

The Case For System Testing with Swift Hierarchical VM Fork

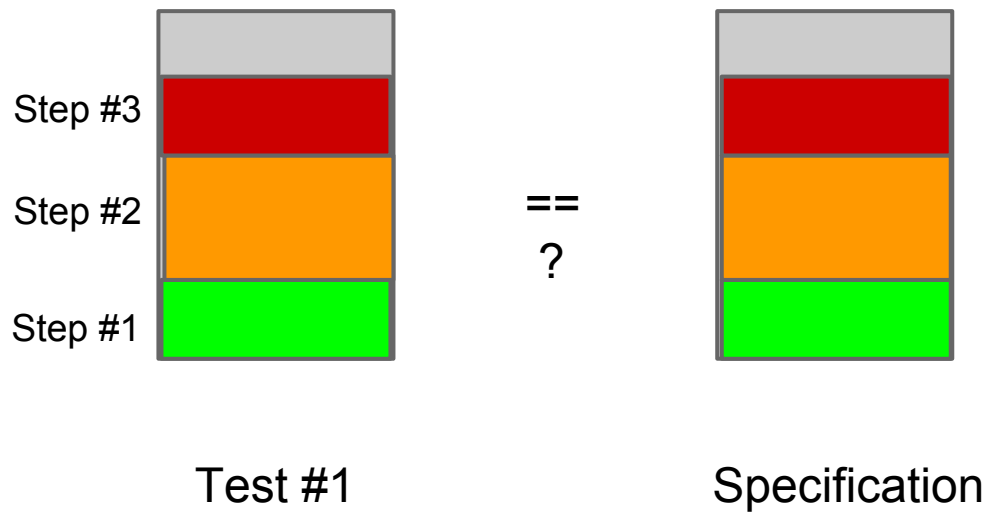
Junji Zhi, Sahil Suneja and Eyal de Lara
Department of Computer Science
University of Toronto



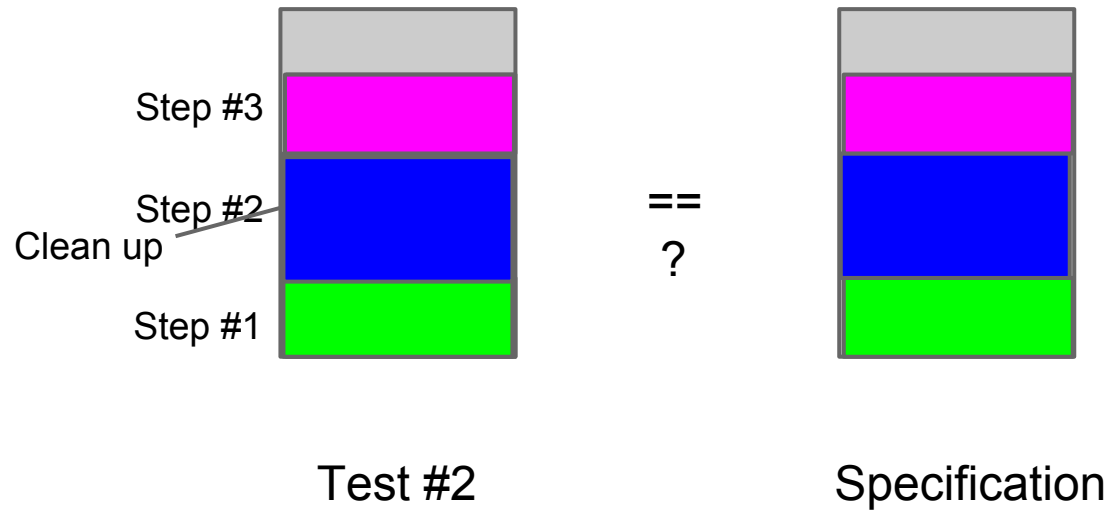
Introduction

- Testing large systems is often difficult
- System configuration takes effort
- Executing tests consumes resources and time

System Testing

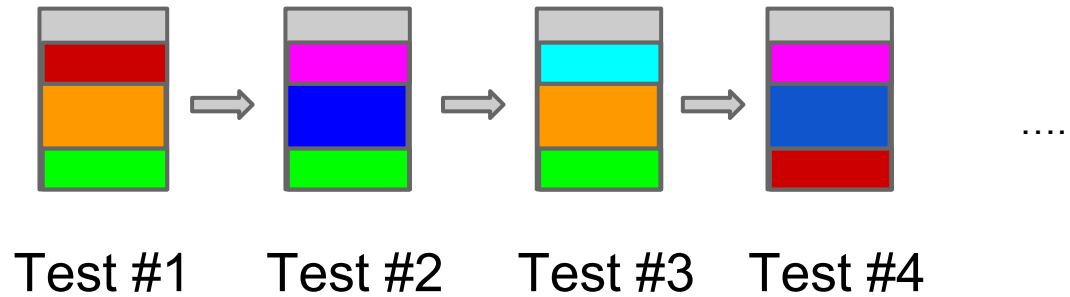


System Testing

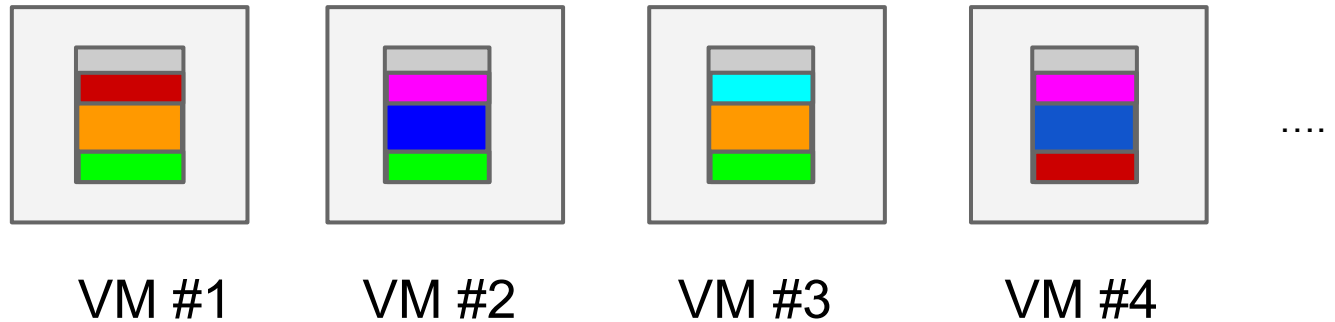


Multiple test cases

Sequential,
One machine



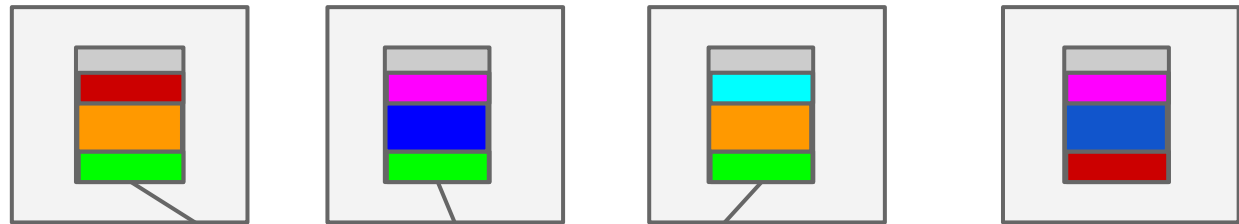
Simple parallel,
multiple VMs



Observations

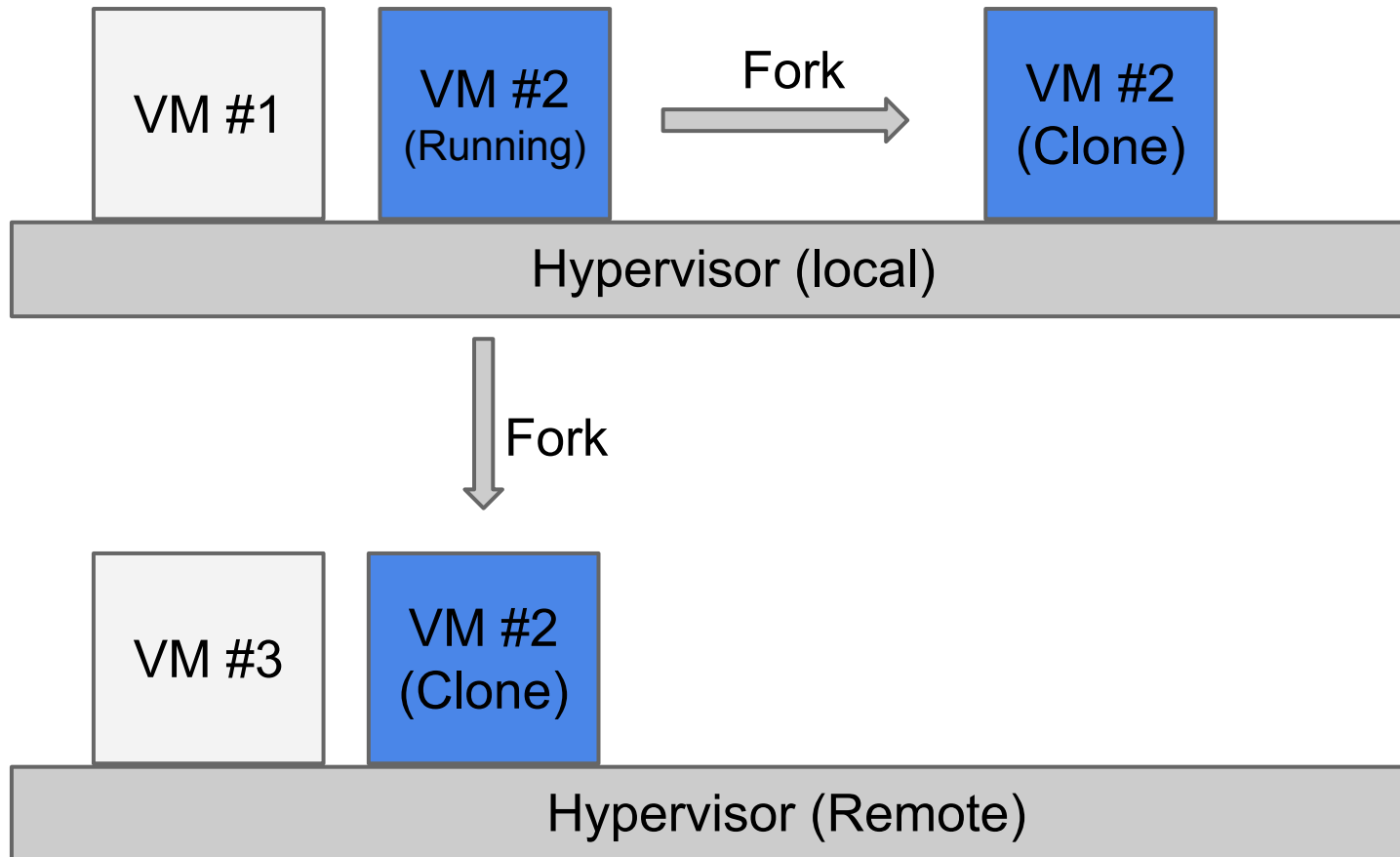
- Many commonalities or overlapping steps exist among tests
- Test cases share the same code base.

Simple parallel,
multiple VMs



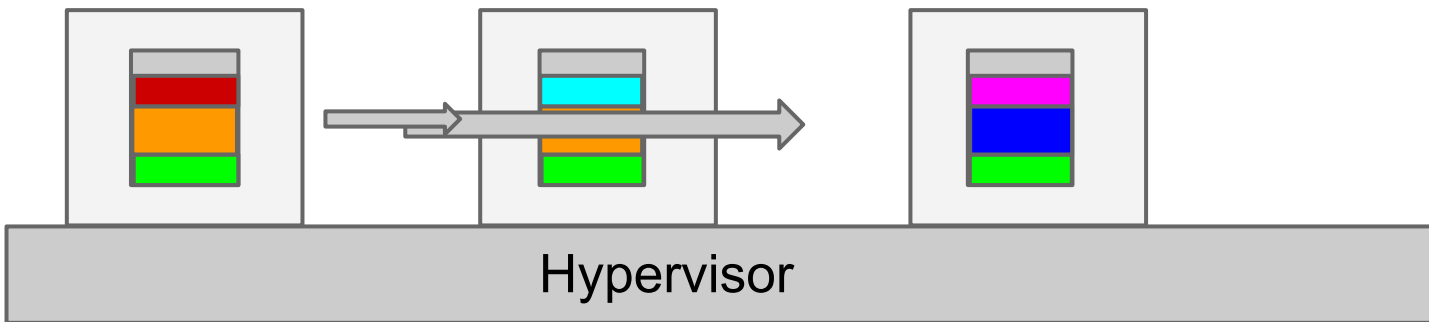
Common Steps are executed in each VM!

VM Fork



Test Execution with VM Fork

- Reuse common steps
- Share memory and disk state



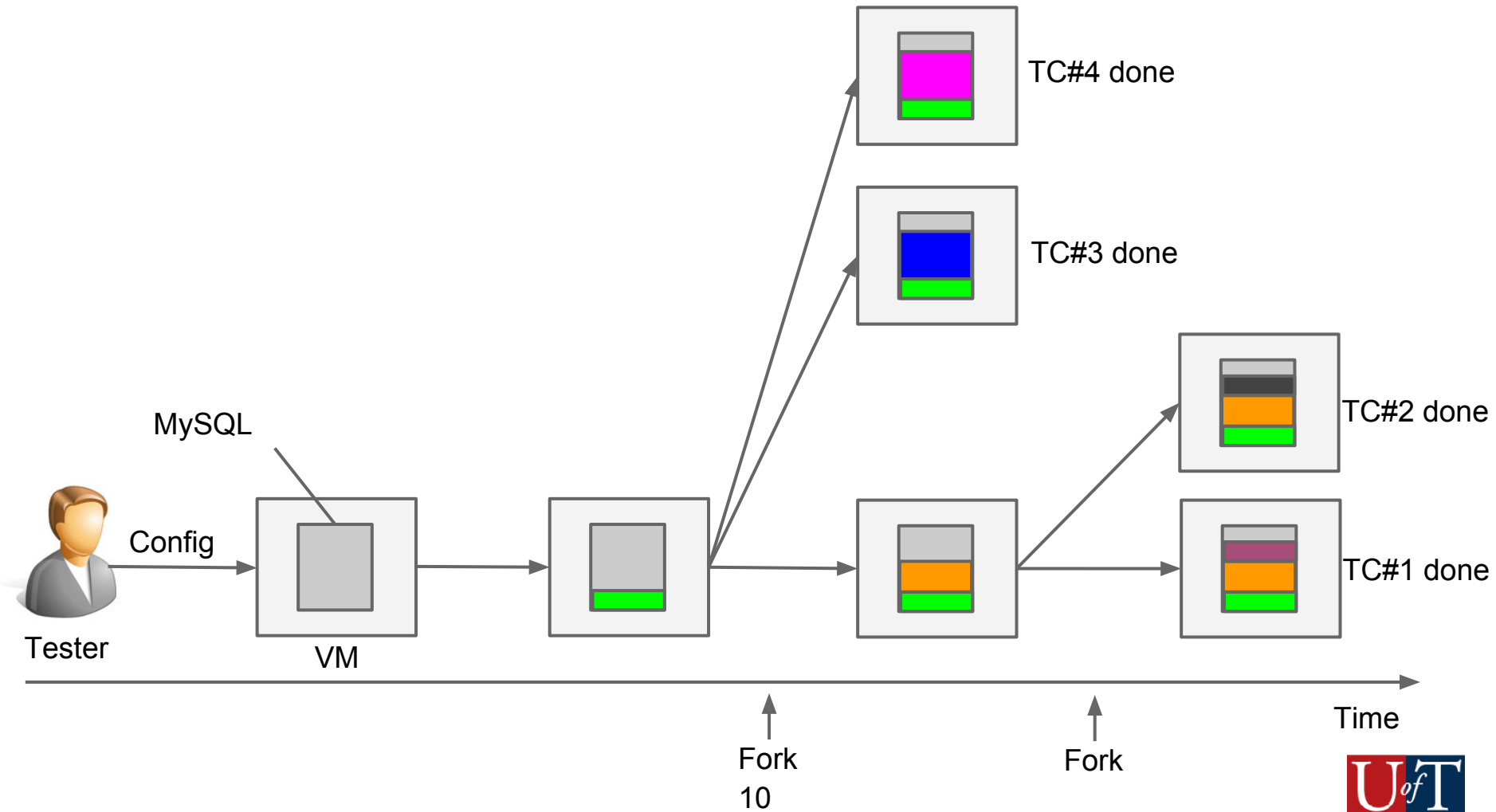
Case Study: Testing MySQL

- MySQL v5.5 “large_tests” suite

TC#	Step 1	Step 2	Step 3
1	Construct*	Delete	Insert
2	Construct	Delete	Add Column
3	Construct	Select	/
4	Construct	Update	/

*Construct: construct a new table and populate rows

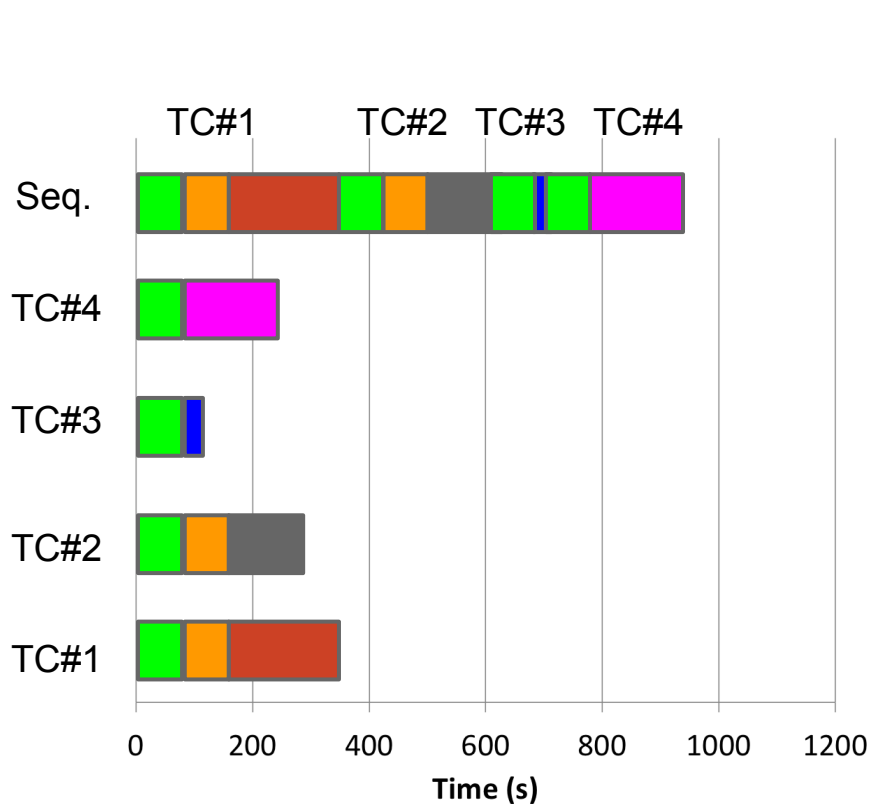
MySQL Testing with VM Fork



Experiment Setup

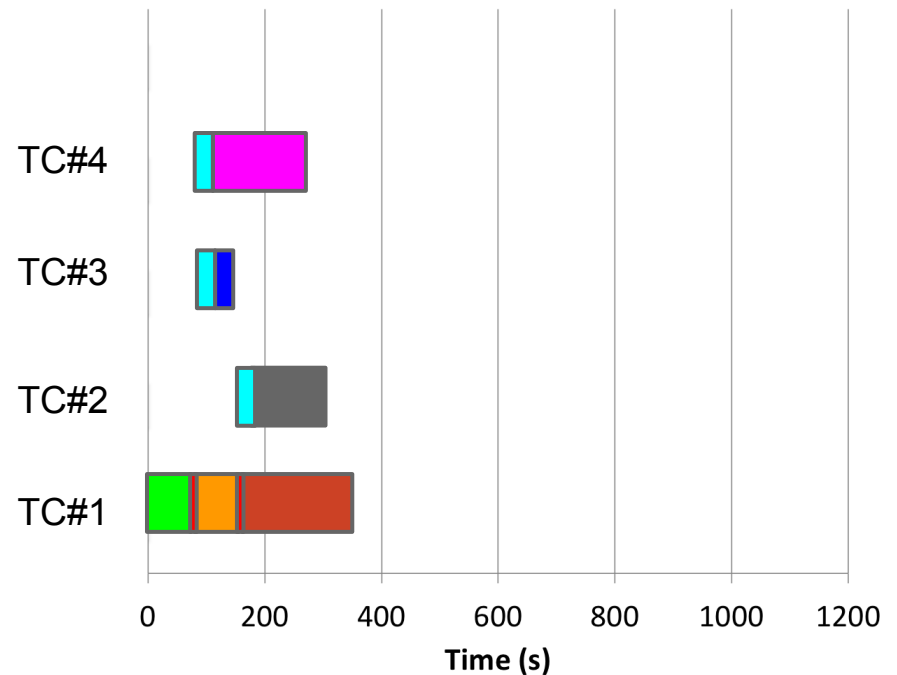
- All experiments on server x86 12GB RAM
 - 64-bit Ubuntu 12.04 (Kernel v3.2.0)
 - Mysql v5.5
 - KVM v3.8
 - QEMU v1.0
- No current hierarchical fork implementation
 - Emulated VM fork with QEMU snapshot
 - Issues: slow

Results: Execution Time



Sequential & Simple Parallel
Total CPU cycles: 971s

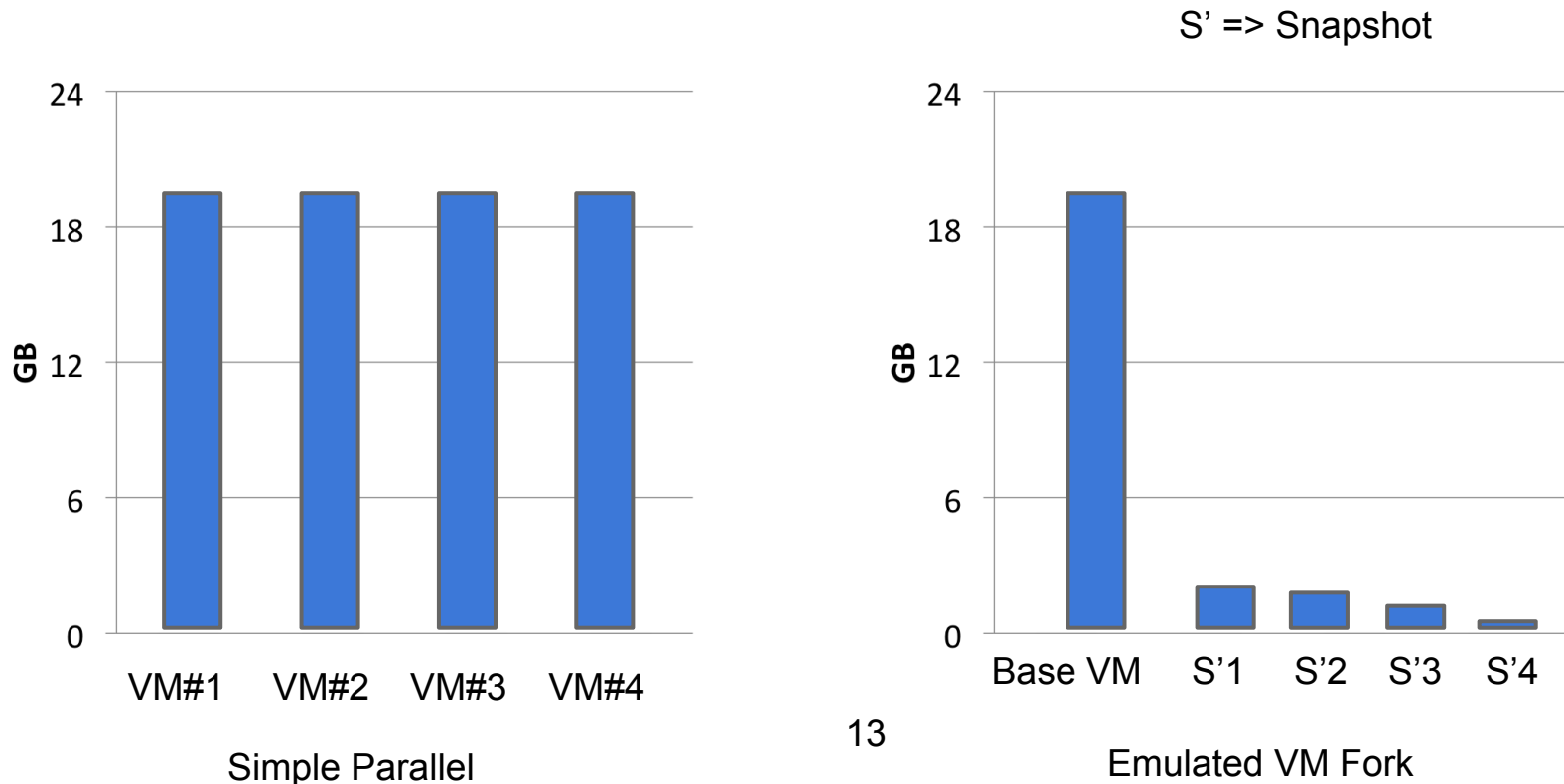
boot from snapshot



With Emulated VM Fork
Total CPU cycles: 690s

Results: Space Usage

- 1 Snapshot < 5% Base VM
- 70% space saving



Challenges

- Fast hierarchical VM fork implementation
- When to fork
- Where to fork
- Transparency

Summary

- Leverage hierarchical VM fork to optimize system testing
- Advantages
 - Reuse common steps among TCs
 - Saving memory and disk space
- 30% reduction in VM runtime
- 70+% Space Savings

Thank you!
&
Questions?

Software Testing on the Cloud

- Cloud creates new opportunities
- Each VM encapsulates an entire test environment
- VM can run in parallel, isolated from each other

Case Study: Testing MySQL

- Structured automated test scripts

```
1 #Test 1
2 drop table if exists t1;
3 create table t1;          #config step
4 insert into t1 SOME_ROWS;
5 delete from t1 where CRITERIA1;
6 insert into t1 DELETED_ROWS;
7 drop table t1;          #clean-up step
8
9 #Test 2
10 drop table if exists t1;
11 create table t1;          #config step
12 insert into t1 SOME_ROWS;
13 delete from t1 where CRITERIA1;
14 alter talbe t1 add column C1;
15 drop table t1;          #clean-up step
```

Testing on the cloud

TABLE I
CATEGORIZATION OF LITERATURE BASED ON TEST LEVEL & TYPE

Problem Domain	Test Level				Test Type			
	Acceptance Testing	System Testing	Unit Testing	Integration Testing	Functional Testing	Performance Testing	Security Testing	Interoperability Testing
Mobile App.s	X	[8], [3]	[9]	X	[3]	[8]	[8]	X
Cloud App.s	X	[10], [11], [12], [13], [14], [15], [16]	[11], [12], [9], [13]	[15], [17]	[10], [11], [12], [13], [14], [15]	[10], [16]	X	X
Desktop App.s	X	[18], [11], [16]	[9]	X	[18], [11]	[16]	X	X
Web Services & App.s	X	[19], [20]	[9]	[5]	[19], [20]	[5], [20]	X	X
Distributed & Parallel App.s	X	[21], [22], [23], [24]	[9]	X	[21], [22], [23], [24]	X	X	X
Cloud Service Dev. & Deployment	X	[25], [26], [27]	[25], [28], [9], [13]	[26], [25]	[27], [28], [13], [29], [30], [31], [32]	[26], [29], [33], [34]	[29]	[29], [30], [35], [36], [37]
Migration to Cloud	X	[38], [39], [40], [41], [6]	[6]	X	[38], [41], [6], [39], [40]	[41]	[41]	X
Cloud Infrastructure & Storage	X	[42], [43], [44], [45]	X	X	[44], [45]	[42], [43]	X	X
Real-Time Systems	X	[46], [47], [48]	[47], [49], [9]	X	[46], [47], [48]	[50]	X	X
Network Config.	X	[51]	X	X	[51]	X	X	X
Test Task Mang.	[52]	[53], [7], [52]	[7], [52]	X	[53], [7], [52]	X	X	X