Instructor Notes

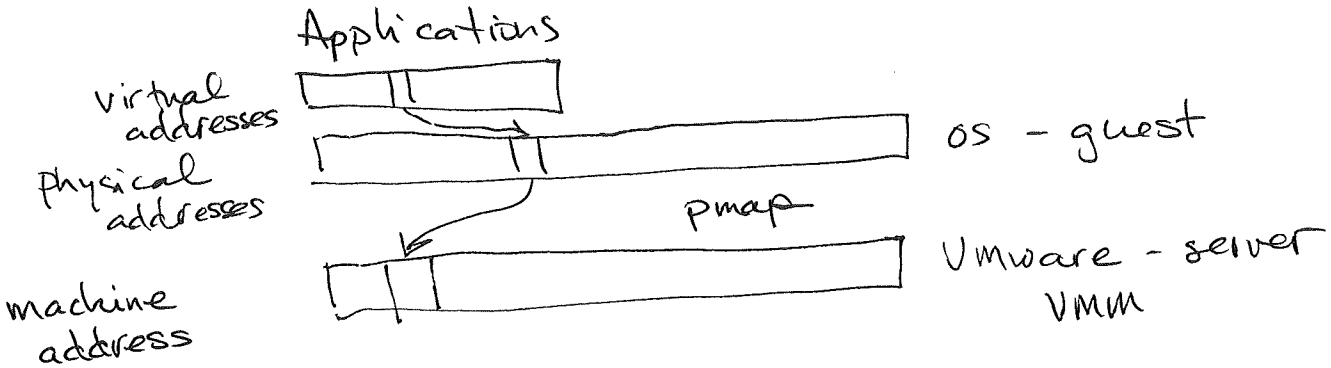
Carl A. Waldspurger
Memory Resource Management in VMware ESX Server
In Proc. Fifth Symposium on Operating Systems Design and Implementation (OSDI '02), Dec. 2002

1. What are the motivations given in this paper for using Virtual Machines? What new requirements does VMware have that Disco did not?

- Individual servers often underutilized
  → consolidate as virtual machines on single physical machine w/ no performance penalty
- Use fewer machines
  → simplify management + reduce costs

Must run completely unmodified OSes
(can't even influence design of guest OS like IBM)

2. Review: Draw a diagram showing how physical memory is mapped to machine memory. What does the pmap data structure do?



pmap: maintained by server - for each VM guest
tracks PPN to MPN

3. What is one problem that server consolidation introduces? One solution is to have the VMM move one of the VM pages to a swap area on disk. What are the problems with this?

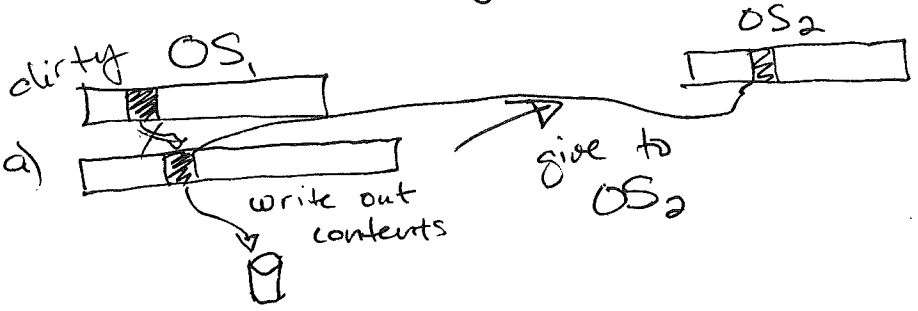
Machine
- Memory may be overcommitted (fine in common case, but not for busy times)

- Each is given illusion of "max size" memory, but don't have Σ max

---

1) Which page to swap out to disk?
Don't have info in server about usage in OS

2) Double paging



b) OS1 runs, has memory pressure, decides to page out P.
- must read/touch page to write it out to its swap device
- extra I/O traffic for nothing

* Not working with the guest to ~~replace~~ yoke page

4. Observation: Can't change OS, but can load a new driver into it. When the VMM server wants to reclaim memory, what does it do? How will the guest OS respond if memory is plentiful? If memory is scarce? What information does the VMM server receive? How can the server ensure that the guest does not touch the returned page?

- Influence/trick OS to revoke most appropriate memory - even uses OS's own policies

- Idea: Balloon module (device driver or kernel service)
  Server communicates directly w/ balloon

Want memory: Inflate balloon
- driver allocates pinned physical pages
- guest OS must find best physical pages

1) Lots of free mem?
   - Give pages from free list

2) Scarce?
   - OS pages some out itself (avoid double, choose most approp)

Balloon passes back physical pages that were allocated to it

- Server deallocates entries in pmap + machine pages can be given to another guest

---

Guest access to ppn goes thru pmap
- generate special fault, give guest new ppn → mpn

5. What does Figure 2 show?

- performance of memory-intensive application (dbench) very similar w/ ballooning to size X as running w/ size X !

- slightly worse because guest OS used more resources when give more memory initially

6. What is one opportunity that server consolidation introduces? How did Disco previously find these opportunities?

- Sharing!

Disco: required changes + intrusion

- change "bcopy()" to copy-on-write sharing
- interpose on disk accesses + network (to look at addresses)
- changed some alignment of data structures

7. How does VMware determine if multiple pages can be shared? What are the advantages of this general approach? How is hashing used to check with pages that have been marked copy-on-write? What happens if a match is found? What happens if a match is not found?

- Identical content → can be shared
  - + no changes
  - + finds more sharing!
- Use hashing as quick check that 2 pages might be identical
  - Key: look @ copy-on-write pages so know that contents match hash

periodically scan for copies:
- hash contents of page
- look up in hash table for match w/ COW pages

match? do full compare

match full? use COW remapping for ppn to mpn

no match? turn to COW page?

no - would make writes too expensive

instead - use as hint

if match w/ hint, must double check that hash still matches contents (could have changed)

8. How much memory savings do these techniques lead to?

Figure 4. Best case - identical VMs

Reclaimed line is savings

Even w/ 1 VM have sharing (~12%)

Fig. 5: 3 production deployments

7.2, 18.7, 32.9% savings

low?

9. What is an additional requirement for competing guests in a consolidated environment? What is a basic approach for giving each guest its fair share of memory? Which guest will have a page revoked? (What is the formula?)

- Want performance isolation, QoS guarantees

- Use tickets (or shares)
  - Get memory $\propto$ Shares
  - Use more memory when underutilized

min-funding revocation

compute $\frac{\text{shares}}{\text{pages}} = \frac{S}{P} = r$

revoke from process w/ lowest r

(paying the least for resources)

10. Why doesn't the completely fair approach lead to the best aggregate, system-wide performance? How is the shares-per-page formula modified? What does the server need to know to apply this formula?

Idle clients w/ many shares can hoard under-used resources
Busy clients would get more benefit from that memory
Performance isolation + efficient utilization conflict

Idle Memory Tax
- Reclaim from clients not actively using memory

$$r = \frac{S}{P \cdot (f + k(1-f))}$$

f: active fraction
k: idle page cost

set k to get different tax rates

How do we know f??

11. How is the amount of idle memory in a guest obtained without modifying the guest?

- Sample accesses to memory over some interval

At some interval:

- Pick small # n of VMs physical pages @ random
- Invalidate mappings (TLB) so Server sees accesses
  - (re-establish mappings on access)
  - increment counter, t (pages touched)

(active) $f = \frac{t}{n}$

sample 100 pages / 30 secs → 100 minor page faults

relatively low cost

2 smoothing functions over estimates

12. What do Figures 6 and 7 show?

Fig 6:

- toucher app
- Fast ave: leads on increases
- Slow ave: lags on decreases
- Max: picks fast or slow to not penalize apps that are changing

Fig 7:

1) Boot up
2) No idle mem tax
3) Tax imposed

13. What memory parameters need to be set by a system administrator for each VM? How much disk swap space must be reserved for each VM?

- Min - guaranteed even if overcommitted → admission control
- Max - physical mem. configured for guest
- Shares - relative importance vs. other guests

Swap?

Max-min

14. How does the ESX server change its policies as the amount of free memory in the system changes? For example, what if there is more than 6% (high) free mem? Below (soft) 4%? Below (hard) 2%? Below (low) 1% What optimization helps a lot when first booting many guests?

High: nothing

Soft: ballooning

Hard: paging

Low: block execution of VMs above target

Opt?

"share before swap"

shared increased rapidly

15. Conclusions?

- Important problem
- Realistic assumptions led to interesting ideas / techniques

- Have to infer information + look for tricks to control guests if can't change them directly