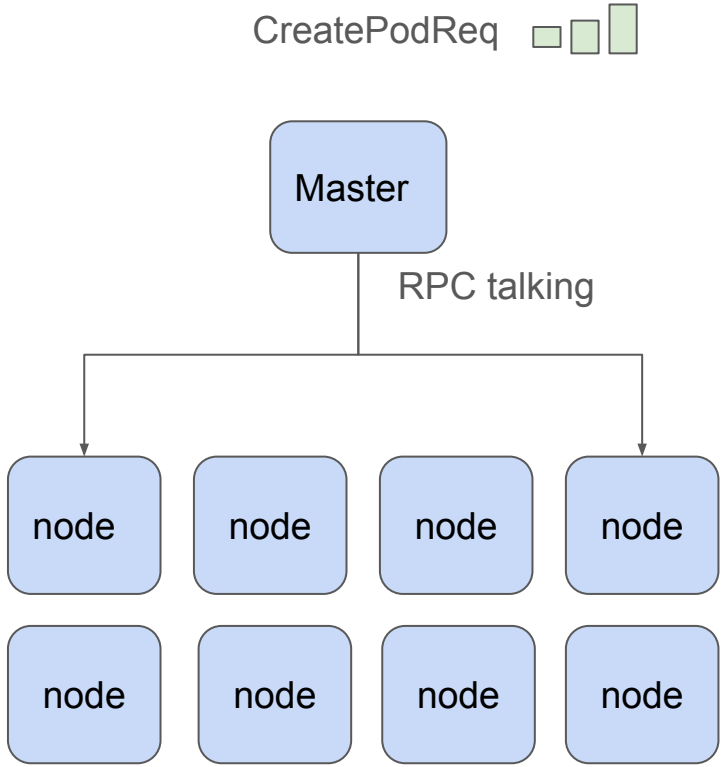
Cloud Computing Engineering Efficiency

Kubernetes

- 7-8 million nodes (physical servers)
- O(billions) of running tasks (containers / VMs)
- Shared Pool (group of nodes):
  - Search, Ads, Youtube
- Dedicated Pool:
  - GCE
- A Pool:
  - Master (Control Plane): Scheduling, Bin-packing, Migrations
  - Nodes (Kubelets): Start / Monitor / Cancel containers, VMs
- Each Job / Pod:
  - A collection of tasks
  - Each task has its own resource requirements (:= limit) of CPU, RAM, SSD, GPU, TPU, etc

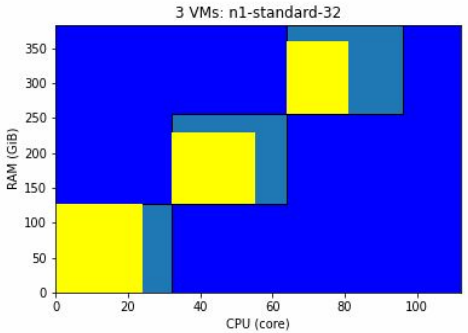


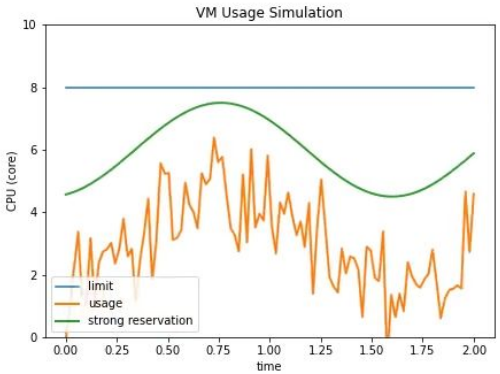
Efficiencies in Kubernetes

- Bin-packing: allocate tasks/VMs across different hosts to minimize stranding and fragmentations
  - Bin-packing is NP hard
  - Online bin-packing is heuristic based, best fit by scoring hosts
  - Multi-resource: lots of improvement space
- Overcommit: when tasks are idling, reclaim the resources to other tasks.

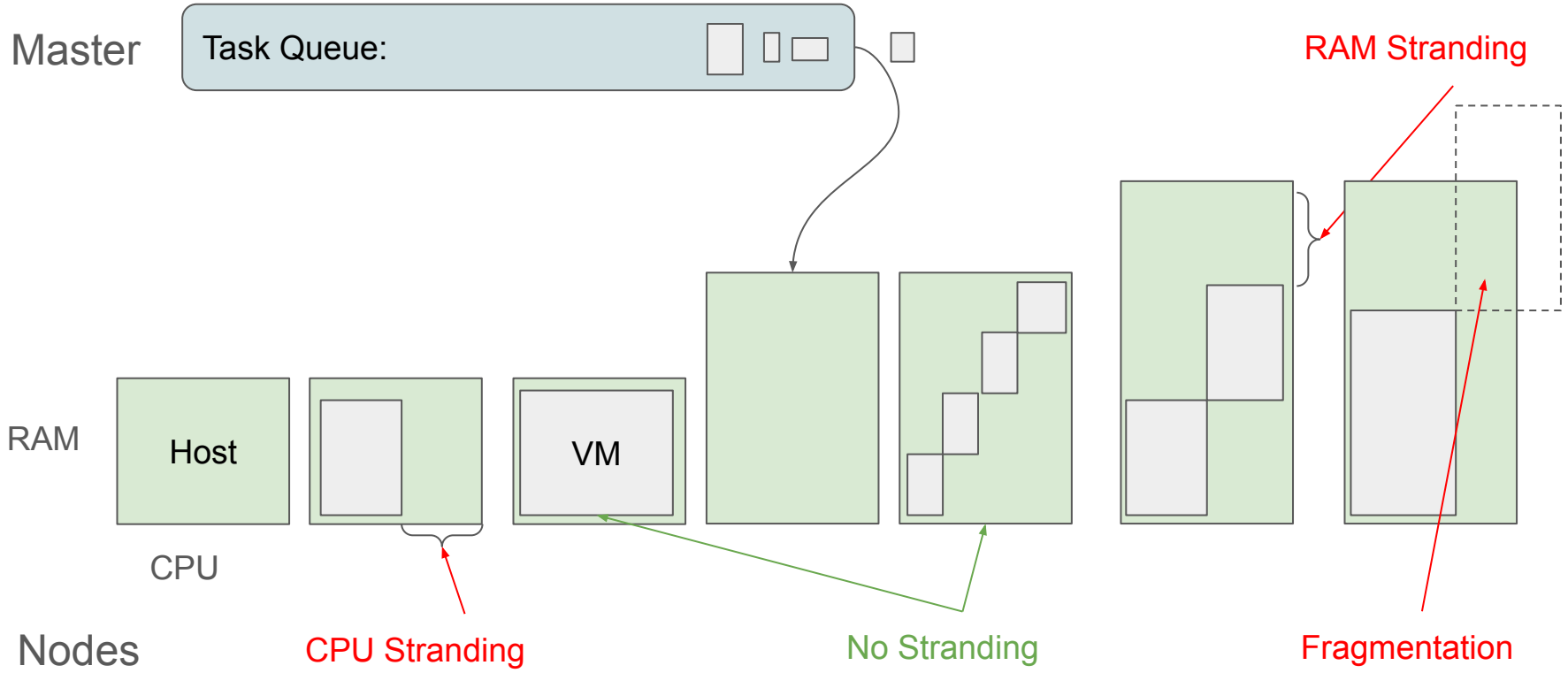
  Peak(Sum) < Sum(Peak)

  - Efficiency and performance tradeoff in general
  - Pivot point of performance degradation → Live Migration
  - Risk control: Statistical capacity buffering and admission control
- Maintenance holdback capacity
  - streaming the maintenance events





Bin-packing Inefficiency: Stranding & Fragmentation



Reduce Stranding: Improve Bin-packing

High level approach

- Define metrics: Stranding & Fragmentation (rigorously computable)
- Measure: Continuous simulations of new VMs until seeing stockouts
- Improve:
  - Bin-packing algo: Adding new scores, shuffle the scores
  - Host shapes: align the RAM:Core ratios between workload and hosts
- Evaluate: Simulations
- Launch: Monitor metrics

We also use integer programming to understand the theoretical upper bound of bin-packing efficiency.

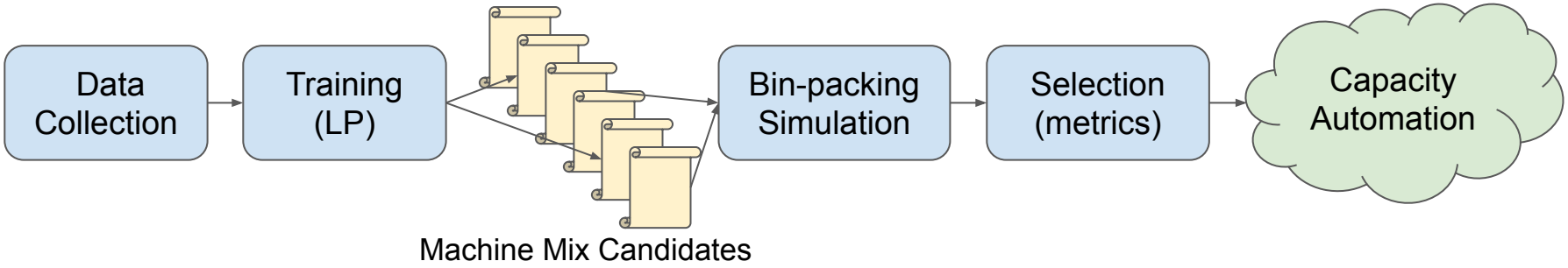
Reduce Stranding: Host Shape Recommendations

Input: The historical workload time series of VM usages, with a given choices of machine shapes

Output: A mix of machine shape to best fit these VMs with minimal stranding

Solution: Domain-based learning pipeline: recommendation system

Impact: Reduce resource stranding by 10%+ of the fleet ($XXX million)



Technical Work

An intersection of software development, system optimization, data engineering, data mining and ML:

- Linear Programming: Stochastic optimizations, modeling, solver implementations
- High-fidelity simulations: C++ Borg scheduler
- Large scale simulations: distributed on K8’s
- Large scale data pipeline: big query SQL, spark-type pipeline (Flume)
- Algorithms: heuristics, LP, ML prediction of lifetime, best-fit scheduling
- RPC service for data serving (Read API)
- ML: VM life-time prediction (published paper on my resume in SysInfra 2025)

Overcommit

- Live migrations based on node side monitoring of performance
- Aggregate capacity control
- Statistical/ML model of peak usage prediction (real-time)
  - Time series forecast
  - Rule based protections
- Risk analysis (statistical)

TPU/GPU Collocation Scheduler

- Bin-packing algo with topological constraints
- Fragmentations
- Migrations to defrag
- Minimization of fragmentations & migrations