

# Revisión: Efficient Virtual Memory for Big Memory Servers, ISCA 2013

Antonio Solana Suárez




October 2023

- 1 Introducción
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment
- 4 Implementacion
- 5 Evaluación
- 6 Análisis del paper

- La memoria es una parte crítica en el rendimiento de los sistemas de cómputo.
- Las aplicaciones cada vez usan más memoria.<sup>1</sup>
- Una de las ventajas de la virtualización es que permite a los procesos usar más memoria que la física disponible. **Swapping**
- **Escasez** de memoria(Antes) vs **Abundancia**(Ahora)

La virtualización de la memoria está presente en estos sistemas pero esto no es gratis. **Traducción de direcciones**

---

<sup>1</sup>Rensel, D. 2018. IDC The Digitization of the World From Edge to Core   

- El TLB es una caché para acelerar el proceso de traducción de páginas virtuales a páginas físicas.
- Trabaja en paralelo con acceso a memoria, solo overhead en fallos del TLB
- ¿Es este el único coste que tiene el TLB?

- El TLB es una caché para acelerar el proceso de traducción de páginas virtuales a páginas físicas.
- Trabaja en paralelo con acceso a memoria, solo overhead en fallos del TLB
- ¿Es este el único coste que tiene el TLB?

Los accesos al TLB consumen  $\approx 6\%^2$  de la energía de la CPU.

---

<sup>1</sup>Sodani, A. 2011. Race to Exascale: Opportunities and Challenges. MICRO 2011 Keynote address.

- 1 Introduccion
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment
- 4 Implementacion
- 5 Evaluación
- 6 Análisis del paper

**Table 1. Test machine configuration**

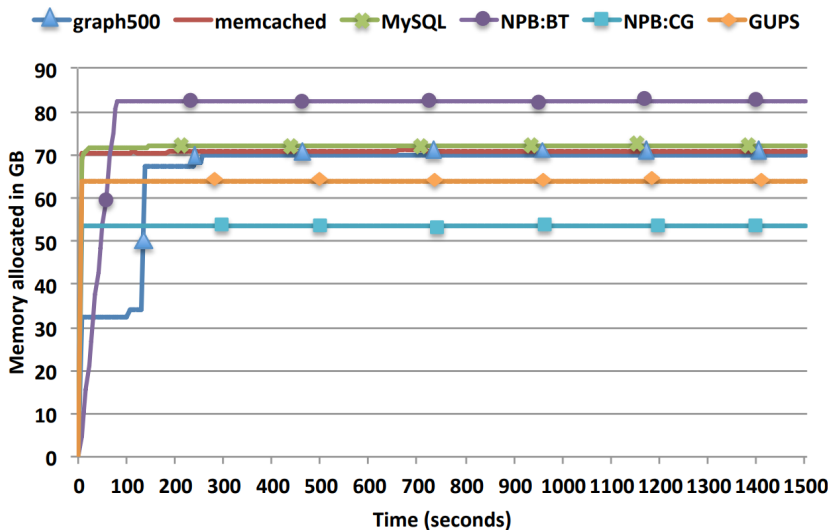
	Description
<b>Processor</b>	Dual-socket Intel Xeon E5-2430 (Sandy Bridge), 6 cores/socket, 2 threads/core, 2.2 GHz
<b>L1 DTLB</b>	4KB pages: 64-entry, 4-way associative; 2MB pages: 32-entry 4-way associative; 1GB pages: 4-entry fully associative
<b>L1 ITLB</b>	4KB pages: 128-entry, 4-way associative; 2MB pages: 8-entry, fully associative
<b>L2 TLB (D/I)</b>	4 KB pages: 512-entry, 4-way associative
<b>Memory</b>	96 GB DDR3 1066MHz
<b>OS</b>	Linux (kernel version 2.6.32)

# Test I Benchmarks ejecutados

- **Graph500:** Generación, compresión(Kernel 1) y búsqueda en anchura(Kernel 2).  
toy(17GB)-mini(140GB)-...-large(140TB)-huge(1.1PB).
- **Memcached:** Bases de datos clave valor en memoria(?)
- **MySQL TPC-C** Transacciones de almacén de ventas: Compras, ventas, gestión del stockage.
- **GUPS**(GigaUpdatesPerSecond): Número de actualizaciones aleatorias por segundo en una matriz gigante.
- **NAS Parallel benchmark:** BP(Block tridiagonal solver), CG(Conjugate gradient)



# Test II Resultados



**Table 3. Page-grain protection statistics**

	<b>Percentage of allocated memory with read-write permission</b>
<b>graph500</b>	99.96%
<b>memcached</b>	99.38%
<b>MySQL</b>	99.94%
<b>NPB/BT</b>	99.97%
<b>NPB/CG</b>	99.97%
<b>GUPS</b>	99.98%

**Table 4. TLB miss cost.**

	Percentage of execution cycles servicing TLB misses			
	Base Pages (4KB)		Large Pages (2MB)	Huge Pages (1GB)
	D-TLB	I-TLB	D-TLB	D-TLB
<b>graph500</b>	51.1	0	9.9	1.5
<b>memcached</b>	10.3	0.1	6.4	4.1
<b>MySQL</b>	6.0	2.5	4.9	4.3
<b>NPB:BT</b>	5.1	0.0	1.2	0.06
<b>NPB:CG</b>	30.2	0.0	1.4	7.1
<b>GUPS</b>	83.1	0.0	53.2	18.3

Las Memory-workloads:

- El uso de memoria es **predecible**
- **No** necesitan las ventajas de la virtualización
- Los fallos del TLB, provocan una pérdida considerable de ciclos de la CPU.

- 1 Introduccion
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment**
- 4 Implementacion
- 5 Evaluación
- 6 Análisis del paper

# Solución propuesta: Direct Segment

- Consiste en realizar una zona contigua en el espacio de direccionamiento del proceso. **Primary Region**
- Esta zona se mapea directamente a memoria a través del uso de 3 registros el **base**, el **limite** y el **offset**.
- Estas zonas no poseen features como fine-grain protection o swapping.

# Direct segment: Características

- Barato en cuanto a hardware
- Mayor escalabilidad
- Coexistencia con la virtualización
- La abstracción de la memoria permanece intacta

# Direct Segment vs Páginas mas grandes Ventajas

- Mejor rendimiento en aplicaciones con accesos aleatorios.
- Menos complejo que modificar el TLB. Hardware más sencillos.
- Mayor flexibilidad
- La abstracción de la memoria permanece intacta. Visión vertical de la memoria.
- Menor fragmentación interna.
- En virtualización reduce el TLB walk al solo tener que realizar el walk del Guest.



# Direct Segment vs Páginas mas grandes Desventajas

- Es inferior en entornos **dinámicos** e **impredecibles**, donde hay muchas procesos que se ejecutan en tiempos cortos
- Puedes llegar a desperdiciar memoria, en software con disperso virtual memory allocation, Thread arena.

# Problemas: ¿Fragmentación?

- La fragmentación externa queda eliminada puesto que la primary region se reserva empleando **múltiplos** del tamaño de página.
- La fragmentación interna puede evitarse debido a la **predictabilidad** del uso de memoria de los programas.

- 1 Introduccion
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment
- 4 Implementacion**
- 5 Evaluación
- 6 Análisis del paper

- **Reservar un rango de direcciones virtuales**, fijo para uso exclusivo del direct segment.
- **Reserva de memoria** paging vs primary region.  
Dos alternativas:
  - **Paging por defecto**, flag en llamada mmap para indicar que pertenece a la primary region
  - **Direct segment por defecto**, regiones anónimas con permisos uniformes van a la primary region.

# Implementación en sistema operativo Hardware

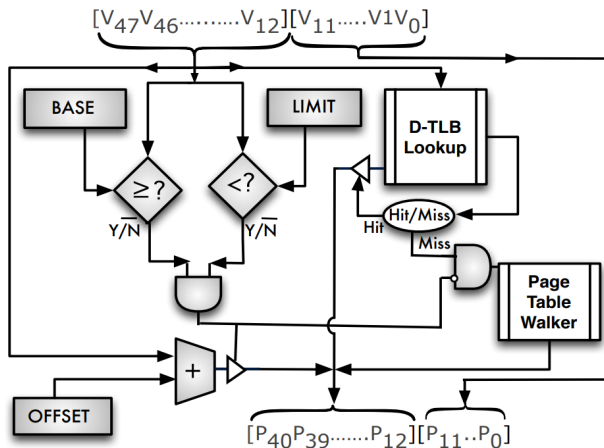


Figure: Esquema funcionamiento del Direct Segment

- 1 Introduccion
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment
- 4 Implementacion
- 5 Evaluación**
- 6 Análisis del paper

- No se ha utilizado un simulador para las pruebas. **Muy costoso**
- Simulacion con gem5 de big-memory workloads puede tardar **meses**
- La **memoria** utilizada en la simulación es de al menos el **doblo**.
- Se implementó un prototipo simple, solo un proceso puede encontrarse usando direct segment. **Primary Process**
- Para la reserva de memoria se ha elegido **paging por defecto**

- Páginas de 4kb, modifica el manejador de fallo de páginas, para introducir el offset.
- Contadores del hardware(**oprofile**) y kernel tweaks para estimar el número de fallos del TLB que serían evitados por direct segment.
- Se hace que el TLB actue como si fuese **software** haciendo que todas las PTE del primary process sea invalido. Modifican el manejador de fallo de páginas, Forcing traps + TLB incoherence
- Se ha estimado lo que tarda el walker en traer la página de memoria al TLB. **Subestima** direct segment por L2 hit



# Evaluacion-Prototipado Forcing traps + TLB incoherence

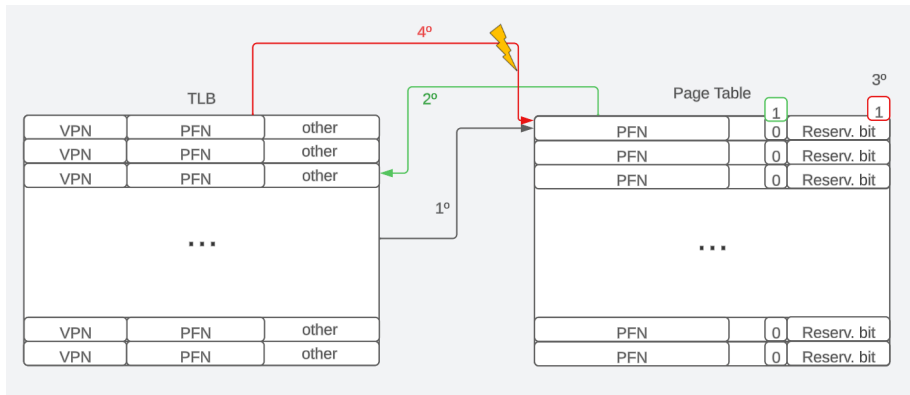


Figure: Detección de TLB misses en el prototipo de evaluación

# Resultados I

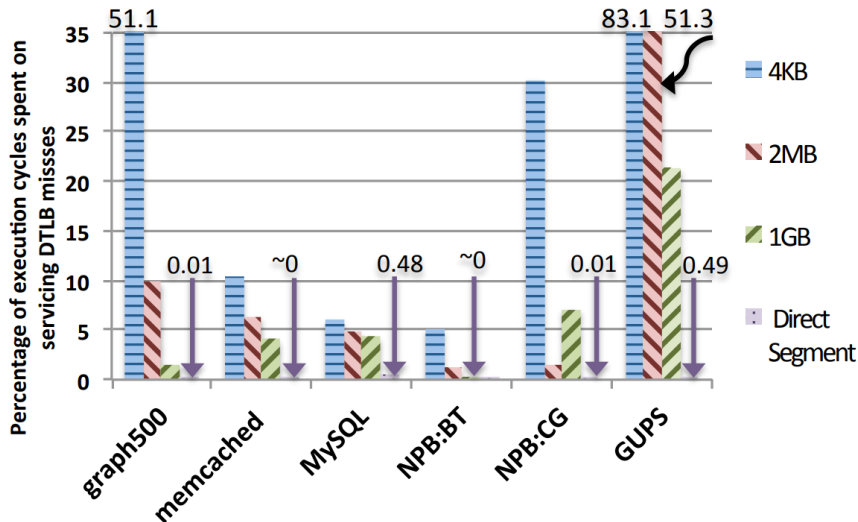
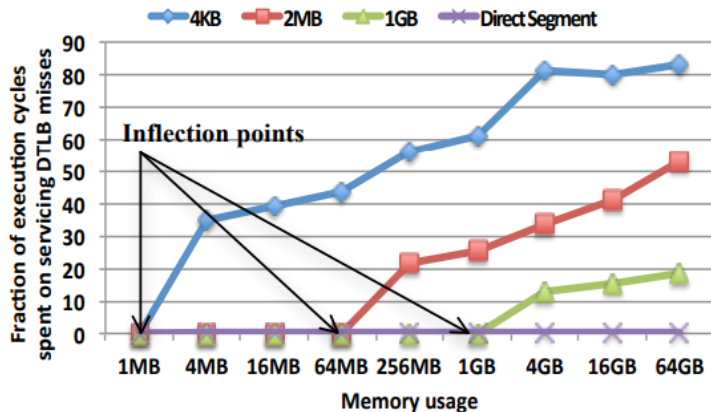


Figure: Resultados Páginas vs Direct Segment Pag.4KB

**Table 5. Reduction in TLB misses**

	<b>Percent of D-TLB misses in the direct segment</b>
<b>graph500</b>	99.99
<b>memcached</b>	99.99
<b>mySQL</b>	92.40
<b>NBP:BT</b>	99.95
<b>NBP:CG</b>	99.98
<b>GUPS</b>	99.99

Figure: Resultados Páginas vs Direct Segment Pag.4KB



**Figure 5. DTLB miss overheads when scaling up GUPS.**

- 1 Introduccion
- 2 Analisis de tareas de servidor intensivas en memoria
- 3 Direct Segment
- 4 Implementacion
- 5 Evaluación
- 6 Análisis del paper**

- Suele realizar una introducción para indicar la estructura de los textos, facilita la lectura.
- Realiza un análisis detallado y estructurado, de las tareas intensas en memoria.
- Las gráficas y las tablas son descriptivas y fáciles de entender.
- Se encuentran recogidas las limitaciones del direct segment

- Uno de los enlaces de la tabla 2 el de la NASA, tiene un typo. No está indicado cuando se visitaron los enlaces.
- La relación entre las gráfica y la tabla de los resultados, no esta justificada.
- El benchmark de memcached no está bien definido.
- La implementación prototipada es algo enrevesada.

# Aprobado



# ¿Preguntas?