

# Large Models and Related Memory Issues

---

Overview

Bill Oakley

# Physical and Virtual Memory

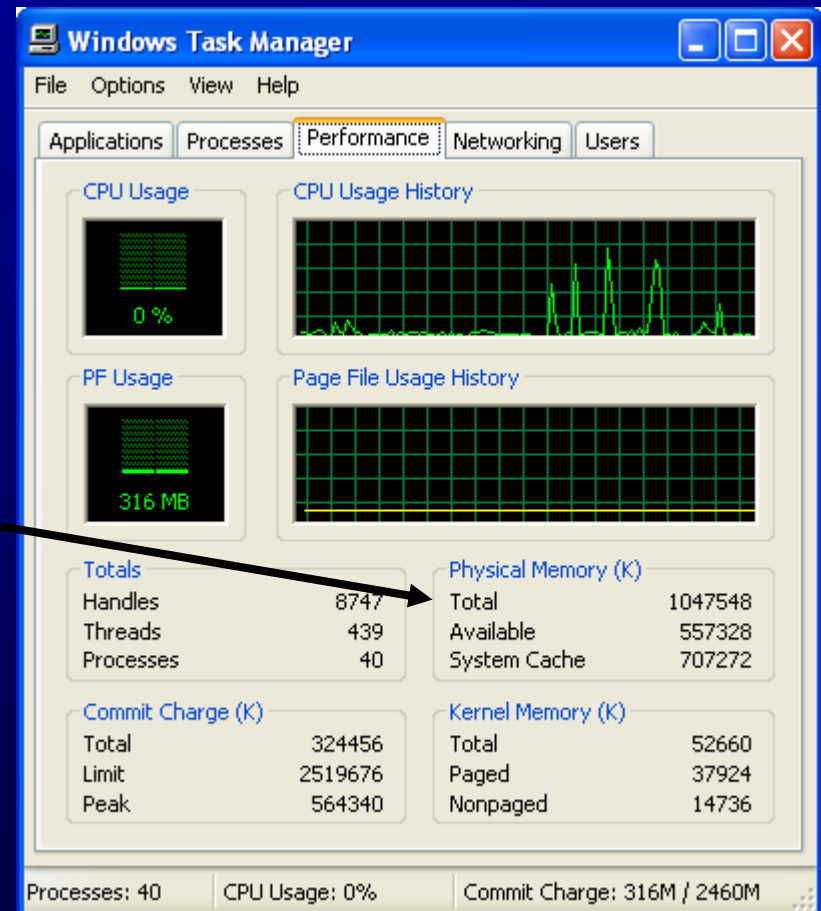
---

- Physical memory (RAM); typically 512M, 1G, 2G or 4G; "fast"
- Virtual memory (on disk, in a page file); expandable; "slow"

# How Much Memory Do I Have?

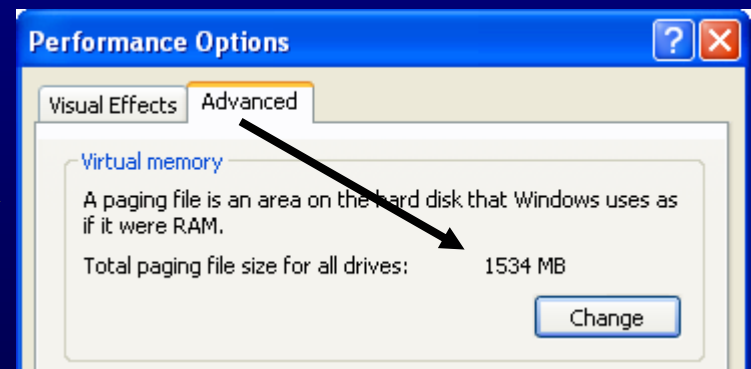
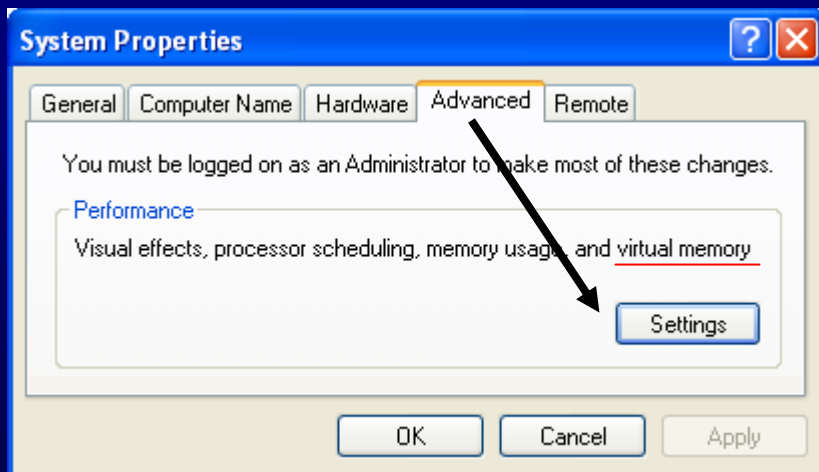
- Windows - Physical
  - Task Manager

Physical



# How Much Memory Do I Have?

- Windows - Virtual
  - Settings ↪ Control Panel ↪ System



# How Much Memory Do I Have?

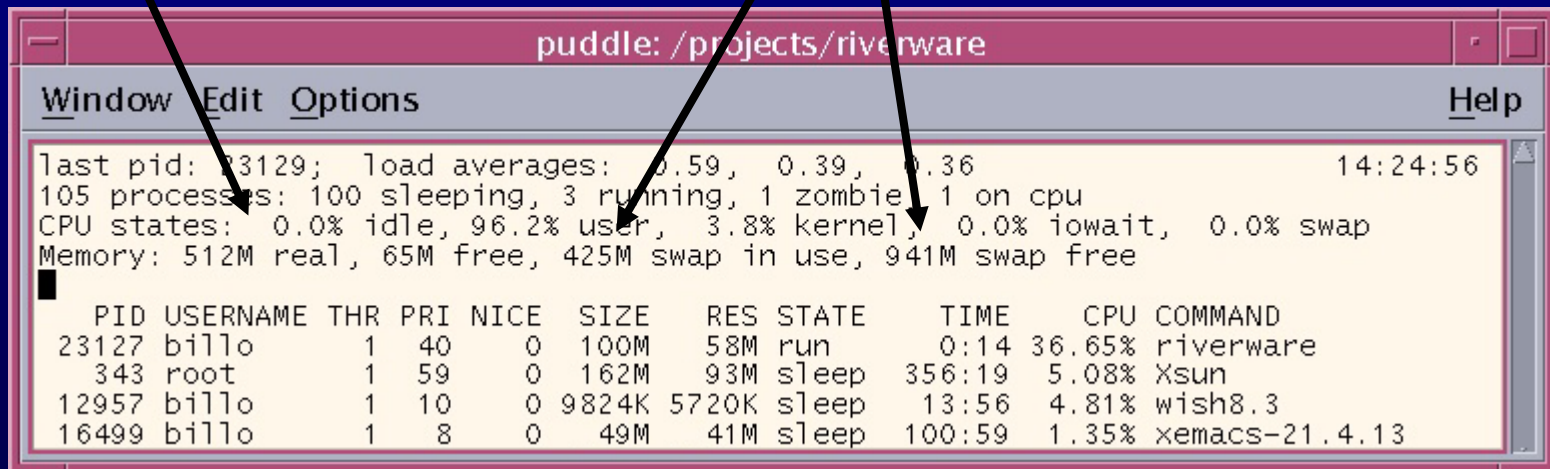
---

- Solaris
  - 'top' command
  - [www.sunfreeware.com](http://www.sunfreeware.com)
  - Precompiled binaries, no registration

# How Much Memory Do I Have?

Physical

Virtual (Sum)



```
puddle: /projects/riverware
Window Edit Options Help
last pid: 23129; load averages: 0.59, 0.39, 0.36 14:24:56
105 processes: 100 sleeping, 3 running, 1 zombie, 1 on cpu
CPU states: 0.0% idle, 96.2% user, 3.8% kernel, 0.0% iowait, 0.0% swap
Memory: 512M real, 65M free, 425M swap in use, 941M swap free
PID USERNAME THR PRI NICE SIZE RES STATE TIME CPU COMMAND
23127 billo 1 40 0 100M 58M run 0:14 36.65% riverware
343 root 1 59 0 162M 93M sleep 356:19 5.08% Xsun
12957 billo 1 10 0 9824K 5720K sleep 13:56 4.81% wish8.3
16499 billo 1 8 0 49M 41M sleep 100:59 1.35% xemacs-21.4.13
```

# Increasing Virtual Memory

---

- On Windows and Solaris virtual memory can be increased

# Physical Versus Virtual Memory

---

- Memory allocated in pages
- Large applications have some pages in physical memory, rest in virtual memory (on disk)
- When application references memory address in page in virtual memory, page fault occurs
- Operating system traps page fault, writes pages in physical memory to disk, reads pages from disk to physical memory



# Physical Versus Virtual Memory

---

- Process is known as paging (or swapping)
- Large applications can spend significant time paging, slowing execution down (a condition sometimes called thrashing)

# How Much Memory Does RiverWare Need?

---

- Highly model dependant
- In general, function of space (number of objects) and time (number of timesteps)
- Ideally would have:  
$$F(N_{\text{res}}, N_{\text{reach}}, \dots, N_{\text{timesteps}})$$
- Instead recommend 1G or 2G, with the ability to expand

# How Much Memory Is RiverWare Using?

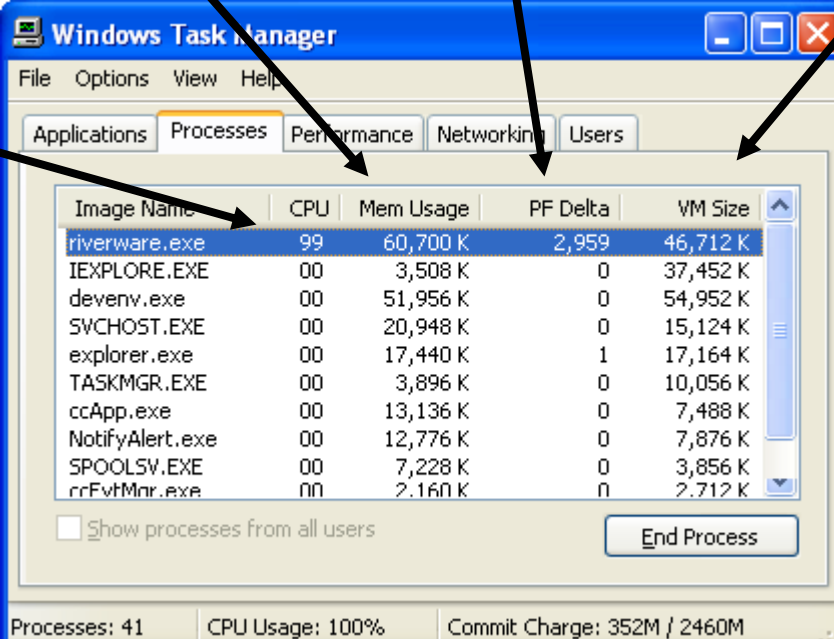
## ■ Windows

Size in physical memory

Paging

Size in virtual memory

CPU Usage



The screenshot shows the Windows Task Manager Performance tab. The 'Processes' tab is selected, and the 'Performance' sub-tab is active. The 'Processes' list is displayed with columns for Image Name, CPU, Mem Usage, PF Delta, and VM Size. The 'riverware.exe' process is highlighted in blue. The status bar at the bottom shows 'Processes: 41', 'CPU Usage: 100%', and 'Commit Charge: 352M / 2460M'.

Image Name	CPU	Mem Usage	PF Delta	VM Size
riverware.exe	99	60,700 K	2,959	46,712 K
IEXPLORE.EXE	00	3,508 K	0	37,452 K
devenv.exe	00	51,956 K	0	54,952 K
SVCHOST.EXE	00	20,948 K	0	15,124 K
explorer.exe	00	17,440 K	1	17,164 K
TASKMGR.EXE	00	3,896 K	0	10,056 K
ccApp.exe	00	13,136 K	0	7,488 K
NotifyAlert.exe	00	12,776 K	0	7,876 K
SPOOLSV.EXE	00	7,228 K	0	3,856 K
rrEvtMgr.exe	00	2,160 K	0	2,712 K

Processes: 41    CPU Usage: 100%    Commit Charge: 352M / 2460M

# How Much Memory Is RiverWare Using?

## ■ Solaris

Total Size

Size in physical memory  
(rest in virtual memory)

CPU Usage

Paging

```
puddle: /projects/riverware
Window Edit Options Help
last pid: 23129; load averages: 0.59, 0.39, 0.36 14:24:56
105 processes: 100 sleeping, 3 running, 1 zombie, 1 on cpu
CPU states: 0.0% idle, 96.2% user, 3.8% kernel, 0.0% iowait, 0.0% swap
Memory: 512M real, 65M free, 425M swap in use, 941M swap free
PID USERNAME THR PRI NICE SIZE RES STATE TIME CPU COMMAND
23127 billo 1 40 0 100M 58M run 0:14 36.65% riverware
343 root 1 59 0 162M 93M sleep 356:19 5.08% Xsun
12957 billo 1 10 0 9824K 5720K sleep 13:56 4.81% wish8.3
16499 billo 1 8 0 49M 41M sleep 100:59 1.35% xemacs-21.4.13
```

# Memory Limits

---

- 32-bit architectures provide 4G of addressable memory
- Solaris separates system memory from application memory – RiverWare has 4G available
- Windows doesn't separate system memory from application memory – RiverWare has (by default) 2G available

# Increasing Memory Limits

---

- Windows – 4 Gig Tuning (4GT)
- Windows – Physical Address Extension (PAE) and Address Windowing Extensions (AWE)
- Windows and Solaris –RiverWare as 64-bit application

# Increasing Memory Limits (cont)

---

- Windows – 4 Gig Tuning
  - 3G for RiverWare, 1G for system
  - Windows XP Professional, Windows Server 2003, Windows NT 4.0 Enterprise
  - May not work with Windows XP SP1 (Microsoft Knowledge Base article 328269)
  - boot.ini switches (/3GT and /USERVA)
  - As of Release 4.4 RiverWare is “large address aware” for 4 Gig Tuning (no code changes)

# Increasing Memory Limits (cont)

---

- Windows – Physical Address Extension and Address Windowing Extensions
  - Allow RiverWare to directly address “huge” amounts of memory while continuing to use 32-bit pointers
  - Windows Server 2003 Enterprise Edition, Windows Advanced Server, Windows 2000 Datacenter/Advanced Server
  - RiverWare would need custom memory allocation routines (substantial code changes)



# Increasing Memory Limits (cont)

---

- Windows and Solaris – RiverWare as 64-bit application
  - Windows Server and Solaris support 64-bit applications
  - RiverWare would have to be “64-bit clean” (unknown code changes)
  - Third-party libraries would have to be either acquired or replaced

# Increasing Memory Limits (cont)

---

- Third-party libraries (cont)
  - ☞ CPLEX – acquire
  - ☞ FlexLM – acquire
  - ☞ Qt – acquire
  - ☞ RogueWave – replace
  - ☞ Galaxy - ???