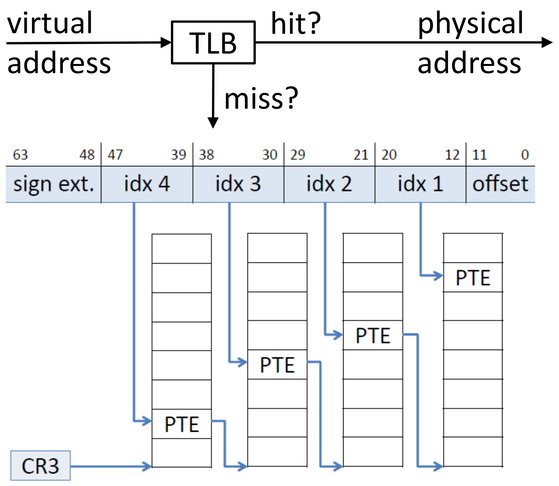
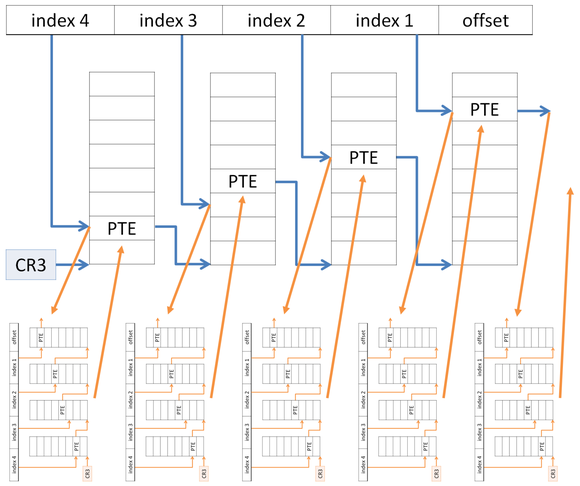
Hash, Don't Cache (the Page Table)

Idan Yaniv, Dan Tsafrir
{idanyani,dan}@cs.technion.ac.il

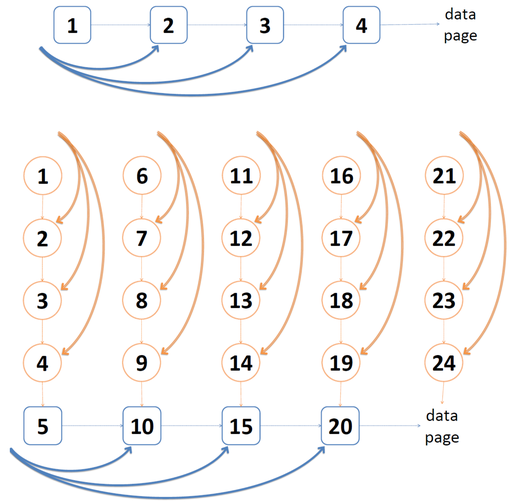
x86-64 processors use radix page tables. TLB misses incur up to 20% overhead.



Virtualization amplifies the page walk overhead---up to 75% of the runtime.



Page walk caches (PWCs) accelerate page walks by caching partial translations.



Hashed page tables yield short page walks without resorting to PWCs.

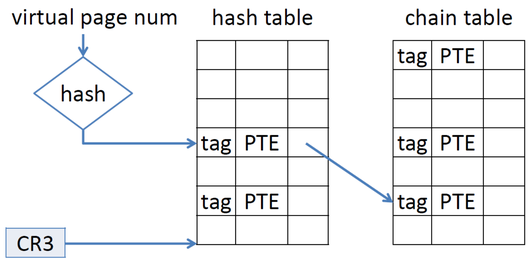
# memory references per walk		
	radix	hashed
bare-metal	4	1
virtual	24	3

Surprisingly, an ISCA'10 study found that:

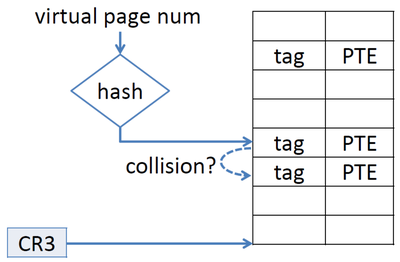
> "[hashed page table] increases the number of DRAM accesses per walk by over 400%"

Contribution #1:
optimizing hashed page tables

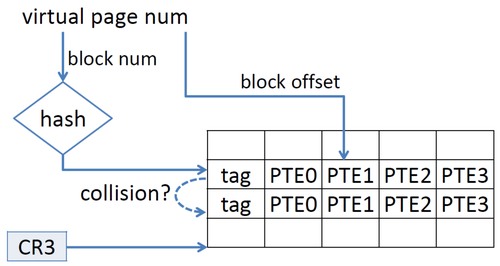
Baseline:



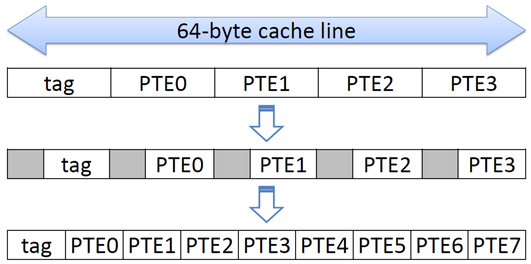
(1) Open addressing:



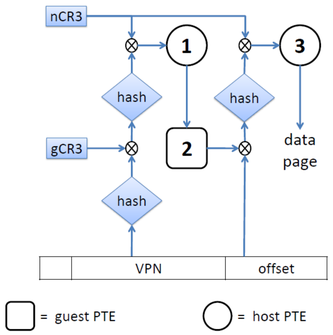
(2) Clustering page table entries:



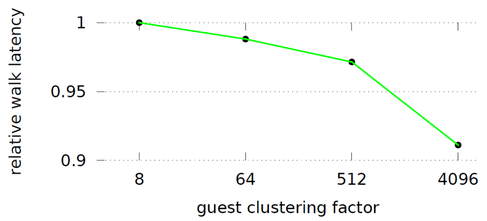
(3) Compacting 8 entries in a cache line (the same as radix):



Contribution #2:
designing 2D hashed for virtualization



The guest and the host page tables should be configured differently.



Contribution #3:
comparing hashed with perfect PWCs

average runtime improvement		
	perfect PWCs	hashed
bare-metal	6 %	8 %
virtual	16 %	17 %

SPEC CPU2006 benchmarks: