Power-aware Virtual Machine Placement and Routing

CSC547/447 Green Computing - Junxiao Shi, Gavin Simons, Nick Crutchfield

VMPR Introduction

J. Jiang, et al, Joint VM Placement and Routing for Data Center Traffic Engineering

VMPR problem

Assign VMs onto PMs, so that every VM has enough local resources, and network congestion is minimized.
A job has several VMs. Resource demands of each VM and traffic matrix between VMs are known and fixed. VMs from different jobs do not communicate.

VMPR is formulated as an optimization problem.
Objective: minimize network cost and node cost. Network cost is an increasing function of link utilization; node cost is the number of active machines.
Constraints: PM must have enough local resources to run all VMs. Link utilization must be below threshold.

VMPR problem is a combinational optimization and is NP-hard. VMPR(β) is a Markov chain approximation of VMPR problem. Upon job arrival and job departure, a new configuration is chosen probabilistically, so that we can operate on close-to-optimal configurations.

VMPR-online

Upon job arrival, the heuristic searches the feasible configurations (adding new VMs into old configuration), and estimates the system performance (higher for lower host utilization and lower network utilization) of each configuration; a configuration is chosen with probability proportional to the exponential of the system performance.

Upon job departure, the heuristic selects the most congested network edge, explores the feasible configurations by migrating one VM using that edge, and estimates the system performance; a configuration is chosen probabilistically.

Note: we cannot simply choose the configuration of highest system performance, because we would be stuck in the local maximum.

VMPR discussion

Strengths

PM and VM Heterogeneity

Data center servers are heterogeneous, due to upgrade cycles and cost considerations. Virtual machine resource demands are also heterogeneous, because different workloads have different requirements.
The physical resources present on a PM and the resource demands of a VM are both described as vectors.

Can add restriction easily

Reliability considerations may require two VMs to be placed onto different PMs or even different racks. This kind of restriction is easy to implement.

Limitations

Resource reservation

Upon job arrival, resource demands and traffic matrix are both known. This is only true for HPC workload.
Resource demands and traffic matrix for web service workload will change overtime, and the change is not always predictable.

No communication between jobs

VMs from different jobs cannot communicate. This is only true for HPC workload.

No accounting of migration cost

VM migration has significant network cost in the general case. This cost is not considered in the solution.

Applicability

VMPR is applicable to a data center running HPC workload. For example, Google Compute Cluster is suitable to use VMPR.

Add Power-awareness to VMPR

Data center power consumption is mainly server power, network power, and cooling power.
New data center's PUE is becoming lower, because efficient cooling systems such as outside

air cooling system are installed. Therefore, server power and network power will be the focus of power optimizations in new data centers, and special arrangement for cooling is no longer necessary.

Recall that VMPR's objective is to minimize network cost and node cost. Network cost is an increasing function of link utilization; node cost is the number of active machines. To add power-awareness, we can simply replace them with the estimated network power and server power.

Optimization Model

This section describes an optimization model that applies VMPR in power-aware aspect. It represents a VMPR snapshot problem, and could be solved with the VMPR-online heuristic.

The model is a combination of VMPR (physical resource constraint) and GreenTE (candidate paths and sleeping links / line cards). A major change from GreenTE is: candidate paths are pre-computed for a pair of physical machines, but the model decides on a path for each pair of VMs, instead of combining traffic onto physical machine pair and deciding a split ratio. Therefore, the path for a pair of VMs will not change during the lifetime of the job unless a VM is migrated.

There are up to k paths between two different physical machines. When a pair of VMs are allocated onto the same physical machine, the traffic between them are routed within the host; for every physical machine h, we define 1 path h-to-h that doesn't contain any link, to account for this case.

maximize	$\sum_{l \in E} P_l x_l + \sum_{m \in M} B_m y_m + \sum_{h \in H} S_h Y_h$	(1)
s.t.	$\sum_{j \in J} \sum_{v \in W_j} Z_{j,v}^{h} \cdot S_{j,v} \preceq Y_h \cdot H^h, h \in H$	(2)
	$\sum_{s,t \in H} \sum_{0 \le i < k} A_{j,v,v'}^{s,t,i} = 1, j \in J, v, v' \in W_j$	(3)
	$\sum_{t \in H} \sum_{0 \le i < k} A_{j,v,v'}^{s,t,i} = Z_{j,v}^{s}, j \in J, v, v' \in W_j, s \in H$	(4)
	$\sum_{s \in H} \sum_{0 \le i < k} A_{j,v,v'}^{s,t,i} = Z_{j,v'}^{t}, j \in J, v, v' \in W_j, t \in H$	(5)
	$f_l^{j,v,v'} = \sum_{s,t \in H} \sum_{0 \le i < k} Q_i^{s,t}(l) \cdot R^j[v,v'] \cdot A_{j,v,v'}^{s,t,i}$	(6)
	$u_l = \frac{1}{C_l} \sum_{j \in J} \sum_{v,v' \in W_j} f_l^{j,v,v'}, l \in E$	(7)

	$u_l < U_T$	(8)
	$x_l = x_{r(l)}$	(9)
	$x_l + u_l \leq 1$	(10)
	$\lvert S_m \rvert y_m \leq \sum_{l \in S_m} x_l$	(11)

The objective (1) is to maximum the total power saving by putting links, line cards, and physical machines to sleep; here we assume every link / line card / physical machine consumes the same power regardless of utilization. Physical resource constraint (2) ensures every physical machine has enough physical resources of each type to run virtual machines. Constraint (3) ensures there's a path between a pair of VMs, and constraints (4)(5) further limits the path to originate from the source physical machine and destinate to the destination physical machine. Constraint (6) calculates the traffic placed on a link by a pair of VMs. Link utilization constraint (7) calculates the link utilization, and (8) restricts the utilization to be below a network-wide threshold. Paired link constraint (9) puts a bidirectional link to sleep only if both direction can be put to sleep. Link usage constraint (10) puts a link to sleep only if it does not carry any traffic. Line card usage (11) puts a line card to sleep only if all links connected to it are sleeping.

Notations

	power	
const	E	set of links
const	M	set of line cards
const	H	set of physical machines
const	P_l	power saving for putting link l to sleep
const	B_m	power saving for putting line card m to sleep
const	S_h	power saving for putting physical machine h to sleep
decision	x_l	1=link l is sleeping, 0=otherwise
decision	y_m	1=line card m is sleeping, 0=otherwise
decision	Y_h	1=physical machine h is sleeping, 0=otherwise
	job	
input	J	set of jobs
input	W_j	set of VMs in job j

input	R^j	traffic matrix between VMs in job j
input	$S_{j,v}$	physical resource demand of job j VM v
decision	$Z_{j,v}^{h}$	1=job j VM v is placed on physical machine h, 0=otherwise
	network	
const	S_m	set of links connected to line card m
const	C_l	capacity of link l
const	r(l)	reverse link of l
const	k	maximum number of candidate paths for each IE pair
const	$Q_i^{s,t}(l)$	1=i-th candidate path from s to t contains link l, 0=otherwise
config	U_T	threshold for the MLU
calculate	$f_l^{j,v,v'}$	job j from VM v to VM v' that is routed through link l
calculate	u_l	utilization of link l
decision	$A_{j,v,v'}^{s,t,i}$	1=job j traffic from VM v to VM v' is routed on i-th candidate path from s to t, 0=otherwise