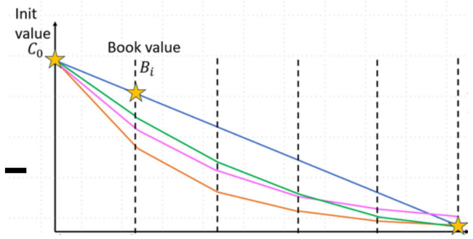
https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Apply Depreciation Model to Cost Quantification

Server Specs
CPU core num,
Mem size
Disk size
Year, tech node,…

SCARIF
Framework

Total embodied Carbon C_{em}



Depreciation Models

Yearly Embodied Carbon $F(A, C_{em}, L)$

App. Specs
Application
Target throughput

Nvidia Triton

Server Utilization
U

t_{task}
Max throughput
P_a, P_i

Grid Specs
CI_a, CI_i

Per task embodied Carbon
$$= \frac{t_{task}}{t_{year} \cdot U} \cdot F$$

Secondary Carbon included
Utilization aware

Per task Operational Carbon
$$= P_a \cdot t_{task} \cdot CI_a + P_i \cdot t_{task} \cdot \frac{1-U}{U} \cdot CI_i$$

Carbon in active state

Secondary carbon in idle state: amortize to every task

https://github.com/arc-research-lab/SCARIF
https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html

Implication of the Depreciation Models

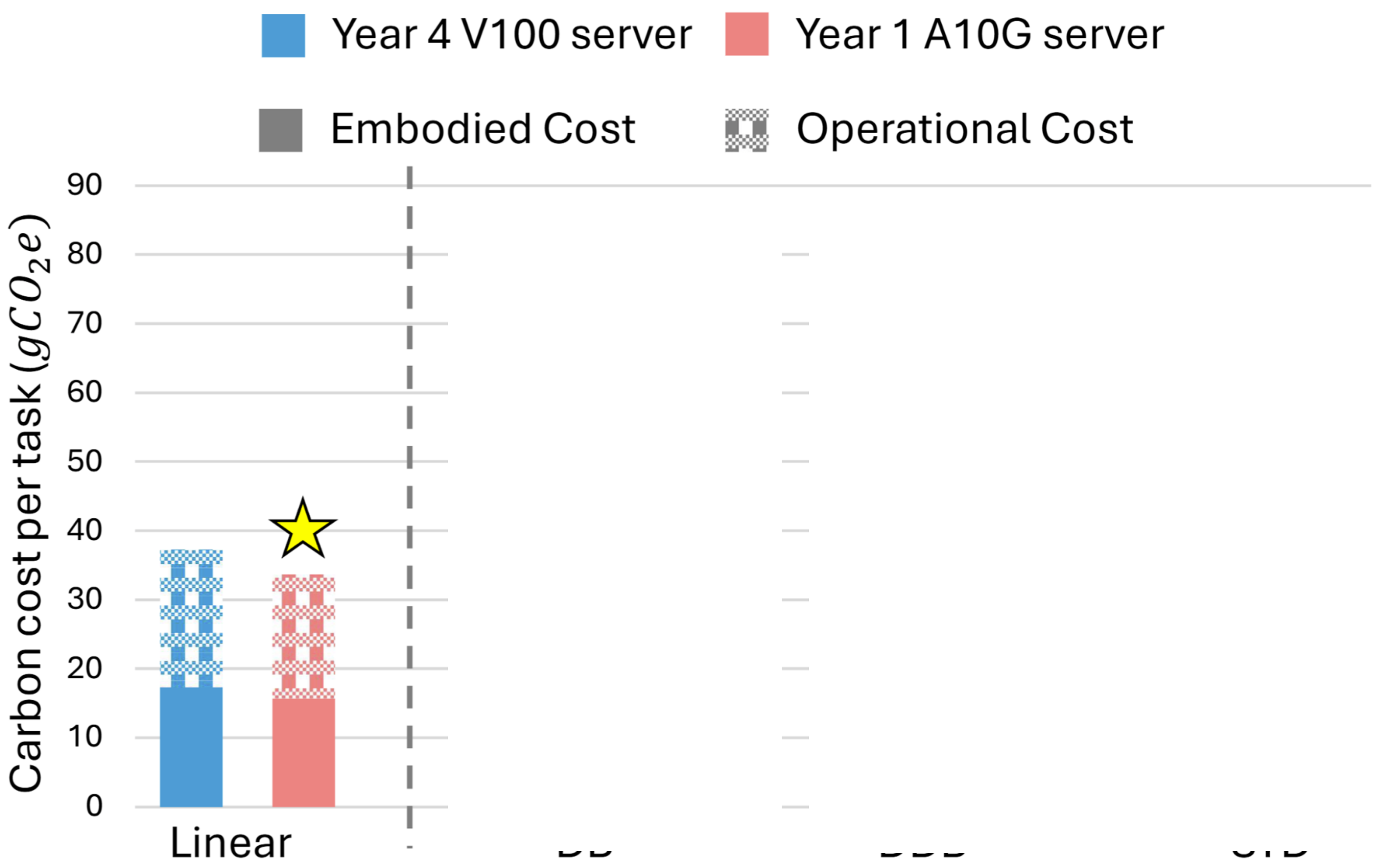
in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

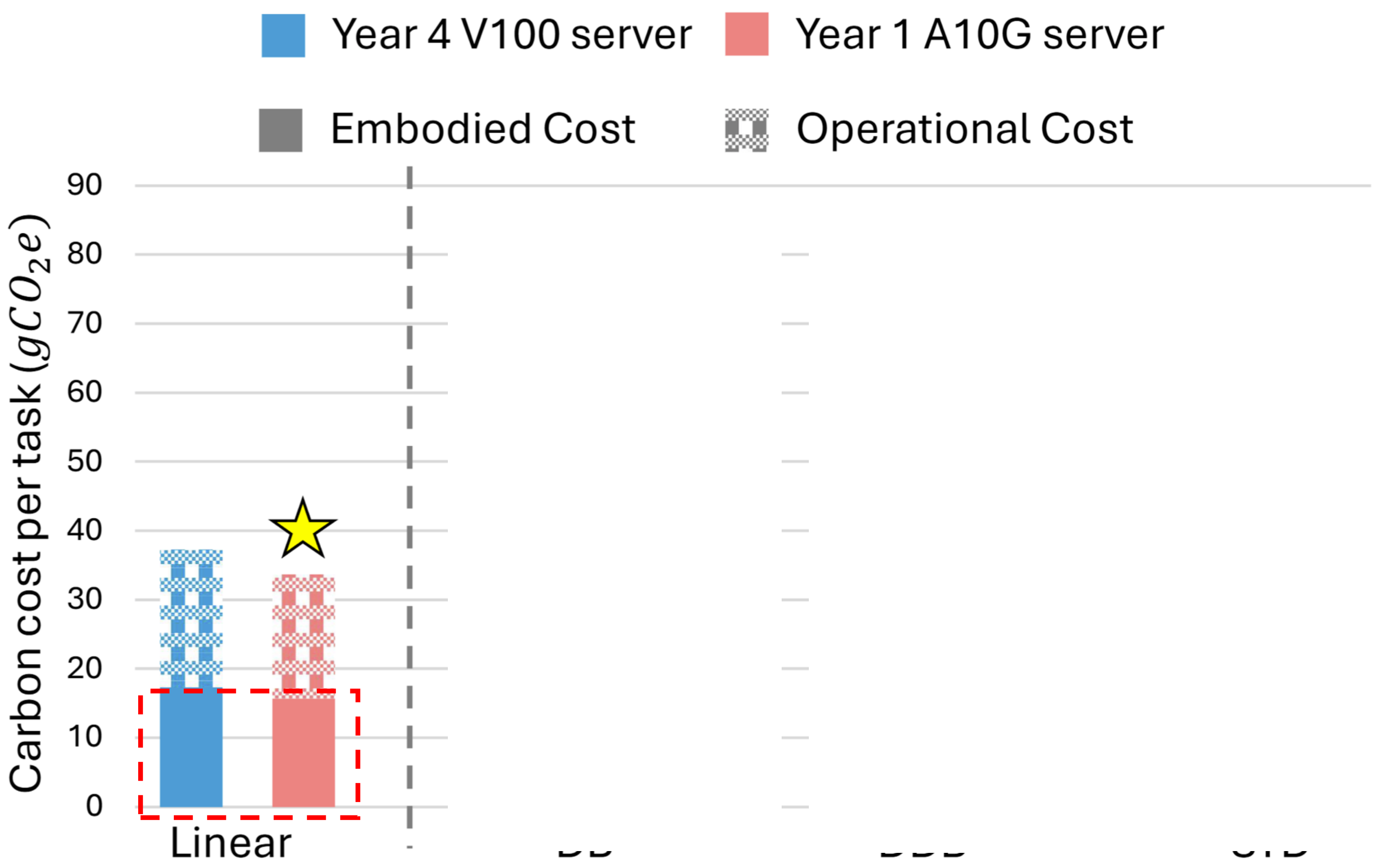
Set the target throughput at 200 infer/s



Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



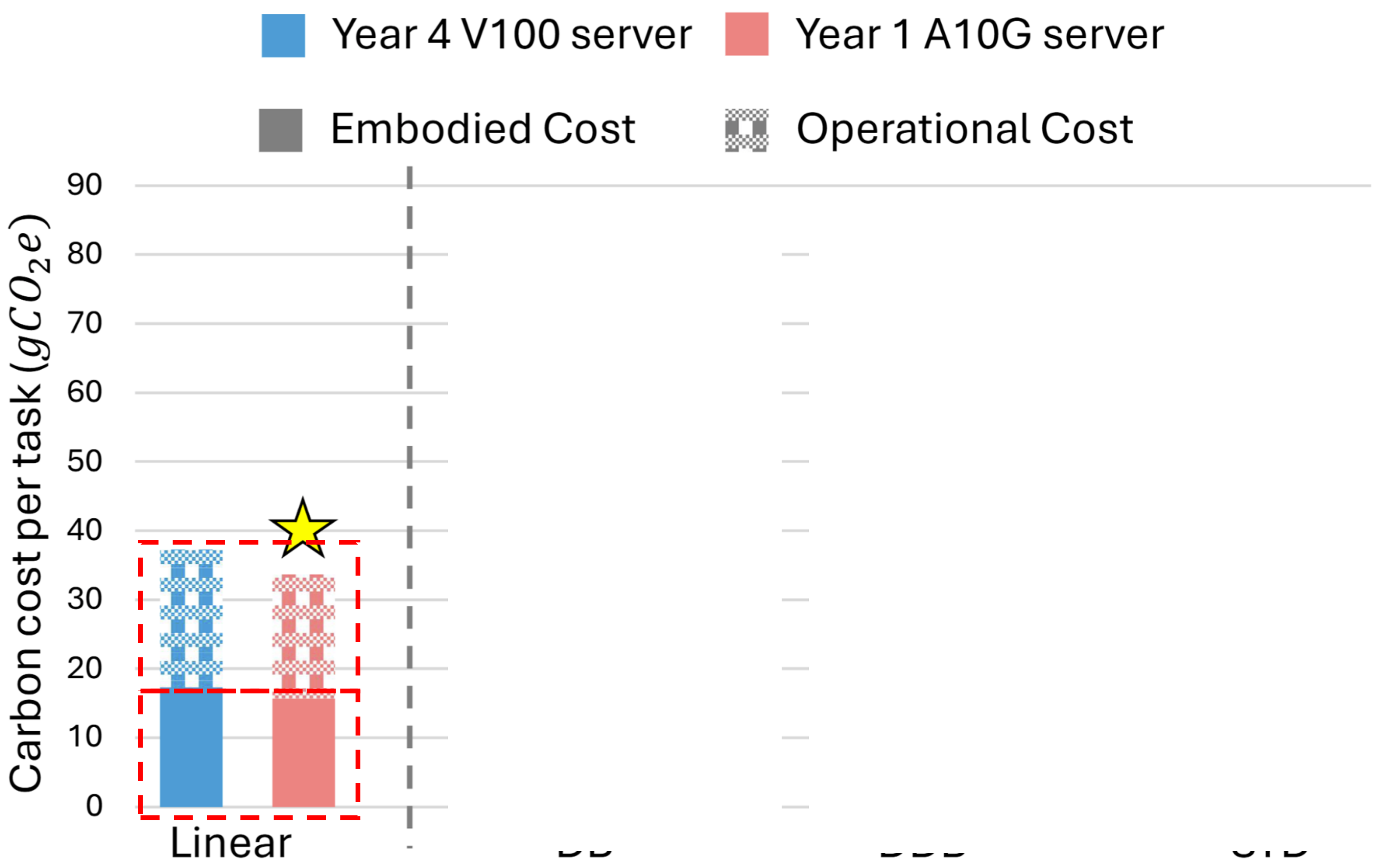
Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s

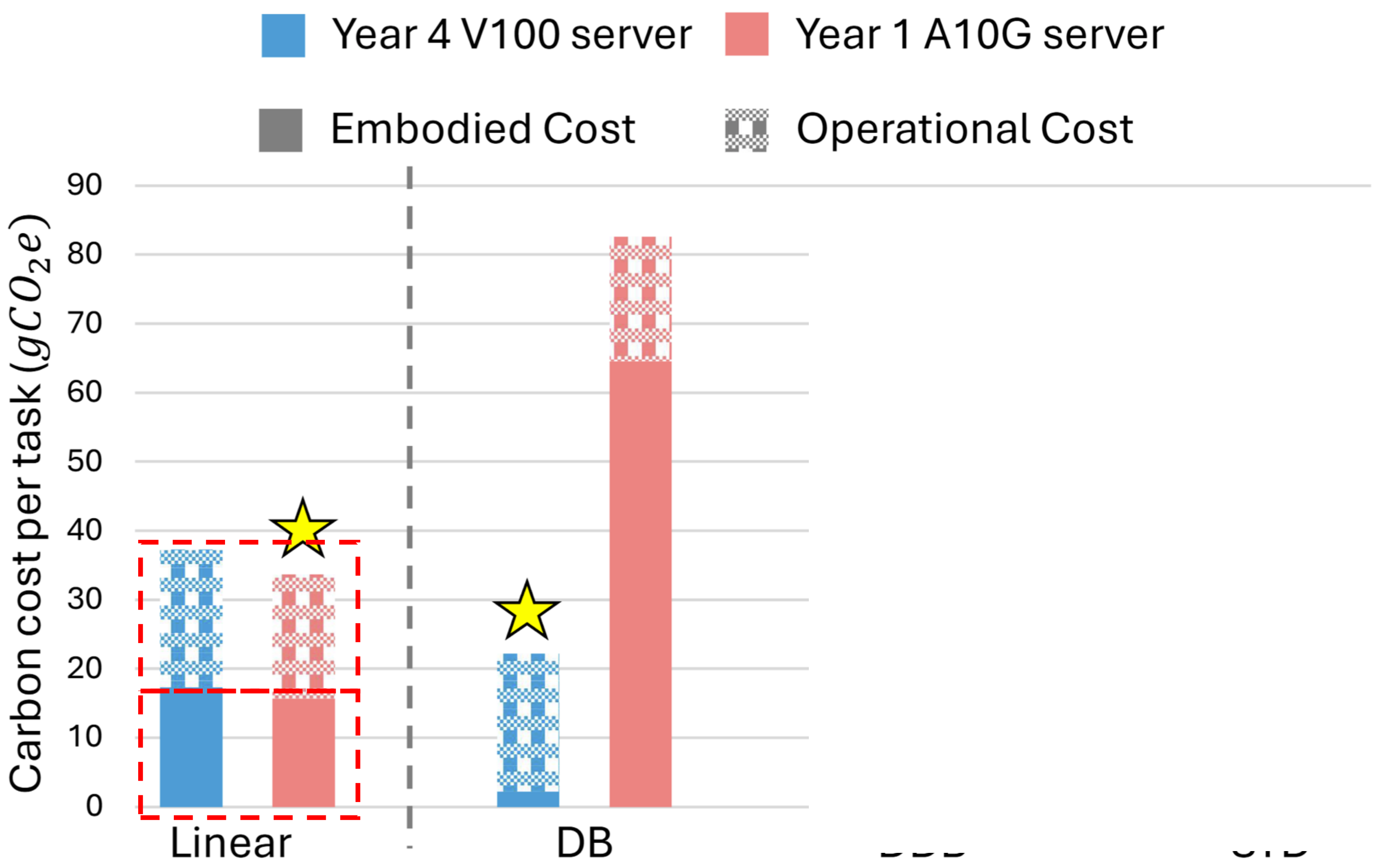
Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

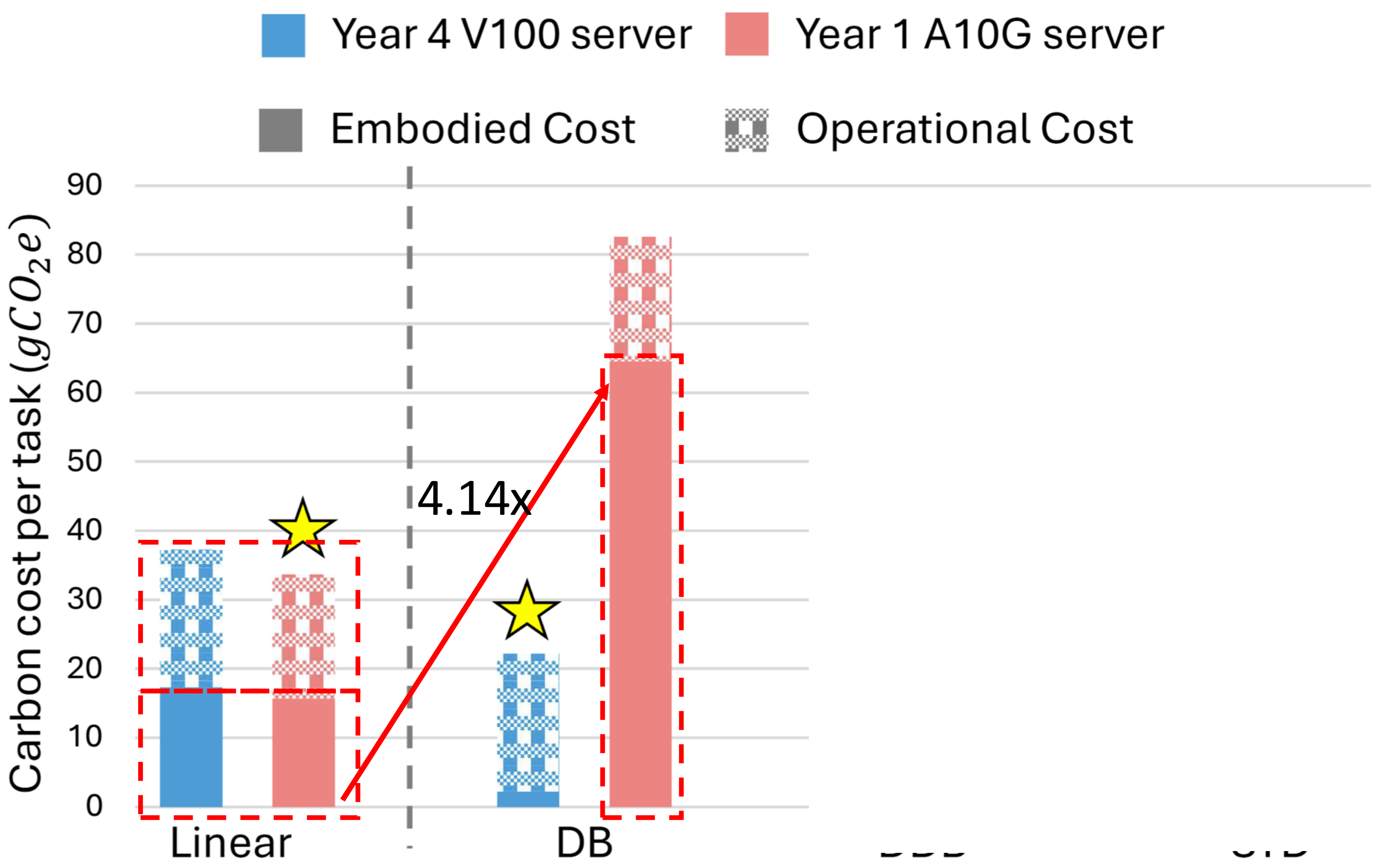
Non-linear depreciation Model:

- Highlight the importance of C_{em}
- New device: Large C_{em} assigned for 1st layer
- Old device: Enjoy low C_{em} in later years
- Other models have a similar flavor but are less aggressive in the earlier years

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

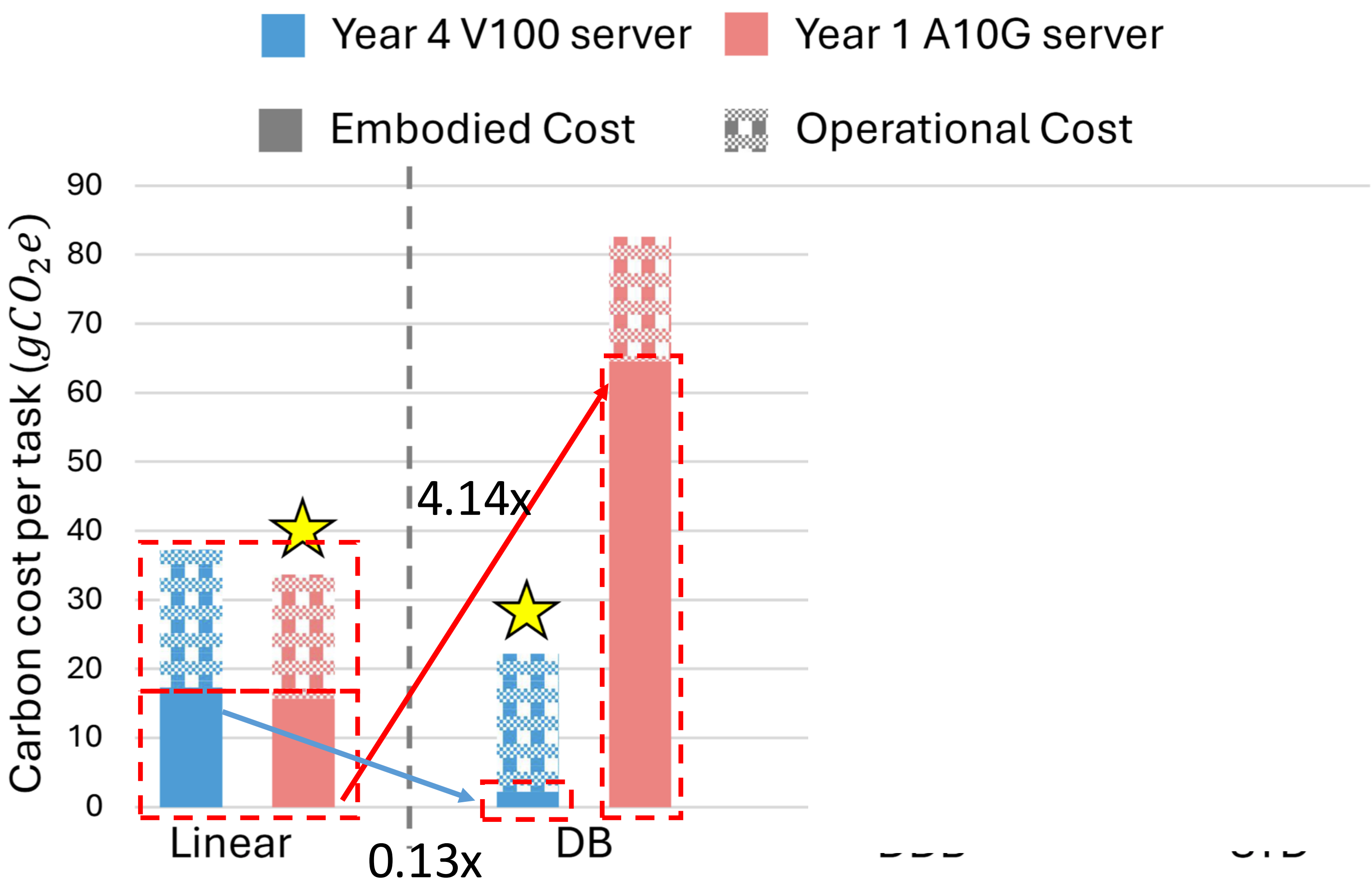
Non-linear depreciation Model:

- Highlight the importance of C_{em}
- New device: Large C_{em} assigned for 1st layer
- Old device: Enjoy low C_{em} in later years
- Other models have a similar flavor but are less aggressive in the earlier years

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

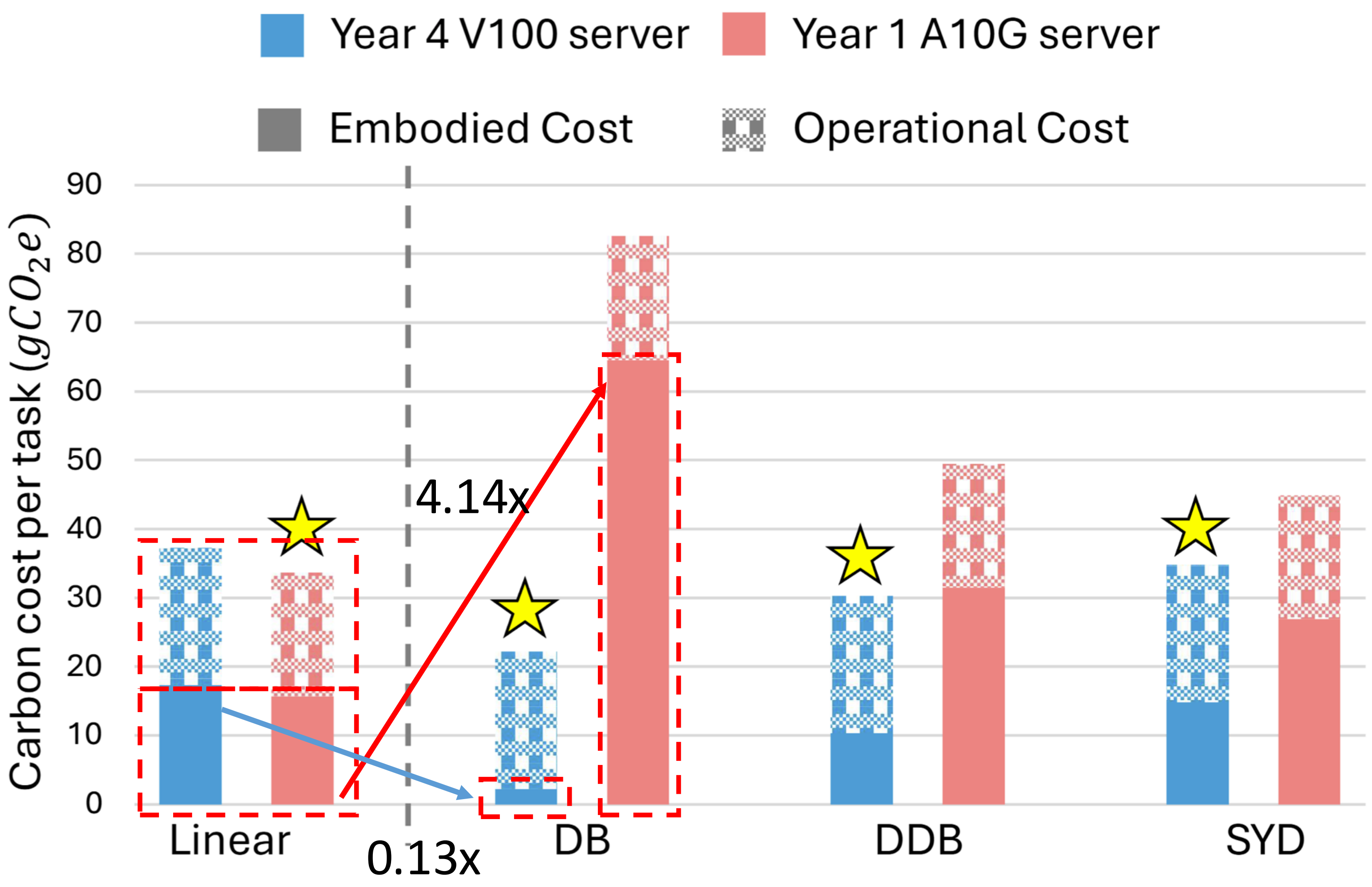
Non-linear depreciation Model:

- Highlight the importance of C_{em}
- New device: Large C_{em} assigned for 1st layer
- Old device: Enjoy low C_{em} in later years
- Other models have a similar flavor but are less aggressive in the earlier years

Implication of the Depreciation Models

in 2020, consider replacing a V100 server with an A10G server (ResNet 50 as the application)

Set the target throughput at 200 infer/s



Linear Model:

- attaches importance to C_{op}
- prefers faster and newer devices.
- incompatible w/t reusing principle

Non-linear depreciation Model:

- Highlight the importance of C_{em}
- New device: Large C_{em} assigned for 1st layer
- Old device: Enjoy low C_{em} in later years
- Other models have a similar flavor but are less aggressive in the earlier years

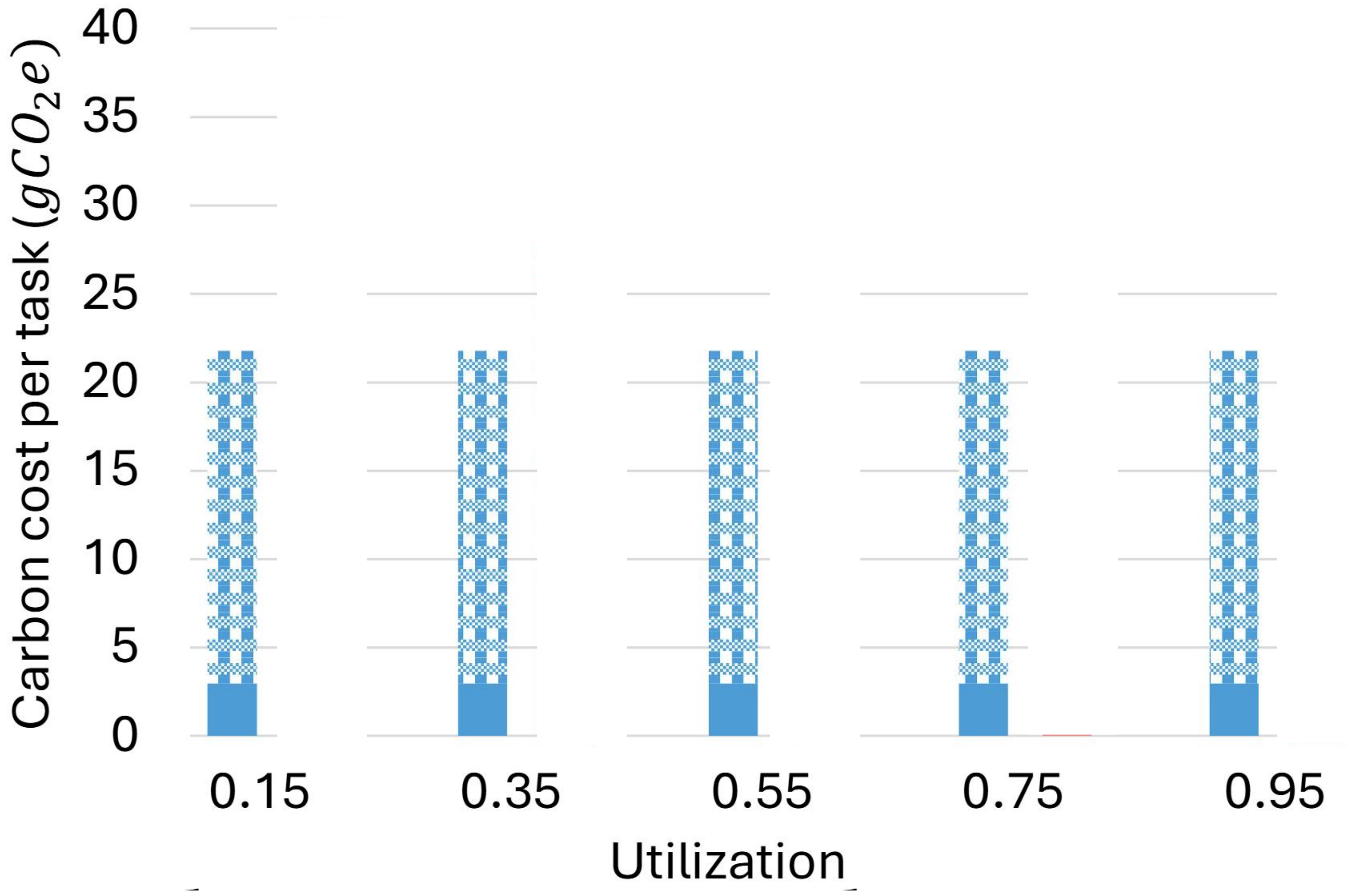
😀 **Reuse is encouraged**

Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations

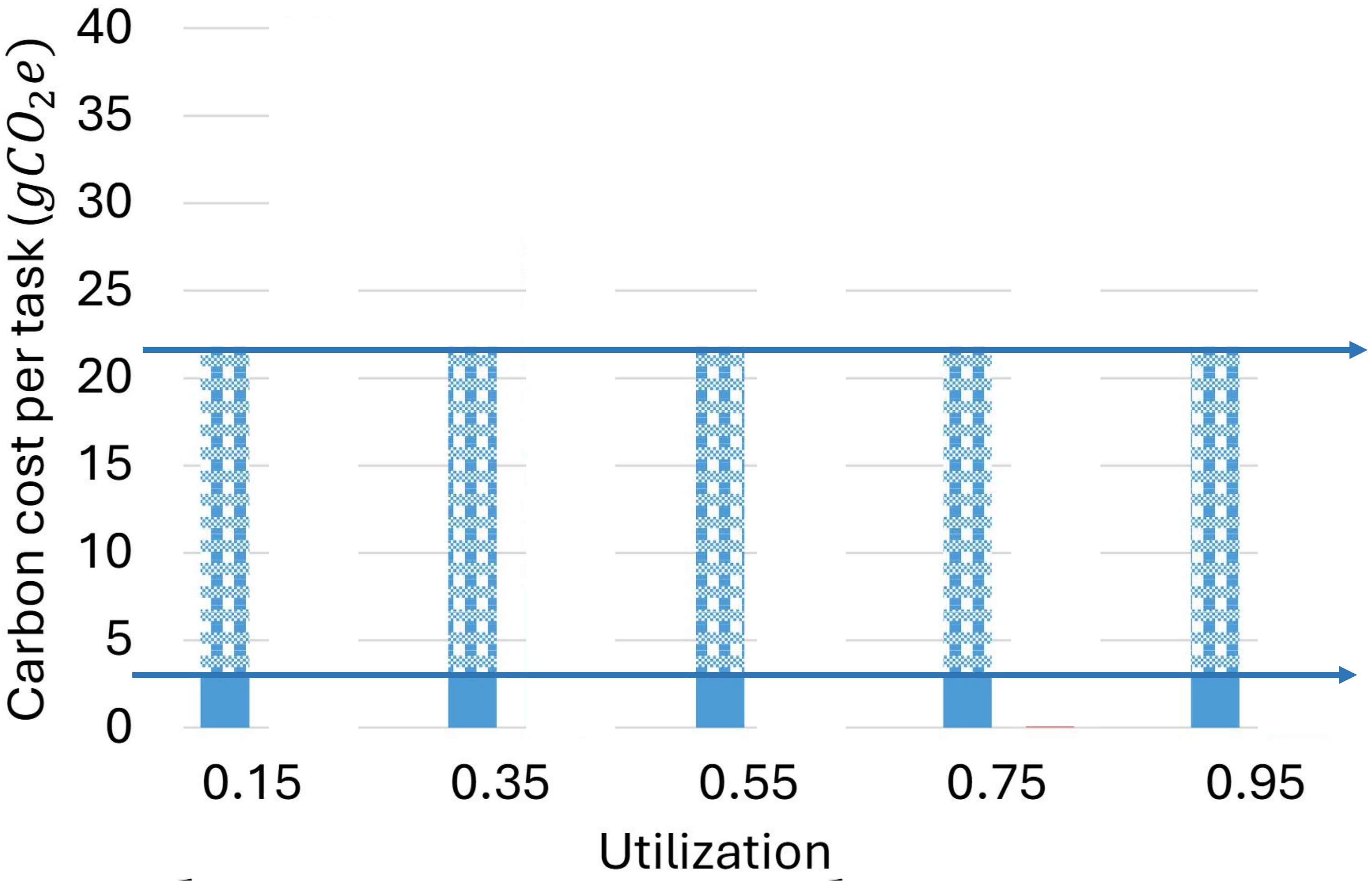
Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations



Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations

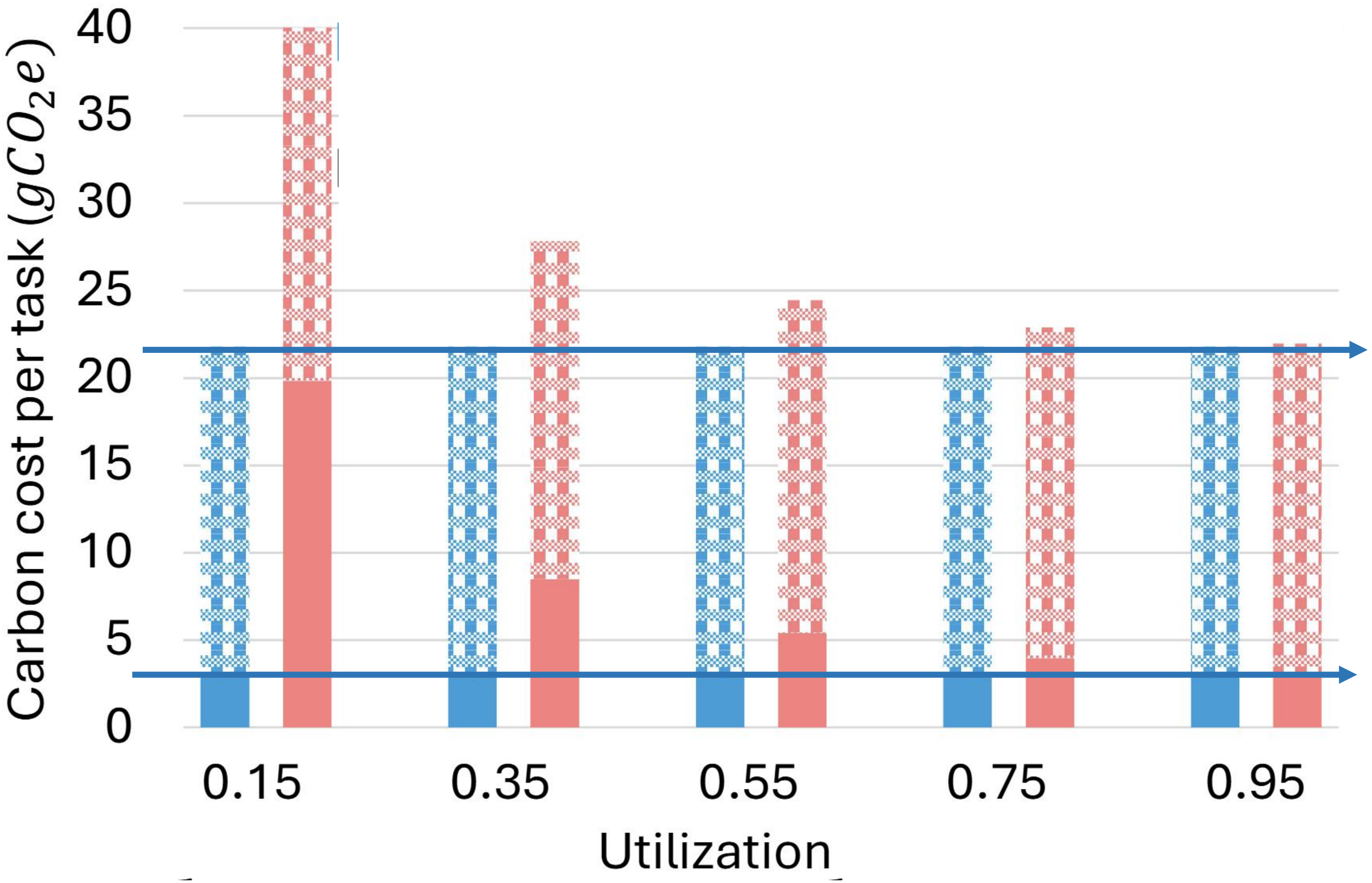


W/o Secondary carbon recovery

- Incomplete coverage of overall carbon cost
- Unaware of server utilization

Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations

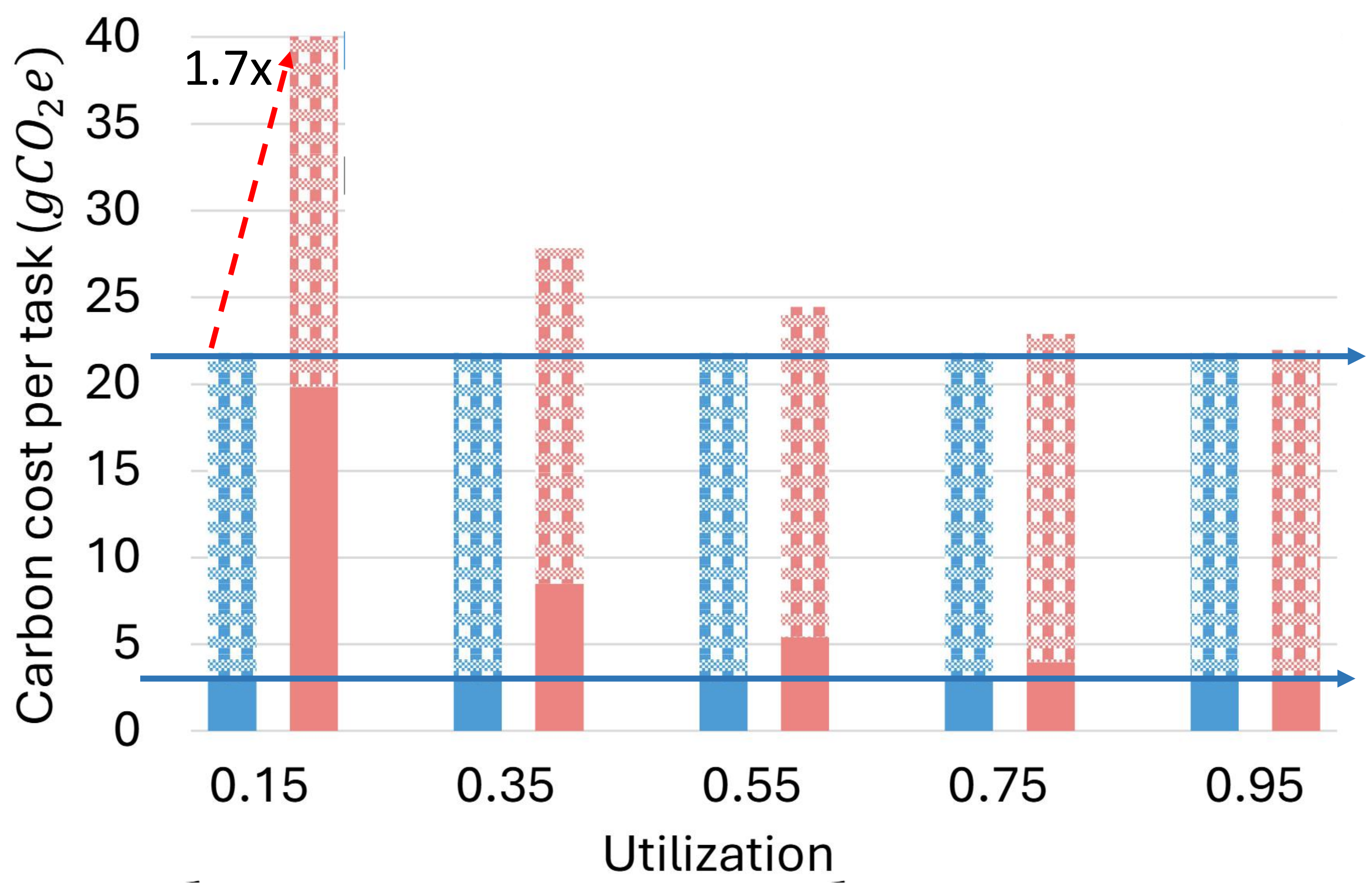


W/o Secondary carbon recovery

- Incomplete coverage of overall carbon cost
- Unaware of server utilization

Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations



W/o Secondary carbon recovery

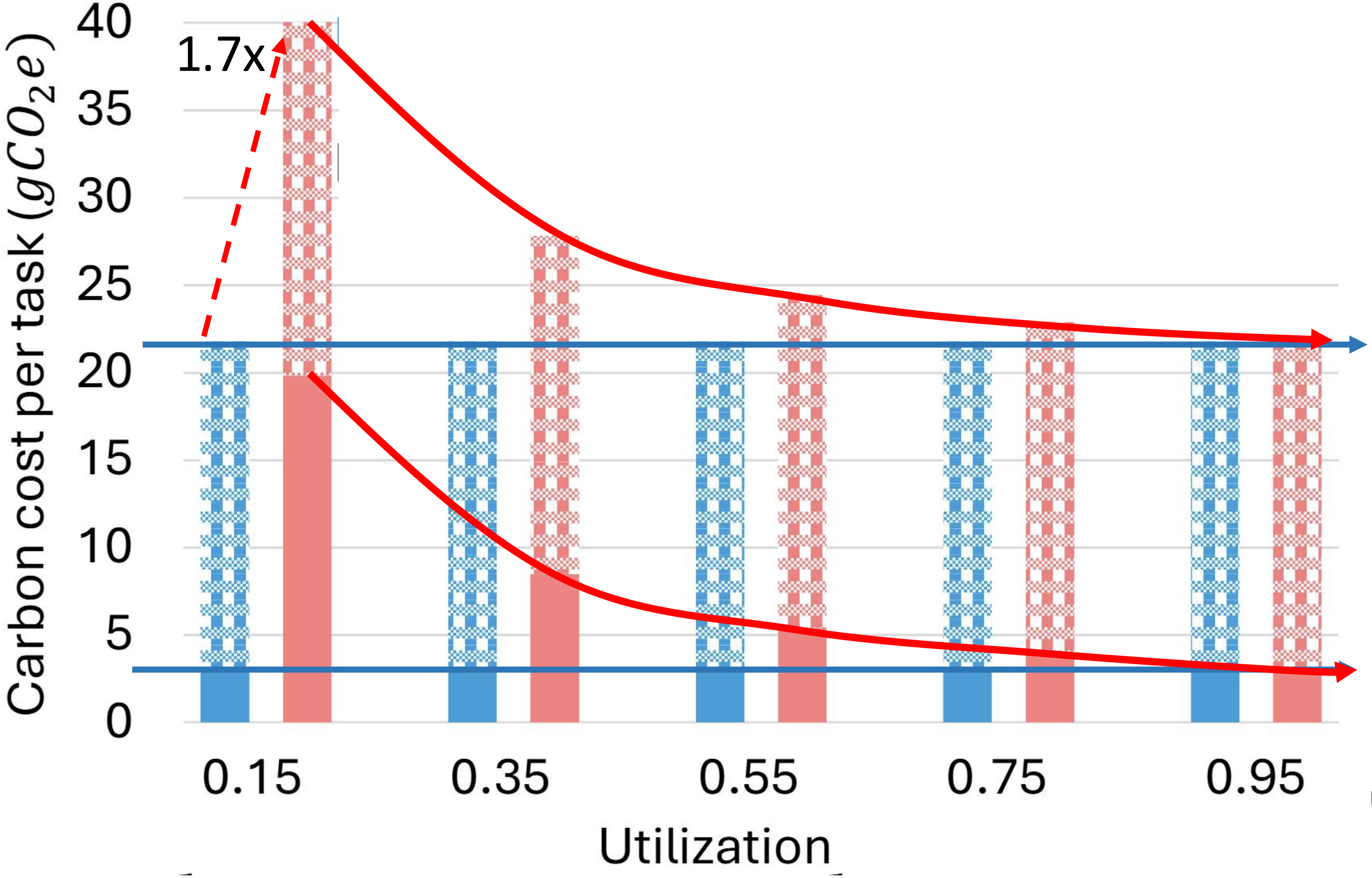
- Incomplete coverage of overall carbon cost
- Unaware of server utilization

W/t Secondary carbon recovery

- Covers the cost at idle time
- Low Util: larger amount of idle carbon, fewer tasks to amortize
- High Util: smaller amount of idle carbon cost, more tasks

Implication of the Secondary Carbon Recovery

In 2020, a year 4 V100 server running on different Utilizations



W/o Secondary carbon recovery

- Incomplete coverage of overall carbon cost
- Unaware of server utilization

W/t Secondary carbon recovery

- Covers the cost at idle time
- Low Util: larger amount of idle carbon, fewer tasks to amortize
- High Util: smaller amount of idle carbon cost, more tasks

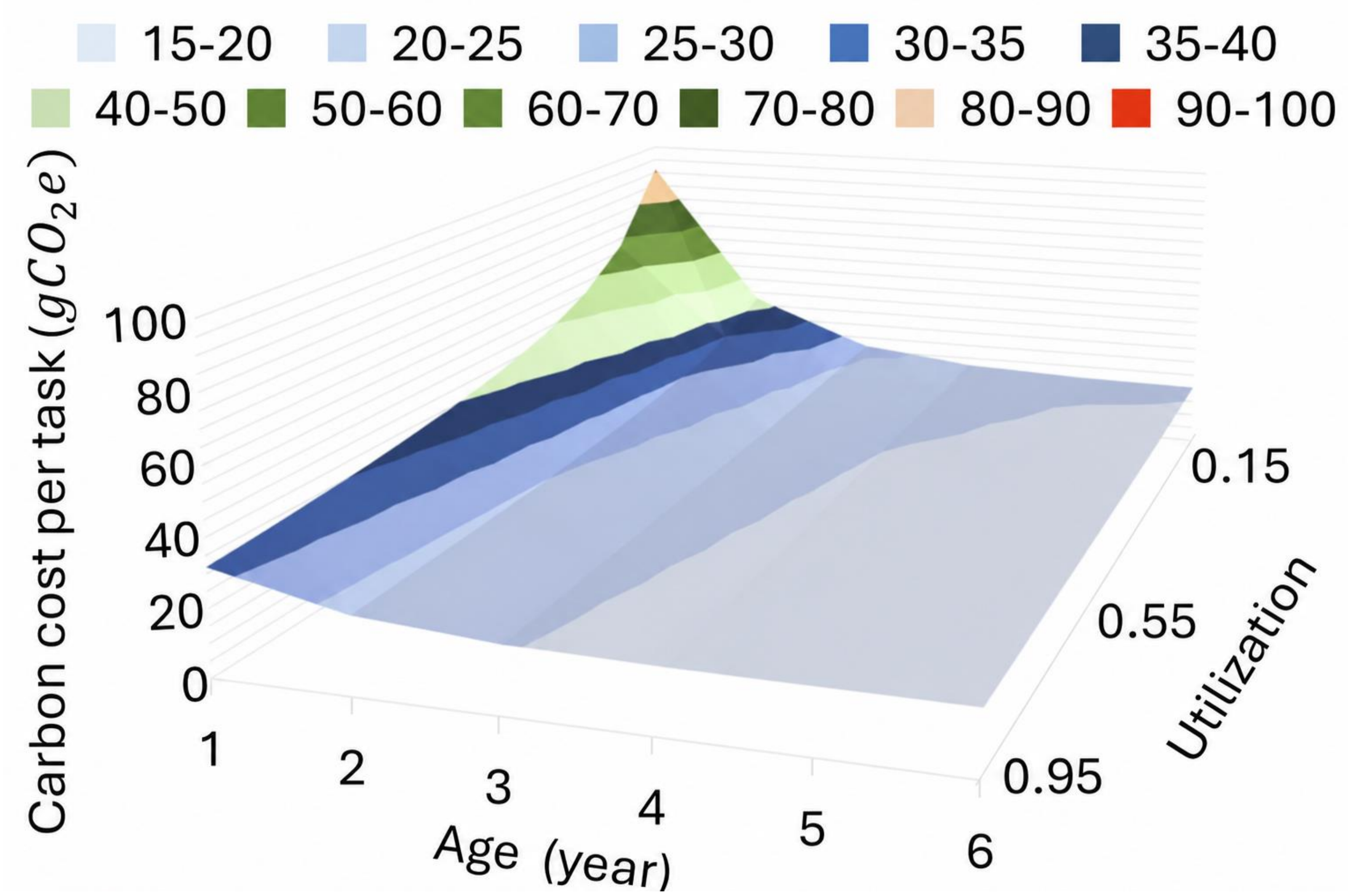
😀 **Keeping high util is promoted**

Combing them together

Per-task carbon cost on V100 GPUs on different utilizations and ages

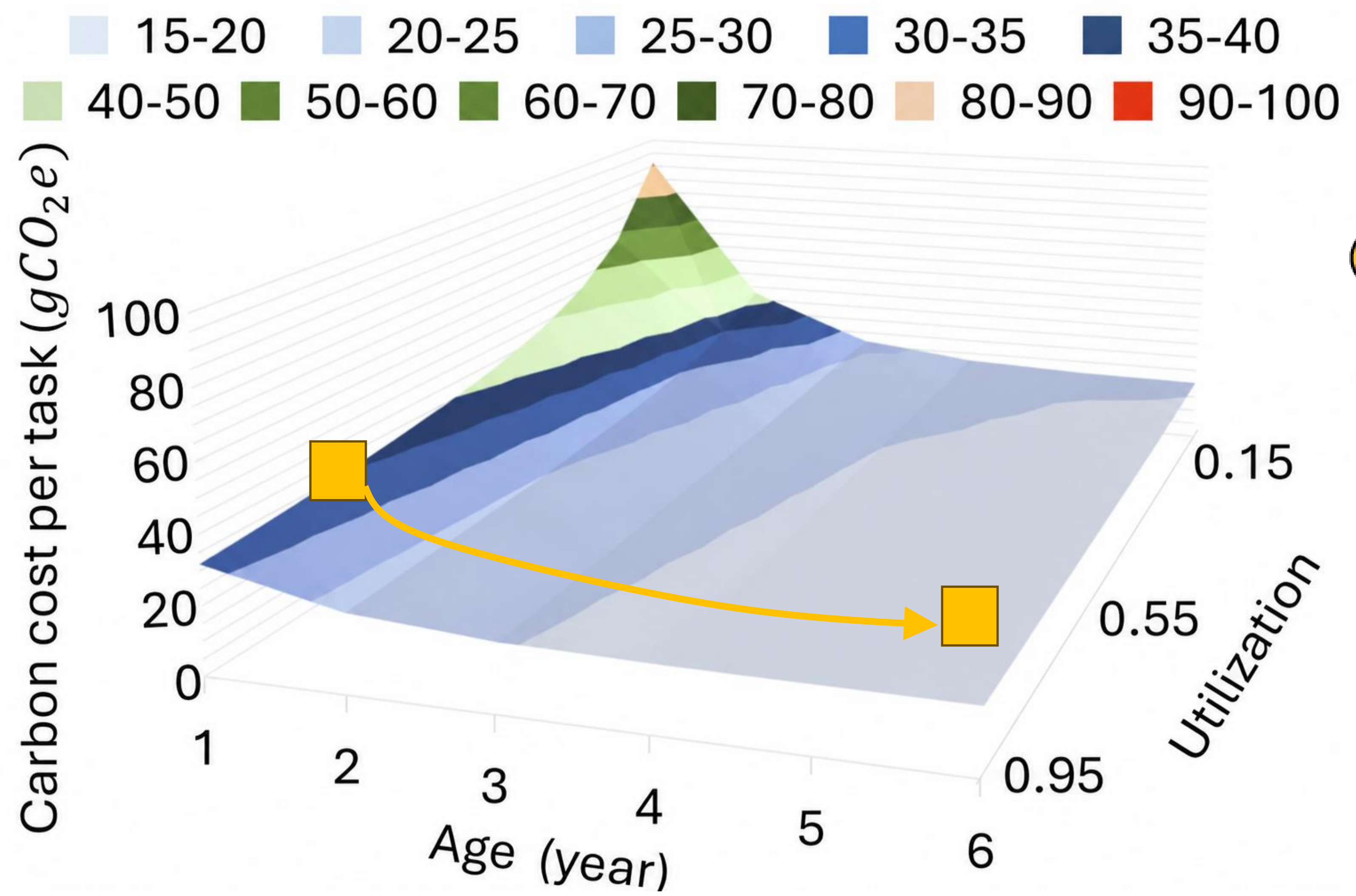
Combing them together

Per-task carbon cost on V100 GPUs on different utilizations and ages



Combing them together

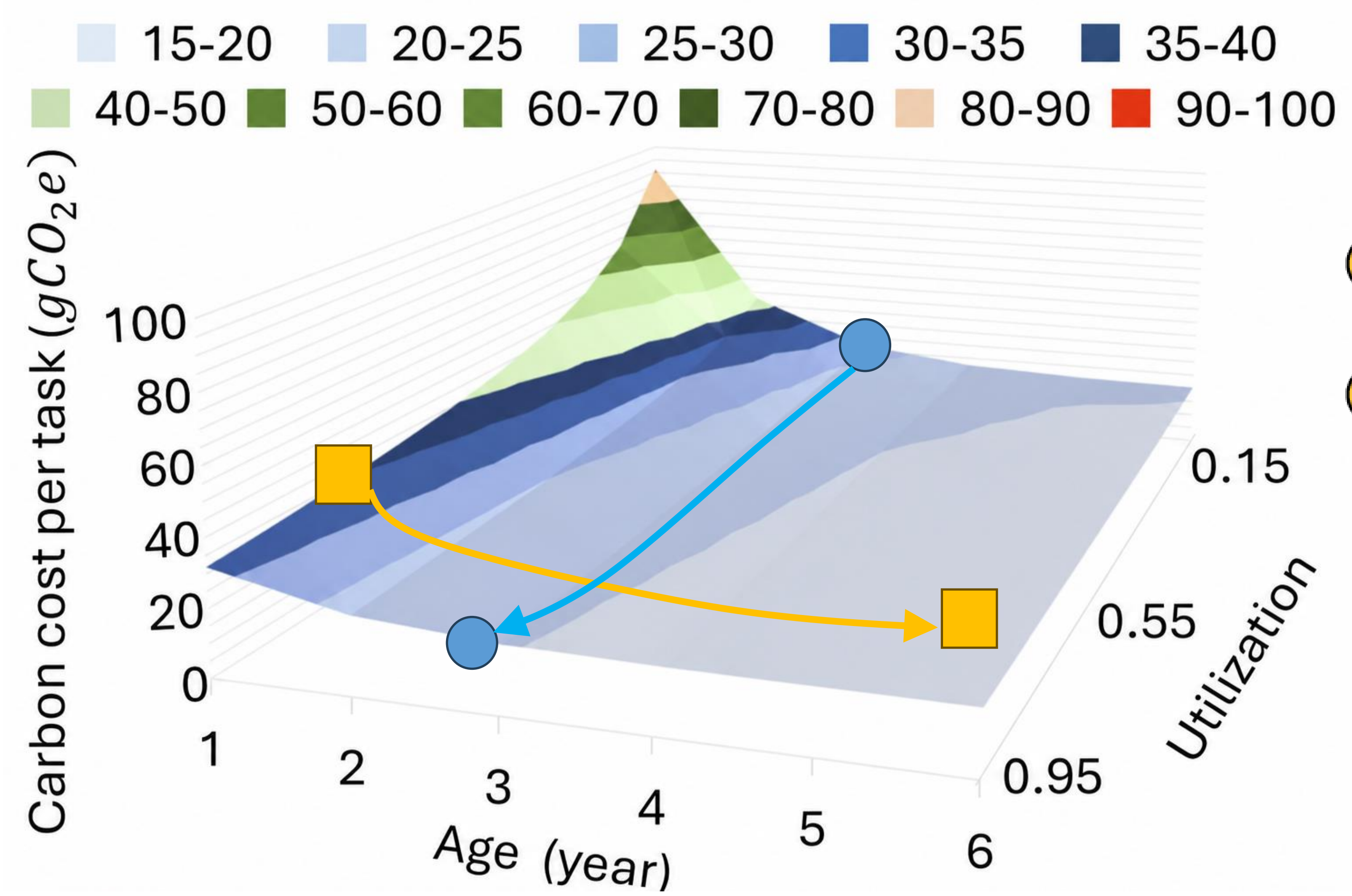
Per-task carbon cost on V100 GPUs on different utilizations and ages



😀 **Reuse is encouraged**

Combing them together

Per-task carbon cost on V100 GPUs on different utilizations and ages

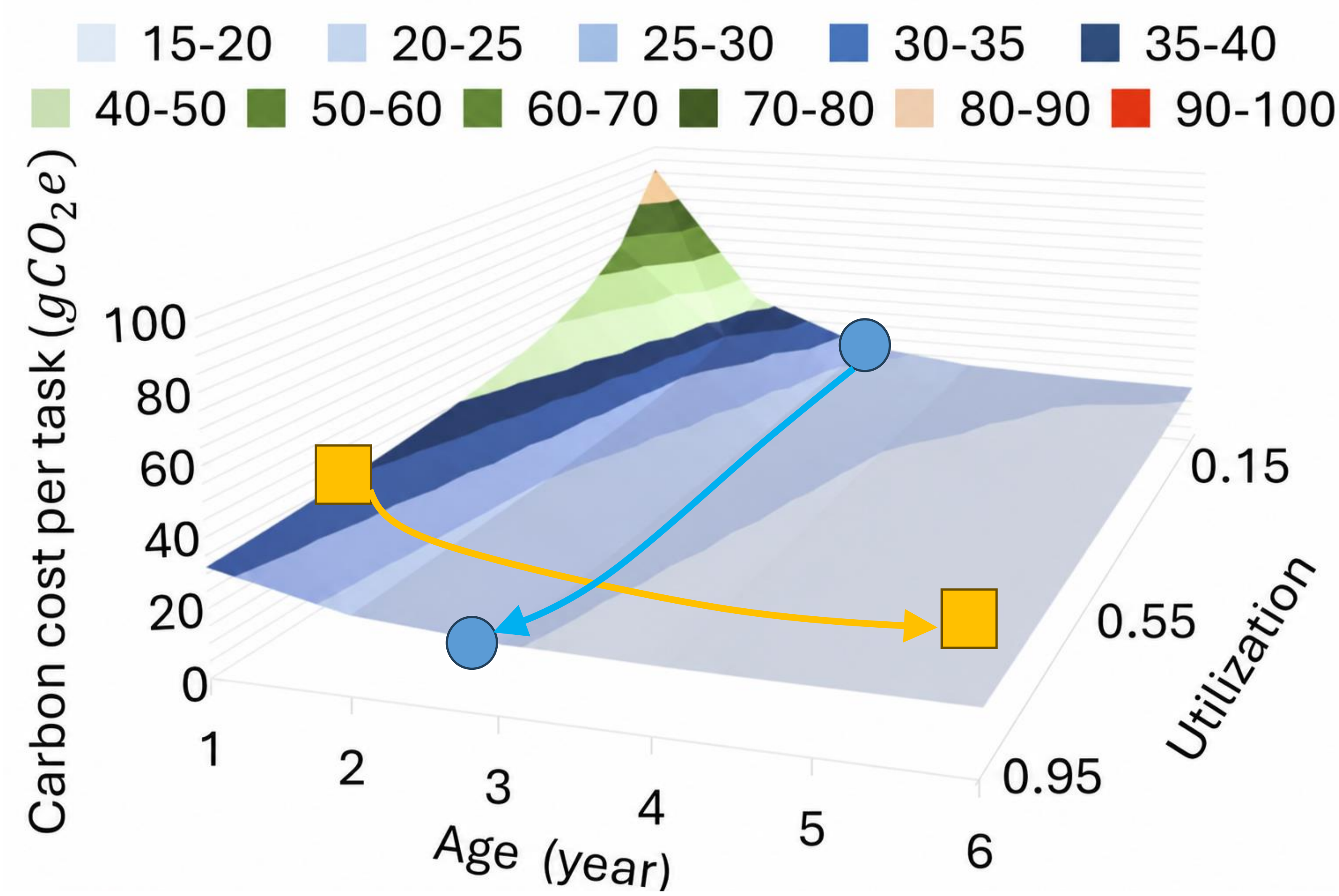


😀 **Reuse is encouraged**

😀 **Keeping high util is promoted**

Combing them together

Per-task carbon cost on V100 GPUs on different utilizations and ages



Reuse is encouraged

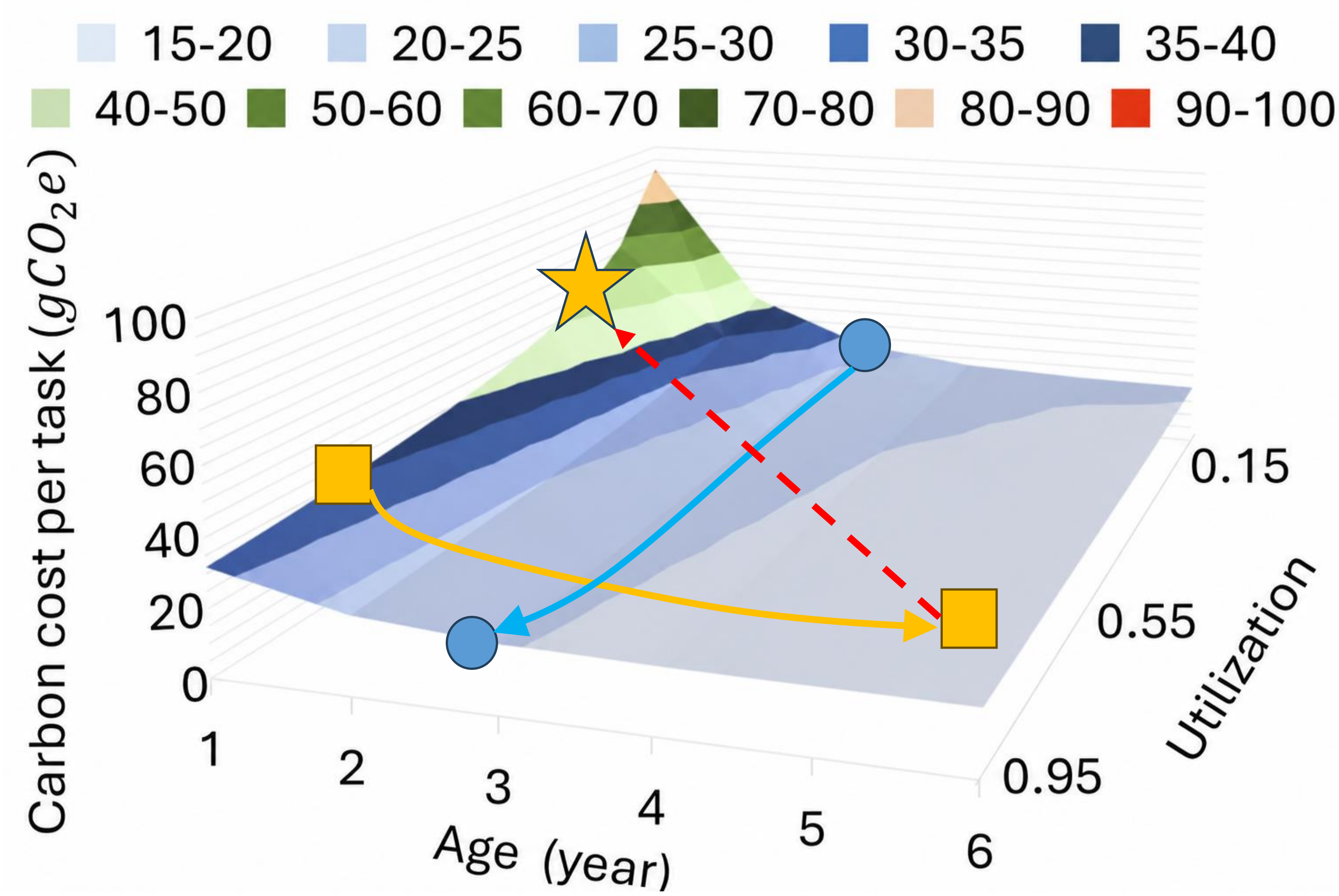
Keeping high util is promoted

Over-provisioning is punished

- Replaces usable old devices with new ones
- Lowering down total utilization

Combing them together

Per-task carbon cost on V100 GPUs on different utilizations and ages



😀 **Reuse is encouraged**

😀 **Keeping high util is promoted**

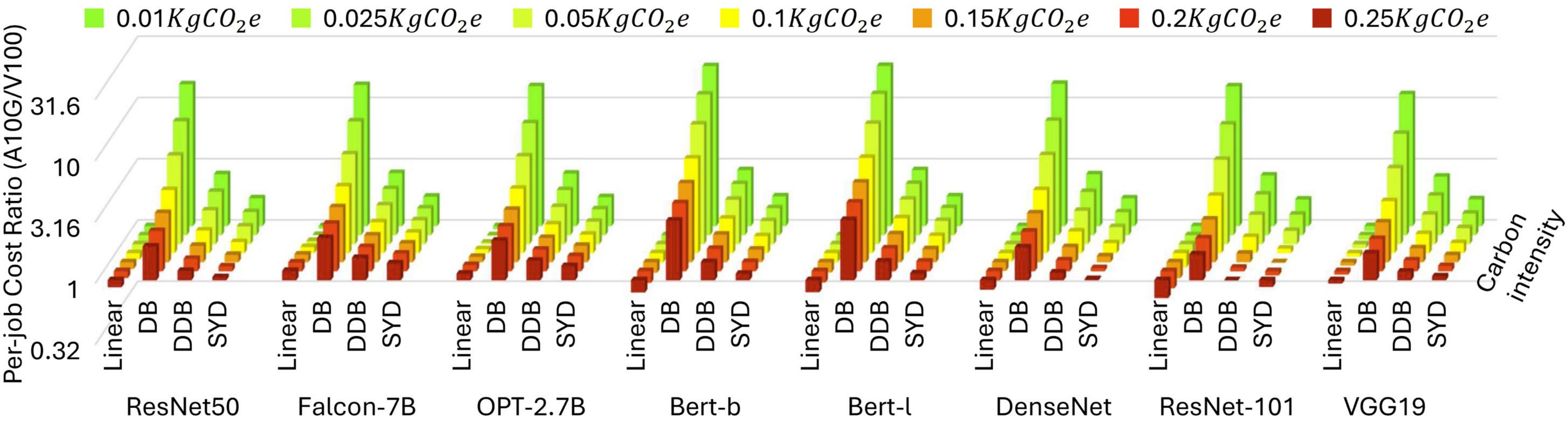
😀 **Over-provisioning is punished**

- Replaces usable old devices with new ones
- Lowering down total utilization

Sensitivity Study

Different Applications and Carbon intensities could affect design choices

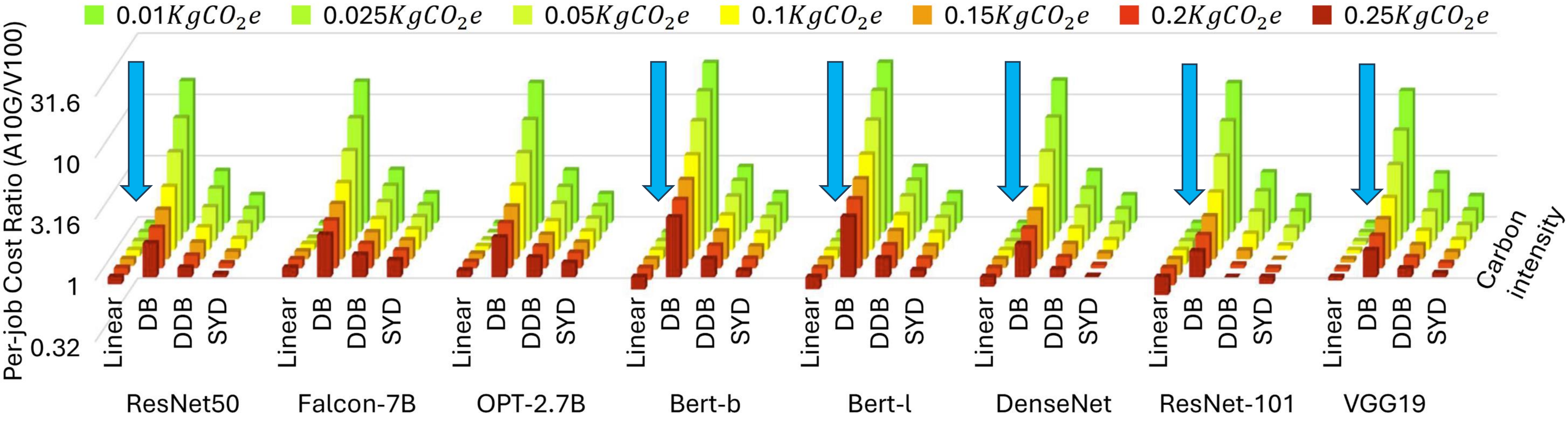
Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



Sensitivity Study

Different Applications and Carbon intensities could affect design choices

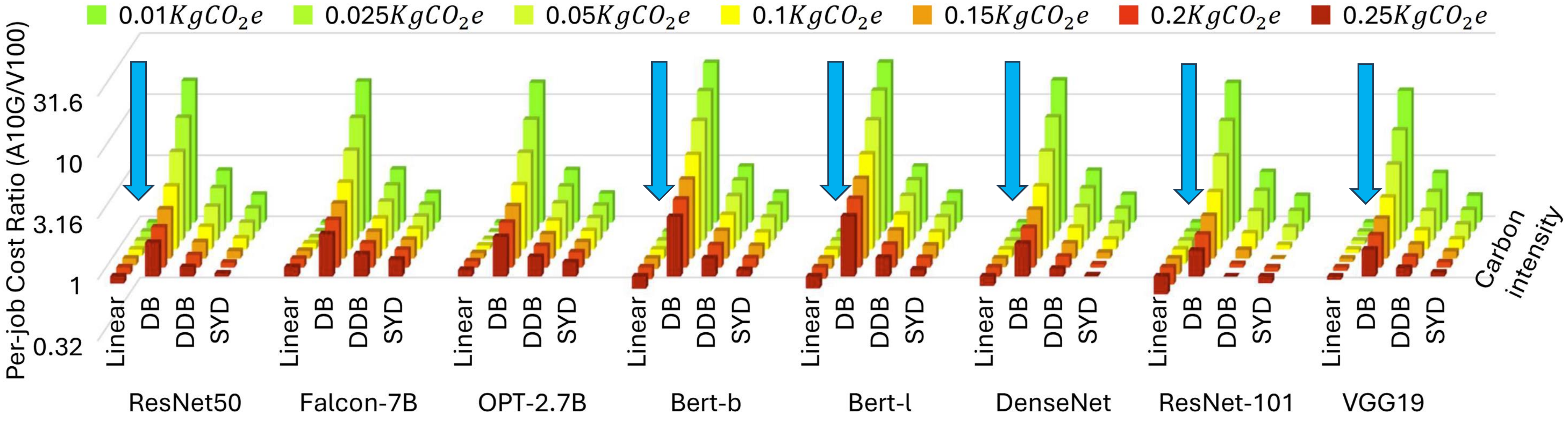
Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



Sensitivity Study

Different Applications and Carbon intensities could affect design choices

Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



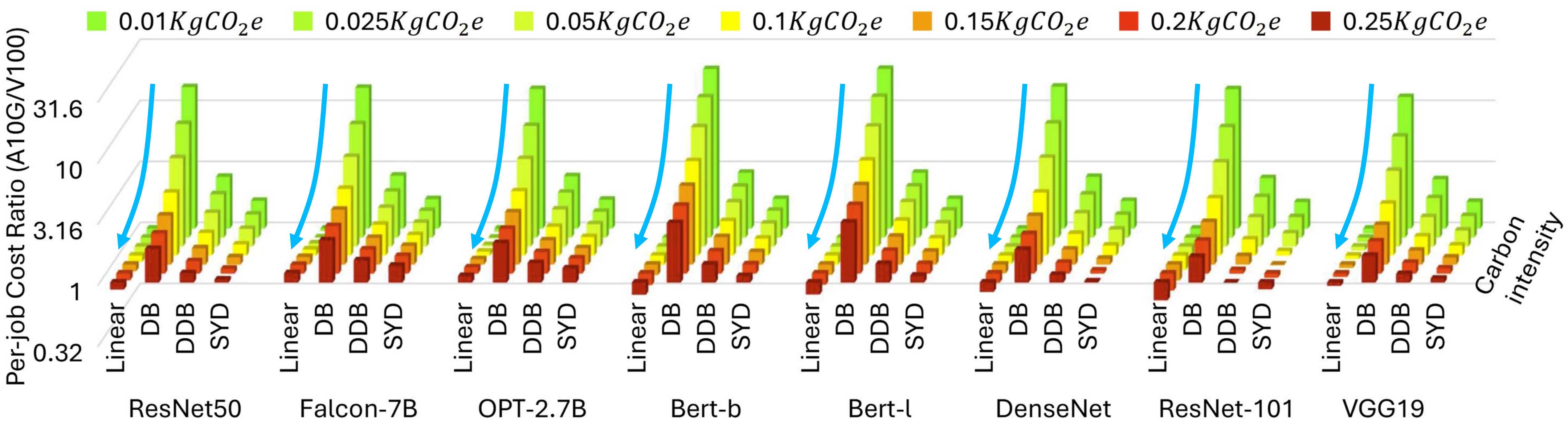
Under linear depreciation model:

- A10G is better in most of the applications
- Higher CI cast importance on the operational carbon → even more towards A10G

Sensitivity Study

Different Applications and Carbon intensities could affect design choices

Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



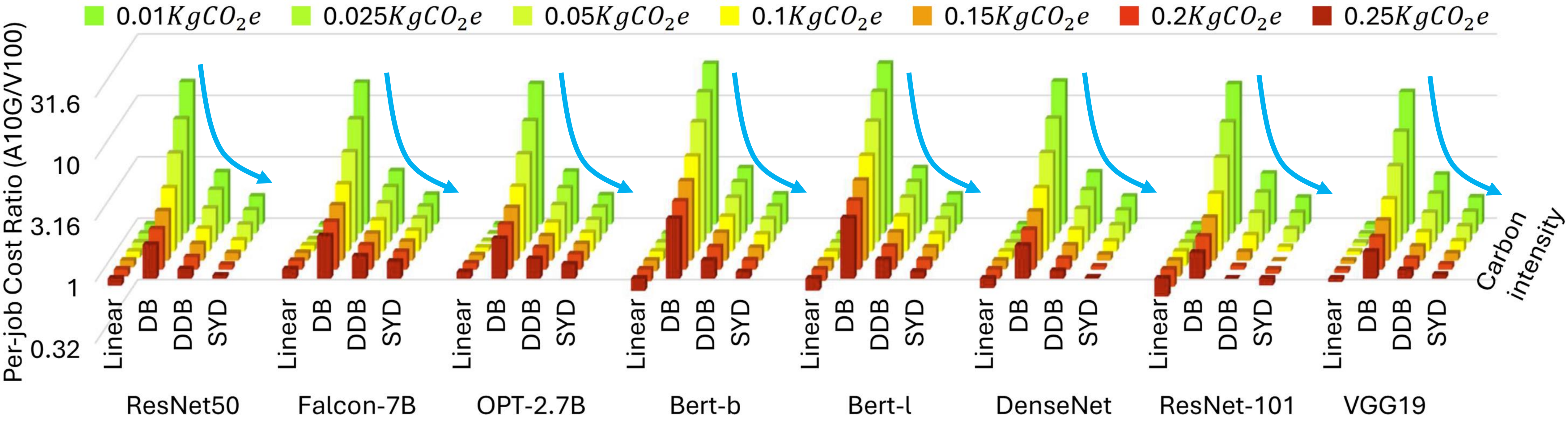
Under non-linear depreciation model:

- The DB models flips the design choices in all applications
- The higher the CI, the smaller the gain for V100 over A10G

Sensitivity Study

Different Applications and Carbon intensities could affect design choices

Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



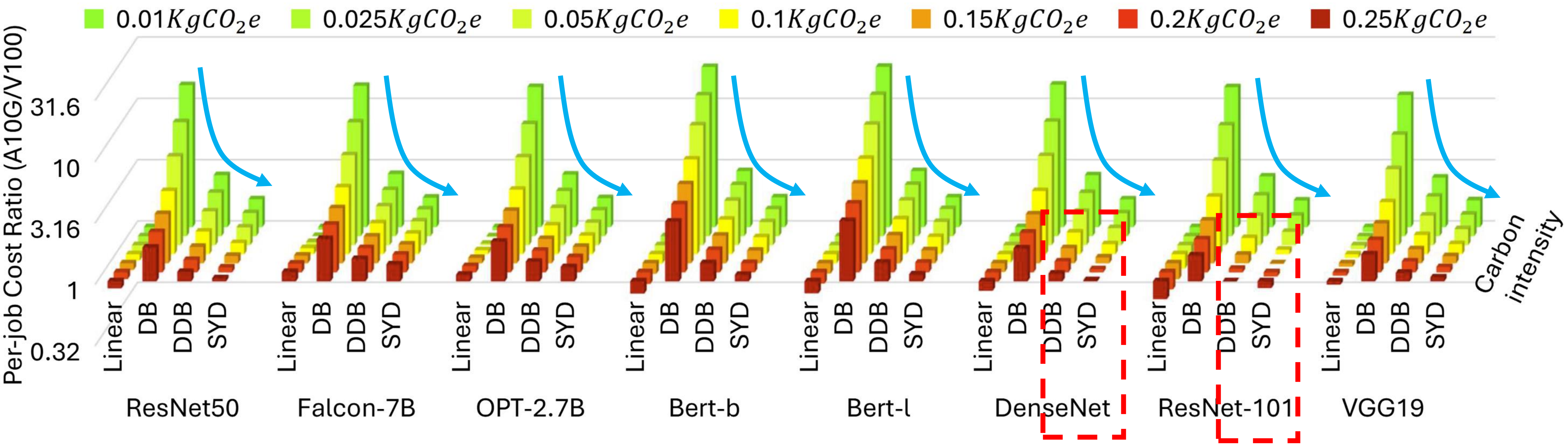
Compare different depreciation models

- The DDB and SYD models are less aggressive
- As a result, the design choices may be flipped again when the CI is high, and the new device have a significant improvement in performance and data efficiency

Sensitivity Study

Different Applications and Carbon intensities could affect design choices

Year 4 V100 server vs. Year 1 A10G server; Y axis > 1 means V100 is better



Compare different depreciation models

- The DDB and SYD models are less aggressive
- As a result, the design choices may be flipped again when the CI is high, and the new device have a significant improvement in performance and data efficiency

Thank You

Advancing Environmental Sustainability in Data Centers via Carbon Depreciation Models

Shixin Ji*, Zhuoping Yang*, Xingzhen Chen*, Alex K. Jones§, Peipei Zhou*

Brown University*; Syracuse University§

shixin_ji@brown.edu

peipei_zhou@brown.edu

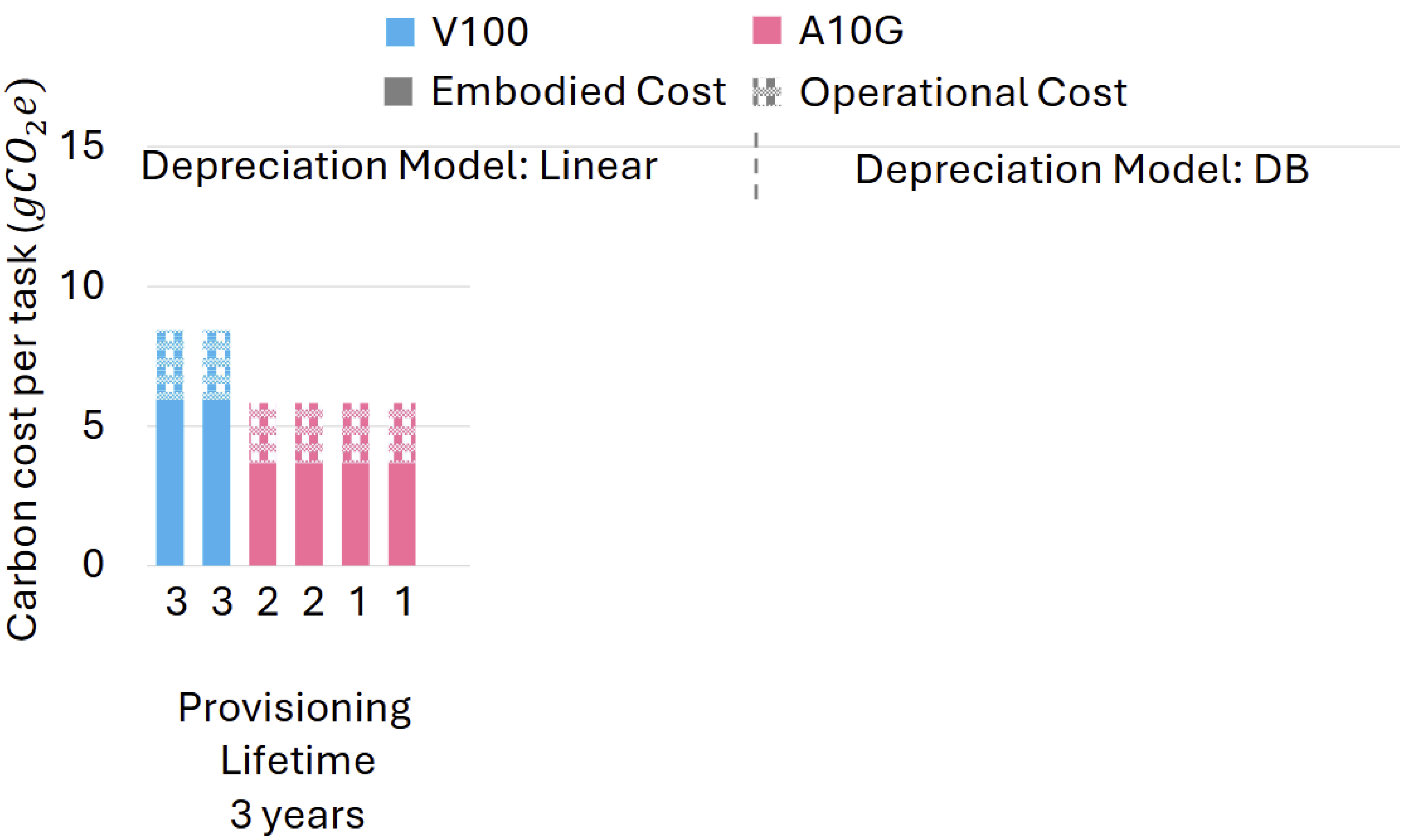
https://peipeizhou-eecs.github.io/

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported

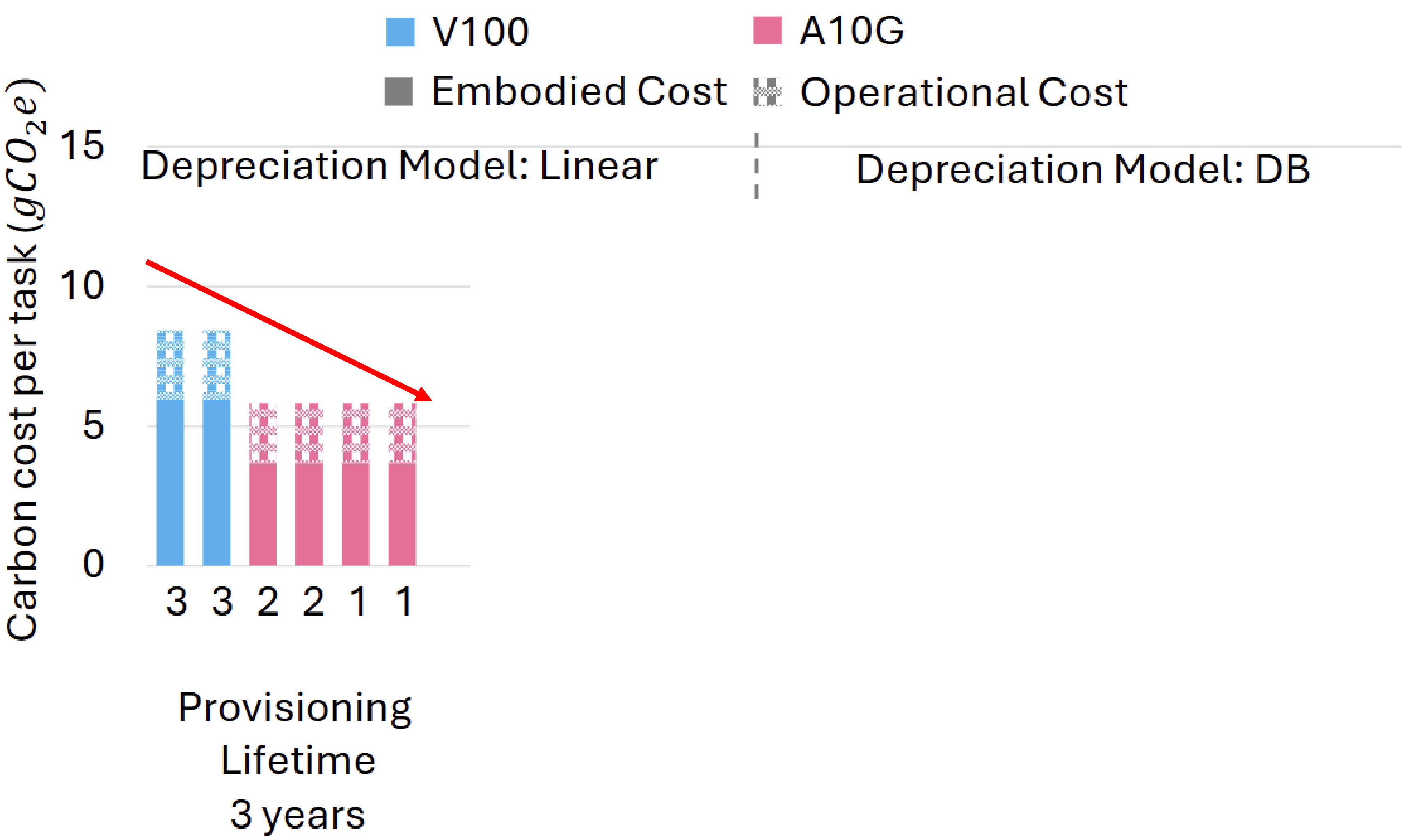


Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported



W/t linear model:

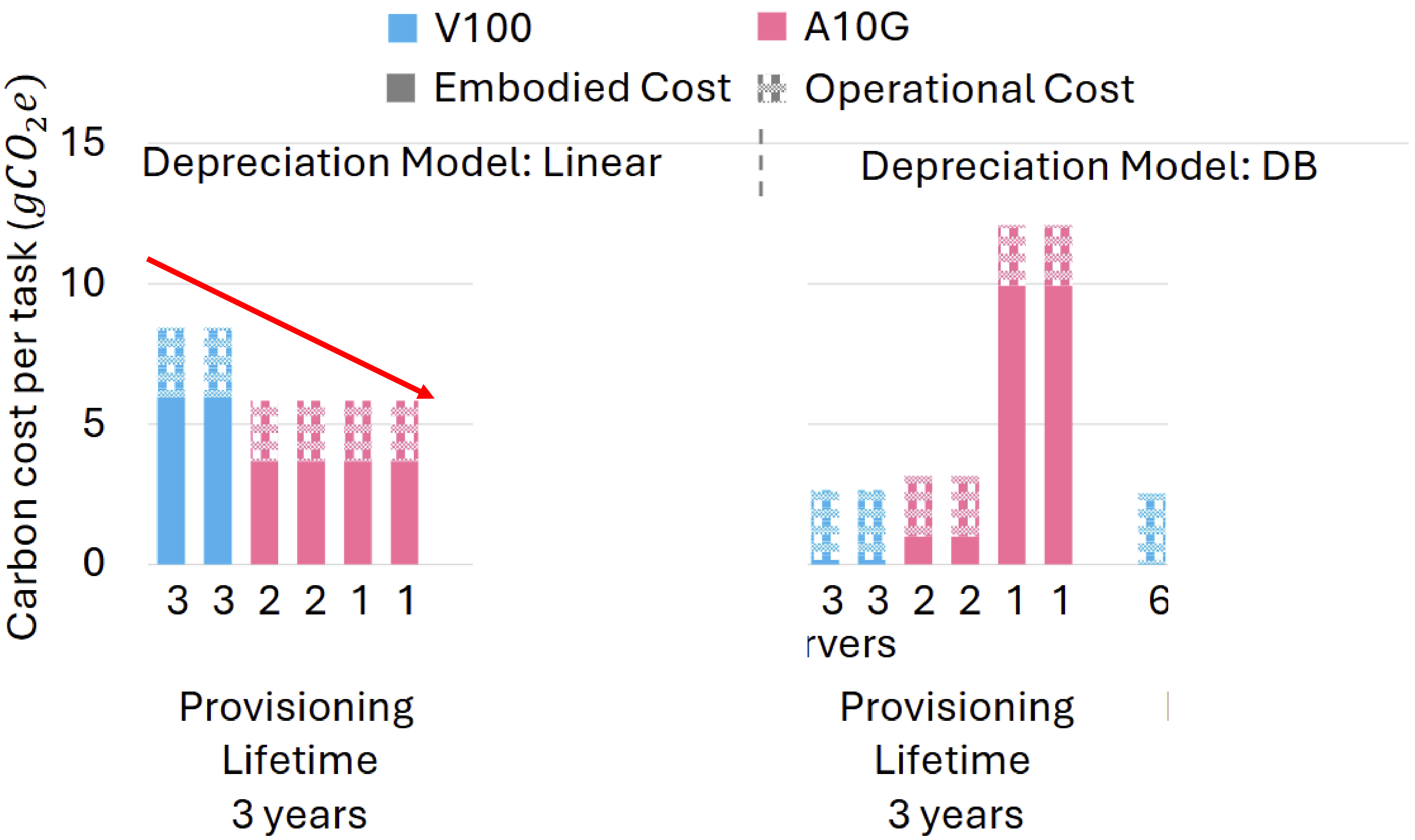
- Newer GPU is continuously better
- Fails to quantify the 'sunk' carbon of the new A10G model in the trend

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported



W/t linear model:

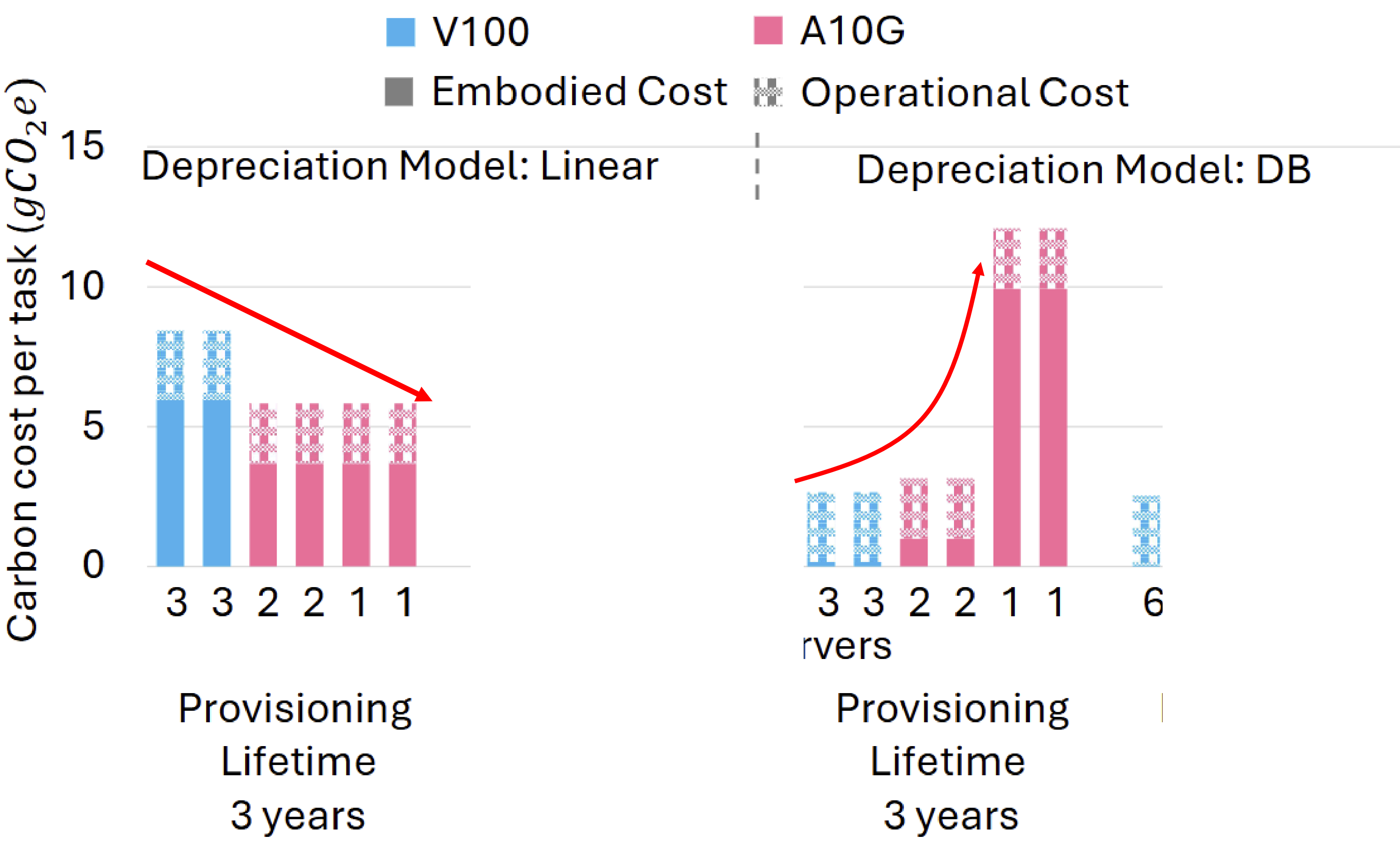
- Newer GPU is continuously better
- Fails to quantify the 'sunk' carbon of the new A10G model in the trend

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported



W/t linear model:

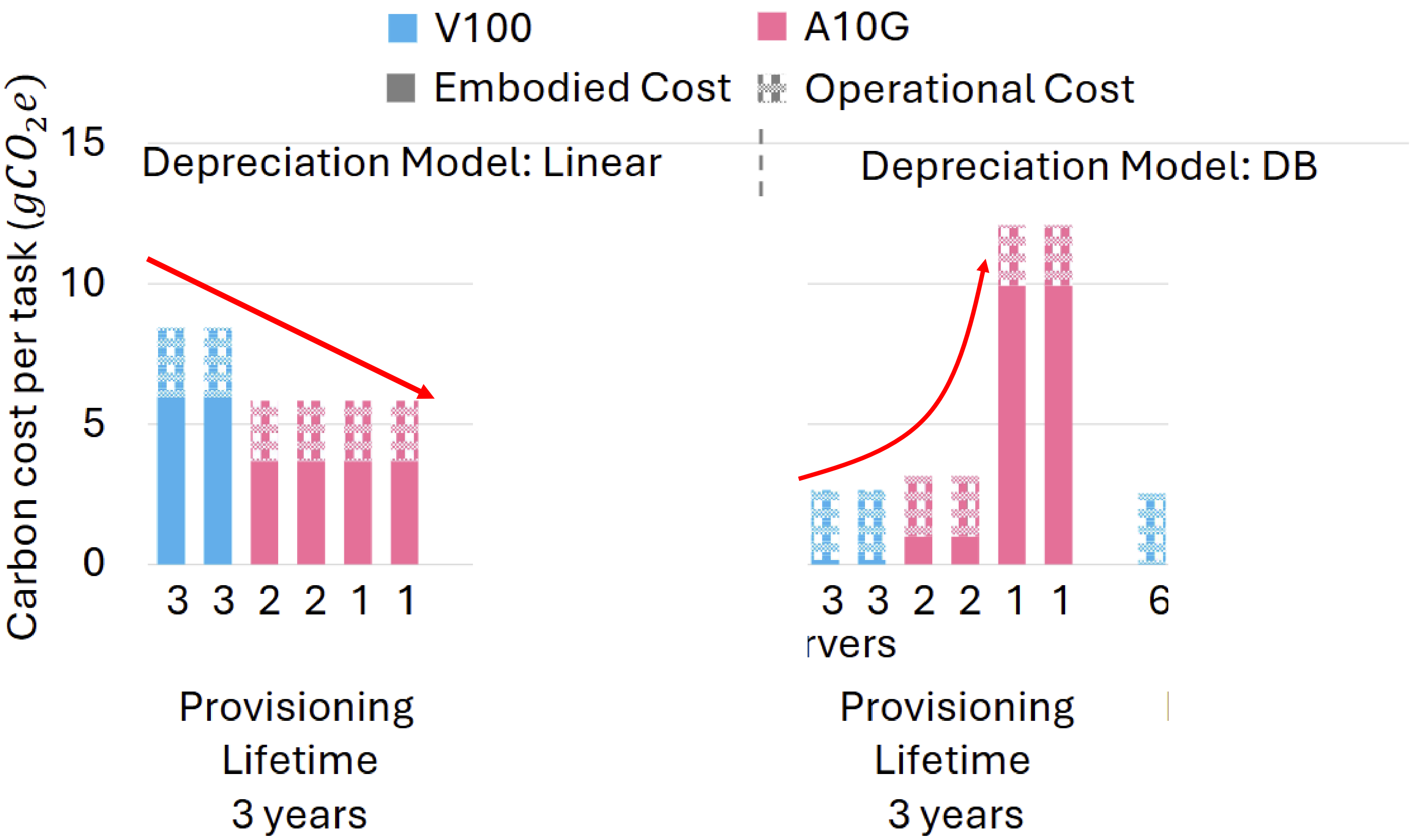
- Newer GPU is continuously better
- Fails to quantify the 'sunk' carbon of the new A10G model in the trend

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported



W/t linear model:

- Newer GPU is continuously better
- Fails to quantify the 'sunk' carbon of the new A10G model in the trend

W/t DB model:

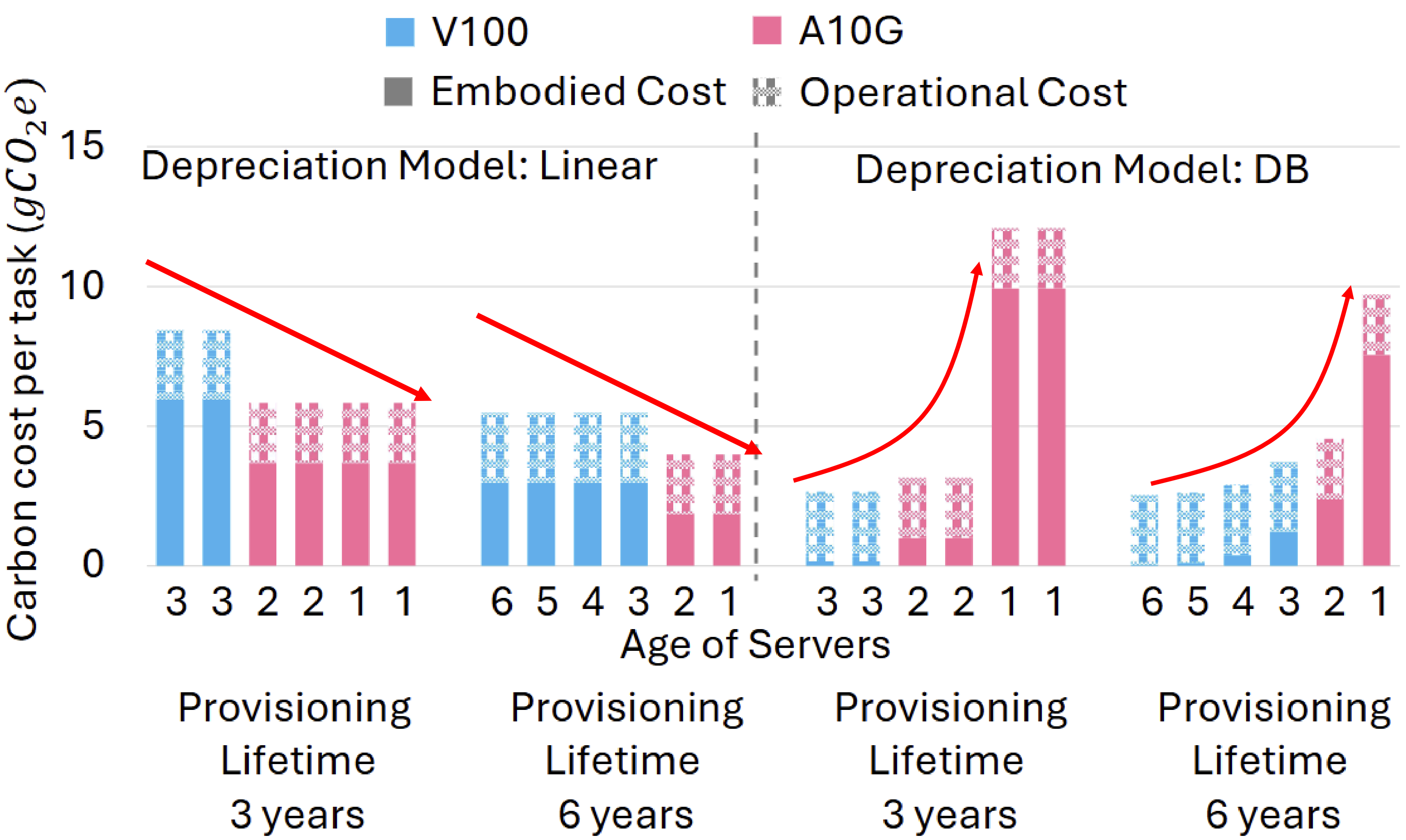
- Older GPUs are better with no additional C_{em}
- Reflect the 'sunk' in the trendline: new A10G models will be much higher than existings

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 or 6-year lifetime

Replacement cycle: 3-year lifetime → replace the oldest 1/3 servers every year → 2 year 1, 2, and 3 servers

The per-task carbon cost is reported



W/t linear model:

- Newer GPU is continuously better
- Fails to quantify the 'sunk' carbon of the new A10G model in the trend

W/t DB model:

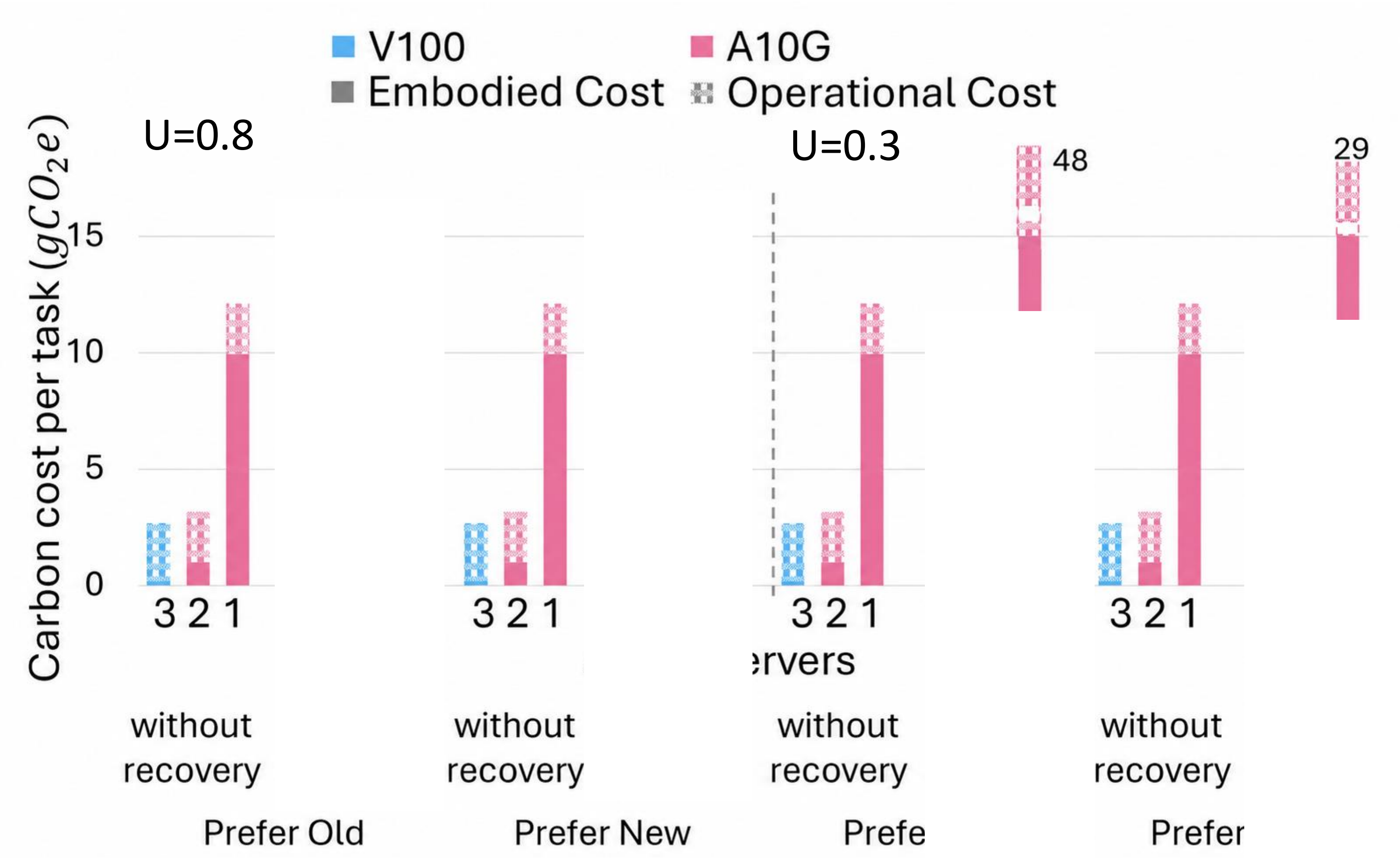
- Older GPUs are better with no additional C_{em}
- Reflect the 'sunk' in the trendline: new A10G models will be much higher than existings

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)

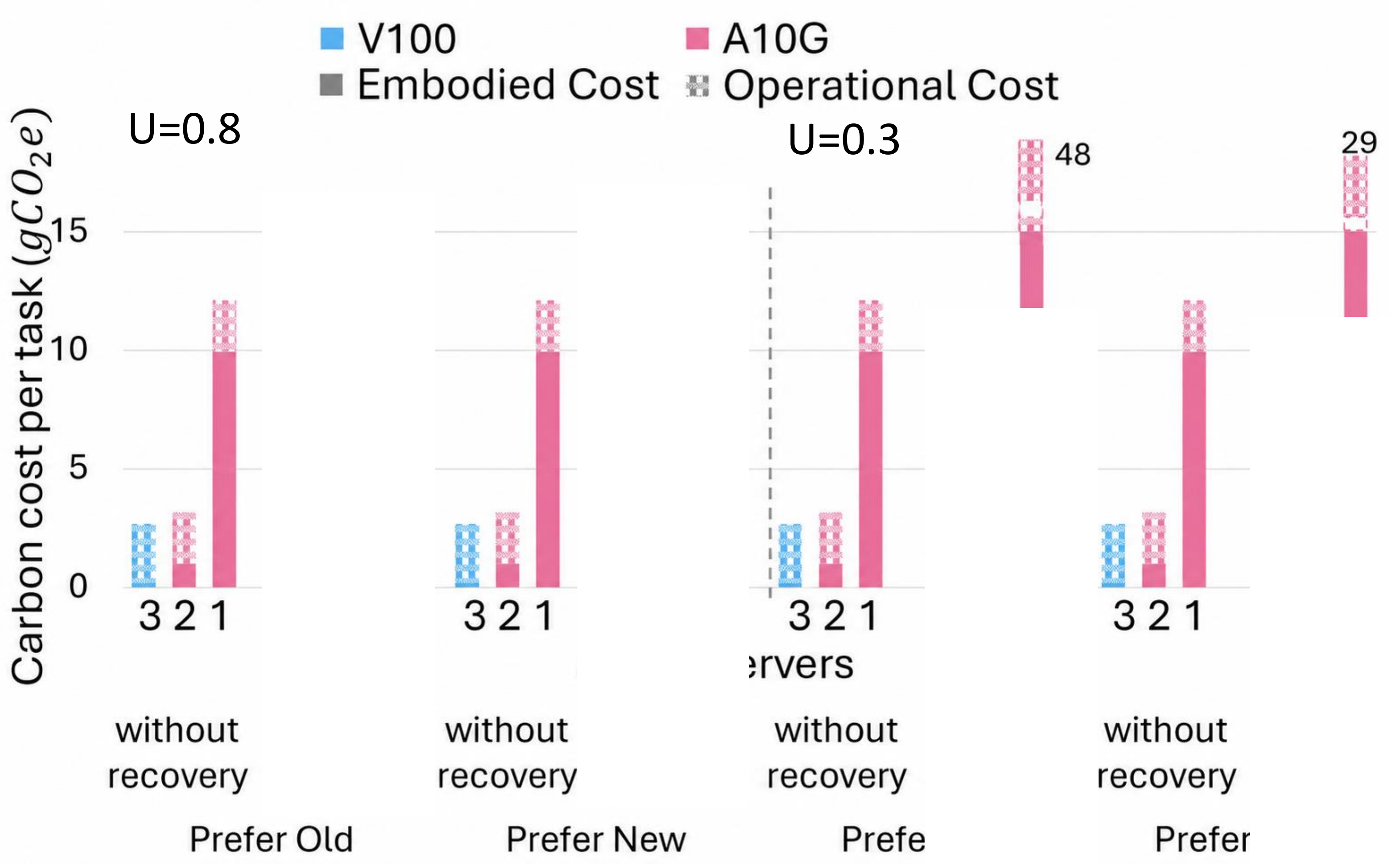


Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

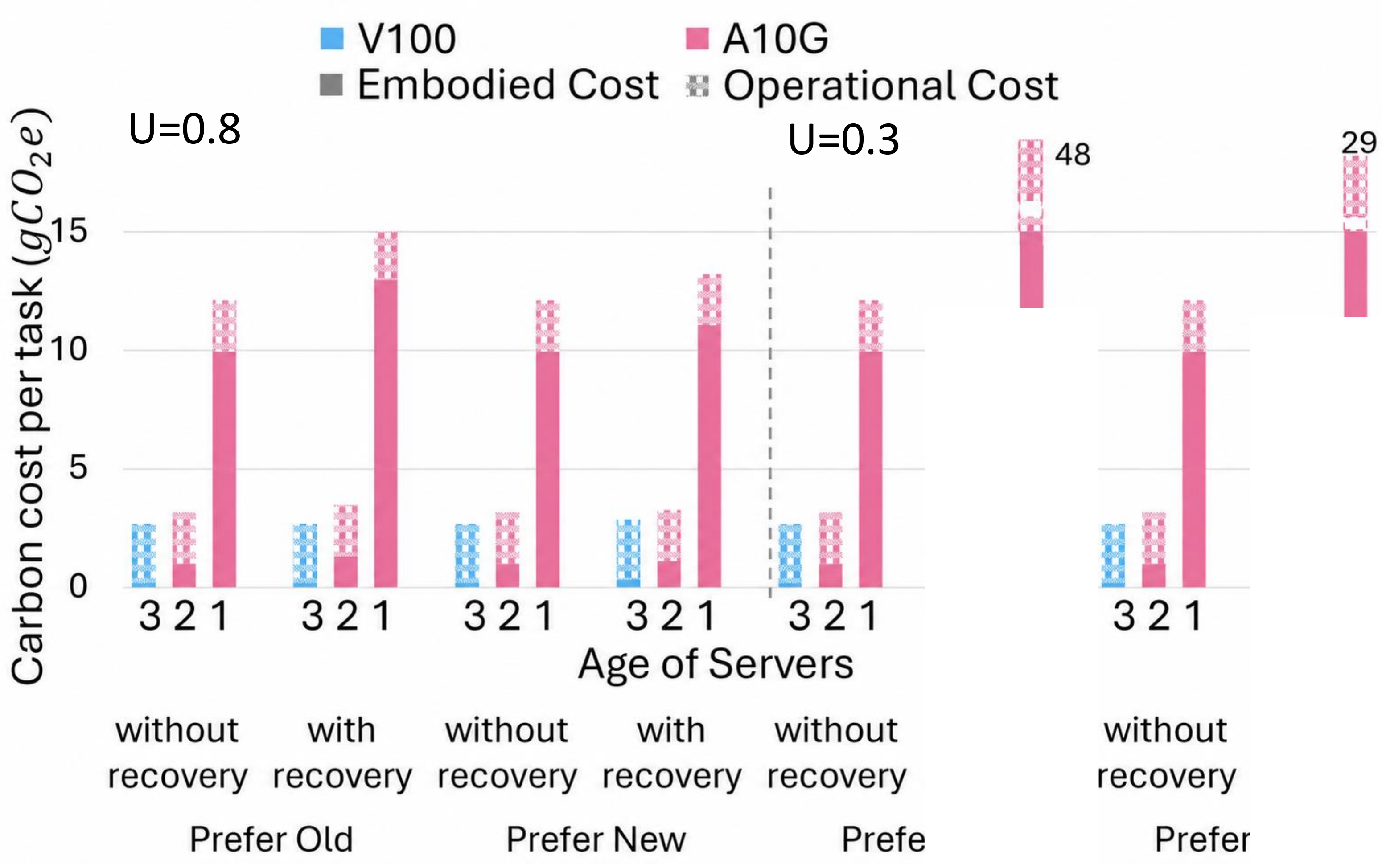
- Unaware about provisioning
- Unaware about Scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning

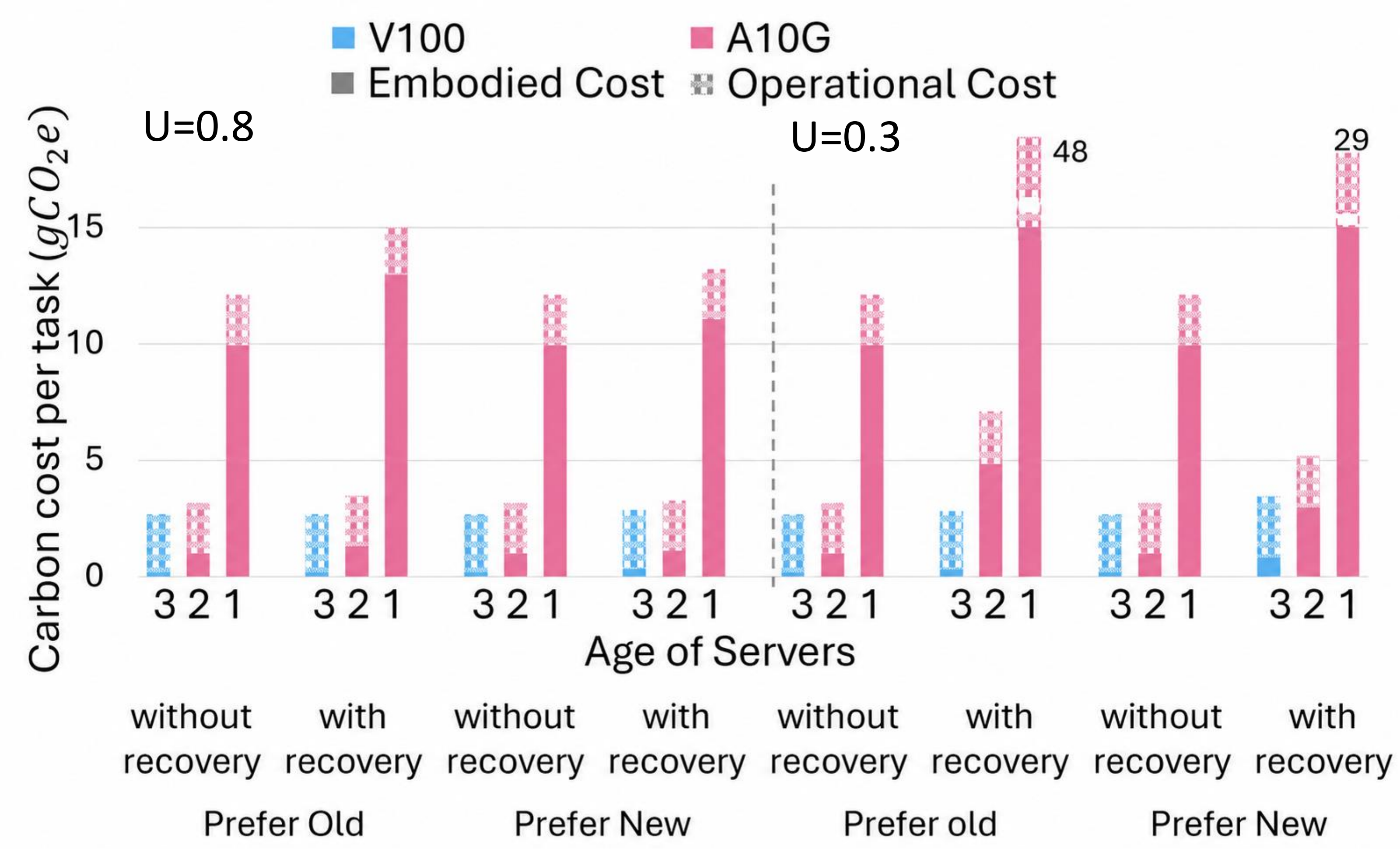
- Slight difference in the carbon cost between different scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning

- Slight difference in the carbon cost between different scheduling

W/t over provisioning

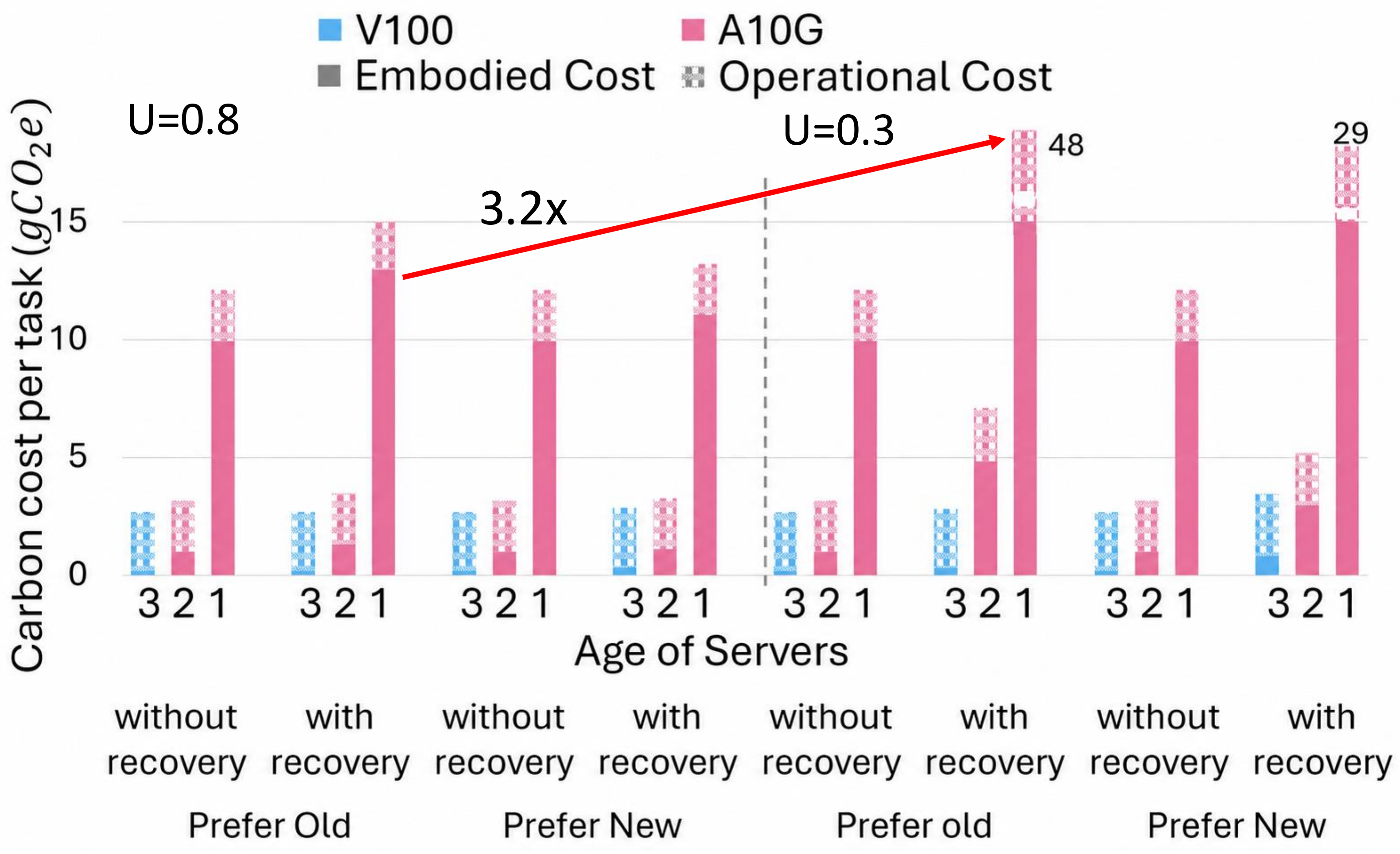
- Huge carbon cost increase
- Large difference between different scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning

- Slight difference in the carbon cost between different scheduling

W/t over provisioning

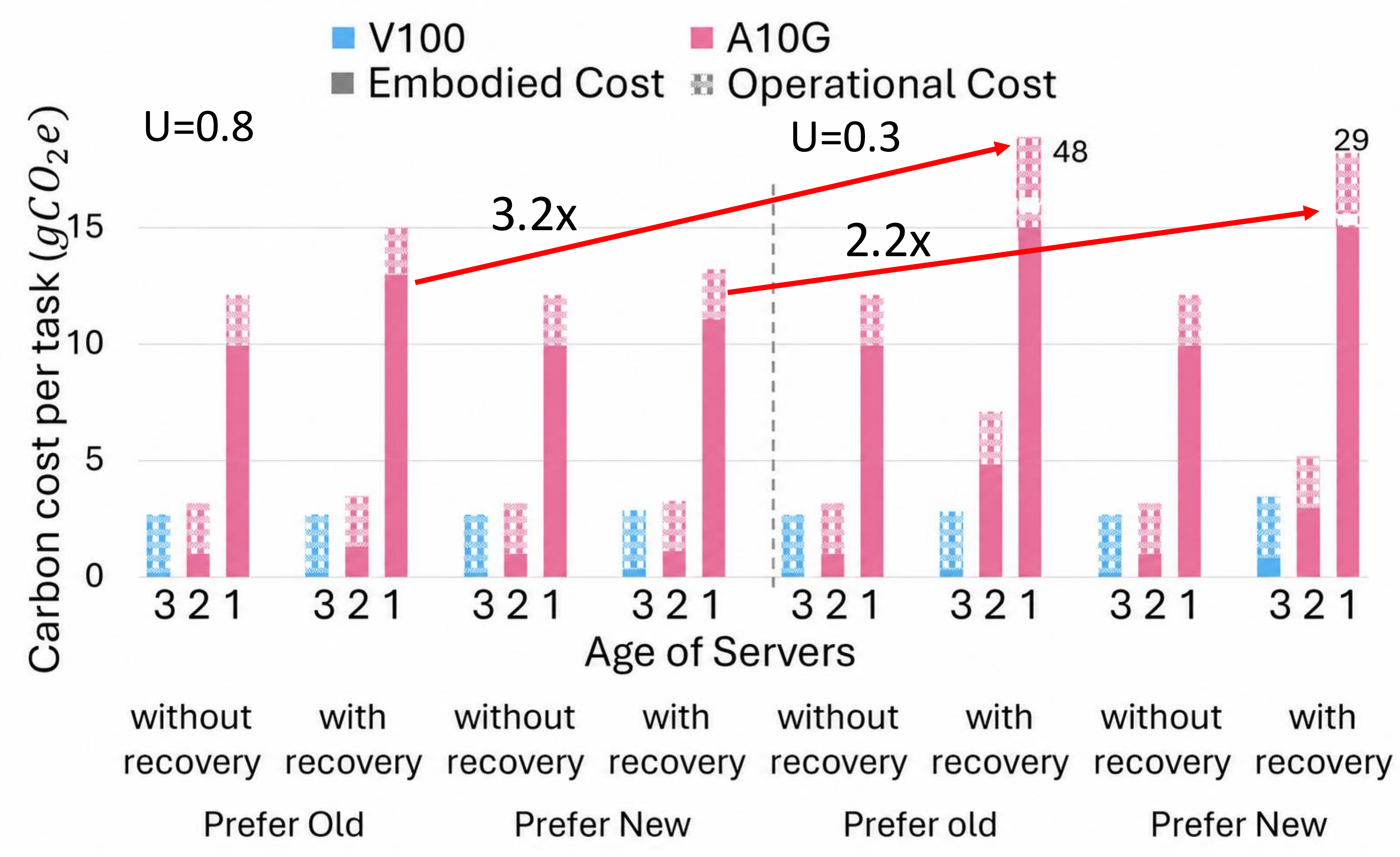
- Huge carbon cost increase
- Large difference between different scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning

- Slight difference in the carbon cost between different scheduling

W/t over provisioning

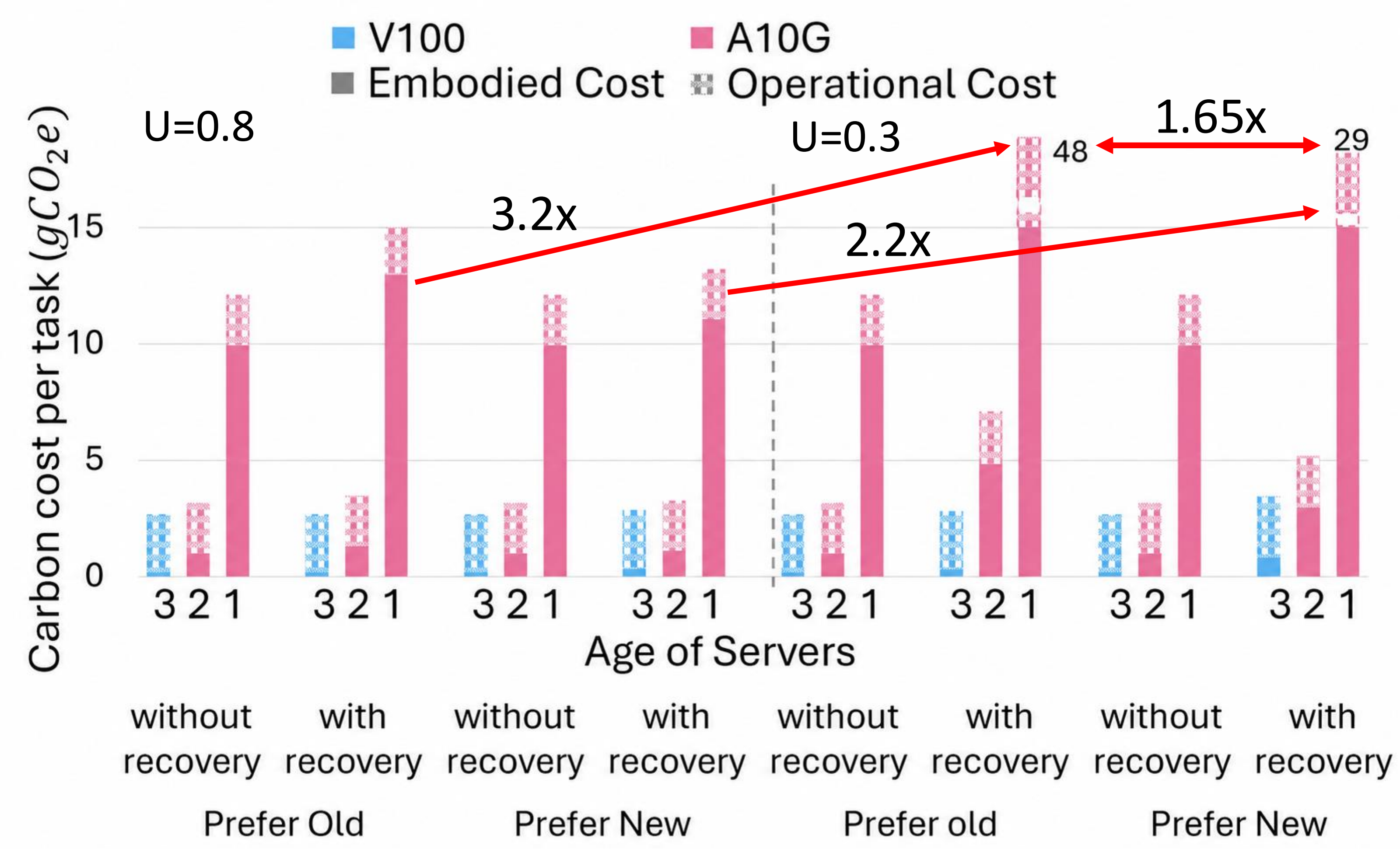
- Huge carbon cost increase
- Large difference between different scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery
- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning
- Slight difference in the carbon cost between different scheduling

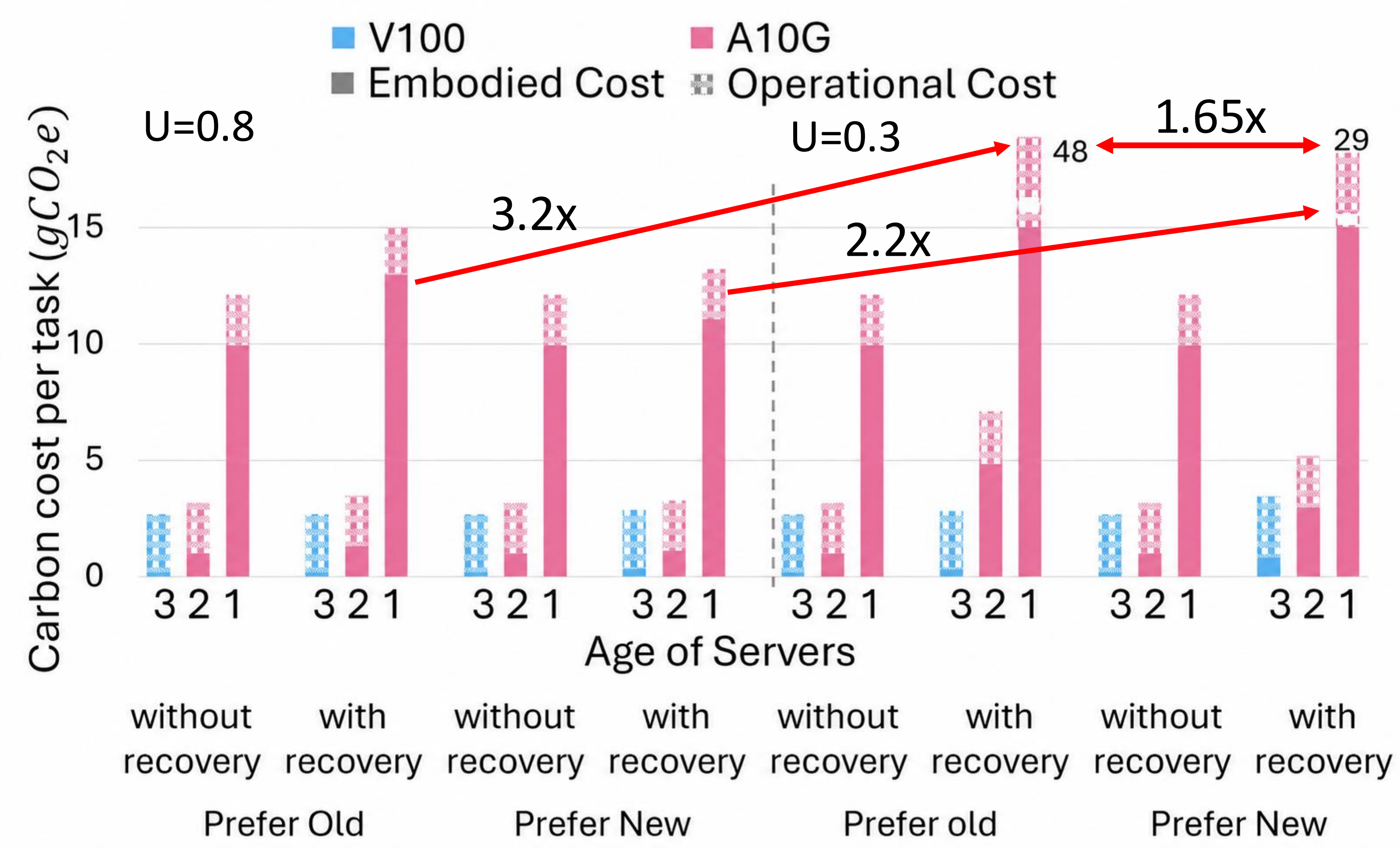
W/t over provisioning
- Huge carbon cost increase
- Large difference between different scheduling

Multi-server scenario: Case Study for Cluster Provisioning

Cluster of 6 servers; 3 year lifetime: 2 x year 1 (2,3) servers, respectively

2 Greedy scheduling algorithms: preferring new servers (better) ; preferring old servers (worse)

2 Utilization Scheme: U=0.8 (good provisioning) and 0.3 (over provisioning)



W/o Secondary carbon recovery

- Unaware about provisioning
- Unaware about Scheduling

W/t good provisioning

- Slight difference in the carbon cost between different scheduling

W/t over provisioning

- Huge carbon cost increase
- Large difference between different scheduling

The secondary carbon makes it clear between good and bad provisioning, and between good and bad scheduling