Microsoft

Virtual Machine Allocation with Lifetime Predictions

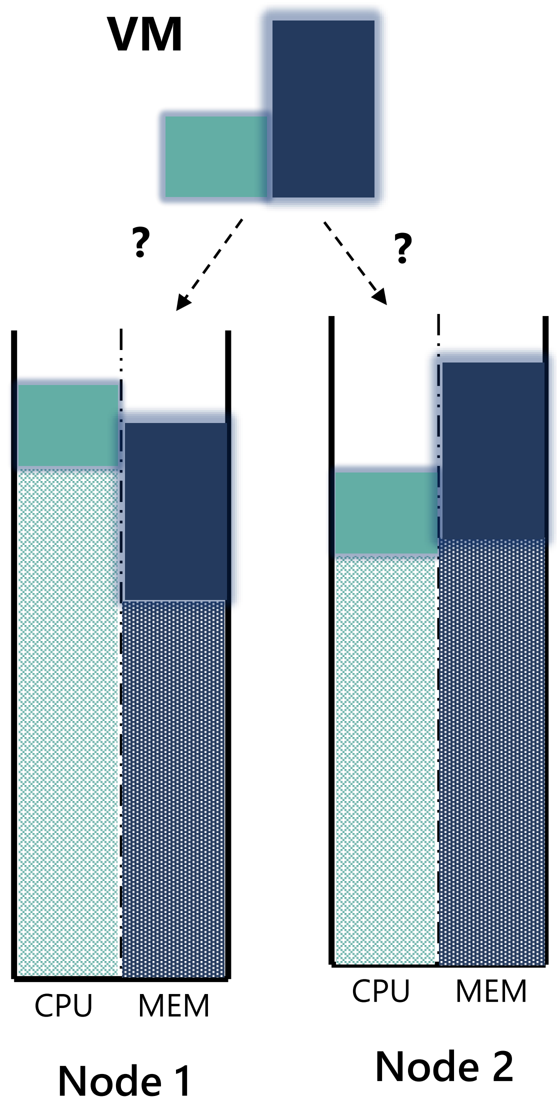
Hugo Barbalho, **Patricia Kovaleski**, Beibin Li, Luke Marshall, Marco Molinaro, Abhisek Pan, **Eli Cortez**, Matheus Leao, Harsh Patwari, Zuzu Tang, Tamires V. C. Santos, Larissa R. Gonçalves, David Dion, Thomas Moscibroda, **Ishai Menache**

Motivation

- Allocation decisions have a direct impact on resource efficiency
- Inefficient placement might result in fragmentation and unnecessary over-provisioning
- Improvements of **1%** in packing efficiency can lead to cost savings of **hundreds of millions of dollars** (Hadary et al., 2020)

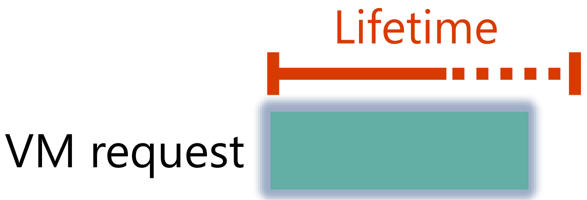
Goal: Increase Azure's packing efficiency with lifetime-aware algorithms

Problem: *Dynamic* multi-dimensional bin packing problem



Motivation

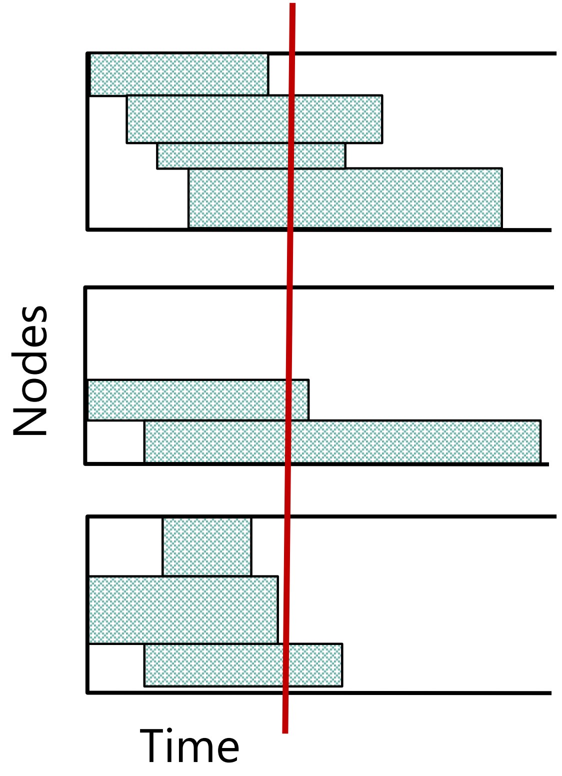
- Allocation decisions have a direct impact on resource efficiency
- Inefficient placement might result in fragmentation and unnecessary over-provisioning
- Improvements of **1%** in packing efficiency can lead to cost savings of **hundreds of millions of dollars** (Hadary et al., 2020)



Goal: Increase Azure's packing efficiency with lifetime-aware algorithms

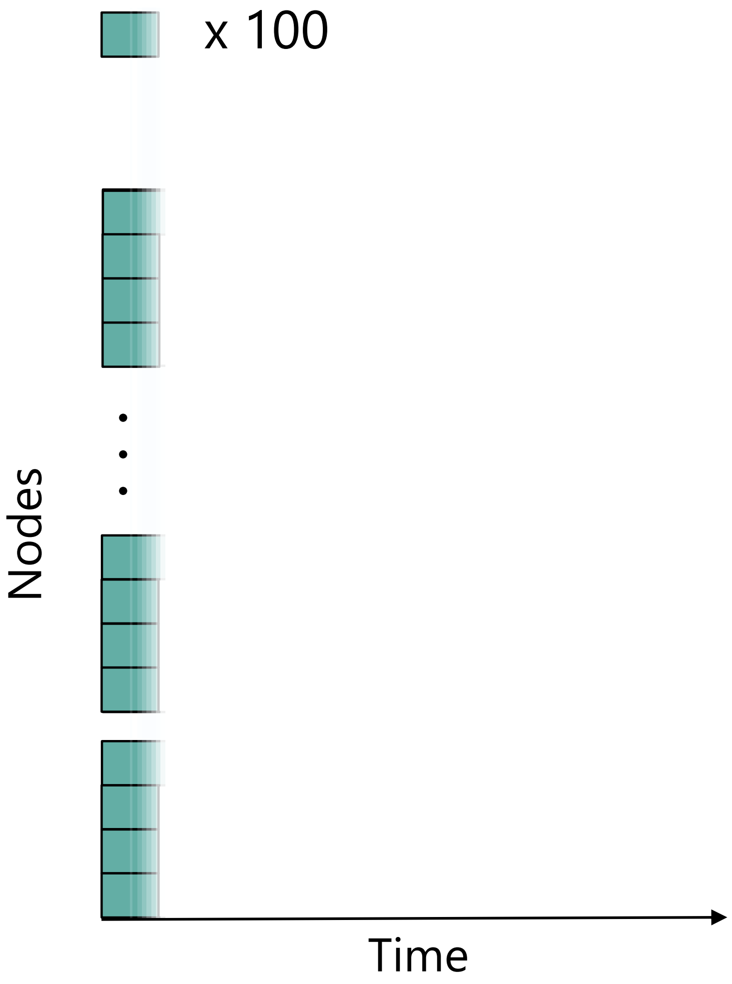
Problem: *Dynamic* multi-dimensional bin packing problem

lifetime



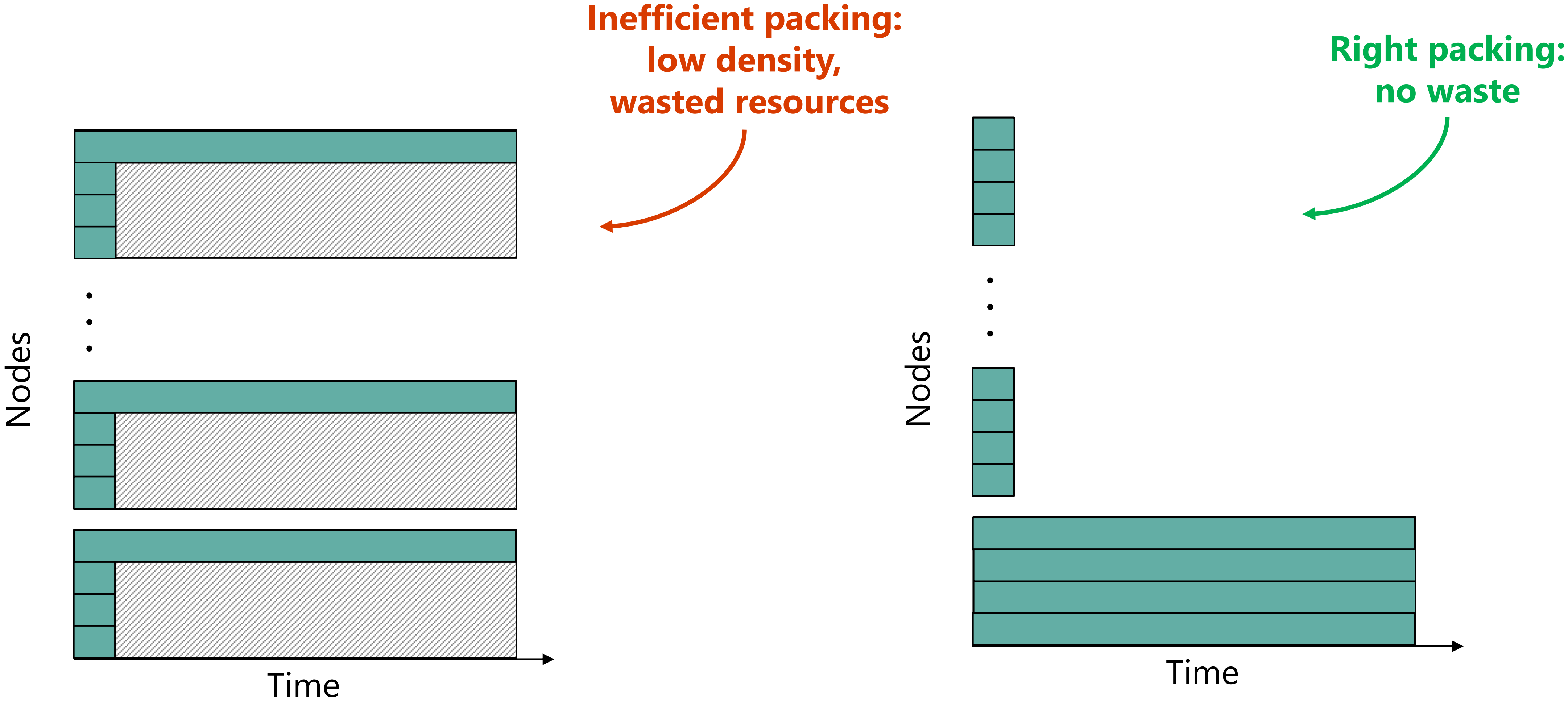
Example

Why lifetime-aware allocations?



Example

Why lifetime-aware allocations?

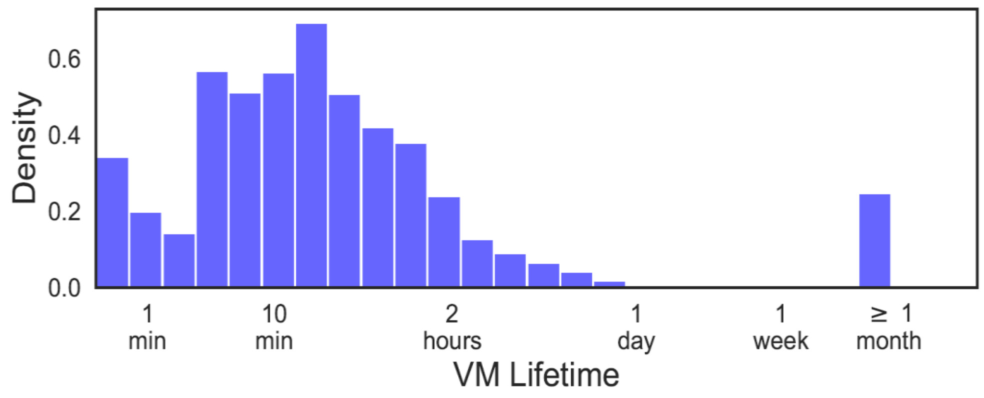


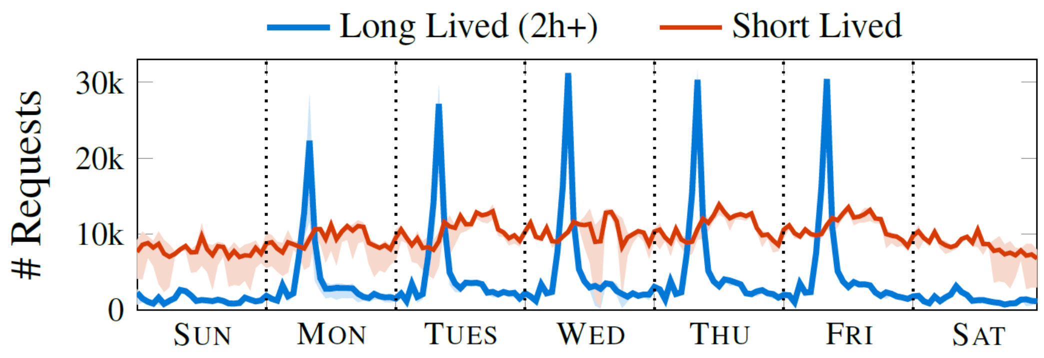
VM lifetime characterization

- How are lifetimes in our system?

- High variance of lifetimes
  - Median: 16 minutes
  - Average: +1 day

- Lifetime temporal patterns
  - Feasibility of VM lifetime prediction





Our contributions

1. Lifetime-aware **algorithm**
2. **ML model** for VM lifetime predictions
3. **System** to support it on real-time

Lifetime Alignment (LA) algorithm

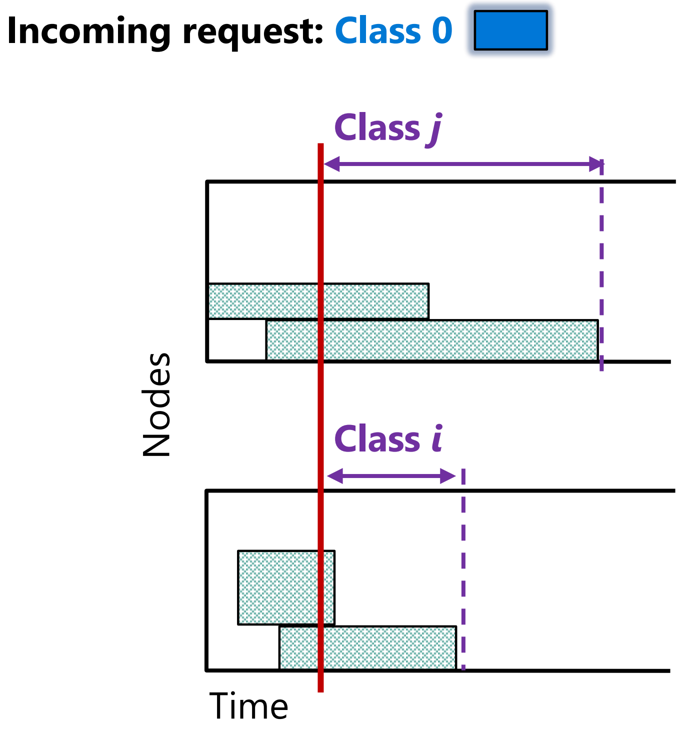
Idea: "Prioritize putting jobs with similar lifetimes together"

- Lifetime ranges are partitioned into classes (where class 0 contains the smallest lifetimes)

For each incoming request:

- If the request is predicted **class 0:**
  - assign to **any** node using Best Fit

- If the request is predicted **class *j*:**
  - assign to a **class *j*** node (if exists), using Best Fit, else,
  - assign to **any** node using Best Fit



- Dynamically updates lifetime classification of nodes
  - Predicted remaining lifetime

- Theoretical indication that LA is **robust to prediction errors**

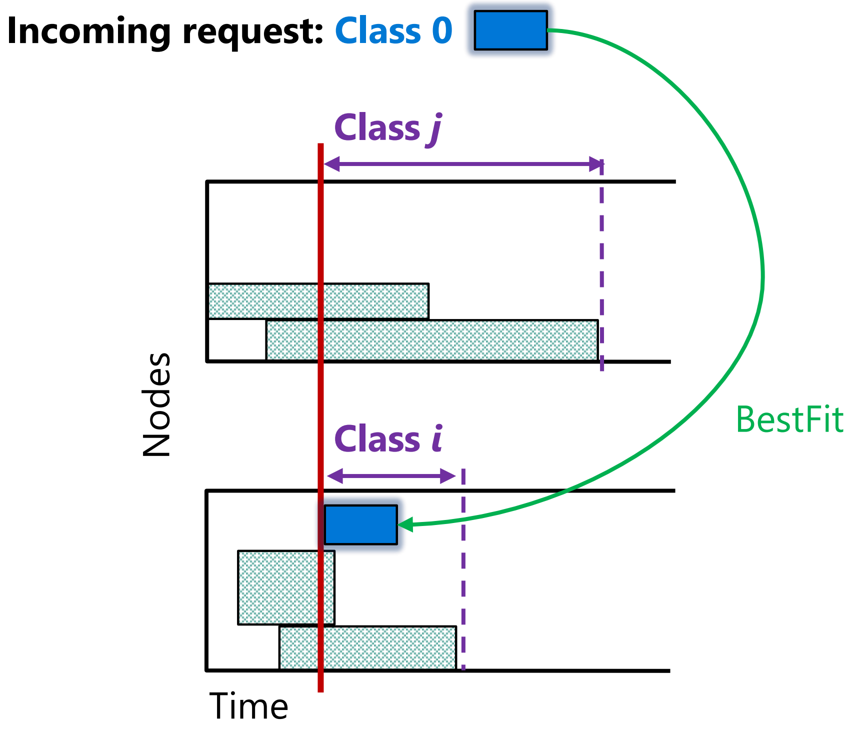
Lifetime Alignment (LA) algorithm

Idea: "Prioritize putting jobs with similar lifetimes together"

- Lifetime ranges are partitioned into classes (where class 0 contains the smallest lifetimes)

For each incoming request:

- If the request is predicted **class 0:**
  - assign to **any** node using Best Fit
- If the request is predicted **class *j*:**
  - assign to a **class *j*** node (if exists), using Best Fit, else,
  - assign to **any** node using Best Fit



- Dynamically updates lifetime classification of nodes
  - Predicted remaining lifetime
- Theoretical indication that LA is **robust to prediction errors**

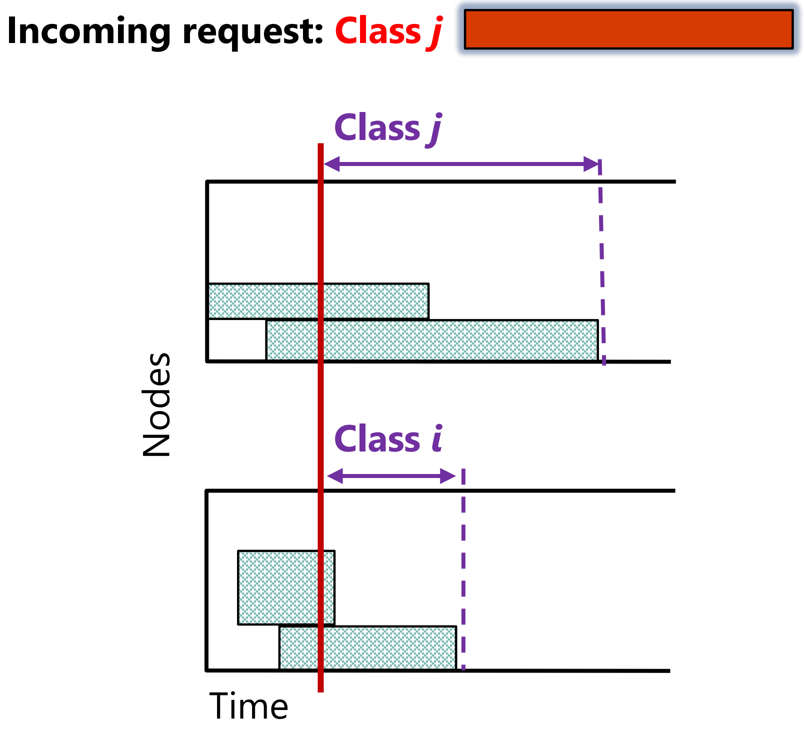
Lifetime Alignment (LA) algorithm

Idea: "Prioritize putting jobs with similar lifetimes together"

- Lifetime ranges are partitioned into classes (where class 0 contains the smallest lifetimes)

For each incoming request:

- If the request is predicted **class 0**:
  - assign to **any** node using Best Fit
- If the request is predicted **class *j***:
  - assign to a **class *j*** node (if exists), using Best Fit, else,
  - assign to **any** node using Best Fit



- Dynamically updates lifetime classification of nodes
  - Predicted remaining lifetime
- Theoretical indication that LA is **robust to prediction errors**

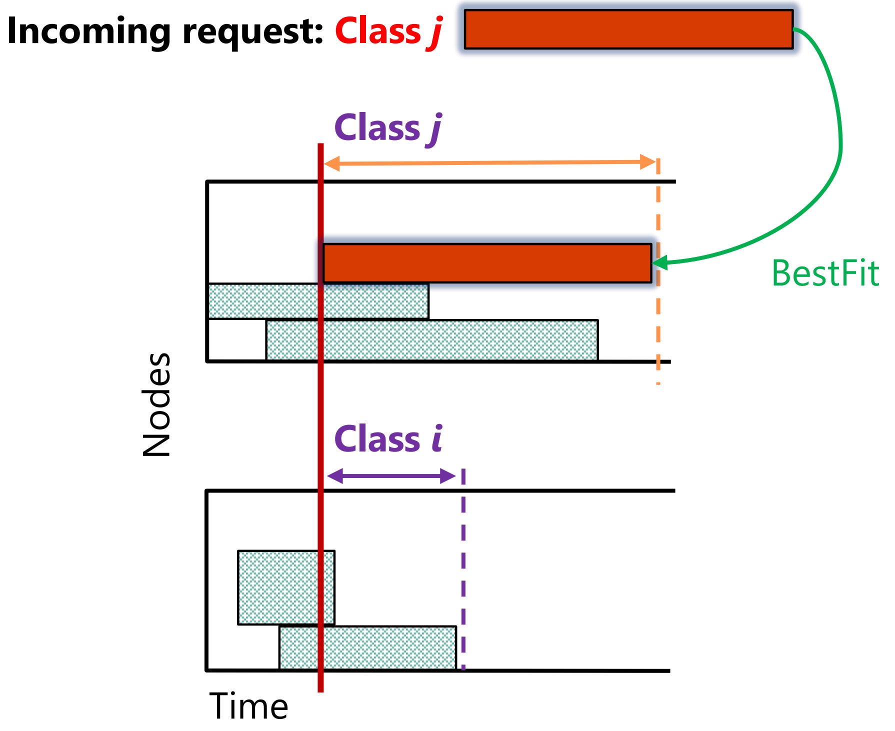
Lifetime Alignment (LA) algorithm

Idea: "Prioritize putting jobs with similar lifetimes together"

- Lifetime ranges are partitioned into classes (where class 0 contains the smallest lifetimes)

For each incoming request:

- If the request is predicted **class 0:**
  - assign to **any** node using Best Fit
- If the request is predicted ***class j*:**
  - assign to a ***class j*** node (if exists), using Best Fit, else,
  - assign to **any** node using Best Fit



- Dynamically updates lifetime classification of nodes
  - Predicted remaining lifetime
- Theoretical indication that LA is **robust to prediction errors**

Predicting lifetime

Challenges:

- Small feature set
- Fast inference time
- Missing data (loss or pruning)
- Skewed and long-tailed lifetime distribution

→

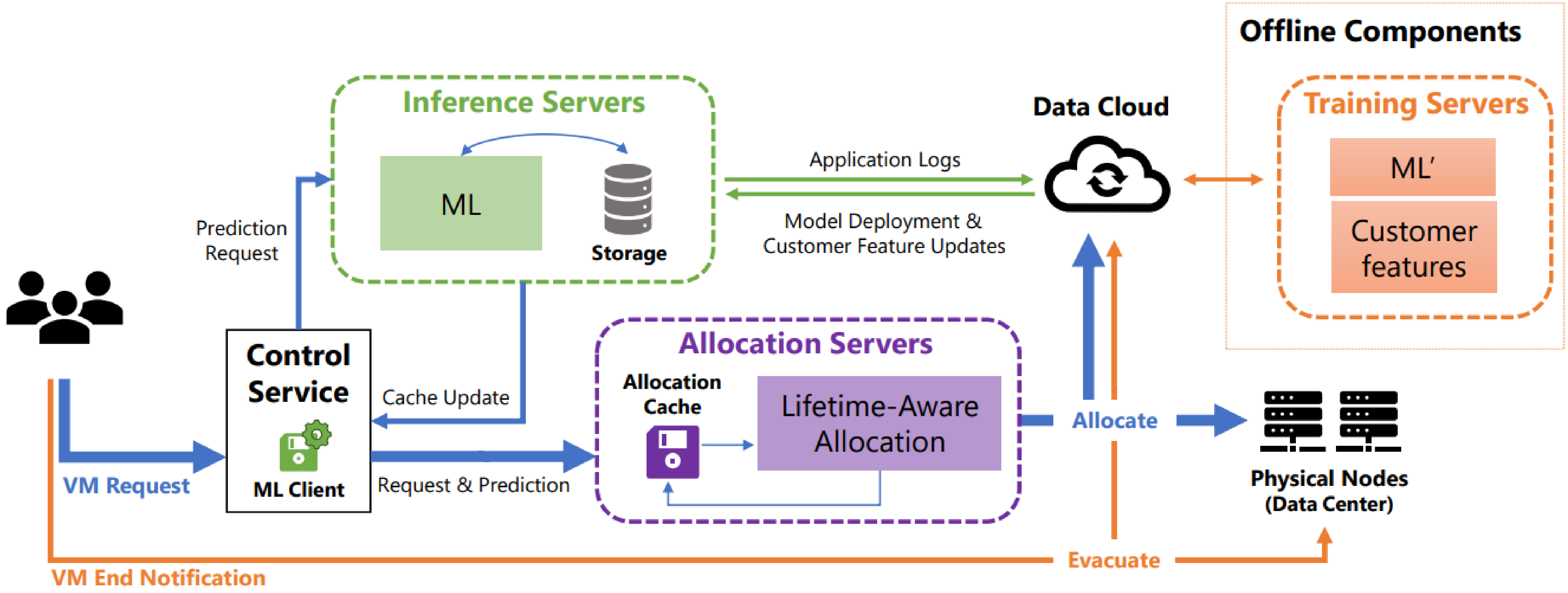
LightGBM model
Binary classification
Short/long threshold

Features:

- VM centric (VM type, OS, request time)
- Customer centric (temporal distribution)

System architecture

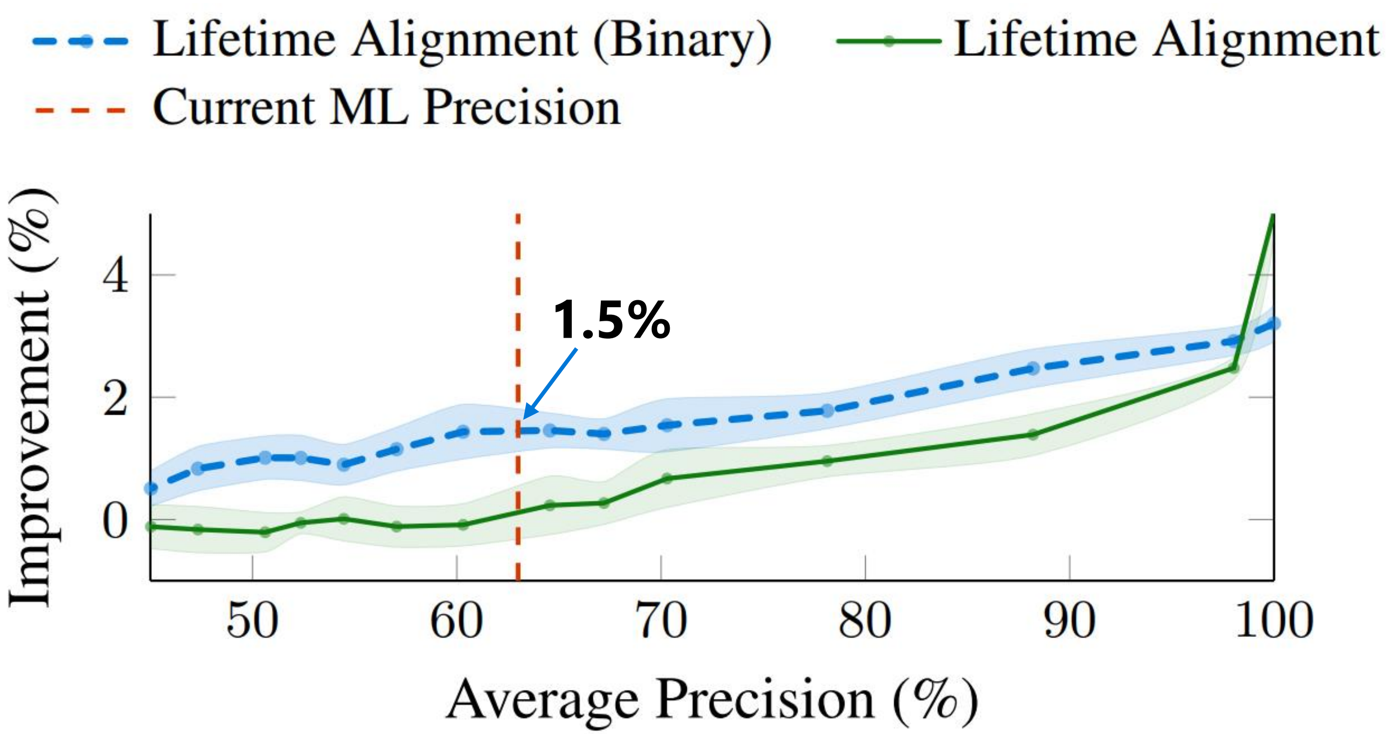
Challenge: How to predict on real time without causing delays?



Real-world production results

- Initial version (ML model + algorithm) in production
- 20 Million daily prediction requests
  - 200+ datacenters
  - 60+ regions
- 60% cache hit on inference results
- 99.2% predictions within time budget
  - Limit of 30ms
- ML model on production achieves expected performance

Experiments



Packing Density: Measures the average number of allocated cores on non-empty machines

Conclusion

We designed and implemented:

- **Lifetime-aware packing algorithm** robust to prediction errors
- **ML model** for VM lifetime predictions
- **System infrastructure** to support ML predictions in the critical path

➢ Packing improvements expected to save hundreds of millions of dollars per year

General methodology for resource management:

1. Produce data-driven intelligence (ML training, simulations) – offline, slower time-scale
2. Utilize the intelligence at real-time (“inference”)
3. Applies to other scenarios, e.g., admission control (OSDI’23)

References

Hadary, O., Marshall, L., Menache, I., Pan, A., Greeff, E. E., Dion, D., Dorminey, S., Joshi, S., Chen, Y., Russinovich, M., et al. Protean: VM Allocation Service at Scale. In 14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20), pp. 845–861, 2020.

Azar, Y. and Vainstein, D. Tight bounds for clairvoyant dynamic bin packing. ACM Trans. Parallel Comput., 6 (3), oct 2019. ISSN 2329-4949. doi: 10.1145/3364214.

Buchbinder, N., Fairstein, Y., Mellou, K., Menache, I., and Naor, J. Online virtual machine allocation with lifetime and load predictions. ACM SIGMETRICS Performance Evaluation Review, 49(1):9–10, 2021.

Microsoft

Thank you

Appendix

Lifetime Alignment (LA) algorithm

- Theoretical indication that LA is **robust to prediction errors**

Simplified setting:

- 1 resource (e.g., #CPU cores)
- Defined lifetime intervals (e.g., $I_j \in [2^{j-1}, 2^j)$)
- **Objective:** Minimize average number of nodes used over a time horizon

Theorem: Assume the predicted lifetimes are in expectation within a factor of α from true lifetimes. Then

avg #nodes used by "theoretical" algorithm $\leq O(\alpha^2 \cdot \log \mu) \cdot$ optimal number of nodes

μ = Ratio longest/shortest lifetime

System architecture

- Prefer Best Fit Rule (PBFR)
  - Scores the nodes based on how well they will be packed after the insertion of the requested VM
  - Output score is quantized in a small number of buckets

V1

Dynamic PBFR (DPBFR)

If long-lived, “pack better”, be more selective

V2

Lifetime Awareness Rule (LAR)

LA algorithm → PBFR

Simpler & Closer to default

- Safeguard
  - Returns to default PBFR if distribution change

Evaluation

ML performance:

- Limited inference time

- LGB exhibits:
  - Low latency
  - Smallest memory footprint (40X less)
    - 20 MB (small set) and 51 MB (large set)
    - Does not load temporal features embedding
  - Competitive prediction accuracy

Features	ML	t(μs)	AP	F1	AUC
Small	LGB	**0.1**	46%	45%	89%
	CAM	0.3	73%	47%	84%
	LGB + GRU	0.2	47%	45%	89%
	LGB + CAM	0.2	50%	47%	90%
Large	LGB	0.2	62%	62%	94%
	CAM	0.4	63%	63%	93%
	LGB + GRU	0.2	63%	63%	94%
	LGB + CAM	0.2	63%	64%	95%

Table 1. Machine learning performance over 3 months. Random coin flip would result in a F-1 score of 17%.

Evaluation

Potential benefits under unrealistic setting:

- Perfect lifetime predictions
- PBFR at the limit
  - No quantization
- Offline heuristic as upper bound

Method	**Density Avg. (STD)**	**Improvement (%)**
PBFR (no quant.)	82.12% (+/- 1.80%)	-
Lifetime Alignment	85.06% (+/- 0.05%)	**3.58%**
Offline heuristic	90.11%	9.73%

Table 2. Performance under idealized setting. Results are averaged over ten different instances.

- **Packing Density** (PD): Average number of allocated cores on non-empty machines