CPSC 513: Integrated System Design at CS, UBC

Class project presentation

Constraint-based data center resource allocation

Nodir Kodirov

Dec. 15, 2014

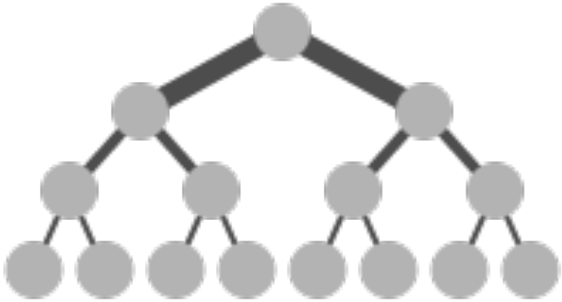
Motivation

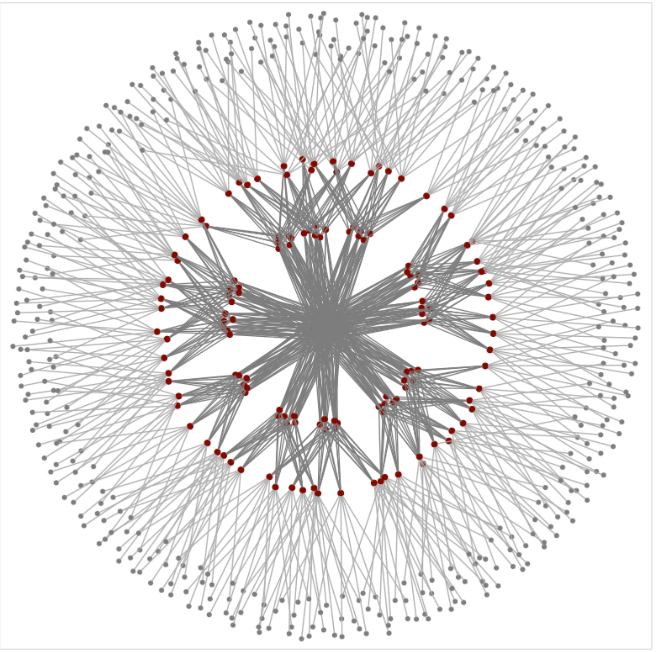
Motivation – user examples

- Who?
  - Instagram run on Amazon EC2 before purchased by Facebook
  - Zynga (up to 12K VMs, largest Facebook game app, e.g., Farmville) initially run on Amazon EC2
  - DropBox, Reddit, NetFlix, and etc.
- Why?
  - fast provisioning, easy to deploy and administer, scale up/down, reliable, cheap compute resource, and etc.

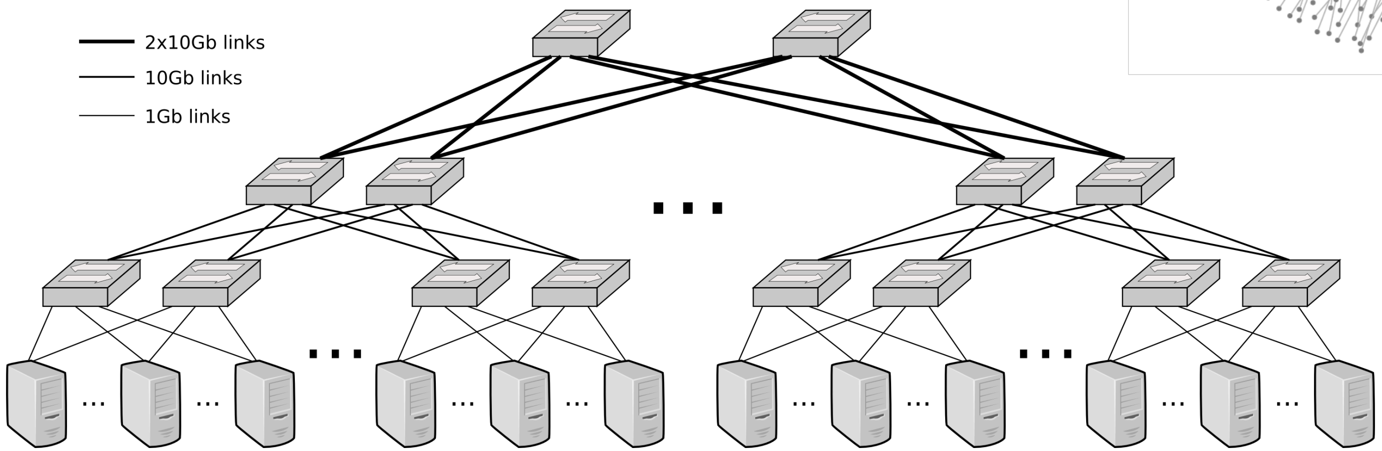

Google
google.com/datacenters

Example topology: Fat tree

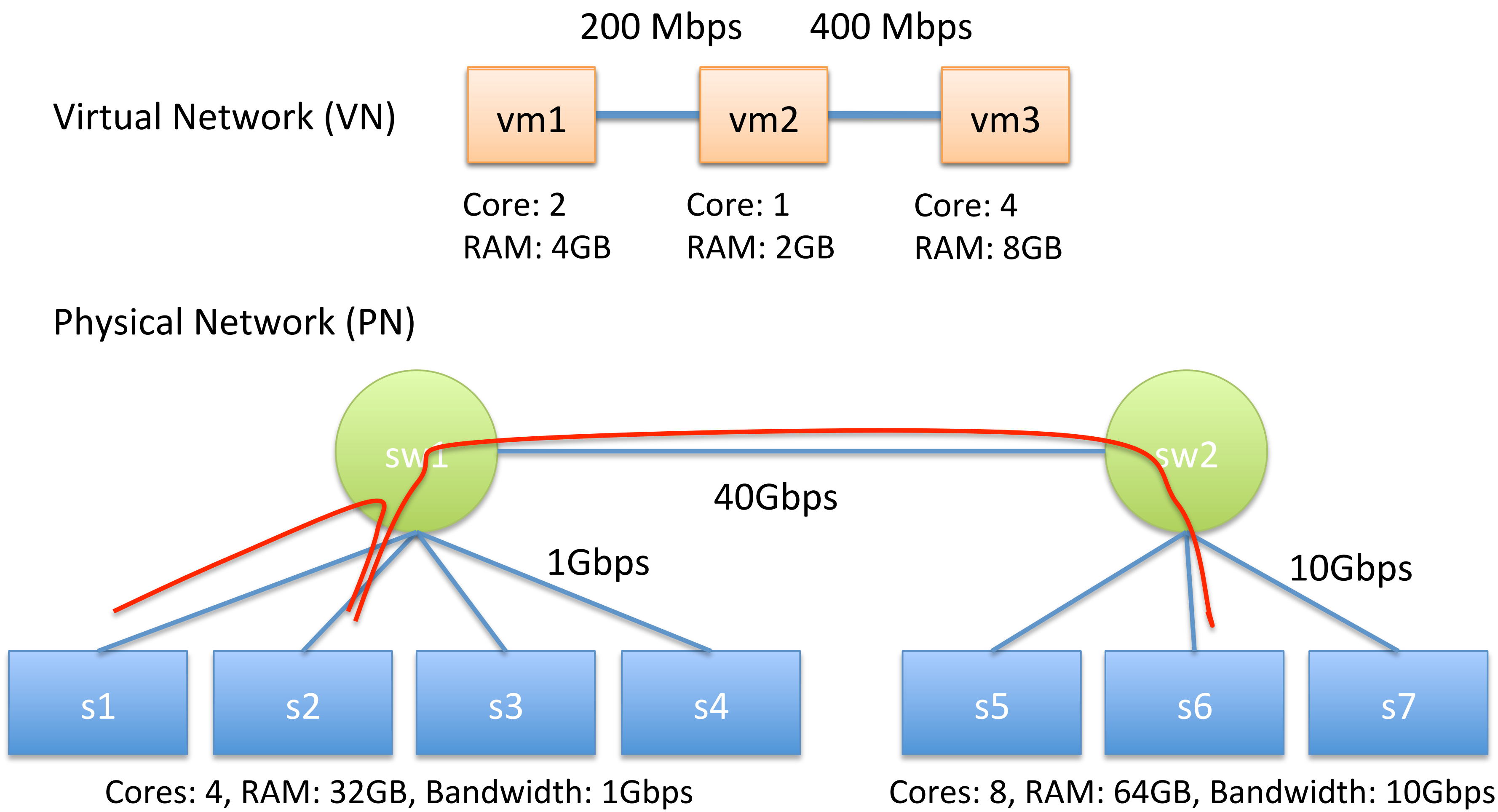




3-level fat-tree
432 servers
180 switches



Motivation: how?



Content

- Motivation
- Different approaches
- My implementation
  - SMT solver: MSR Z3
  - Demo
- Future work and conclusion

Different approaches

- Naïve: bin packing (e.g., greedy)
- Mixed-Integer Linear Problem solvers: e.g., IBM CPLEX
- Constraint solver: SOCC'11
- NEW! SMT solvers: FMCAD'13, paper draft from UBC

On the Feasibility of Automation for Bandwidth Allocation Problems in Data Centers

Yifei Yuan, Anduo Wang, Rajeev Alur, and Boon Thau Loo
University of Pennsylvania

SAT Modulo Monotonic Theories

Sam Bayless
sbayless@cs.ubc.ca
Univ. of British Columbia

Noah Bayless
nbayless@pgmini.org
Point Grey Mini Sec. School

Holger H. Hoos
hoos@cs.ubc.ca
Univ. of British Columbia

Alan J. Hu
ajh@cs.ubc.ca
Univ. of British Columbia

Implementation: constraints

$$\alpha_s : \bigwedge_{v \in V} \left(\sum_s X(v,s) = 1 \right).$$

$$\alpha_r : \bigwedge_{e,k} \left(\sum_{l \in L} R(l,e,k) \leq 1 \right).$$

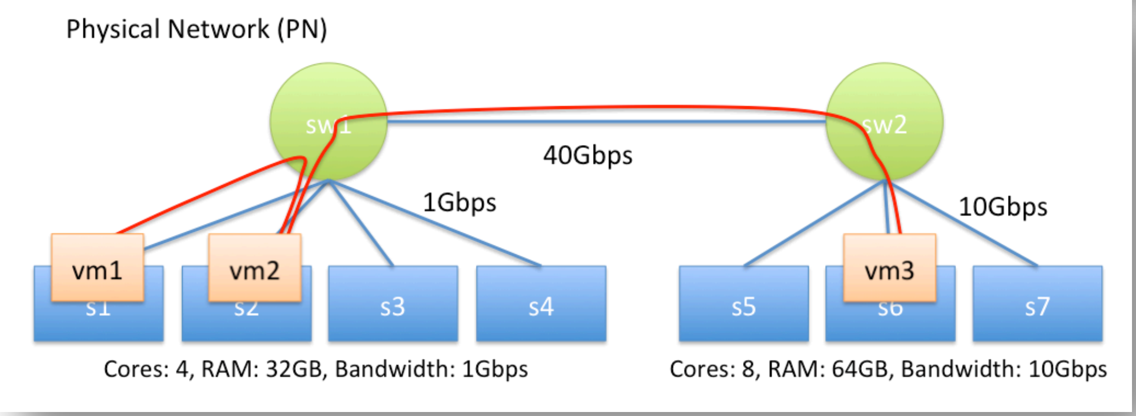
$$\alpha_v : \bigwedge_{\substack{(v_1,v_2) \in E, \\ s_1,s_2 \in A, s_1 \neq s_2}} \Bigg((X(v_1,s_1) = 1 \wedge X(v_2,s_2) = 1) \rightarrow \bigvee_{\substack{l_1 : s_1 \in l_1 \\ l_2 : s_2 \in l_2}} (Y(l_1,e) = r(e) \wedge Y(l_2,e) = r(e)) \Bigg).$$

$$\alpha_c : \bigwedge_{e,k} \left(\bigvee_{l_1,l_2 : l_1,l_2 \text{ are adjacents}} R(l_1,e,k) \wedge R(l_2,e,k+1) \right).$$

$$\beta_{server} : \bigwedge_{s \in A} \left(\sum_v X(v,s) \leq c(s) \right)$$

$$\alpha_y : Y(l,e) = r(e) \Leftrightarrow \bigvee_k R(l,e,k) = 1.$$

$$\beta_{link} : \bigwedge_{l \in E} \left(\sum_e Y(l,e) \leq b(l) \right).$$



Implementation: constraints

$$\alpha_s : \bigwedge_{v \in V} \left(\sum_{s} X(v,s) = 1 \right).$$

$$\alpha_r : \bigwedge_{e,k} \left(\sum_{l \in L} R(l,e,k) \leq 1 \right).$$

$$\alpha_v : \bigwedge_{\substack{(v_1,v_2) \in E, \\ s_1, s_2 \in A, s_1 \neq s_2}} \Bigg((X(v_1,s_1) = 1 \wedge X(v_2,s_2) = 1) \rightarrow \bigvee_{\substack{l_1 : s_1 \in l_1 \\ l_2 : s_2 \in l_2}} (Y(l_1,e) = r(e) \wedge Y(l_2,e) = r(e)) \Bigg).$$

$$\alpha_c : \bigwedge_{e,k} \left(\bigvee_{l_1,l_2 : l_1,l_2 \text{ are adjacents}} R(l_1,e,k) \wedge R(l_2,e,k+1) \right).$$

$$\beta_{server} : \bigwedge_{s \in A} \left(\sum_{v} X(v,s) \leq c(s) \right)$$

$$\alpha_y : Y(l,e) = r(e) \Leftrightarrow \bigvee_{k} R(l,e,k) = 1.$$

$$\beta_{link} : \bigwedge_{l \in E} \left(\sum_{e} Y(l,e) \leq b(l) \right).$$

$$\Phi_{PN,VN} = \alpha_s \wedge \alpha_r \wedge \alpha_c \wedge \alpha_y \wedge \alpha_v \wedge \beta_{server} \wedge \beta_{link}.$$

Demo

Limitations

- My implementation does not support routing direction
- Assumes bandwidth is unlimited when deployed to the same server
- Handles only one level of switch connection
  - one-level fat tree
- Did not check for larger PN and VNs
  - Worst case might be too slow
  - Should be able to terminate within some deadline, and get the relative best solution

Future work

- Implement abstraction
- Implement with SMMT
- See how MILP works
- Compare different approaches to recommend the best one

Take aways

- SMT solvers can be used to solve wide range of problems
  - especially when tool is mature
- Specialized solvers can bring huge performance win (SMMT)
- SMT solvers (FM in general) guarantee correctness
- Bridge systems research to FM

Acknowledgements

- Huge thanks to Sam Bayless
  - inspirational warnings: "*you are definitely going to get it completely wrong the first time …*" = True
  - describing SMMT, and agreeing to collaborate
  - saving me from getting stuck on Z3 statements
- Thanks to Penn. State folks for FMCAD 13 paper; MSR for making Z3 available
- Thanks to Alan for awesome class and "Whistler bonus"
- Thanks to everyone in class for being together!

- End note: do not write buggy code, use FM!

Backup slides

Discussion: challenges

- Do not confuse Z3 and Python variables (Int, Bool, IntVal, BoolVal)
- If possible, assign value to Z3 matrices only once
- Z3 primitives used
  - return_value = If(condition, true, false)
  - Implies(condition1, condition2)
  - And(condition1, condition2, …)
    - Or(…), Not(…)
  - *Possible to nest these statements*
  - solver.add(*constraints*)
- Only use primitives you are sure about
  - common sense programming approach might not be the same
  - slightly misused primitive gives lots of trouble (silently!)

Different approaches

- Naïve: bin packing (e.g., greedy)
- Constraint solver: SOCC'11

Modeling and Synthesizing Task Placement Constraints in Google Compute Clusters

Bikash Sharma *
Pennsylvania State University
University Park 16802
bikash@cse.psu.edu

Victor Chudnovsky
Google Inc.
Seattle 98103
vchudnov@google.com

Joseph L. Hellerstein
Google Inc.
Seattle 98103
jlh@google.com

Rasekh Rifaat
Google Inc.
Seattle 98103
rasekh@google.com

Chita R. Das
Pennsylvania State University
University Park 16802
das@cse.psu.edu

Different approaches

- Naïve: bin packing (e.g., greedy)
- Constraint solver: SOCC'11
- Mixed-Integer Linear Problem solvers: e.g., IBM CPLEX
  - NSDI 2012, CMU tech report 2013

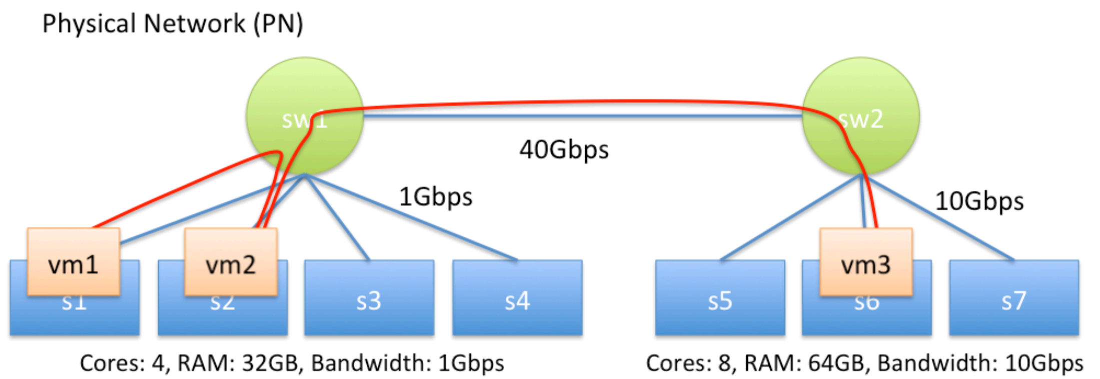
Design and Implementation of a Consolidated Middlebox Architecture

Vyas Sekar[*], Norbert Egi[††], Sylvia Ratnasamy[†], Michael K. Reiter[*], Guangyu Shi [††]
[*] Intel Labs, [†] UC Berkeley, [*] UNC Chapel Hill, [††] Huawei

Tetrisched: Space-Time Scheduling for Heterogeneous Datacenters

Alexey Tumanov[*], Timothy Zhu[*]
Michael A. Kozuch[†], Mor Harchol-Balter[*], Gregory R. Ganger[*]
Carnegie Mellon University[*], Intel Labs[†]

Implementation: notations



- X(v,s): VM v is mapped to server s
- Y(l,e): physical link l is reserved bandwidth virtual link e
- R(l,e,k): physical link l is the k-th edge on the routing path for virtual link e
- Server capacity:
  - $\sum_v X(v,s) < c(s)$, for every server s
- Link capacity:
  - $\sum_e Y(l,e) < b(l)$, for every physical link l