ArcGIS Server
Performance and Scalability – Optimization and Testing
Andrew Sakowicz
Objective

• Overview:
  - Key performance factors
  - Optimization techniques
  - Performance Testing
Introduction

- Andrew Sakowicz
  - Esri Professional Services, Redlands
  - asakowicz@esri.com
Audience

- Audience
  - Developers
  - Architects
  - GIS Administrators
  - DBA’s

- Level:
  - Intermediate
Performance factors
Performance Factors: ArcGIS Server Services

*Map Service – Source document optimization*

- Keep map symbols simple
- **Scale dependency**
  - Optimize spatial index
  - Simplify data
  - Avoid re-projections on the fly
  - Optimize map text and labels for performance
  - Use annotations
  - Avoid wavelet compression-based raster types (MrSid, JPEG2000)
  - Use fast joins (no cross db joins)
Performance Factors: ArcGIS Server Services

Map Service – Source map document optimizations, scale dependency

- Performance linearly related to number of features

![Response Time vs. Number of Features](chart.png)

- SQL Query - SDE.ST_ENVINTERSECTS
- Synthetic Polygon Dataset

\[ y = 0.3666x \]

\[ R^2 = 0.9941 \]
Performance Factors: ArcGIS Server Services

Map Service – Output image format choices

- **PNG8/24/32**
  - Transparency support
  - 24/32 good for anti-aliasing, rasters with many colors
  - Lossless: Larger files (> disk space/bandwidth, longer downloads)

- **JPEG**
  - Basemap layers (no transparency support)
  - Much smaller files
Performance Factors: ArcGIS Server Services

- Optimized services create caches significantly faster than “Classic”
Performance Factors: ArcGIS Server Services

Geoprocessing Service

- Pre-compute intermediate steps when possible
- Use local paths to data and resources
- Avoid unneeded coordinate transformations
- Add attribute indexes
- Simplify data

Detailed instructions on the Resource Center
Performance Factors: ArcGIS Server Services

**Image Service**

- Tiled, JPEG compressed TIFF is the best (10-400% faster)

- Build pyramids for raster datasets and overviews for mosaic datasets

- Tune mosaic dataset spatial index.

- Use JPGPNG request format in Web and Desktop clients
  - Returns JPEG unless there are transparent pixels (best of both worlds).

Help Topic: “Optimization Considerations for ArcGIS Image Server”
Performance Factors: ArcGIS Server Services

Geocode Service

- Use local instead of UNC locator files.

- Services with large locators take a few minutes to “warm-up”

- New 10.0 Single Line Locators offer simplicity in address queries but might be slower than traditional point locators.
Performance Factors: ArcGIS Server Services

Mobile Service

- **Document Preparation**
  - Minimize operational layers
  - Cache basemap layers

- **Service Configuration**
  - Try to keep total service cache size under 250 MB

- **Usage considerations**
  - Avoid batch postings in favor of frequent updates
Performance Factors: ArcGIS Server Services

*Feature/Geodata Service – Database maintenance is key*

- **Database Maintenance/Design**
  - Keep versioning tree small, compress, schedule synchronizations, rebuild indexes and have a well-defined data model
- **Geodata Service Configuration**
  - Server Object usage timeout (set larger than 10 min default)
  - Upload/Download default IIS size limits (200K upload/4MB download)
- **Feature Service**
  - Trade-off between client-side rendering and sending large amounts of data over the wire.
Performance Factors: CPU Type

Select adequate hardware to support desired performance/load

• CPU
  - Select for intended use
    - Mapping: highest Baseline CINT Rate/Core
    - GP: highest Baseline CFP Rate/Core
  - Sizing
    - Published CPU benchmarks: http://www.spec.org/cpu2006/results/cint2006.html
    - Published CPU-limited ESRI benchmarks: http://resources.esri.com/enterprisegis/index.cfm?fa=codeGallery
Performance Factors: Hardware Resources

Ensure sufficient CPU, Memory and Network resources

- User load: concurrent users or throughput
- Operation CPU service time (model) - performance
- CPU type

\[
\# CPU_t = \frac{S T_b \times TH_t \times 100}{3600 \times %CPU_t} \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}
\]

subscript  \( t \) = target
subscript  \( b \) = benchmark
\( ST \) = CPU service time
\( TH \) = throughput
\( %CPU \) = percent CPU
Performance Factors: CPU Type

*Select adequate hardware to support desired performance/load*

- System Designer
Performance Factors: CPU Type

CPU type impacts response time (performance) and capacity (scalability)

- Amazon instance: XXL vs. XL
Performance Factors: Data Sources

Select data storage format that provides optimal performance

Data storage format
- RDBMS, FileGDB, Shapefile, SDC
Performance Factors: Data Location

*Select data location that provides optimal performance*

- Local to SOC machine
- UNC (protocol + network latency/bandwidth penalties)

- All disks being equal, locally sourced data results in better throughput.
Performance Factors: DB Management

Optimize DB configuration and conduct maintenance

- DBMS configuration
- Create and maintain (rebuild) attribute indexes
- Updating DBMS statistics
- Versioning management
  - Reconcile and post
  - Compress

*Non-optimal DBMS may be a source of significant performance degradation*
Performance Factors: ArcGIS Server Framework

**SOC**

Optimal number of instances/core departs from CPU-limited value of 1 by choice of source data type/location.
Tuning primer
Tuning Steps

- Optimize ArcGIS Services
- Profile individual user operations and tune if needed
- Drill down through software stack
  - Application
  - Service
  - Mxd
  - Layer
  - DBMS query
- Correlate your findings between tiers
- **Performance and load test**
Performance tuning

• Benefits
  - Improved performance - user experience
  - Optimal resource utilization – scalability

• Tools
  - Fiddler
  - Map Service Publishing Toolbar
  - DBMS trace
A test is executed at the web browser. It measures web browser call’s elapsed time (roundtrip between browser and data source).
Tuning Primer

Web diagnostic tools: Fiddler, Tamperdata, Yslow
Tuning Primer

Web diagnostic tools: Fiddler

- Can validate image returned
Tuning Primer

Web diagnostic tools: Fiddler

- Understand each request URL
- Verify cache requests are from virtual directory, not dynamic map service
- Validate host origin (reverse proxy)
- Profile each transaction response time
Tuning Primer

Web diagnostic tools: Fiddler

HTTP Request

Inspector Tab

HTTP Response
Tuning Primer

Web diagnostic tools: Fiddler (NeXpert Report)
Tuning Primer

Analyze SOM/SOC statistics

- Total Response Time (\(t_1-t_2\))
- Wait Time
- Usage Time
- Search & Retrieval Time

Analyze AGS context server statistics using ArcCatalog, Manager or logs. They provide aggregate and detailed information to help reveal the cause of the performance problem.
Tuning Primer

Analyze SOM/SOC statistics

- ArcCatalog

- Detailed log - set to verbose

<Msg time="2009-03-16T12:23:22" type="INFO3" code="103021" target="Portland.MapServer" methodName="FeatureLayer.Draw" machine="myWebServer" process="2836" thread="3916" elapsed="0.05221">Executing query.</Msg>


### Tuning Primer

**mxdperfstat**


<table>
<thead>
<tr>
<th>Item</th>
<th>At Scale</th>
<th>Layer Name</th>
<th>Refresh Time (sec)</th>
<th>Recommendations</th>
<th>Features</th>
<th>Vertices</th>
<th>Labeling</th>
<th>Geometry Phase (sec)</th>
<th>Graphics Phase (sec)</th>
<th>Cursor Phase (sec)</th>
<th>DBMS CPU</th>
<th>DBMS LIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>1,000</td>
<td>TaxlotDenseLabel</td>
<td>1.93</td>
<td>Simplify labeling, symbology: GraphicsPhase=1.42; simplify geometry and/or set label scale; convert polygon to polyline: vertices fetched=200001; simplify geometry and/or set label scale: vertices fetched=200001;</td>
<td>1</td>
<td>200,001</td>
<td>TRUE</td>
<td>0.45</td>
<td>1.42</td>
<td>1.04</td>
<td>0.02</td>
<td>266</td>
</tr>
<tr>
<td>42</td>
<td>1,000</td>
<td>TaxlotDenseNoLabel</td>
<td>0.53</td>
<td>simplify geometry: vertices fetched=200001;</td>
<td>1</td>
<td>200,001</td>
<td>FALSE</td>
<td>0.45</td>
<td>0.02</td>
<td>0.9</td>
<td>0.02</td>
<td>140</td>
</tr>
</tbody>
</table>
Tuning Primer

ArcGIS Services

Heat Map based on response times from ArcGIS Server
Observe correlation between feature density and performance
Tuning Primer

Data sources

Browser

Web Server

SOM

SDE/DBMS

Total Response Time (t1-t2)

Wait Time

Usage Time

Search & Retrieval Time
**Tuning Primer**

**Data Sources – Oracle Trace**

```sql
select username, sid, serial#, program, logon_time from v$session where
username='STUDENT';
```

<table>
<thead>
<tr>
<th>USERNAME</th>
<th>SID</th>
<th>SERIAL#</th>
<th>PROGRAM</th>
<th>LOGON_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>STUDENT</td>
<td>132</td>
<td>31835</td>
<td>gsrvr.exe</td>
<td>23-OCT-06</td>
</tr>
</tbody>
</table>

SQL> connect sys@gis1_andrews as sysdba

Enter password:

Connected.

SQL> execute

sys.dbms_system.set_ev(132,31835,10046,12,'');

**DBMS trace is a very powerful diagnostic tool**
Private Sub OracleTrace_Click()
    . . .
    Set pFeatCls = pFeatLyr.FeautreClass
    Set pDS = pFeatCls
    Set pWS = pDS.Workspace
    sTraceName = InputBox("Enter <test_name><email>")
    pWS.ExecuteSQL ("alter session set tracefile_identifier = " & sTraceName & ")
    pWS.ExecuteSQL ("ALTER SESSION SET events '10046 trace name context forever, level 12"")
    . . .
End Sub
Tuning Primer

Data Sources – Oracle Trace (continued)

SQL ID : 71py6481sj3xu
SELECT  1 SHAPE, TAXLOTS.OBJECTID,  TAXLOTS.SHAPE.points,TAXLOTS.SHAPE.numpts, 
        TAXLOTS.SHAPE.entity,TAXLOTS.SHAPE.minx, TAXLOTS.SHAPE.miny,  
        TAXLOTS.SHAPE.maxx,TAXLOTS.SHAPE.maxy,TAXLOTS.rowid  
FROM  SDE.TAXLOTS TAXLOTS WHERE SDE.ST_EnvIntersects(TAXLOTS.SHAPE,:1,:2,:3,:4) =  1

<table>
<thead>
<tr>
<th>call</th>
<th>count</th>
<th>cpu</th>
<th>elapsed</th>
<th>disk</th>
<th>query</th>
<th>current</th>
<th>rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse</td>
<td>0</td>
<td>0.00</td>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Execute</td>
<td>1</td>
<td>0.07</td>
<td>0.59</td>
<td>115</td>
<td>1734</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Fetch</td>
<td>242</td>
<td>0.78</td>
<td>12.42</td>
<td>2291</td>
<td>26820</td>
<td>0</td>
<td>24175</td>
</tr>
</tbody>
</table>

| total  | 243   | 0.85 | 13.02   | 2406 | 28554 | 0       | 24175|

Elapsed times include waiting on following events:

<table>
<thead>
<tr>
<th>Event waited on</th>
<th>Times</th>
<th>Max. Wait</th>
<th>Total Waited</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL*Net message to client</td>
<td>242</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>db file sequential read</td>
<td>2291</td>
<td>0.39</td>
<td>11.69</td>
</tr>
<tr>
<td>SQL*Net more data to client</td>
<td>355</td>
<td>0.00</td>
<td>0.02</td>
</tr>
<tr>
<td>SQL*Net message from client</td>
<td>242</td>
<td>0.03</td>
<td>0.54</td>
</tr>
</tbody>
</table>

********************************************************************************
Tuning Primer

Data Sources – Oracle Trace (continued)

- Definitions
  - Elapsed time [sec] = (CPU + wait event)
  - CPU [sec]
  - Query (Oracle blocks e.g. 8K read from memory)
  - Disk (Oracle blocks read from disk)
  - Wait event [sec], e.g. db file sequential read
  - Rows fetched
**Example (cost of physical reads):**
- Elapsed time = 13.02 sec
- CPU = 0.85 sec
- Disk = 2291 blocks
- Wait event (db file sequential read ) = 11.69 sec
- Rows fetched = 24175
**Tuning Primer**

**Data Sources – SQL Profiler**

### SQL Profiler Output

**EventClass**: ArcSDE_trace (ANDR11WS2)

<table>
<thead>
<tr>
<th>EventClass</th>
<th>Login</th>
<th>Application</th>
<th>TextData</th>
<th>CPU</th>
<th>Duration</th>
<th>RowCounts</th>
<th>Reads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace Start</td>
<td>sde</td>
<td>sde</td>
<td>&lt;ShowPlanXML xmlns=&quot;<a href="http://schemas">http://schemas</a>....</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Showplan XML Statistics P...</td>
<td>sde</td>
<td>sde</td>
<td>SELECT state_id,owner,creation_time...</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>SP:StmtCompleted</td>
<td>sde</td>
<td>sde</td>
<td>SELECT lineage_name, time_last_modi...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Showplan XML Statistics P...</td>
<td>sde</td>
<td>sde</td>
<td>SELECT s_.eminx,s_.emin,y,s_.emaxx,y...</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Showplan XML Statistics P...</td>
<td>sde</td>
<td>sde</td>
<td>SELECT s_.eminx,s_.emin,y,s_.emaxx,y...</td>
<td>921</td>
<td>2624</td>
<td>36231</td>
<td>11</td>
</tr>
</tbody>
</table>

### Diagram

- **Index Scan**: Scan a nonclustered index, entirely or only a range.
- **Physical Operation**: Index Scan
- **Logical Operation**: Index Scan
- **Actual Number of Rows**: 51629
- **Estimated I/O Cost**: 1.11424
- **Estimated CPU Cost**: 0.183394
- **Estimated Operator Cost**: 1.29763 (63%)
- **Estimated Subtree Cost**: 1.29763
- **Estimated Number of Rows**: 21.77
- **Estimated Row Size**: 590
- **Actual Row Size**: 0
- **Actual Rebounds**: 0
- **Actual Row Revisions**: 0
- **Ordered**: False
- **Node ID**: 3
Summary

• Optimize ArcGIS Services
• Profile individual user operations and tune if needed
• Drill down through software stack
  - Application
  - Service
  - Mxd
  - Layer
  - DBMS query
• Correlate your findings between tiers
• Performance and load test
Performance testing
Test Objectives

- Contractual Service Level Agreement
- Bottlenecks
- Capacity
- Benchmark
Test Data
Test Data

Bbox (Using Fiddler)

Selected Extent
From HTTP Debugging Proxy

Area of Interest

QueryString

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>f</td>
<td>image</td>
</tr>
<tr>
<td>dpi</td>
<td>96</td>
</tr>
<tr>
<td>transparent</td>
<td>true</td>
</tr>
<tr>
<td>format</td>
<td>png8</td>
</tr>
<tr>
<td>bbox</td>
<td>&quot;[&quot;min&quot;:62196593.737018872,&quot;ymin&quot;:2303862.765275147,&quot;xmax&quot;:6231478.566248391,&quot;ymax&quot;:2311468.277924415,&quot; spatialReference&quot;:{&quot;wkid&quot;:2230}]&quot;</td>
</tr>
<tr>
<td>bboxSR</td>
<td>2230</td>
</tr>
<tr>
<td>imageSR</td>
<td>2230</td>
</tr>
<tr>
<td>size</td>
<td>1222.782</td>
</tr>
</tbody>
</table>
Test Data

Attribute Data
Test Data

Generate Bboxes

One simple example of Python script to generate Bboxes

```python
from random import random

def generateBoxes(fullExtent, gridControl):
    bBoxes = []
    bBoxes.append(fullExtent)
    width = fullExtent[2] - fullExtent[0]
    for grid in gridControl:
        nWidth = width/grid
        nHeight = height/grid
        for row in range(0, grid):
            for column in range(0, grid):
                minX = fullExtent[0]+(column*nWidth)
                minY = fullExtent[1]+(row*nHeight)
                maxX = minX+nWidth
                maxY = minY+nHeight
                bBoxes.append((minX, minY, maxX, maxY))
    return bBoxes

def writeTuple(path, arr):
    try:
        f = open(path, 'w')
        for item in arr:
            f.write('
'.join([str(x) for x in item]) + '
')
        f.close()
    except IOError, (errno, strerror):
        print path
        print "writeTuple I/O error(%s): %s" % (errno, strerror)
    except:
        print "writeTuple Unexpected error:", sys.exc_info()[0]
        raise

if __name__=='__main__':
    extent = (6219593.737018972, 2303862.765275147, 6231478.566248391, 2311468.277924415)
    grid = [2, 0, 16]
    bBoxes = generateBoxes(extent, grid)
    for item in bBoxes:
        print item
    writeTuple("C:\test.csv", bBoxes)
```
Test Data

Heat Map based on response times from ArcGIS Server
Test Data

Observe correlation between feature density and performance
TEST SCRIPTS
Test Scripts

- Record user workflow based on application user requirements
- Create single user web test
  - Define transactions
  - Set think time and pacing based on application user requirements
  - Parameterize transaction inputs
  - Verify test script with single user
LOAD TEST
Load Test

- Create load test
  - Define user load
  - Create machine counters to gather raw data for analysis
- Execute
Visual Studio Quick Introduction – Load Test

Scenarios:
- Test Mix (WebTest or Unit Test),
- Browser Mix,
- Network Mix,
- Step Loads

Perfmon Counter Sets:
Available categories that may be mapped to a machine in the deployment

Run Settings:
- Counter Set Mappings – Machine metrics
- Test duration
Visual Studio - Load Test Run

Threshold rules violated
Testing with GIS Test Tool
GIS Test Tool– Performance Test Capabilities

- Define web tests including QA step for verification
- Define transactions including think time
- Define load tests
- Execute load tests
- Capture system metrics for multiple machines
- View results and export to Excel
ANALYSIS
Tips and Tricks

Analyze Results - Validation

- Compare and correlate key measurements
  - Response Time (increasing, higher than initially profiled for single user)
  - Throughput
  - CPU on all tiers
  - Network on all tiers
  - Disk on all tiers
  - Passed tests
  - Failed test
Tips and Tricks

Analyze Results - Validation

• Lack of errors does not validate a test
  - Requests may succeed but return zero size image
  - Spot check request response content size
Tips and Tricks

Analyze Results – Reporting and Analysis

• Exclude failure range, e.g. failure rate > 5% from the analysis
• Exclude excessive resource utilization range
Tips and Tricks

Analyze Results

Validation Example – Unexpected response time decrease under heavy load

Unexpected curve shape:
Response time should be increasing.

Likely root cause:
failed or 0 size image requests.
Tips and Tricks

**Validation Example – Expected CPU and Response Time Correlation**

Max CPU utilization correlates with:
- User load
- CPU utilization

Response Time increase correlates with:
- User load
- CPU utilization
Tips and Tricks

Validation Example – Test failure due to w3wp memory bottleneck

Symptom: System available memory is decreasing

Root cause: System memory bottleneck at Web Server process
**Tips and Tricks**

**Determining System Capacity**

- Maximum number of concurrent users corresponding to, e.g.:
  - Maximum acceptable response time
  - First failure or 5%
  - Resource utilization greater than 85%, for example CPU

- Different ways of defining acceptance criteria (performance level of service), e.g.
  - 95% of requests under 3 sec
  - Max request under 10 sec
Tips and Tricks - Execute

- Ensure
  - Only target applications are running
  - Application data is in the same state for every test
  - Good configuration management is critical to getting consistent load test results
Report

- Executive Summary
- Test Plan
  - Workflows
  - Work load
- Deployment documentation
- Results and Charts
  - Key Indicators, e.g. Response Time, Throughput
  - System Metrics, e.g. CPU %
  - Errors
- Summary and Conclusions
  - Provide management recommendations for improvements
- Appendix
TEST TOOLS
# Testing - Selecting Load Test Tool

<table>
<thead>
<tr>
<th>Test Tools</th>
<th>Open Source</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadRunner</td>
<td>No</td>
<td>• Industry Leader&lt;br&gt;• Automatic negative correlations identified with service level agreements&lt;br&gt;• Http Web Testing&lt;br&gt;• Click and Script&lt;br&gt;• Very good tools for testing SOA&lt;br&gt;• Test results stored in database&lt;br&gt;• Thick Client Testing&lt;br&gt;• Can be used for bottleneck analysis</td>
<td>• High Cost&lt;br&gt;• Test Development in C programming language&lt;br&gt;• Test metrics difficult to manage and correlate&lt;br&gt;• Poor user community with few available examples</td>
</tr>
<tr>
<td>Silk Performer</td>
<td>No</td>
<td>• Good solution for testing Citrix&lt;br&gt;• Wizard driven interface guides the user&lt;br&gt;• Can be used for bottleneck analysis</td>
<td>• Moderate to High Cost&lt;br&gt;• Test metrics are poor&lt;br&gt;• Test Development uses proprietary language&lt;br&gt;• Test metrics difficult to manage and correlate&lt;br&gt;• Poor user community with few available examples</td>
</tr>
<tr>
<td>Visual Studio Test Team</td>
<td>No</td>
<td>• Low to moderate cost&lt;br&gt;• Excellent Test Metric reporting&lt;br&gt;• Test Scripting in C# or VB.NET&lt;br&gt;• Unit and Web Testing available&lt;br&gt;• Blog support with good examples&lt;br&gt;• Very good for bottleneck analysis</td>
<td>• No built in support for AMF&lt;br&gt;• No Thick Client options&lt;br&gt;• Moderate user community</td>
</tr>
<tr>
<td>JMeter</td>
<td>Yes</td>
<td>• Free&lt;br&gt;• Tool</td>
<td>• Provides only response times&lt;br&gt;• Poor User community with few available examples</td>
</tr>
</tbody>
</table>
GIS Test tool

Quick Preview

Coming Soon
Tool selection depends on objective
- Commercial tools all have system metrics and correlation tools
- Free tools typically provide response times and throughput, but leave system metrics to the tester to gather and report on
CAPACITY PLANNING
Input Capacity Planning

- Find Input for Capacity Planning
  - Test Report
    - Includes Throughput (Transactions per hour)
    - Includes System Metrics – %CPU Utilization, #Cores
    - Spec Rate from the machines tested
  - Use this information to calculate Service Time for Transactions
Input Capacity Planning

- Capacity model expressed as Service Time

\[ ST = \frac{\#CPU \times 3600 \times \%CPU}{TH \times 100} \]
Capacity Planning

- Estimate capacity for a different hardware platforms
  - Find your target (t) server on
    http://www.spec.org/cpu2006/results/rint2006.html

\[
\#CPU_t = \frac{ST_b \times TH_t \times 100}{3600 \times \%CPU_t} \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}
\]
Capacity Planning

- SPEC.org
  - Find your server from the results
## Capacity Planning

### SPEC® CINT2006 Result

**Dell Inc.**  
**PowerEdge 2950 (Intel Xeon processor X5355, 2.66 GHz)**

<table>
<thead>
<tr>
<th>Test</th>
<th>Copies</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>perlbench</td>
<td>8</td>
<td>143</td>
</tr>
<tr>
<td>bzip2</td>
<td>8</td>
<td>74.3</td>
</tr>
</tbody>
</table>

**CPU2006 license:** 55  
**Test sponsor:** Dell Inc.  
**Tested by:** Dell Inc.  

**Test date:** Mar-2007  
**Hardware Availability:** Dec-2006  
**Software Availability:** Nov-2006  

**SPECint® rate base2006 = 80.9**  
**SPECint® rate2006 = Not Run**
Capacity Planning

- Additional examples on capacity planning can be found here: