Forecasting MySQL Scalability

Baron Schwartz

O'Reilly MySQL Conference & Expo 2011
Percona Server

- Replaces MySQL
- Faster Queries
- More Consistent
- More Measurable
- More Features
Percona XtraBackup

- Backs Up InnoDB
- Non-Blocking
Forecasting Performance Scalability

- Performance == Response Time
- Scalability is a mathematical equation (function)
- This is about scalability, sorry about the bad title in the conference program.
The Scalability Function

![Graph showing the relationship between throughput and threads or nodes. The graph is linear, with an upward trend as the number of threads or nodes increases.]
This is Linear Scalability

Throughput vs. Threads or Nodes
This is Not Linear Scalability

Throughput vs. Threads or Nodes
What Causes Non-Linearity?

What's this about?
Factor #1: Serialization

- Some portion of the work cannot be done in parallel
- “Sigma” is the serial fraction
- It grows linearly
- This is Amdahl's Law

\[ C(N) = \frac{N}{1 + \sigma(N - 1)} \]
Factor #2: Coherency

- Some portion of the work relies on IPC, cross-node communication, etc
- “Kappa” is the synchronized fraction
- It grows quadratically
- This is Neil Gunther's University Scalability Law

\[ C(N) = \frac{N}{1 + \sigma(N - 1) + \kappa N(N - 1)} \]
Real Systems Usually Have Both

- Most systems have serialization & coherency. Coherency causes retrograde scaling.
How To Forecast Scalability

- Measure throughput -vs- nodes or concurrency
- Plot the points
- Perform curve-fitting to find sigma, kappa
- Examine results carefully, throw out bad points, tweak, etc etc.
Is it Cheating to Cull Bad Data?

- The model correctly describes the factors involved in scalability.
- It is a reference without which there is nothing to discuss.
Case Study #1

- Percona Server on Cisco UCS Server

![Graph showing the relationship between concurrency (N) and throughput (C)](image)
Applying the Model

Peak capacity is $C=10991$ at $N=26$

$\sigma = 0.013616$
$kappa = 0.001414$
$R^2 = 0.999971$
How Good Was the Model?
What Does Capacity Mean?

- We can't run systems at peak throughput
- Performance (response time) would suck
- Capacity is maximum throughput that maintains acceptable response time
  - Latency is important
  - Consistency is also important
- The Universal Scalability Law doesn't predict response time as used here, only throughput
Case Study #2

- This is a real MySQL server under load tests.
- How close is the server to its limits?
**Measurements**

```bash
code
mysqladmin ext -ri10 \       | grep -e Uptime -e Threads_running -e Questions
Questions 118357171
Threads_running 8
Uptime 614909
Questions 118364376
Threads_running 6
Uptime 614920
Questions 118370320
Threads_running 4
Uptime 614930
Questions 118377196
```
Transforming the Data

- We need Throughput Versus Concurrency
- Throughput is simple: Queries Per Second
- Concurrency? That's tougher
  - I averaged Threads_running over each sample
Plotting The Result

Peak capacity is $C=1338$ at $N=20$.

$$\sigma = 0.072252$$

$$\kappa = 0.002268$$

$$R^2 = 0.972464$$
That Doesn't Look Usable

- Peak throughput prediction is too low
- Peak concurrency prediction is too high
- This data is too messy to work with
What's The Problem?

- Threads_running is instantaneous samples.
- We need to know the average.
Averaged over 150-sec Intervals

Peak capacity is $C=1329$ at $N=15$

- $\sigma = -0.034749$
- $\kappa = 0.004590$
- $R^2 = 0.993142$

Modelled
Measured
Better, But Not Good Enough

- There are clearly outliers
- The plotted points don't "point at the axis"
What's Wrong?

- SHOW STATUS increments Threads_running
- There are 3 replication slaves connected
- We need to subtract these to get concurrency closer to reality
- Let's try again with “Threads_running - 4”
Adjusted Concurrency

Peak capacity is $C = 1301$ at $N = 12$

\[ \sigma = 0.021675 \]
\[ \kappa = 0.006568 \]
\[ R^2 = 0.993525 \]
Take-Away

- This server is approaching its peak capacity
- Don't count on sustained QPS over 1000 or so
- If Threads_running > 10, you're in trouble
Important Background Info

• This is a complex workload...
• On a virtualized server...
• With 8 cores...
• Running MySQL 5.0.51dogslow
• MySQL can do a lot better. This MySQL can't.
Existing System

- This technique models the existing workload on the existing system.
- It doesn't model what happens if you change things in the system.
- We might be able to optimize queries and get a different outcome, for example.
Once You've Learned This, It's Lots Of Fun.
Does it scale linearly?
Benchmark at the Clustrix Booth

<table>
<thead>
<tr>
<th>#nodes</th>
<th>TPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>58344</td>
</tr>
<tr>
<td>6</td>
<td>115193</td>
</tr>
<tr>
<td>9</td>
<td>167831</td>
</tr>
<tr>
<td>12</td>
<td>218004</td>
</tr>
<tr>
<td>15</td>
<td>266178</td>
</tr>
<tr>
<td>18</td>
<td>315842</td>
</tr>
<tr>
<td>20</td>
<td>343838</td>
</tr>
</tbody>
</table>
Looks Pretty Linear To Me!

• But it's not. Do the math.
  • 3 nodes = 58344 TPS
  • 18 nodes = 6 * 58344 = 350064?  
  • No, 18 nodes = 315842
• Not linear scaling.
• But it's still impressive. Let's plot it.
Using “usl” Tool from Aspersa

```
ginger $ usl -e -o model-vs-actual clustrix-scalability.txt
# Command-line: /home/baron/bin/usl -e -o model-vs-actual clustrix-scalability.txt
# Using gnuplot 4.2 patchlevel 6
# Parameters to the model:
min(N)  3
max(N)  20
max(C)  343838
C(1)    19448 (pre-adjustment by 1)
N=1 ??? no
# Fitting the transformed data against a 2nd-degree polynomial.
 a       0.000154677 +/- 6.938e-05 (44.85%)
 b       0.00406757 +/- 0.001111 (27.3%)
 R^2     0.991981
# Re-fitting against the USL with (a, b-a) as a starting point.
# Treating (1, 19448) as a point in original measurements.
 sigma  0.00508683 +/- 0.0008785 (17.27%)
 kappa  8.79207e-05 +/- 4.883e-05 (55.54%)
 C(1)    19448     (not a regression parameter)
 R^2     0.999978
```
Clustrix is Very Scalable.

Peak capacity is $C=820435$ at $N=106$

$\sigma = 0.005087$

$kappa = 0.000088$

$R^2 = 0.999978$
Important Notes

• Clustrix didn't pay me for this.
• I just did a drive-by shooting at their booth.
• These benchmarks are over a year old.
• They have done a lot of work since then and the system “should be much higher performance.”
• Scaling to 106 nodes is extremely good.
Further Study

- Learn the underlying theory
- Learn how to apply the model
- Read the white paper on percona.com
- You can use Aspersa's “usl” tool to help

http://www.perfdynamics.com/  http://perfdynamics.blogspot.com/  @DrQz
We're Hiring!  www.percona.com/about-us/careers/