How to Find the Optimal Configuration for Your Virtualized Environment

Iordan K. Iotzov

Blog: http://iiotzov.wordpress.com/

About me

- 15+ years of database administration and development experience
- MS in Computer Science, BS in Electrical Engineering
- Presented at RMOUG,Hotsos, NYOUG and Virta-Thon
- Active blogger and OTN participant

Agenda

- Overview of virtual server consolidation
- Modeling the problem
  - Summary and Definitions
  - Constraints
  - Implementation
- Solving the model
  - Discrete optimization
- Conclusion

Overview

Competing Goals

- Minimize the computational footprint of your enterprise through virtualization
  - pay less for hardware and licenses
- While making sure performance and business requirements are met
  - ensure that end user experiences and business processes do not suffer due to the consolidation

Overview

Target Audience

- Medium to large size enterprises
  - For small systems, the efforts to get a great virtual to physical mapping probably do not justify the benefits
- Reasonable load volatility
  - The mapping decisions are based on past performance, so the future load should not significantly deviate from the past. Do not attempt for systems that can go “viral”.
- Ability to measure, store and process various performance metrics
  - Performance data should be made available in a single repository, ideally in a relational database

Overview
Optimization

Brief Introduction to Optimization:

Minimize $f(x)$

Subject to constraints $x \in S$

Where $f: D \rightarrow R$, D is the domain of f
and $S \in D$ is the set of feasible solution x

Constraints S can by represented as

$$C_i(x) = 0$$
$$C_j(x) \geq 0$$

Overview
Continuous Optimization

Continuous optimization:
variable x has real values

Example:

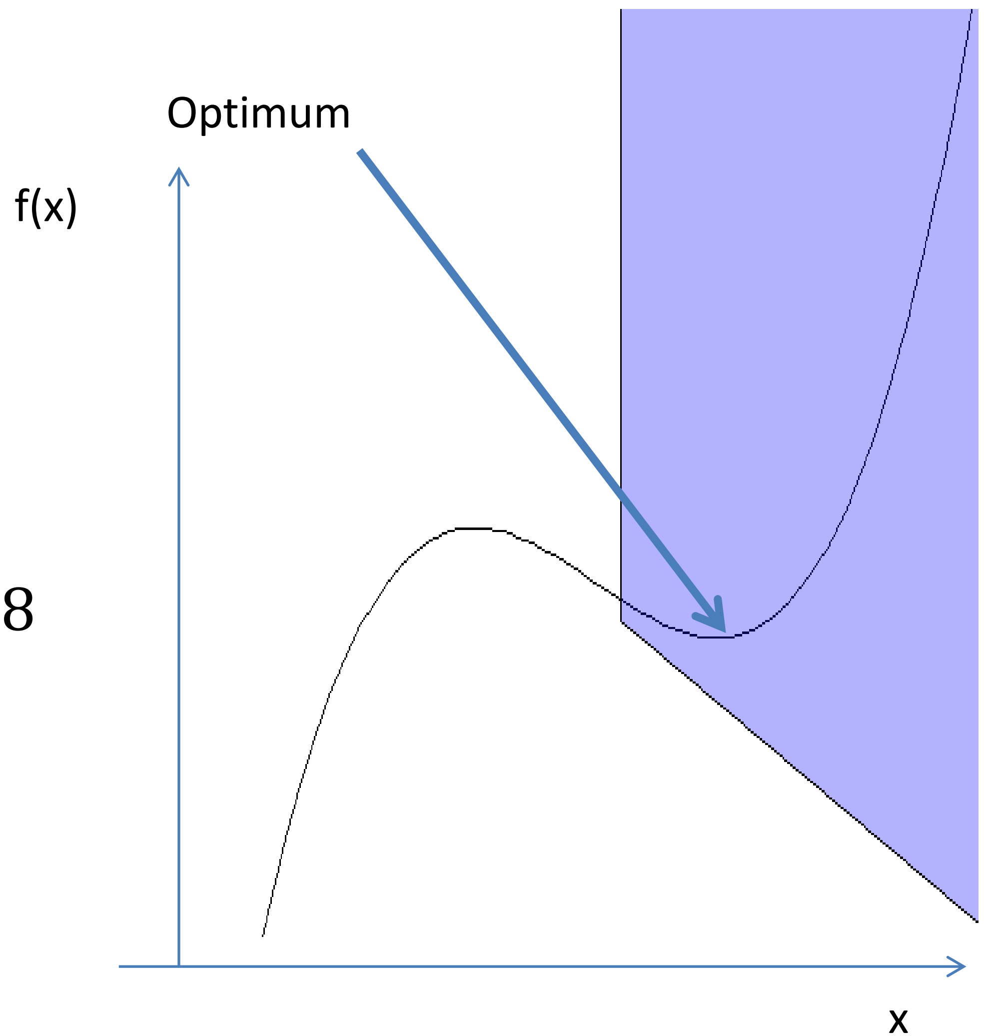
Minimize:

$$f(x) = x^3 - 2x^2 - 31x + 28$$

Subject to :

$$x > 0$$

$$40x + f(x) > 0$$

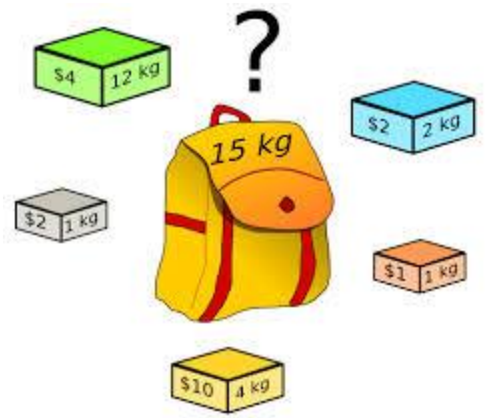


Overview
Discrete Optimization

Discrete optimization: variables x_i are discrete

Example: Knapsack problem

There are n items, each with value v_i and weight w_i. The goal is to maximize the sum of the values of the items in a bag with capacity W



Maximize: $\sum_{i=1}^{n} v_i x_i$

Subject to : $\sum_{i=1}^{n} w_i x_i$ < W

http://commons.wikimedia.org/wiki/File:Knapsack.svg

Modeling the Problem
Summary and Definitions

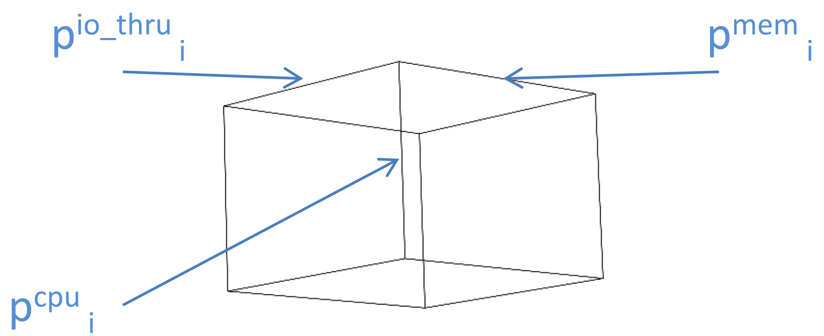
Mapping of virtual to physical servers
as an optimization problem

Minimize:
Total Cost of Ownership (TCO)
Need to account for sunk costs
It is OK to simplify

Subject to:
Technology and Business Requirements
Minimal CPU oversubscription during certain hours (based on historic patterns)
Guaranteed level of performance even when a virtual server goes “berserk”
Certain virtual servers have to run on CPUs with at least x GHz
and many more…

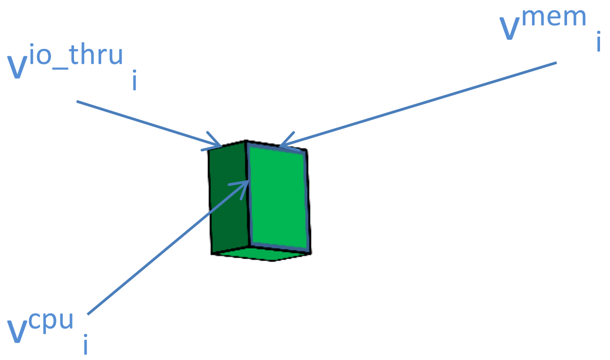
Modeling the Problem
Summary and Definitions

- Physical servers
  - p_i , where i in [1,n]
    - each server has p^{cpu}_i number of CPUs, each with p^{speed}_i speed in GHz
    - each server has p^{mem}_i memory (GB)
    - each server has $p^{io\_thru}_i$ IO throughput(GB/sec)



Modeling the Problem
Summary and Definitions

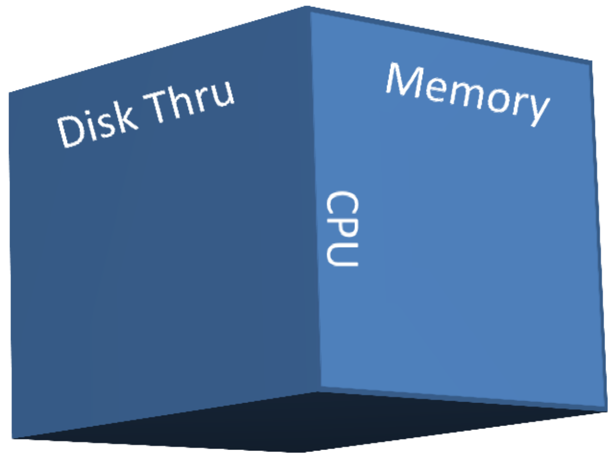
- Virtual servers
  - v_i , where i in [1,m]
    - each server has been allocated v^{cpu}_i number of CPUs
    - each server has been allocated v^{mem}_i memory (GB)
    - each server has used no more than $v^{io\_thru}_i$ IO throughput(GB/sec)



Modeling the Problem
Summary and Definitions

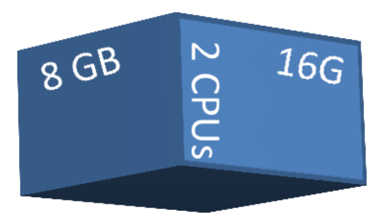
Graphical Representation:

Number of CPUs
Amount of RAM
HBA throughput



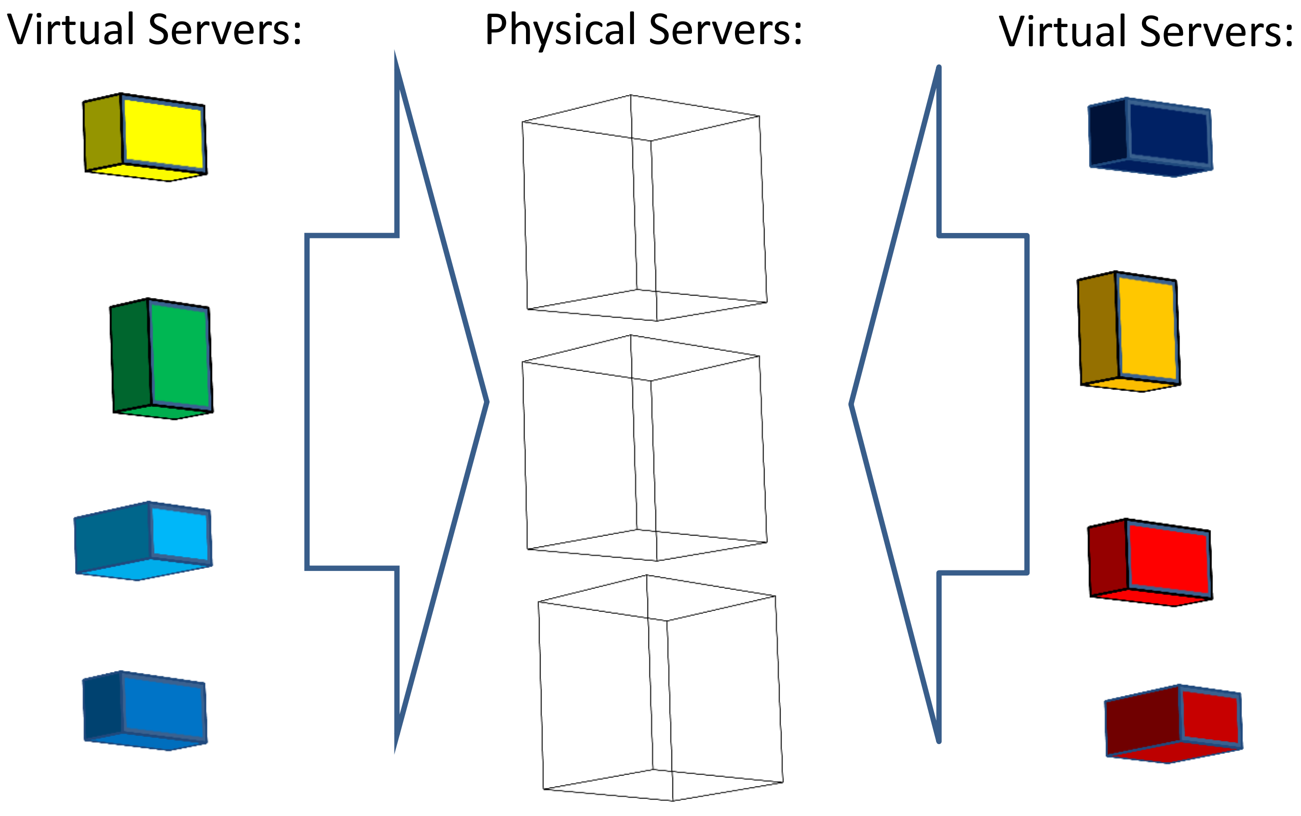
4 CPUs
8G RAM
8GB HBA

2 CPUs
16G RAM
8GB HBA



Modeling the Problem
Summary and Definitions

Mapping of virtual servers to physical ones

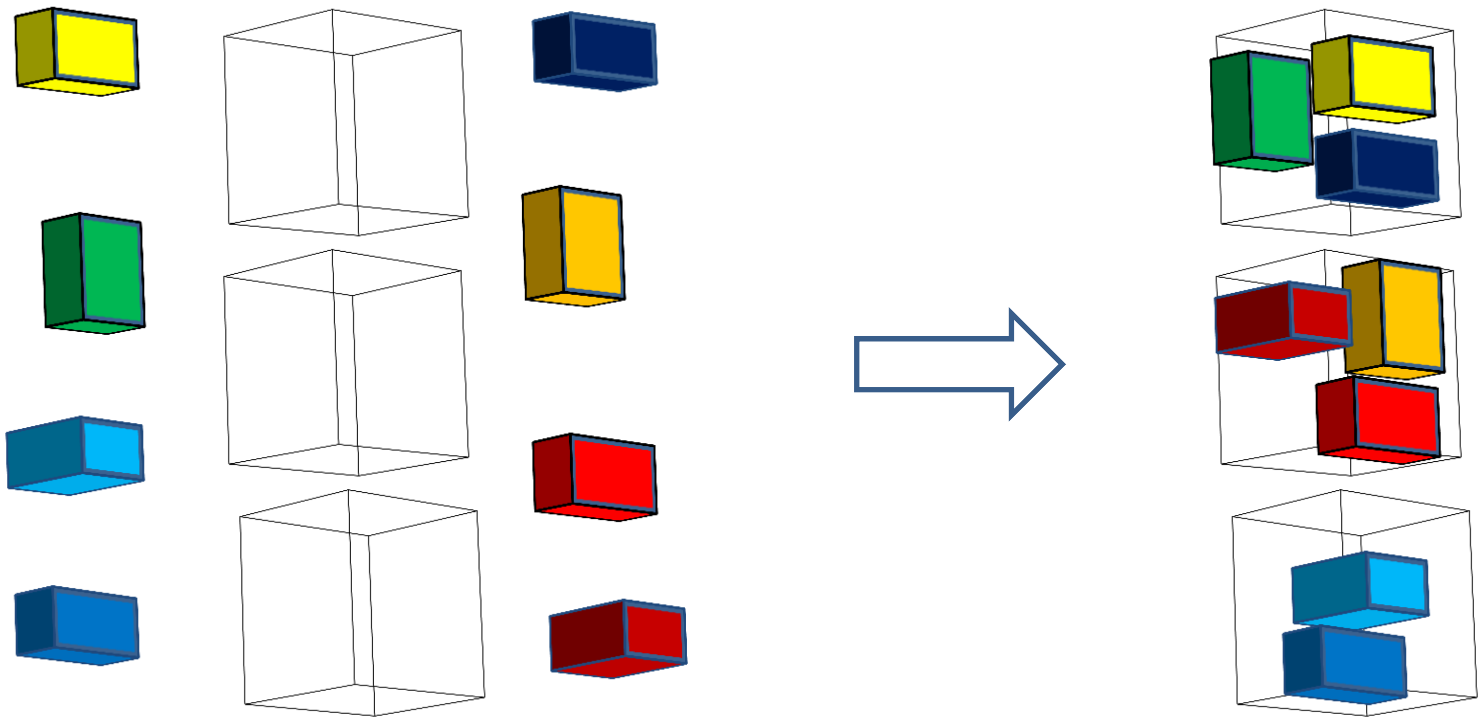


Modeling the Problem

Summary and Definitions

Possible mapping shortfalls : Too spread out

Low chance of performance issues due to interference from other VMs, but possibly using more hardware/licenses than needed

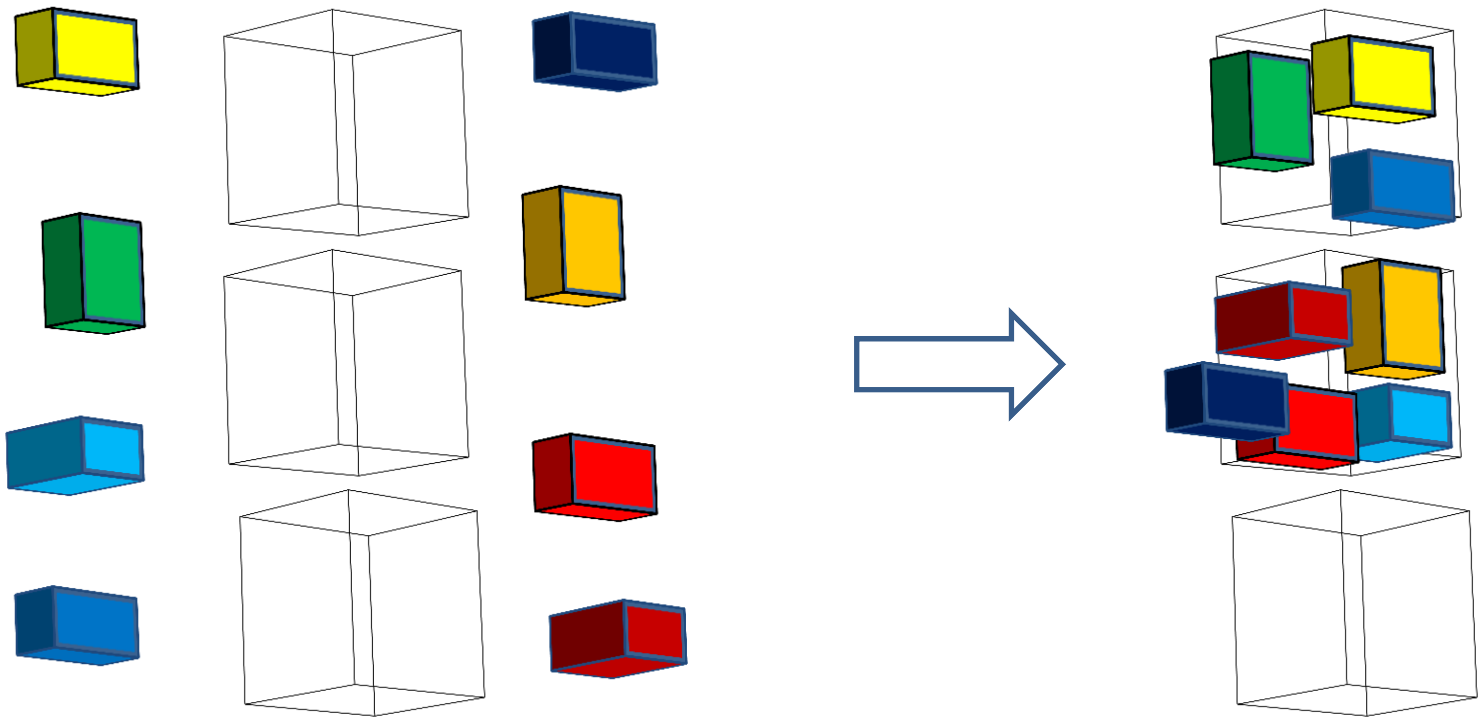


Modeling the Problem

Summary and Definitions

Possible mapping shortfalls : Too tight

Minimal hardware/licenses costs, but with significant chance of performance issues due to interference from other VMs

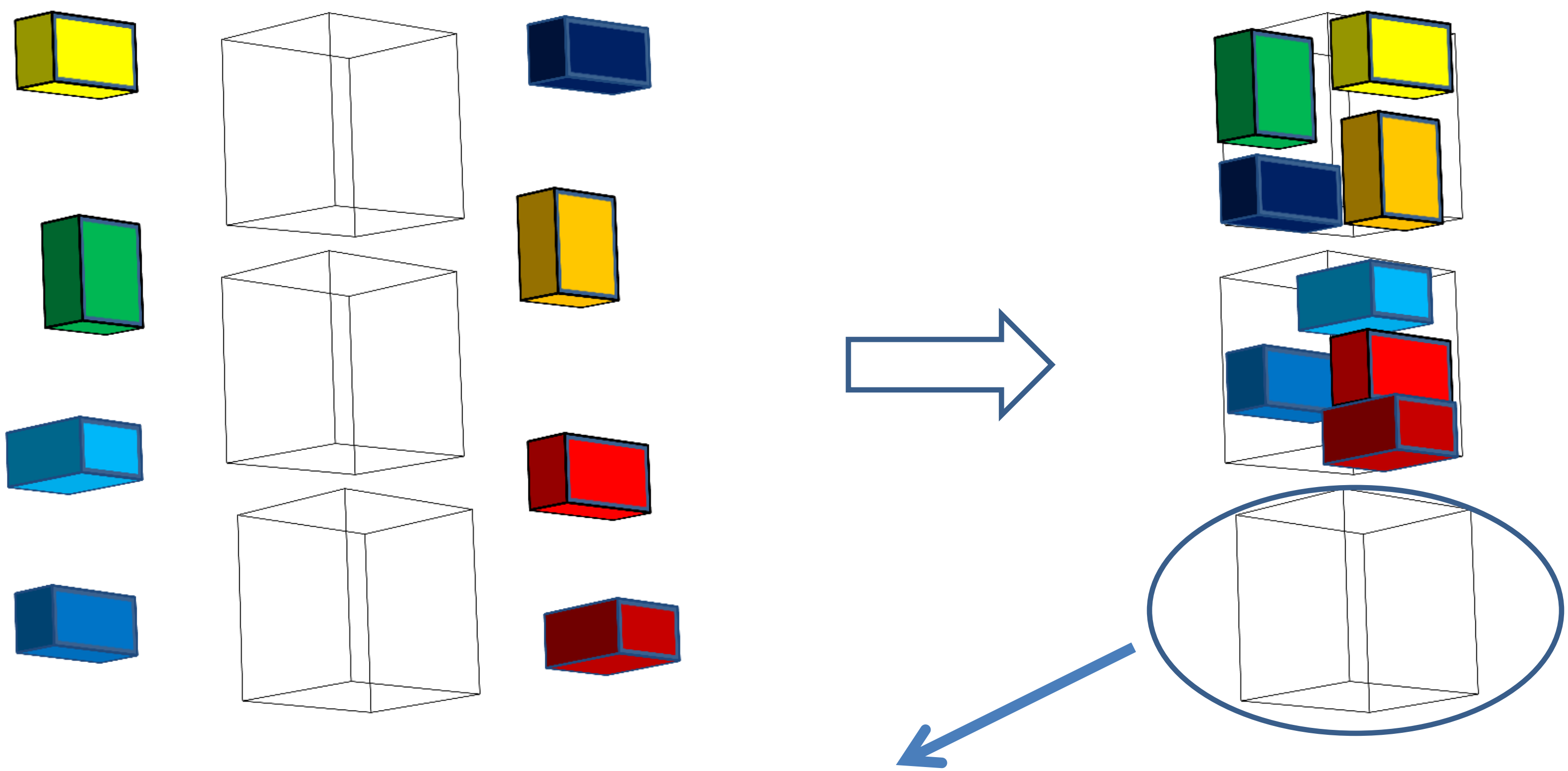


Modeling the Problem

Summary and Definitions

Just right!

Minimal hardware/licenses costs, with low chance of performance issues due to interference from other VMs



Less Hardware/ Licensing!

Modeling the Problem
Summary and Definitions

Find a mapping between virtual and physical servers $x_{i,j}$

$x_{i,j}$ = 1 if virtual server i will reside on physical server j , 0 otherwise

Minimize the number of CPUs (Since license cost are typically tied to the number of CPUs):

$$\sum_{j=1}^{n} x_j \, p^{cpu}{}_j$$

x_j = 0 iff $x_{i,j}$ = 0 for all I (physical server p_i not used)

x_j = 1 iff $x_{i,j}$ = 1 for al least one I (physical server p_i used)

Note*: The function to minimize can be modified if using Oracle approved hard partitioning*

Subject to :

For each virtual server is in one and only one physical server

For each virtual server j,

$\sum_{i=1}^{n} x_{i,j}$ = 1

Modeling the Problem
Constraints

Constraints allow us to specify performance and business requirements that the system must adhere to.

A common requirement is to allow no overallocation of computing resources

No CPU overallocation:

For each physical server i ,
$\sum_{i=1}^{m} x_{i,j} * v^{cpu}{}_{i} \leq = p^{cpu}{}_{i}$

No memory overallocation:

For each physical server I ,
$\sum_{i=1}^{m} x_{i,j} * v^{mem}{}_{i} \leq = p^{mem}{}_{i}$

No IO throughput overallocation:

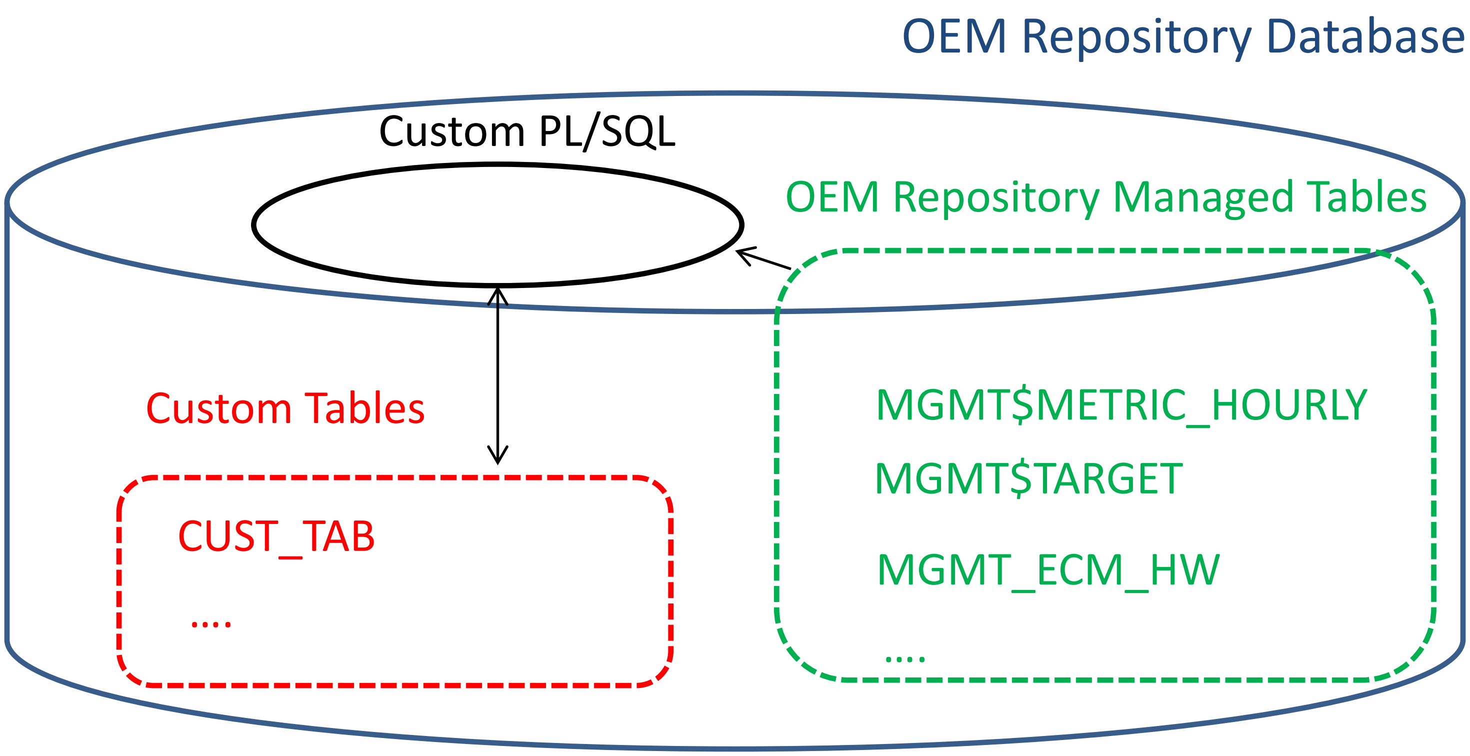
For each physical server I ,
$\sum_{i=1}^{m} x_{i,j} * v^{io\_thu}{}_{i} \leq = p^{io\_thru}{}_{i}$

Modeling the Problem
Implementation

- Custom Solutions (PL/SQL package)
  - ability to incorporate virtually any information
  - ability to customize to any specific environment/licensing need
- Off the Shelf Solutions (Oracle OEM/ Consolidation Planner)
  - minimal setup
  - easy to use

Modeling the Problem
Implementation/Custom

A repository, ideally in a database, is a great place to host the optimization logic.

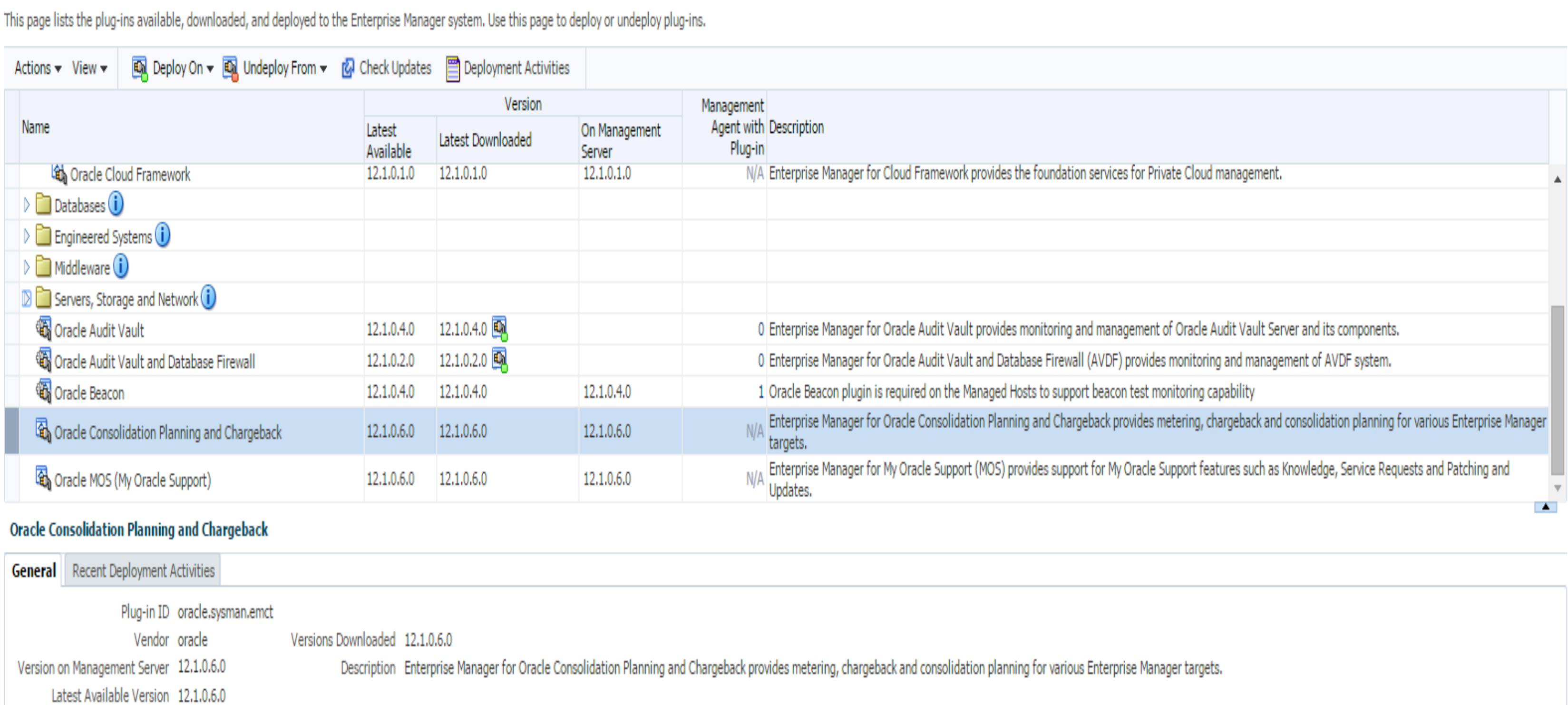


Modeling the Problem

Implementation/ Consolidation Planner

Off the shelf solutions: Oracle Enterprise Manager Consolidation Planner

Setup->Extendibility->Plugins



This page lists the plug-ins available, downloaded, and deployed to the Enterprise Manager system. Use this page to deploy or undeploy plug-ins.

Actions ▾ View ▾ Deploy On ▾ Undeploy From ▾ Check Updates Deployment Activities

Name	Version: Latest Available	Version: Latest Downloaded	Version: On Management Server	Management Agent with Plug-in	Description
Oracle Cloud Framework	12.1.0.1.0	12.1.0.1.0	12.1.0.1.0	N/A	Enterprise Manager for Cloud Framework provides the foundation services for Private Cloud management.
Databases					
Engineered Systems					
Middleware					
Servers, Storage and Network					
Oracle Audit Vault	12.1.0.4.0	12.1.0.4.0		0	Enterprise Manager for Oracle Audit Vault provides monitoring and management of Oracle Audit Vault Server and its components.
Oracle Audit Vault and Database Firewall	12.1.0.2.0	12.1.0.2.0		0	Enterprise Manager for Oracle Audit Vault and Database Firewall (AVDF) provides monitoring and management of AVDF system.
Oracle Beacon	12.1.0.4.0	12.1.0.4.0	12.1.0.4.0	1	Oracle Beacon plugin is required on the Managed Hosts to support beacon test monitoring capability
Oracle Consolidation Planning and Chargeback	12.1.0.6.0	12.1.0.6.0	12.1.0.6.0	N/A	Enterprise Manager for Oracle Consolidation Planning and Chargeback provides metering, chargeback and consolidation planning for various Enterprise Manager targets.
Oracle MOS (My Oracle Support)	12.1.0.6.0	12.1.0.6.0	12.1.0.6.0	N/A	Enterprise Manager for My Oracle Support (MOS) provides support for My Oracle Support features such as Knowledge, Service Requests and Patching and Updates.

Oracle Consolidation Planning and Chargeback

General | Recent Deployment Activities

Plug-in ID oracle.sysman.emct

Vendor oracle

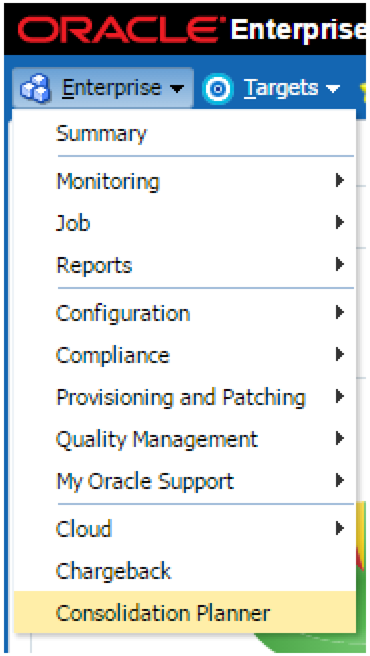
Version on Management Server 12.1.0.6.0

Latest Available Version 12.1.0.6.0

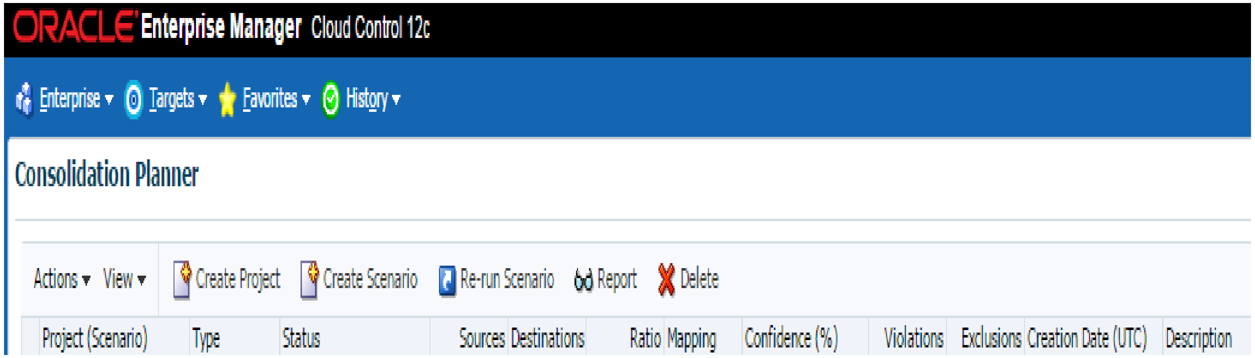
Versions Downloaded 12.1.0.6.0

Description Enterprise Manager for Oracle Consolidation Planning and Chargeback provides metering, chargeback and consolidation planning for various Enterprise Manager targets.

Modeling the Problem
Implementation/ Consolidation Planner



- Consolidation Project
  - defines the scope of the consolidation effort
- Consolidation Scenario
  - specific requirements and constraints



Modeling the Problem
Implementation/Custom

Allocating enough resources, such as virtual CPU (v^{cpu}_i), to be able to sustain maximum load (as per history) would minimize the likelihood of a performance problems related to resource utilization.

OEM Repository query for getting the max number of CPUs used:

```
SELECT MAX(ceil(m))
FROM
(         select
            max((a.maximum*c.cpu_count)/100) m
            , a.rollup_timestamp
          from
            mgmt$metric_hourly a ,
            mgmt$target b ,
            sysman.MGMT_ECM_HW c
          where     a.metric_name = 'Load'
          and       a.column_label = 'CPU Utilization (%)'
          and       a.target_guid = b.target_guid
          and       b.target_name = <hostname>
          and       c.hostname =  <hostname>
          and       c.vendor_name = 'Intel Based Hardware'
          group by a.rollup_timestamp)
```

Modeling the Problem
Constraints

Sizing for max load per day
Lower risk of contention
Larger footprint

Sizing for average load per hour
Higher risk of contention
Smaller footprint

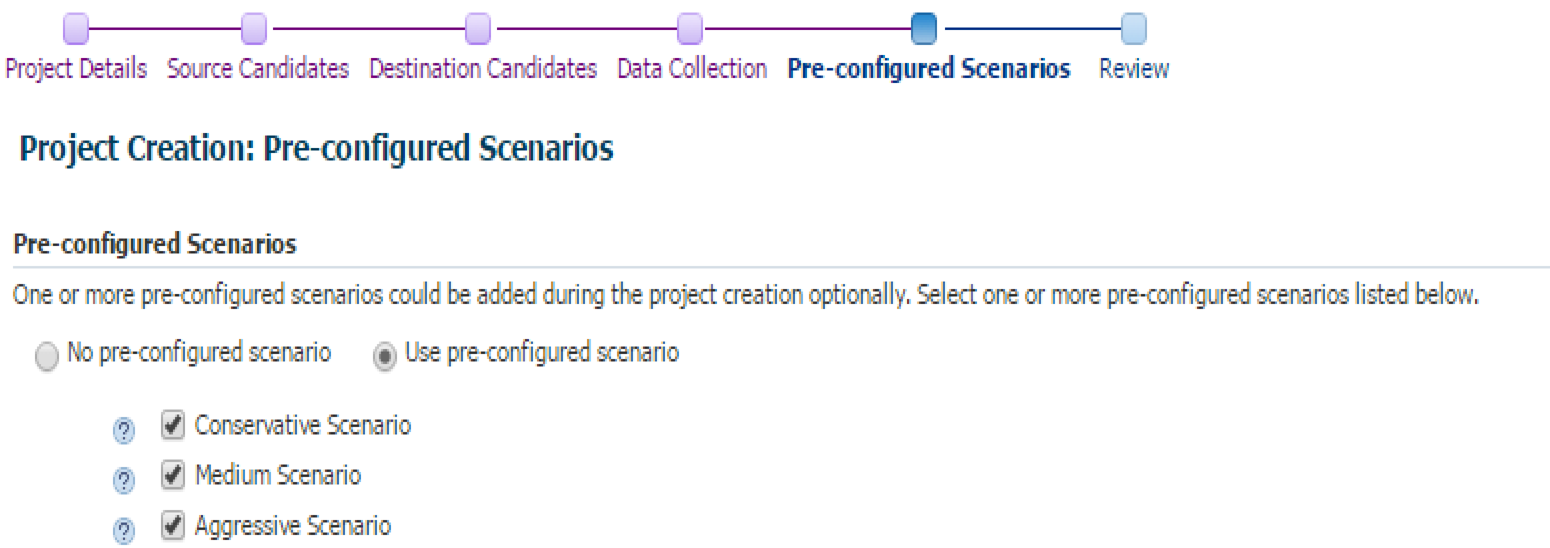
Sizing for max load can be quite conservative i.e. we are likely to get excellent performance, but we are going to allocate substantial resources.

We can switch the balance a little bit – we can slightly increase the chance of performance issues, but reduce the computational footprint.

We can archive that by taking into account the timing of the load. We can come with a configuration that would not have resulted in an overalloaction during any time of the past. Overallocation in future is possible if the timing of the workloads changes.

Modeling the Problem
Implementation/ Consolidation Planner

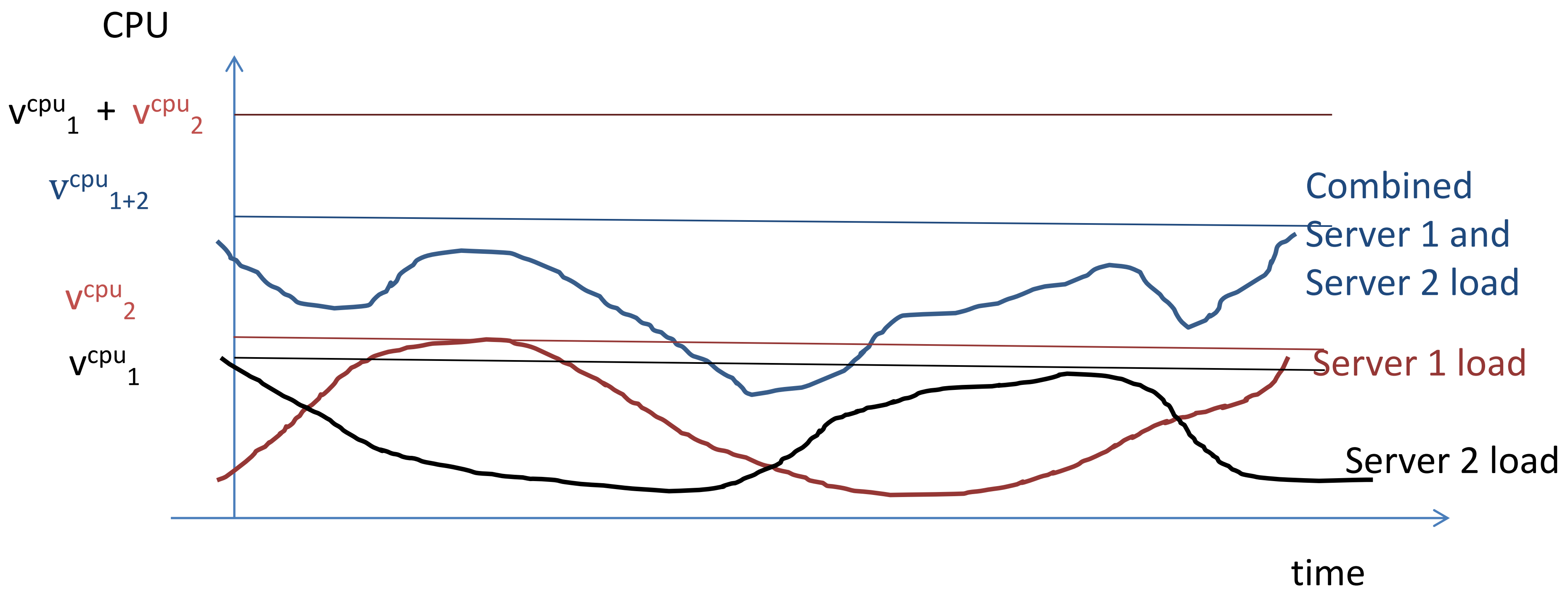
Consolidation planner comes with pre-configured scenarios for three different points on the contention/footprint scale



Modeling the Problem
Constraints

Sizing for max. load : $v^{cpu}_1 + v^{cpu}_2$

Sizing for max. combined load : $v^{cpu}_{1+2} < v^{cpu}_1 + v^{cpu}_2$



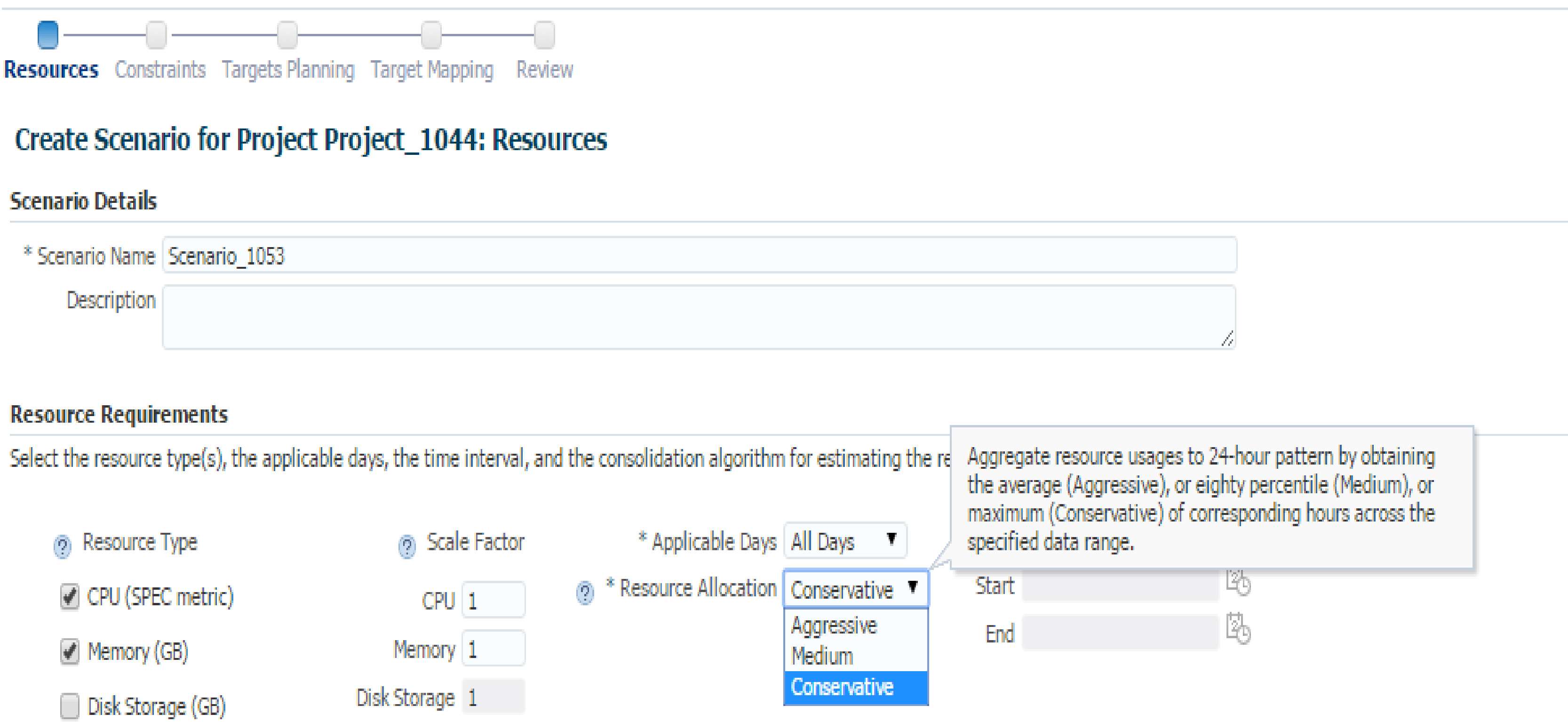
Modeling the Problem
Implementation/Custom

The following query checks if a list of virtual servers would fit in a physical server

```
SELECT COUNT(*)
FROM
  (SELECT SUM((a.average*c.cpu_count)/100) m
  FROM mgmt$metric_hourly a ,
    mgmt$target b ,
    sysman.MGMT_ECM_HW c
  WHERE a.metric_name = ''Load''
  AND a.column_label  = ''CPU Utilization (%)''
  AND a.target_guid   = b.target_guid
  AND b.target_name  IN ('||<list of virt servers>|| ' )
  AND c.hostname
    ||''.''
    ||c.domain      = b.target_name
  AND c.vendor_name = ''Intel Based Hardware''
  GROUP BY a.rollup_timestamp
  HAVING SUM((a.average*c.cpu_count)/100) > 0.9*'||<CPUs of physical server>
  )
```

Modeling the Problem
Implementation/Consolidation Planner

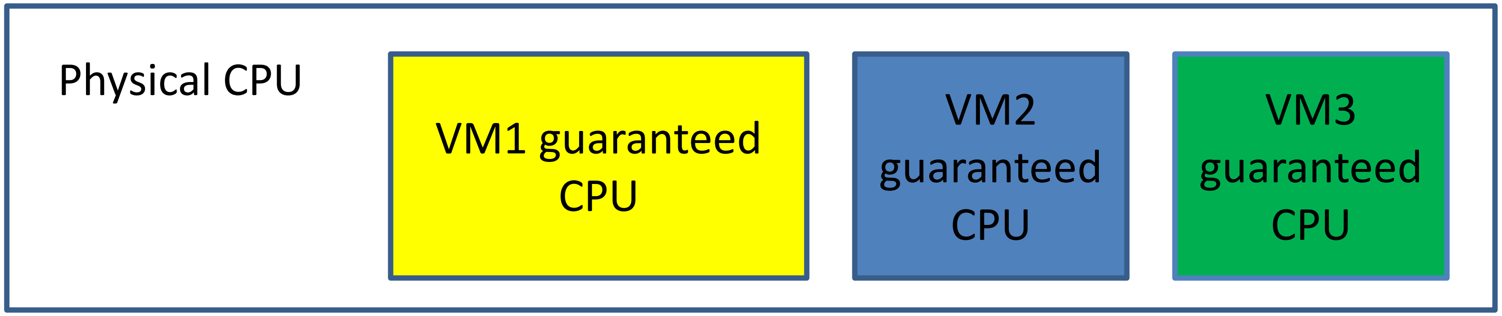
OEM Consolidation Planner can consider either max, 80% or average load.



Modeling the Problem
Constraints

Major drawback of over-allocation – if one of the VMs consumes unplanned amount of resources , the other VMs would suffer.

Some virtualization providers allow us to guarantee each of the VM certain level of resources (CPU/memory) in case of over allocation.

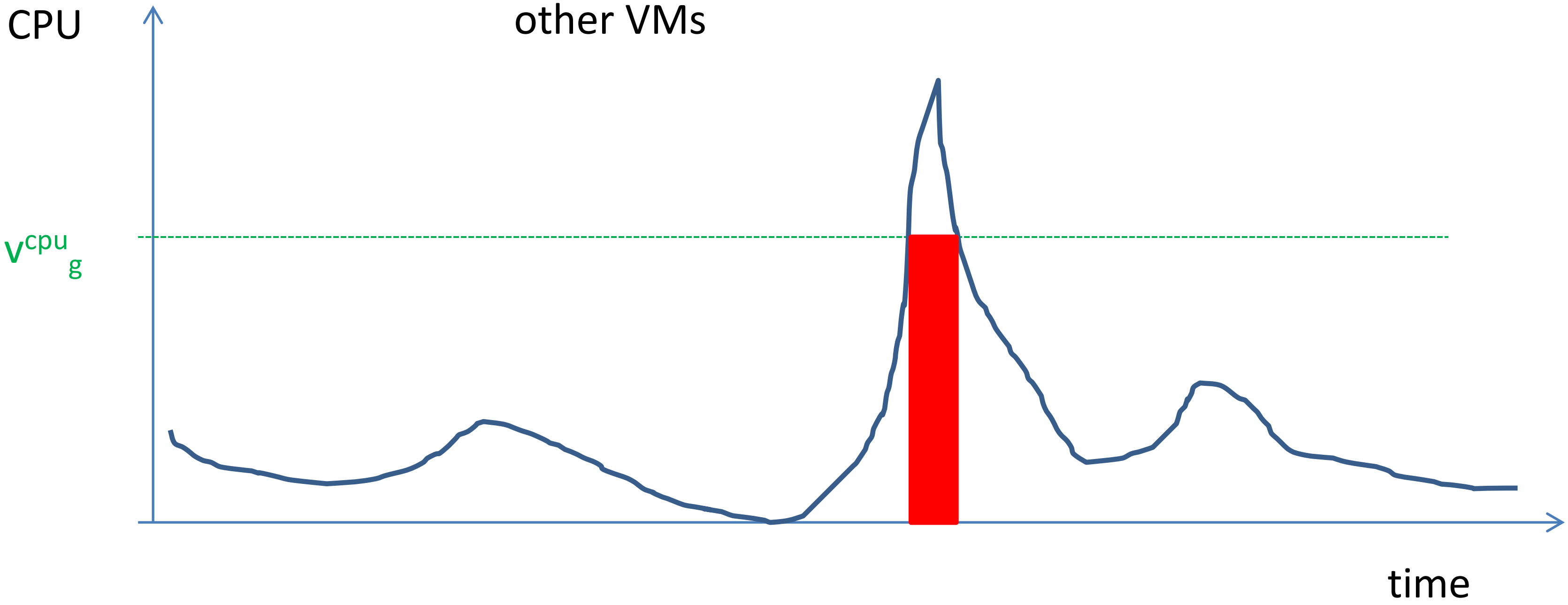


Physical CPU > VM1 guaranteed CPU + VM2 guaranteed CPU + VM3 guaranteed CPU

Modeling the Problem
Constraints

A reasonable compromise is to guarantee that under distress each VM will get resources that would be enough to accommodate the load in 95% of the time

V^{cpu}_{g} - CPU a VM is guaranteed to receive regardless of activities in other VMs



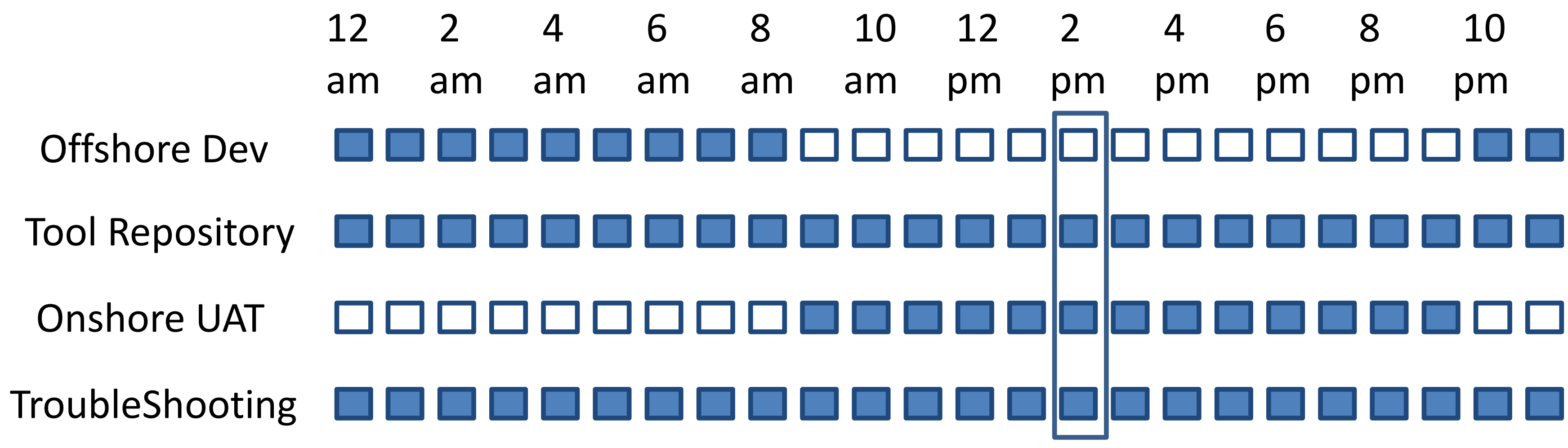
Modeling the Problem
Implementation/Custom

Find CPU level that is enough for the system 95% of the time

```
SELECT MAX(ceil(m))
FROM
      (SELECT MIN(m) m
      FROM
        (SELECT m ,
          percent_rank () over ( ORDER BY m) perc
        FROM
          (SELECT MAX((a.maximum*c.cpu_count)/100) m ,
            a.rollup_timestamp
          FROM mgmt$metric_hourly a ,
            mgmt$target b ,
            sysman.MGMT_ECM_HW c
          WHERE a.metric_name = 'Load'
          AND a.column_label  = 'CPU Utilization (%)'
          AND a.target_guid   = b.target_guid
          AND b.target_name   = i.hostname
            ||'.<domain_name>.com'
          AND c.hostname    = i.hostname
          AND c.vendor_name = 'Intel Based Hardware'
          GROUP BY a.rollup_timestamp
          )
        )
      WHERE perc > 0.95
    )
```

Modeling the Problem
Constraints

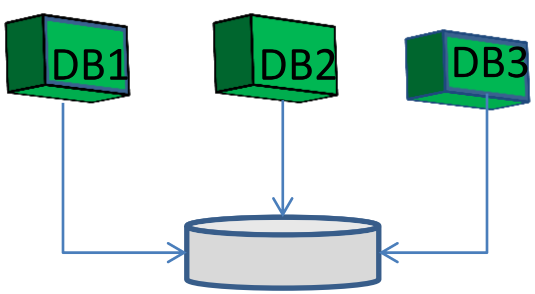
Reducing computational footprint by recognizing that some servers are needed only during certain hours.
Most business requirement checks should be done for every hour of the day, taking into account which servers are active then.

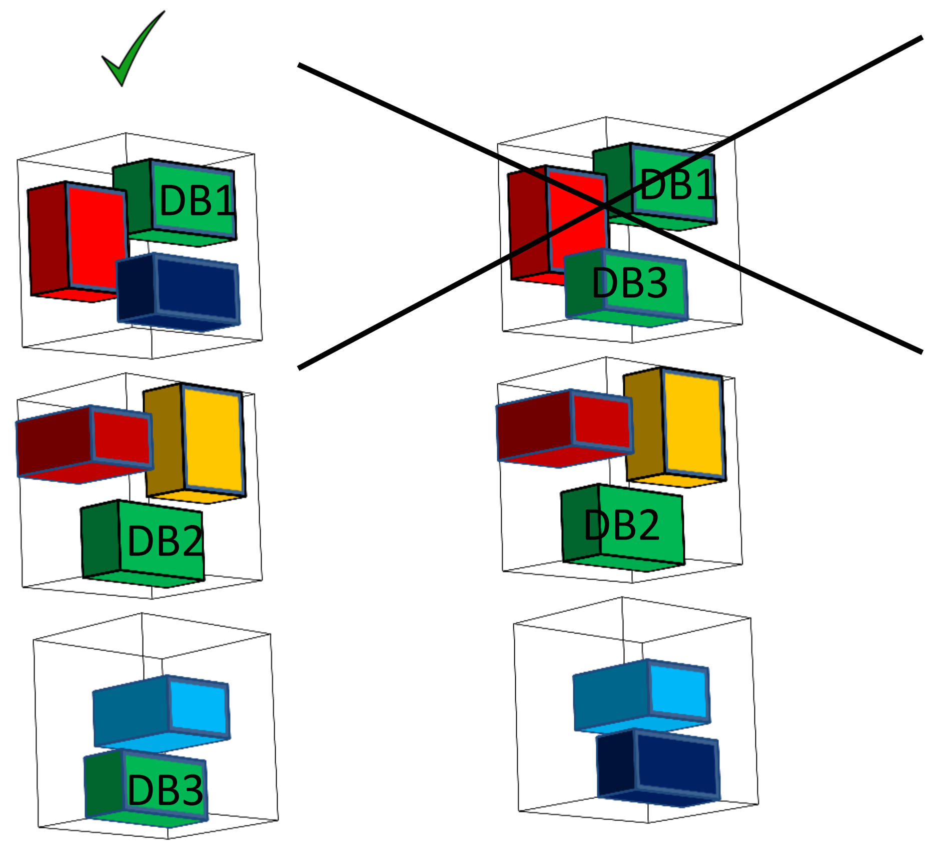


Modeling the Problem
Constraints

No two nodes of a RAC cluster should be on the same physical server

RAC cluster:

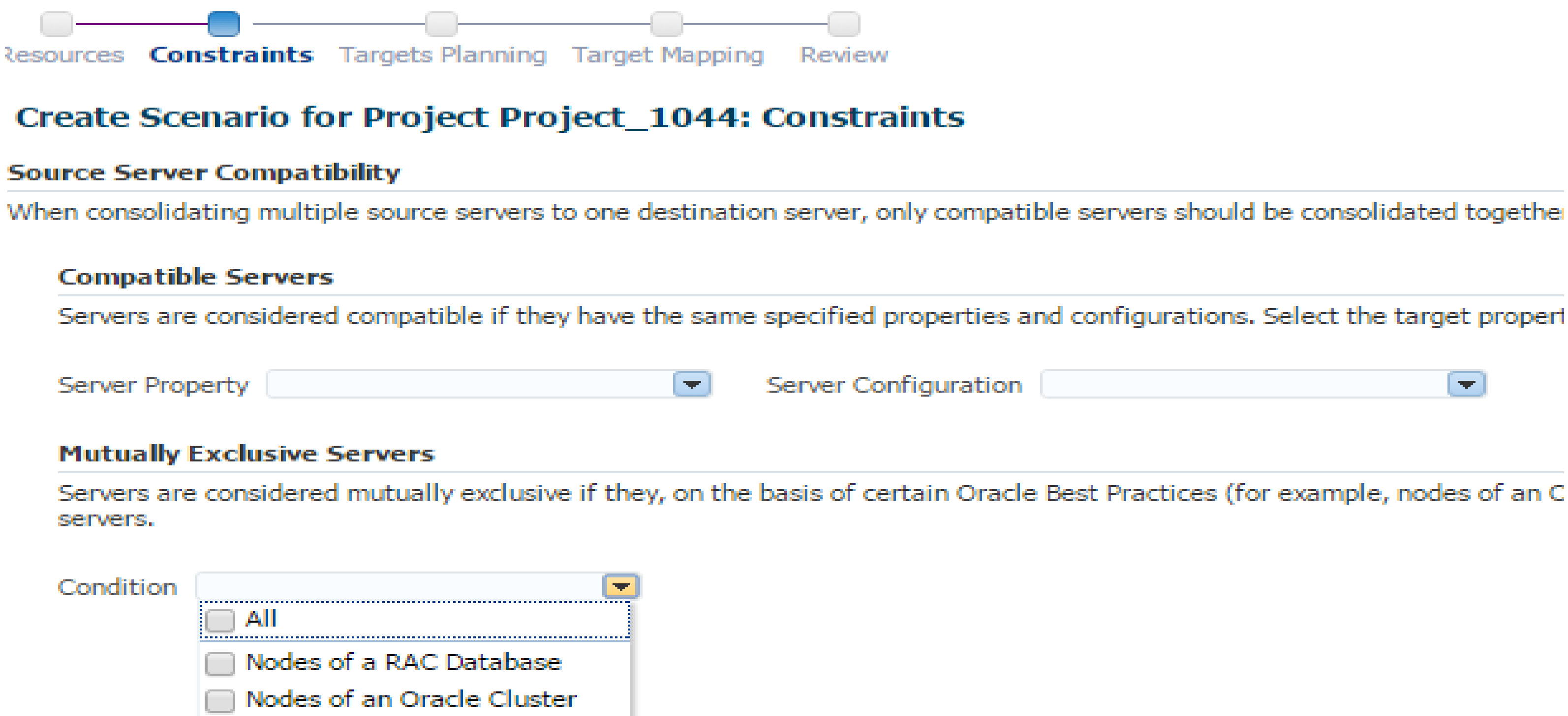




Modeling the Problem

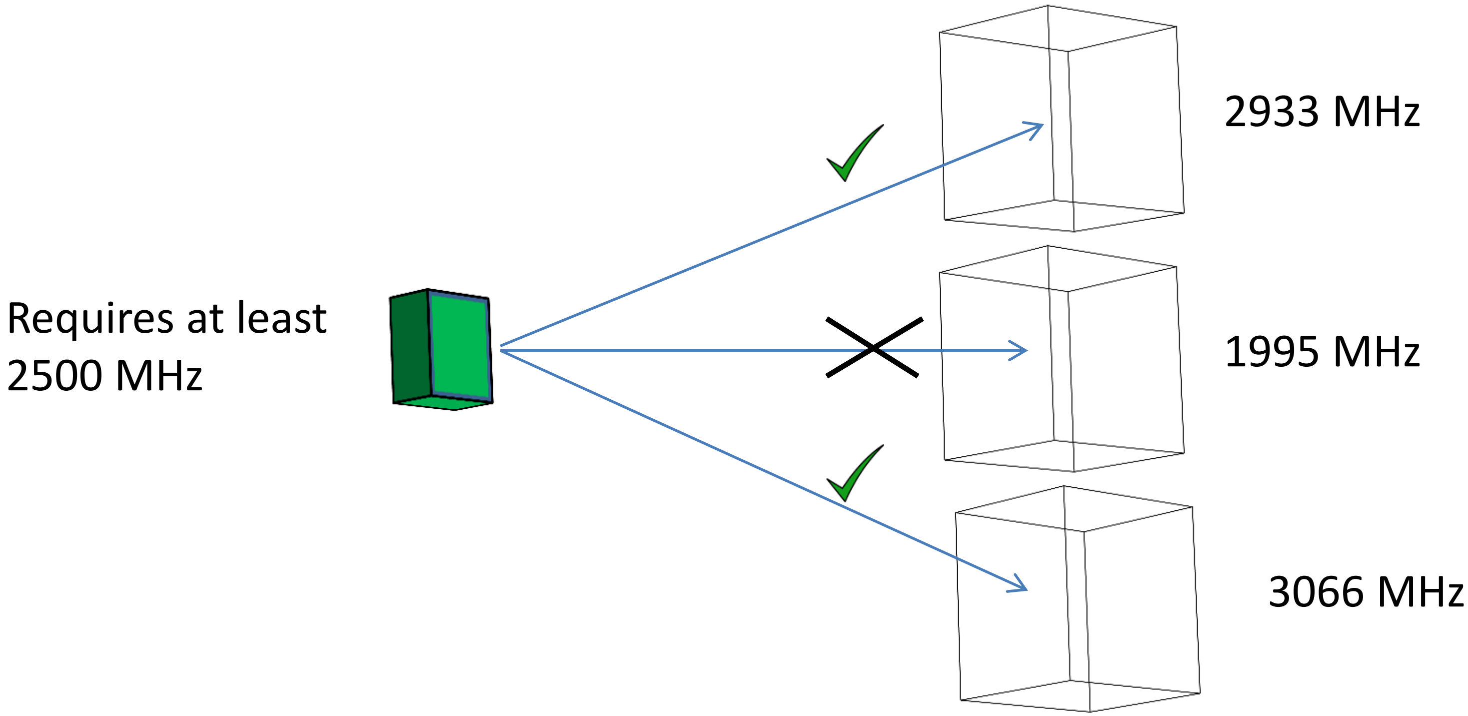
Implementation/ Consolidation Planner

Specifying the RAC nodes constraint in the Consolidation Planner



Modeling the Problem
Constraints

Guarantee that a virtual machine runs on a physical server that has sufficient CPU speed



Solving the Model

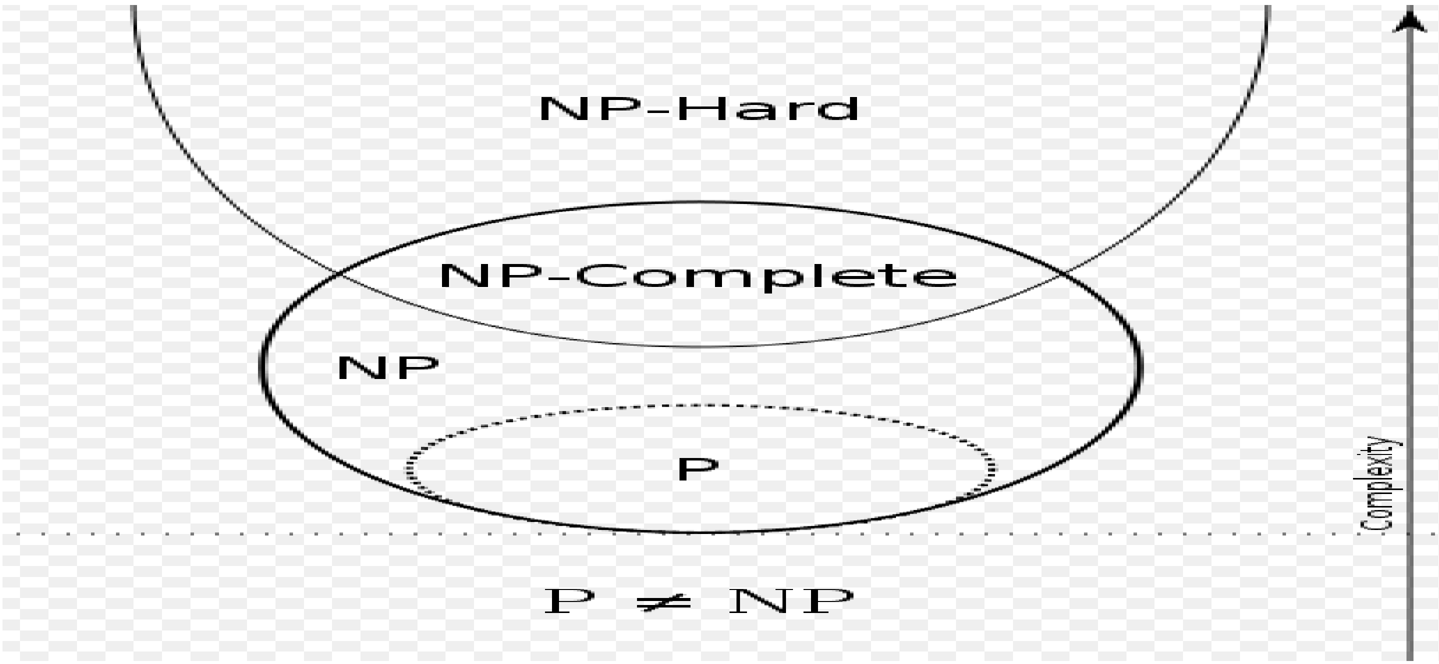
Discrete Optimization

Computational Complexity of Optimization Problems

- P – can be solved in polynomial time
- NP – the solution can be verified in polynomial time
- NP hard – at least as difficult as any problem in NP
- NP complete – NP hard and in NP

Solving the Model
Discrete Optimization

Computational Complexity of Optimization Problems



From:

http://en.wikipedia.org/wiki/NP-hard#mediaviewer/File:P_np_np-complete_np-hard.svg

Solving the Model
Discrete Optimization

Finding the optimal solution for many real world problems may require enormous, frequently impractical, amount of computing resources.

We usually need to settle for good, but not necessary optimal solutions. Here are some major techniques in Discrete Optimization:

- Constraint Programming
- Local Search
- Linear and Integer Programming

Solving the Model
Discrete Optimization

The problem we are trying to solve here can be considered a variation of the offline variable size Bin Packing Problem (BPP).

Given:
N items, each with weight w_j
M bin, each with capacity c_i

Minimize:$\sum c_i$,
for all bins which have at least one item

Subject to:
Each item must be in exactly one bin
$\sum w_j < c_i$, for all items that are in bin i

Solving the Model
Discrete Optimization

Heuristics for solving classic BPPs

- Next-Fit: Put in as many items as possible in a bin, then move to the next one.

- First Fit: Put an item in the first bin that fits it. Start using a new bin only after trying all partially filled bins

- Best-Fit: Assign items in a way that minimizes the residual capacity of a bin

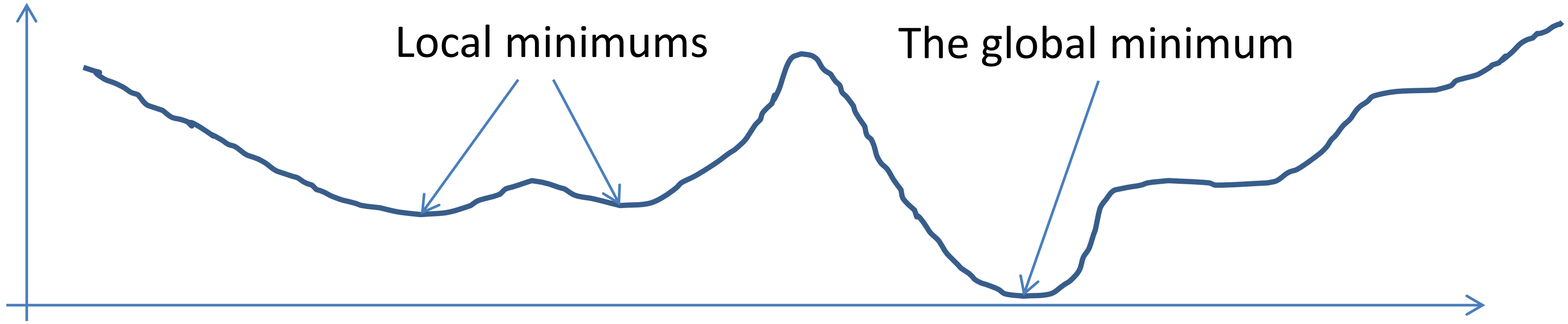
- Next-Fit Decreasing: Same as Next-Fit, but have the items ordered in decreasing order

Solving the Model

Discrete Optimization

Randomization – a simple way to minimize the risk of a bad solution. It has intuitive local search interpretation.

- Starting from (somehow) random starting position
- Random hill-climbing moves
- Simulated Annealing – randomly allowing moves that do not improve the solution



Conclusion

- Getting optimal virtual server consolidation is more of a science than an art
- Doing optimal virtual server consolidation right requires time and efforts, but it can have significant ROI
- There is no need to look for absolute optimality - getting a great, though not optimal solution, can make a huge difference.

Thank you