This paper discusses aspects of memory management from the VMware ESX server.

A very important aspect of any VM layer is how it multiplexes/shares/manages memory. This paper focuses on that aspect as it applies to the commercial VMware ESX server. The ESX server's main advantage is its memory management techniques that allow sharing. One of these techniques is "ballooning," which allows machines to overallocate the total memory between many machines (i.e., 3 machines with 256mb on a machine with only 512mb). Ballooning shows acceptable performance degradation from a machine that actually had the allocated amount of memory. Ballooning can allow sharing or allocation/deallocation of memory between VMs through idle memory allocation changing the percentage of memory that each machine holds on to if not using. This way memory not being used can be used by other machines (the whole idea behind SHARED memory). This aspect shows ok performance. Also, they share memory in the sense that multiple machines can access the same memory if it is the same through doing page signature matching. They do an initial search matching hashes of pages.. and then upon successful hash matches actually scan the entire page to ensure similarity.. this is maybe a good idea... couldn't really tell from the paper, they didn't support it well enough... which brings me to what wasn't great about the paper...

Yes, the biggest drawback seemed to be that everything was done at a user level that knew little of the OS machines above them, they had to be very tricky about how to not lose performance to multi-level paging and such... it seems that this approach or abstraction is far from perfect... the sharing of memory by matching content seems like a good idea... but the whole time I am just wondering if there is a much simpler approach, or at least more proper one. First off, for efficient virtual machines... I think it seems necessary that the OS be protected to run as a virtual machine this could possibly make efficient implementation of a virtual machine much easier... Some standard communication layer or protocol between the virtual machines and the VM management layer seems like a better solution in the long run... and if there was such a thing that was proven effective and the protocol of communication simple enough... maybe developers would make the ports...

In the future... well yeah... I think this approach of running unmodified OS (at least unmodified OS that didn't have virtual machines in mind when designed) is not a final solution... a good way to get a working solution for now... but not for the future...