Approaches to Improving the Efficiency of Data Centers

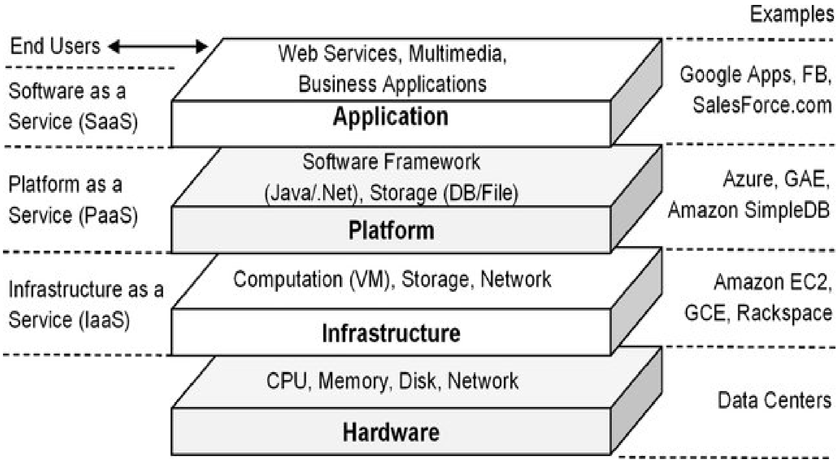
Andrey Chupakhin

Research Fellow, PhD in Computer Science

Lomonosov Moscow State University

Huawei Cloud InnovWave 2021 conference

Data Centers and Clouds



Efficiency of the Cloud – Metrics

- Utilization of the physical resources:
  - utilization of computing and storage resources $\to$ max
  - utilization of network resources (compactly assignment) $\to$ min
- Percentage of assigned Virtual Networks (VNs) relative to the total number of Virtual Networks
- Performance of Virtual Machines (VMs)
- Energy consumption and cost-effective scheduling

Efficiency of the Cloud – Problem & Solutions

- Resources fragmentation – VM Live migration
- “wide” VM – Non-Uniform Memory Access (NUMA)
- Research and setup optimal VM oversubscription coefficient

Virtual Machines Assignment Policies

For the VMs, the following assignment policies may be additionally specified:

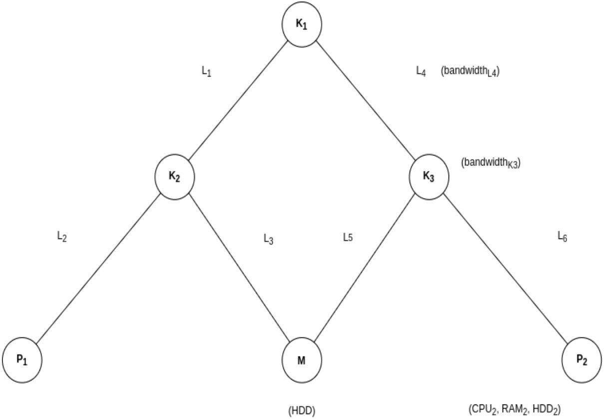
- For each VM, a set of physical servers on one of which it should be located; this is called the **VM-to-PM** affinity rules
- For each VM, a set of physical servers on which it cannot be located; this is called the **VM-to-PM** anti-affinity rules
- The set of VMs that must be located on the same physical server; this is called the **VM-to-VM** affinity rules
- The set of VMs that must be located on different physical servers; this is called the **VM-to-VM** anti-affinity rules

Model of Cloud Physical Network

Physical resources graph $H = (P \cup M \cup K, L)$

- **Nodes**
  - Servers – P
    - $(ph_1, ph_2, ..., ph_{n1}) = vh(p), p \in P$
  - Storages – M
    - $(mh_1, mh_2, ..., mh_{n2}) = uh(m), m \in M$
  - Routers and switches – K
    - $(bh_1, bh_2, ..., bh_{n3}) = bh(k), k \in K$
- **Edges**
  - Physical links – L
    - $(lh_1, lh_2, ..., lh_{n4}) = rh(l), l \in L$

Model of Cloud Physical Network

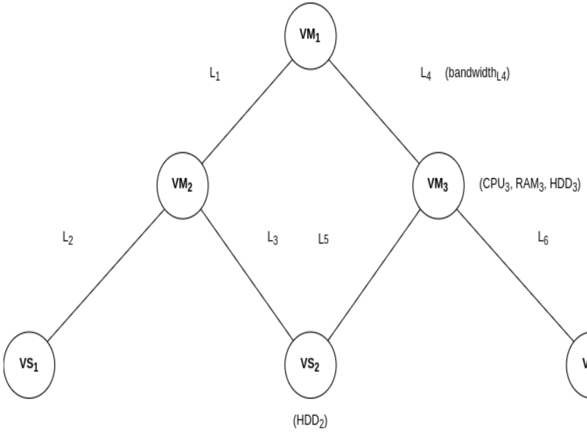


Model of Virtual Network

Virtual resources graph: $G = (W \cup S, E)$

- **Nodes**
  - Virtual machines – W
    - $(wg_1, wg_2, ..., wg_{n1}) = fwg(w), w \in W$
  - Virtual storages – S
    - $(sg_1, sg_2, ..., sg_{n2}) = fsg(s), s \in S$
- **Edges**
  - Virtual links – E
    - $(eg_1, eg_2, ..., eg_{n3}) = feg(e), e \in E$

Model of Virtual Network



Virtual Network Embedding

Virtual network embedding (or virtual network assignment) is a mapping $A : G \to H = \{W \to P, S \to M, E \to \{K, L\}\}$ with the following constraints:

1. The physical resource cannot be overloaded: $\sum_{i \in R_j} x_i \leq y_j$, where R_j is the set of requests assigned to the physical resource j.
2. The types of the physical and virtual resources must match each other: $x_i = y_j$
3. The physical resource must have the required characteristics: $x_i \leq y_j$
4. (**servers only**) Not allowed to overload NUMA blocks on physical servers
5. (**servers only**) Each VM can only be assigned on one NUMA block of a specific physical server

The mapping is correct if the constraints 1-5 and the VM placement policies are satisfied for each request.

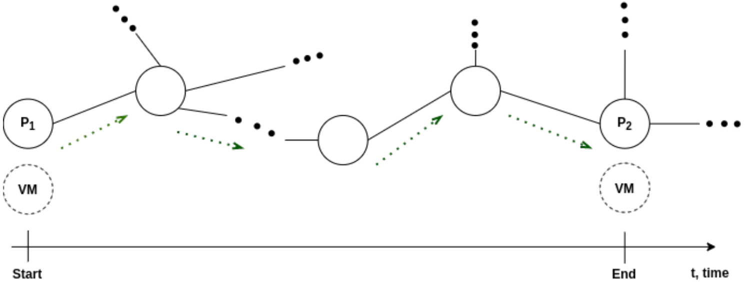
Residual Resources Graph

H_{res} is a copy of H graph with redefined values of characteristics are according constraint 1:

- $vh_{res}(p) = vh(p) - \sum_{w \in W_p} fwg(w), p \in P$
- $uh_{res}(m) = uh(m) - \sum_{s \in S_m} fsg(s), m \in M$
- $bh_{res}(k) = bh(k) - \sum_{k \in R_k} feg(k), k \in K$
- $rh_{res}(l) = rh(l) - \sum_{e \in E_l} feg(e), l \in L$

Migration Plan for Virtual Machines

- The sequence of single migration for each VM
- For each VM migration the start and end of the migration are defined
- All migrations must be completed within the specified directive interval T
- Work intensity for each VM
- For each VM migration the path in the physical resources graph H is found



Virtual Network Assignment Problem

- **Input**
  - New virtual networks $Z = \{G_i\}$
  - Already assigned virtual networks $B = \{G_j\}$ and its mapping $A_B : B \rightarrow H$
  - The residual graph of available resources: H_{res}
  - Migration time limit: T_{dir}
- **Find**
  - Find mappings:
    - $A_L : L \subseteq Z \rightarrow H_{res}$
    - $A_M : M \subseteq B \rightarrow H_{res}$
    - $|L| \rightarrow max$
  - Build a migration plan for a set M without violating the migration time limit T_{dir}

Principles of Constructed Algorithm

Greedy strategies + **limited search**: At each step, the algorithm selects an VN from a set of not assigned VNs and find placement in accordance with the greedy criterion.

1. **Approach** 1. The limited search procedure is called if the selected VN cannot be assigned;
2. **Approach** 2. The limited search procedure is called if, after a trial placement of the selected VN, VNs that cannot be placed appear in the set of unassigned VNs.

For the procedure of limited iteration, the maximum allowable **depth** of iteration is defined. It determines the maximum possible number of VNs that can participate in the search.

V. A. Kostenko. Combinatorial Optimization Algorithms Combining Greedy Strategies with A Limited Search Procedure // Journal of Computer and Systems Sciences International, 2017, Vol. 56, No. 2, pp. 218–226. DOI:10.1134/S1064230717020137

Algorithm

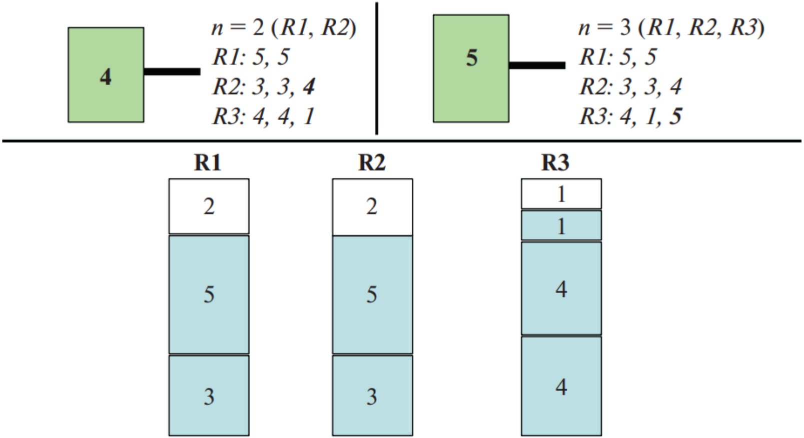
1. Sorting Virtual Networks ($C1$)
2. Choose Virtual Network
3. Policies satisfaction
4. Choose element in Virtual Network ($C2$)
5. Create set of physical servers according to 1-5 constraints (set A)
   - If $A! = \emptyset$
     - Choose physical element ($C3$)
     - If server was chosen then choose NUMA block ($C4$)
     - Assigned element. Change H_{res}
   - If $A = \emptyset$
     - Limited research procedure. If unsuccessful **goto to Step 6**
     - Create migration plan. If unsuccessful **goto to Step 6**
6. Cancel assignment of Virtual Network's elements

$$C_i, i = 1, 2, 3, 4 \text{ – greedy criterias}$$

Greedy Criterias

- Parameters for nodes or edges in graph G and H: $(r_{e,1}, r_{e,2}, ..., r_{e,n})$
- Parameter's deficit: $d(i) = \frac{\sum_G \sum_{e \in E, E \in G} r_{e,i}}{\sum_{ph \in Ph} r_{ph,i}}$
- Element's cost: $r(e) = \sum_{i=1..n} d(i) * \frac{r_{e,j}}{\max_{G, v \in G} r_{v,i}}$
- $weight(e) = r(e) * \sum_l Throughtput(l)$
- C1 – choose Virtual Network G
- C2 – choose virtual element in G
- C3 – choose physical element in H
- C4 – choose NUMA block in server

Limited Research Procedure



Experiments - Description

	PHYSICAL PARAMETERS			VIRTUAL PARAMETERS			
Experiment	CPU	RAM	NETWORK	CPU	RAM	NETWORK	All requests
A1	12046	21073	19581	12610	25156	9806	240
A2	29997	52317	54677	27416	54862	21754	520
A3	59865	105483	115282	58791	117340	46684	1100
A6	4528	10537	14699	6607	13156	5592	120
A7	22444	52236	69821	27874	55736	22549	520
A8	45146	104403	146019	58258	116627	46068	1100
A11	12024	18074	24839	12940	25935	10372	240
A12	30179	44996	69893	28661	57083	23100	520
A13	59941	89850	144761	57020	114198	43732	1100
A16	4483	9067	11748	6466	12847	5374	120
A17	22411	44963	55806	27319	54639	21643	520
A18	45121	90101	114836	58499	116922	46455	1100

Experiments - Results

VCPUs	RAM	NETWORK	ASSIGNED	TIME
0.70 / 0.65	0.80 / 0.75	0.88 / 0.88	145 / 127	3 mins 52 secs / 3 mins 28 secs
0.68 / 0.66	0.79 / 0.76	0.76 / 0.73	345 / 329	7 mins 25 secs / 111 mins 54 secs
0.69 / 0.66	0.78 / 0.75	0.71 / 0.69	688 / 634	50 mins 42 secs / 29 hours 54 mins
0.90 / 0.88	0.76 / 0.75	0.81 / 0.78	62 / 59	50 secs / 26 secs
0.90 / 0.87	0.77 / 0.75	0.65 / 0.60	324 / 300	1 mins 37 secs / 2 hours 1 mins
0.90 / 0.88	0.78 / 0.76	0.59 / 0.55	645 / 603	9 mins 47 secs / 29 hours 48 mins
0.66 / 0.50	0.88 / 0.67	0.77 / 0.53	132 / 89	1 mins 41 secs / 3 mins 54 secs
0.68 / 0.67	0.91 / 0.89	0.66 / 0.62	318 / 293	1 mins 11 secs / 2 hours 6 mins
0.67 / 0.66	0.9 / 0.89	0.56 / 0.54	672 / 642	9 mins 11 secs / 27 hours 30 mins
0.81 / 0.73	0.80 / 0.72	0.87 / 0.81	61 / 51	1 mins 39 secs / 3 mins 3 secs
0.81 / 0.82	0.80 / 0.82	0.73 / 0.70	296 / 286	6 mins 16 secs / 109 mins 48 secs
0.82 / 0.80	0.82/ 0.81	0.66 / 0.66	618 / 554	8 mins 28 secs / 28 hours 30 mins

Properties of Algorithm

- Migration plan for virtual machines
- Ability to define the required set of parameters for physical and virtual elements
- Assignment policies for virtual machines: hard, soft, hybrid mode
- Consider NUMA architecture when assign VM
- Balance between the algorithm's quality and execution time
- Implemented algorithm can work in online mode, i.e. process VNs sequentially

Conclusions and Future Work

Conclusions:

- Algorithm with NUMA, Policies, Migration;
- Tested on different datasets: real and synthetic;
- The algorithm gives goods results on tree and fat-tree topologies.

Future works:

- Improvement of VMs live migration plan;
- VM oversubscription;
- Energy consumption and cost-effective scheduling.

Articles

- Vasil'eva Y. O., Kostenko V. A., Chupakhin A. A. Effect of virtual machine deployment policies on the efficiency data processing centers // Journal of Computer and Systems Sciences International. — 2020. — Vol. 59, no. 3. — P. 400–405.
- Kostenko V. A., Chupakhin A. A. Live migration schemes in data centers // Programming and Computer Software. — 2020. — Vol. 46, no. 5. — P. 312–315.
- Kostenko V. A., Chupakhin A. A. Approaches to improving the efficiency of data centers // Programming and Computer Software. — 2019. — Vol. 45, no. 5. — P. 251–256.
- Kostenko V. A., Plakunov A. V. Ant algorithms for scheduling computations in data processing centers // Moscow University Computational Mathematics and Cybernetics. — 2017. — Vol. 41, no. 1. — P. 44–50.
- Zotov I. A., Kostenko V. A. Resource allocation algorithm in data centers with a unified scheduler for different types of resources // Journal of Computer and Systems Sciences International. — 2015. — Vol. 54, no. 1. — P. 59–68.
- Comparing various approaches to resource allocating in data centers / P. M. Vdovin, I. A. Zotov, V. A. Kostenko et al. // Journal of Computer and Systems Sciences International. — 2014. — Vol. 53, no. 5. — P. 689–701.

Thank You!

- **Email**:
  - Valery Kostenko kost@cs.msu.su
  - Andrey Chupakhin andrewchup@lvk.cs.msu.ru
- **Code**: https://github.com/andxeg/master_course
- **Code**: https://github.com/andxeg/bachelor_diploma_2016