Servet: una suite de benchmarks para el soporte del autotuning en clusters de sistemas multinúcleo

Jorge González-Domínguez*, Guillermo L. Taboada, Basilio B. Fraguela, María J. Martín, Juan Touriño

Grupo de Arquitectura de Computadores
Universidad de A Coruña (Spain)
{jgonzalezd,taboada,basilio.fraguela,
mariam,juan}@udc.es

1. Introducción
   - Autotuning
   - Extracción de los Parámetros Hardware
2. Topología Caché
   - Estimación del Tamaño Caché
   - Determinación de las Cachés Compartidas
3. Caracterización de Sobrecargas de Acceso a Memoria
4. Determinación de los Costes de Comunicaciones
5. Conclusiones

1 Introducción
- Autotuning
- Extracción de los Parámetros Hardware

2 Topología Caché

3 Caracterización de Sobrecargas de Acceso a Memoria

4 Determinación de los Costes de Comunicaciones

5 Conclusiones

Introducción al Autotuning

Códigos que adaptan automáticamente su comportamiento según la máquina donde están siendo ejecutados.

Bibliotecas Secuenciales con Autotuning

- **ATLAS** -> Computación numérica (BLAS)
- **FFTW3** -> Transformadas de Fourier
- **Spiral** -> Algoritmos de procesamiento digital de la señal
- Búsqueda costosa para encontrar el algoritmo más adecuado
- El conocimiento de ciertas características hardware permite reducir estos tiempos de búsqueda

Técnicas de Autotuning

Ejemplos de Técnicas de Autotuning

- Tiling -> Dividir los cómputos en bloques de datos que caben en caché para minimizar los fallos caché
- Comunicaciones eficientes:
  - Minimizando el uso de las redes de interconexión
  - Incrementando el uso de la memoria compartida -> generalmente más rápida
- Políticas de mapeado para minimizar:
  - Los fallos caché a causa de cachés compartidas
  - Las sobrecargas en el acceso a memoria
  - El uso de las redes de interconexión

Técnicas de Autotuning

Ejemplos de Técnicas de Autotuning

- Tiling -> Dividir los cómputos en bloques de datos que caben en caché para minimizar los fallos caché
- Comunicaciones eficientes:
  - Minimizando el uso de las redes de interconexión
  - Incrementando el uso de la memoria compartida -> generalmente más rápida
- Políticas de mapeado para minimizar:
  - Los fallos caché a causa de cachés compartidas
  - Las sobrecargas en el acceso a memoria
  - El uso de las redes de interconexión

Técnicas de Autotuning

Ejemplos de Técnicas de Autotuning

- Tiling -> Dividir los cómputos en bloques de datos que caben en caché para minimizar los fallos caché
- Comunicaciones eficientes:
  - Minimizando el uso de las redes de interconexión
  - Incrementando el uso de la memoria compartida -> generalmente más rápida
- Políticas de mapeado para minimizar:
  - Los fallos caché a causa de cachés compartidas
  - Las sobrecargas en el acceso a memoria
  - El uso de las redes de interconexión

Obtención de los Parámetros Hardware

Opción 1 -> De las especificaciones de la máquina

- ¿Siempre?
- ¿Dónde?
- ¿Sin restricciones?
- ¿Con qué formato?
- ¿Exactitud?

Opción 2 -> Con benchmarks

- General -> Se puede usar siempre sin restricciones
- Portable -> El lugar y el formato no dependen del sistema
- Fiable -> Explora el comportamiento real de la máquina

Obtención de los Parámetros Hardware

Opción 1 -> De las especificaciones de la máquina

- ¿Siempre?
- ¿Dónde?
- ¿Sin restricciones?
- ¿Con qué formato?
- ¿Exactitud?

Opción 2 -> Con benchmarks

- General -> Se puede usar siempre sin restricciones
- Portable -> El lugar y el formato no dependen del sistema
- Fiable -> Explora el comportamiento real de la máquina

Estado del Arte

Limitaciones de los trabajos previos

- Método no portable para estimar los tamaños de las cachés indexadas físicamente
- No tienen en cuenta los diferentes tipos de sobrecargas de acceso concurrente a memoria
- Pobre caracterización de las comunicaciones
- Código no disponible

1 Introducción

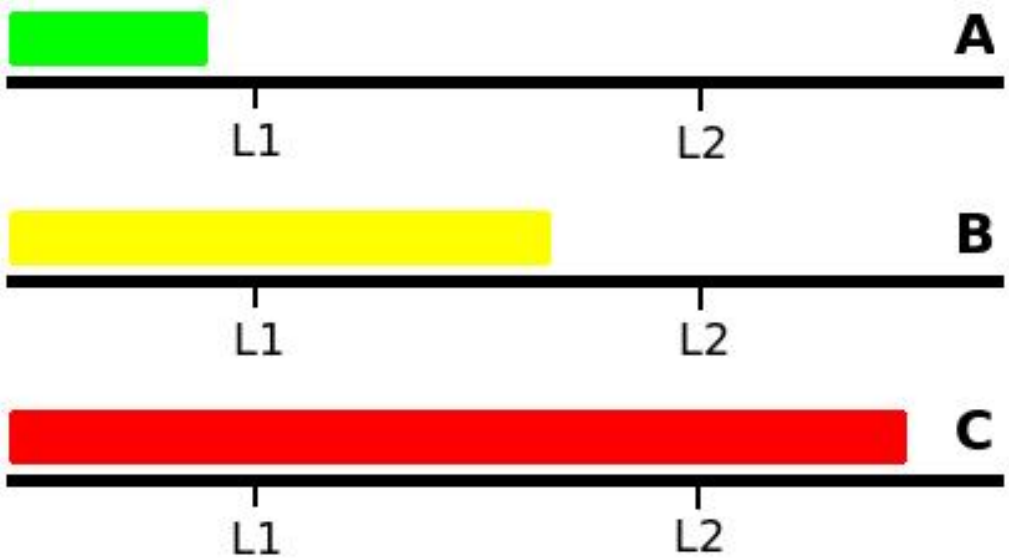
2 Topología Caché
- Estimación del Tamaño Caché
- Determinación de las Cachés Compartidas

3 Caracterización de Sobrecargas de Acceso a Memoria

4 Determinación de los Costes de Comunicaciones

5 Conclusiones

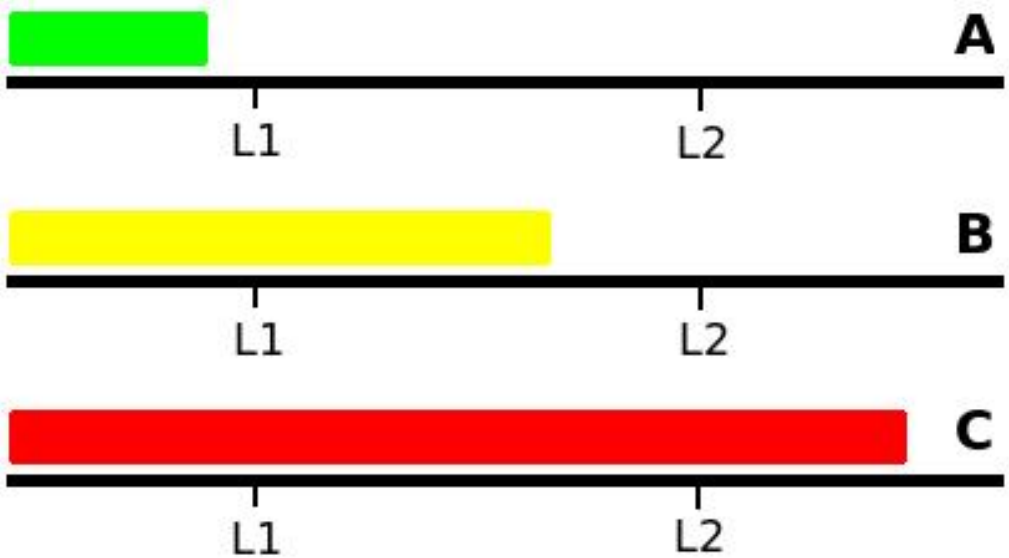
Idea Principal



T -> Tiempo medio de cada acceso

$T_A < T_B < T_C$

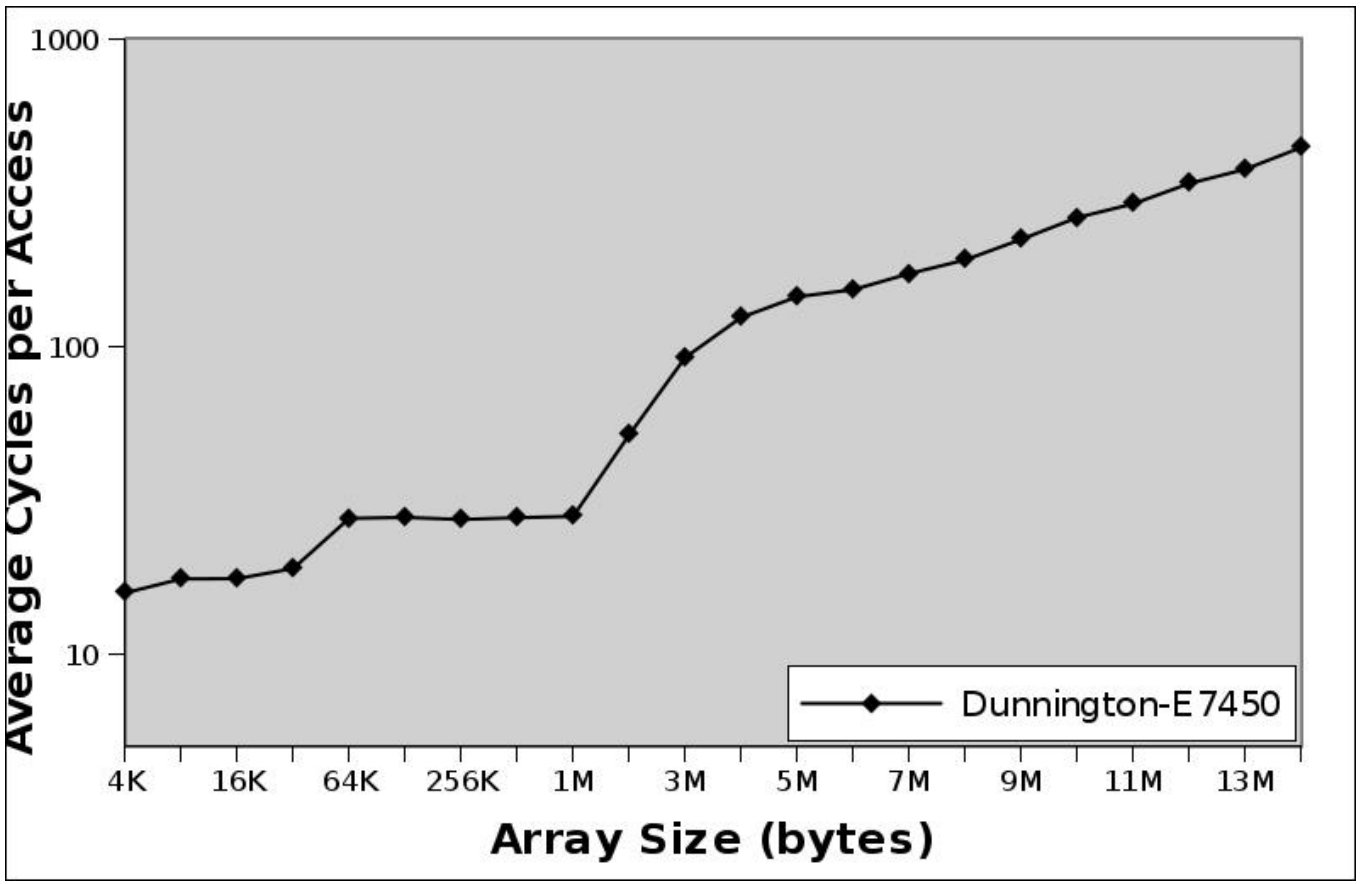
Idea Principal



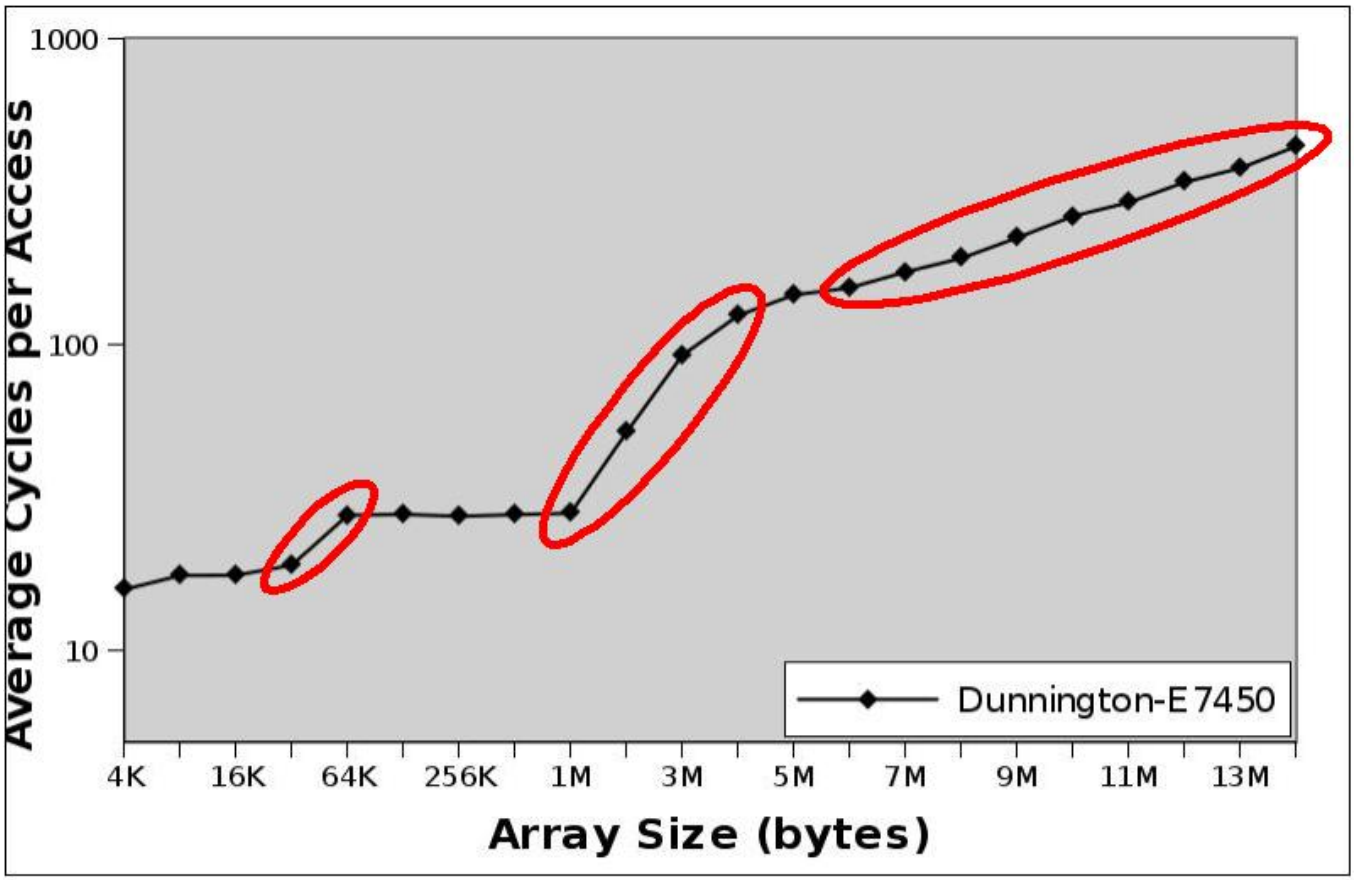
T -> Tiempo medio de cada acceso

$T_A < T_B < T_C$

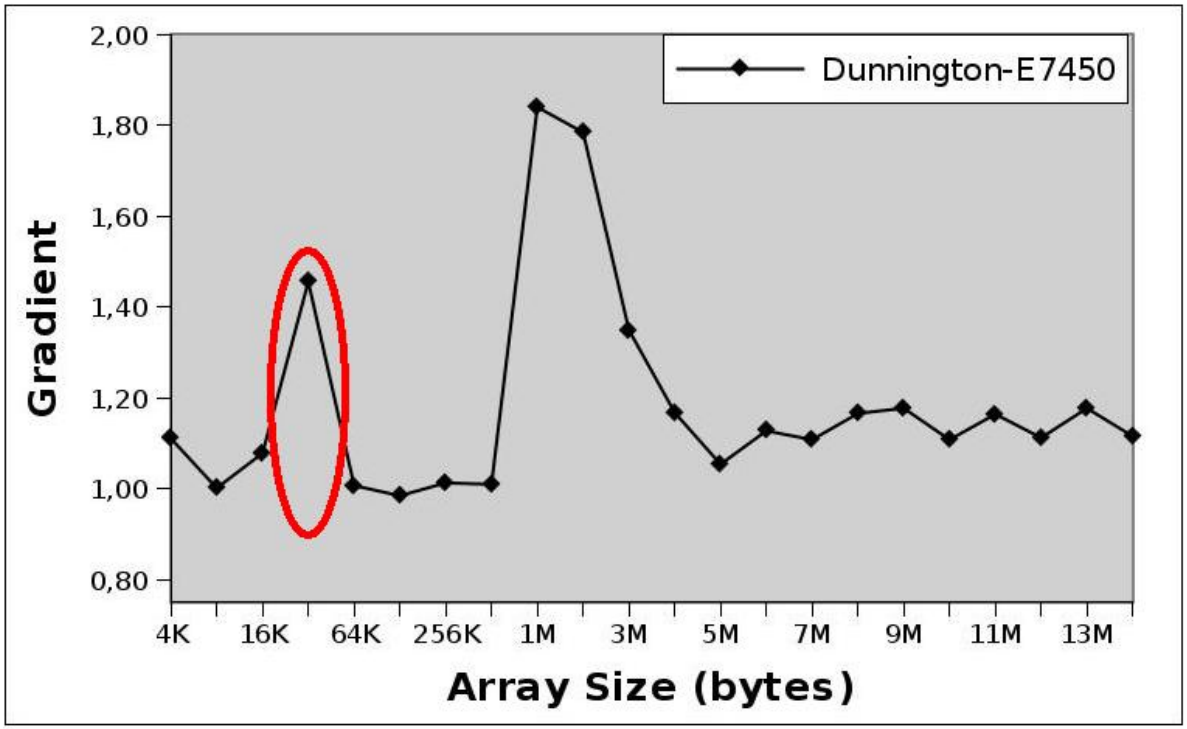
Ejemplo Dunnington: Ciclos



Ejemplo Dunnington: Ciclos

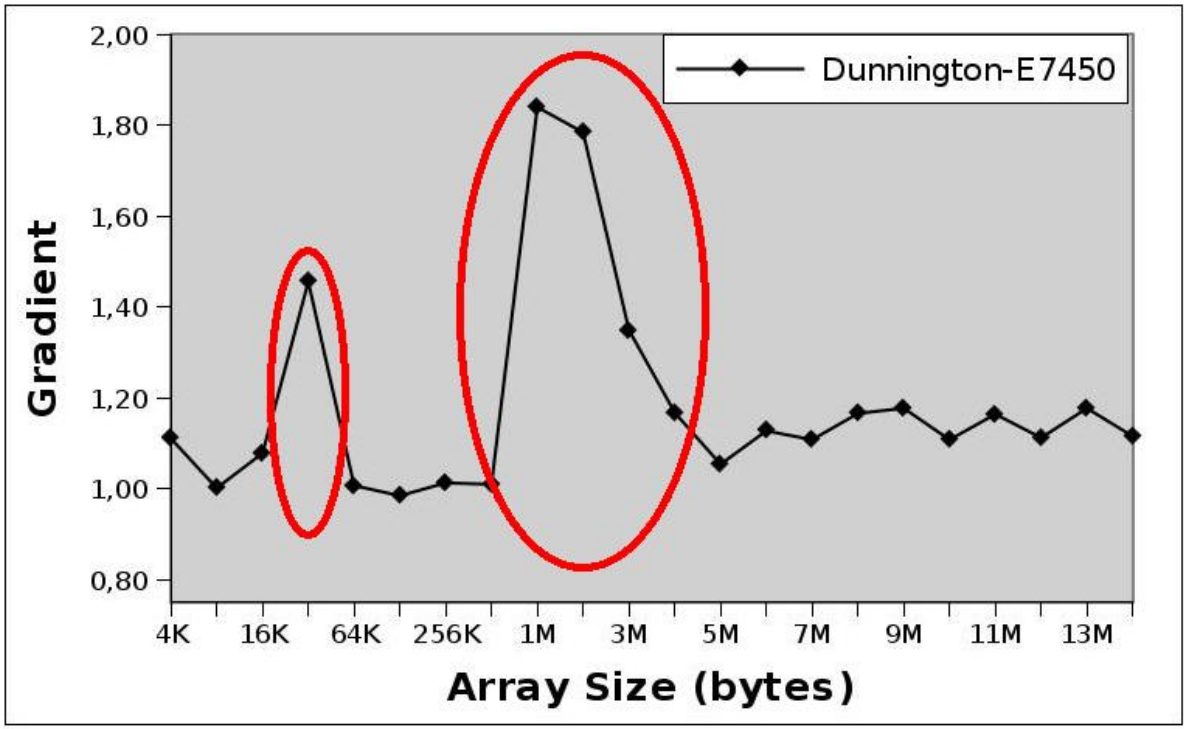


Ejemplo Dunnington: Gradientes (I)



$$L1 = 32KB\checkmark$$

Ejemplo Dunnington: Gradientes (I)



$$L1 = 32KB\checkmark$$
$$L2 = 1MB\ \text{X}$$

Problema con L2

Cachés indexadas físicamente

- La mayoría de las cachés L2 y L3
- Si el tamaño caché es mayor que el tamaño de página -> Contigüidad en memoria virtual no implica adyacencia en memoria física
- Fallos caché en tests con tamaños de array más pequeños que la caché estudiada

Soluciones

- Obligar a trabajar como cachés indexadas virtualmente
  - Coloreado de página por el SO -> No en Linux
  - Llamadas al SO -> Trabajos previos -> No portable
- Estimación desde el comportamiento real -> Servet

Problema con L2

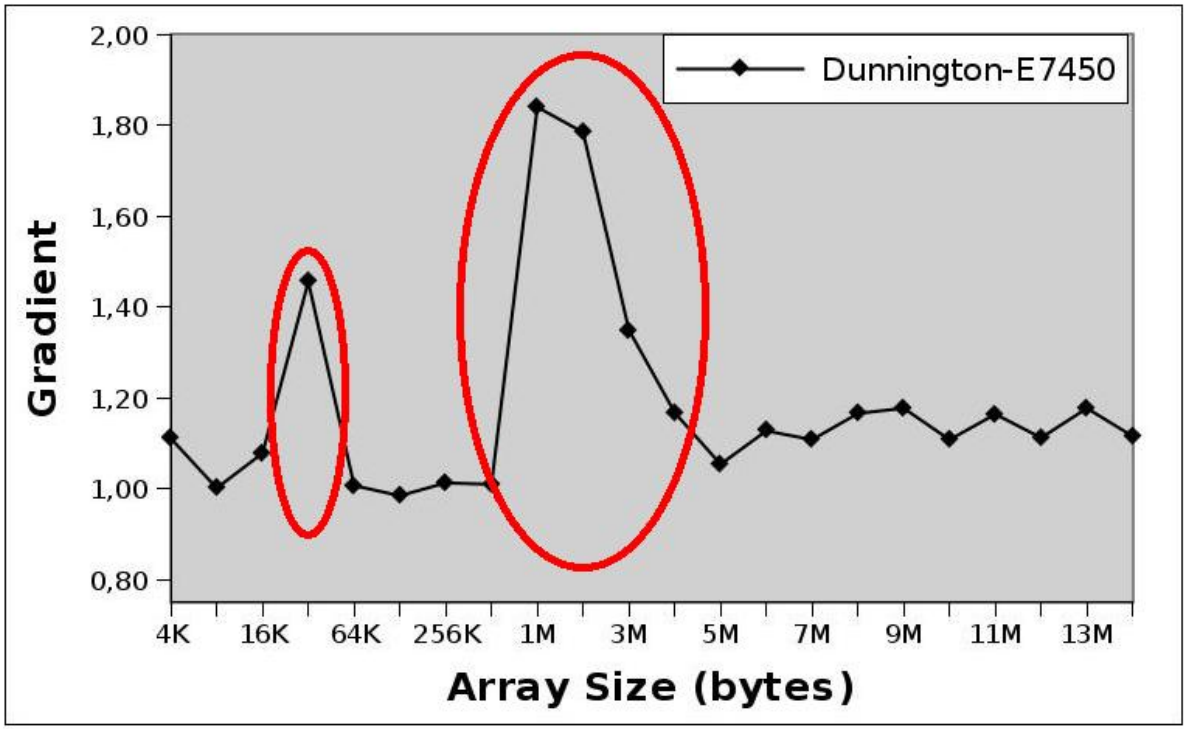
Cachés indexadas físicamente

- La mayoría de las cachés L2 y L3
- Si el tamaño caché es mayor que el tamaño de página -> Contigüidad en memoria virtual no implica adyacencia en memoria física
- Fallos caché en tests con tamaños de array más pequeños que la caché estudiada

Soluciones

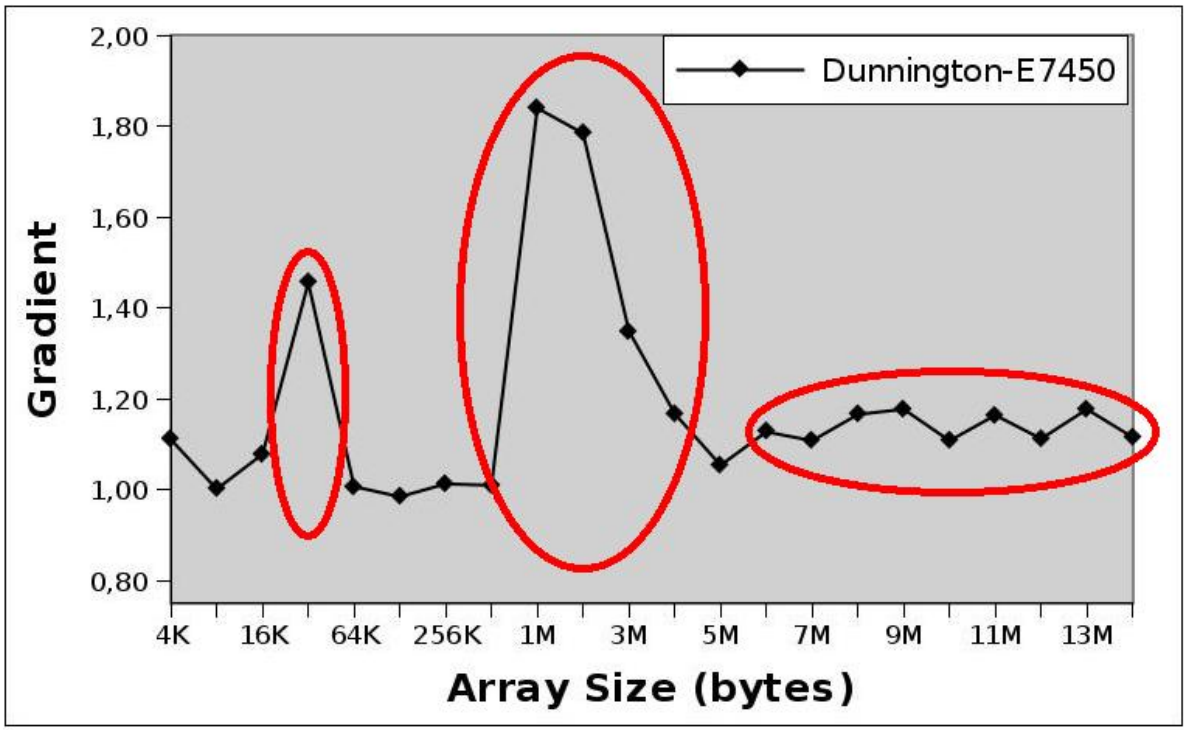
- Obligar a trabajar como cachés indexadas virtualmente
  - Coloreado de página por el SO -> No en Linux
  - Llamadas al SO -> Trabajos previos -> No portable
- Estimación desde el comportamiento real -> Servet

Ejemplo Dunnington: Gradientes (y II)



$$L1 = 32KB\checkmark$$
$$L2 = 3MB\checkmark$$

Ejemplo Dunnington: Gradientes (y II)



$$L1 = 32KB\checkmark$$
$$L2 = 3MB\checkmark$$
$$L3 = 12MB\checkmark$$

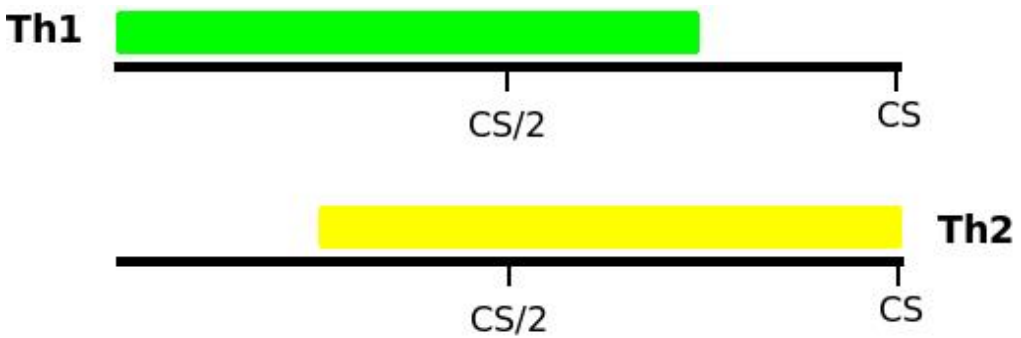
Evaluación Experimental

Entorno académico: Estructura de Computadores 2

- 70 máquinas distintas
- 147 cachés distintas
- 140 estimaciones correctas -> 95%

Idea Principal

1. Caché no compartida



2. Caché compartida

T -> Tiempo medio por acceso

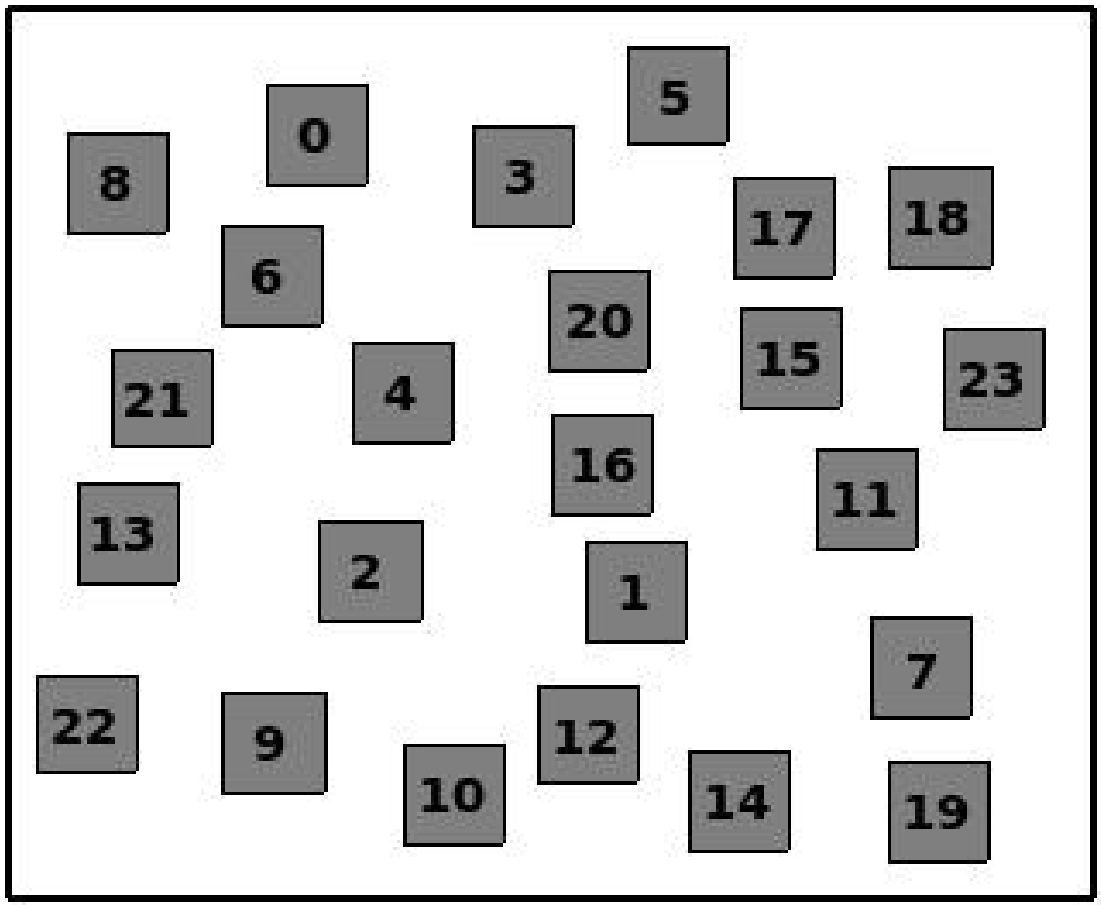
$T_1 < T_2$

Determinación de Cachés Compartidas

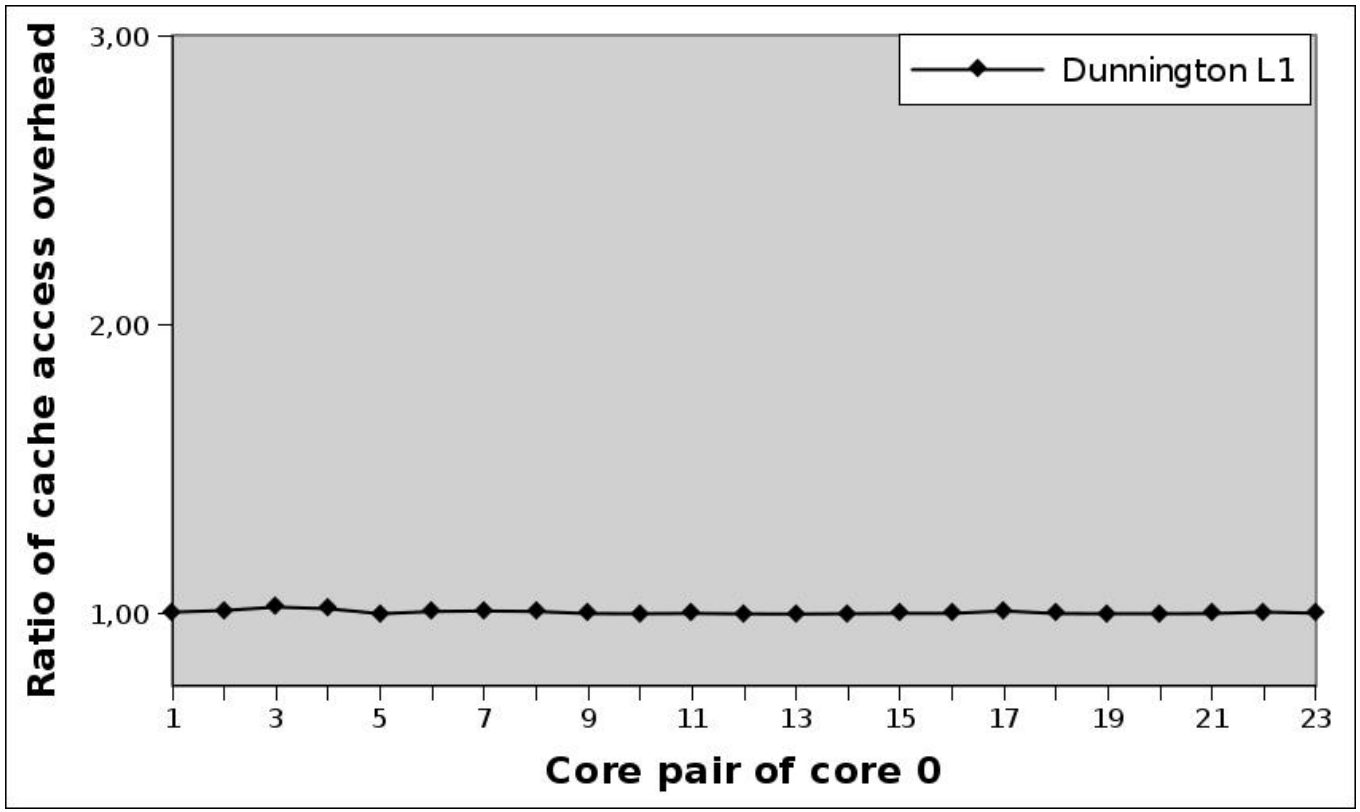
Algoritmo

foreach(CS)
  ref = número de ciclos al acceder un núcleo a un array de tamaño $(CS * 2)/3$
  foreach(par de núcleos)
    c = número de ciclos al acceder ambos núcleos simultáneamente a un array de tamaño $(CS * 2)/3$
    $ratio = c/ref$
    if $ratio > 2$ then **CACHÉ COMPARTIDA**

Topología Inicial

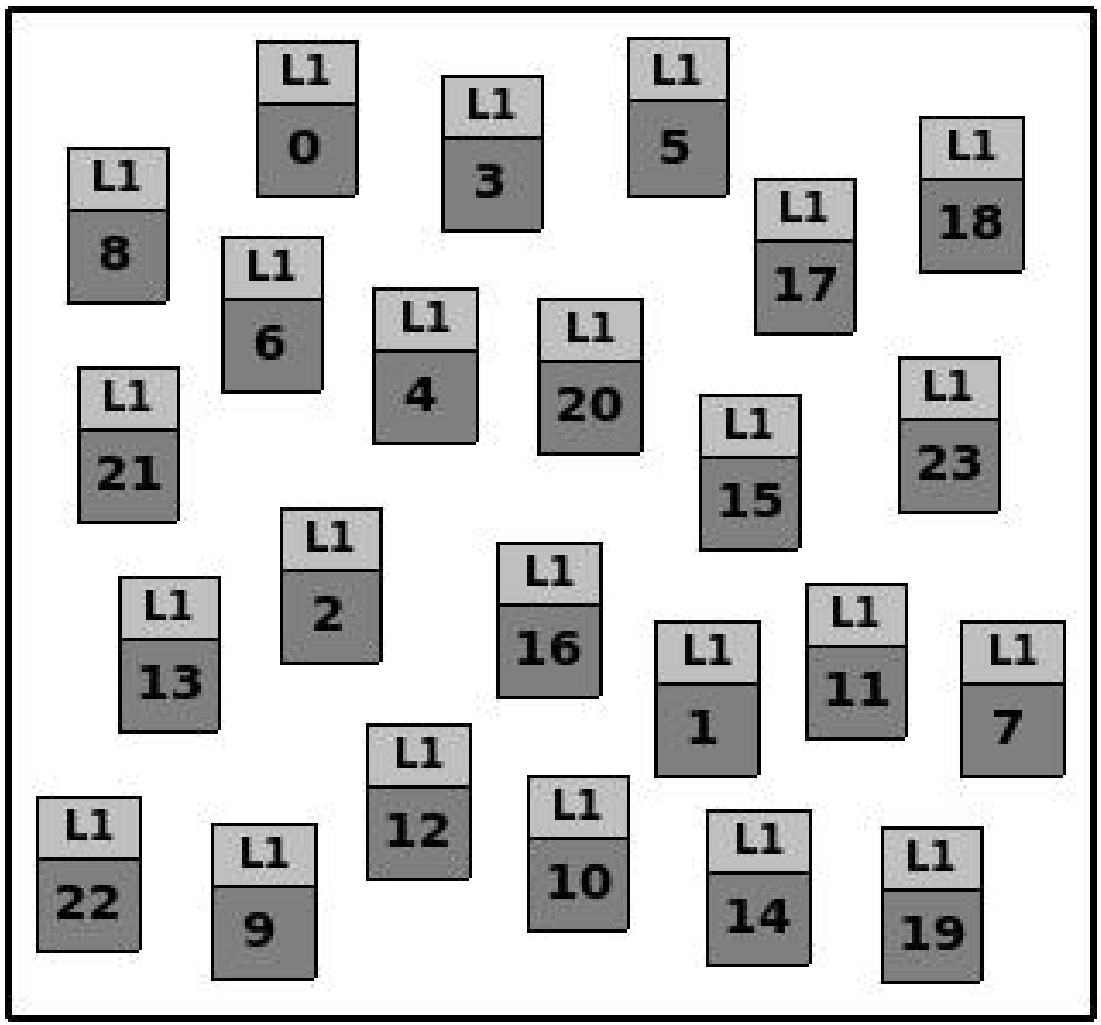


Ejemplo Dunnington: Resultados para L1

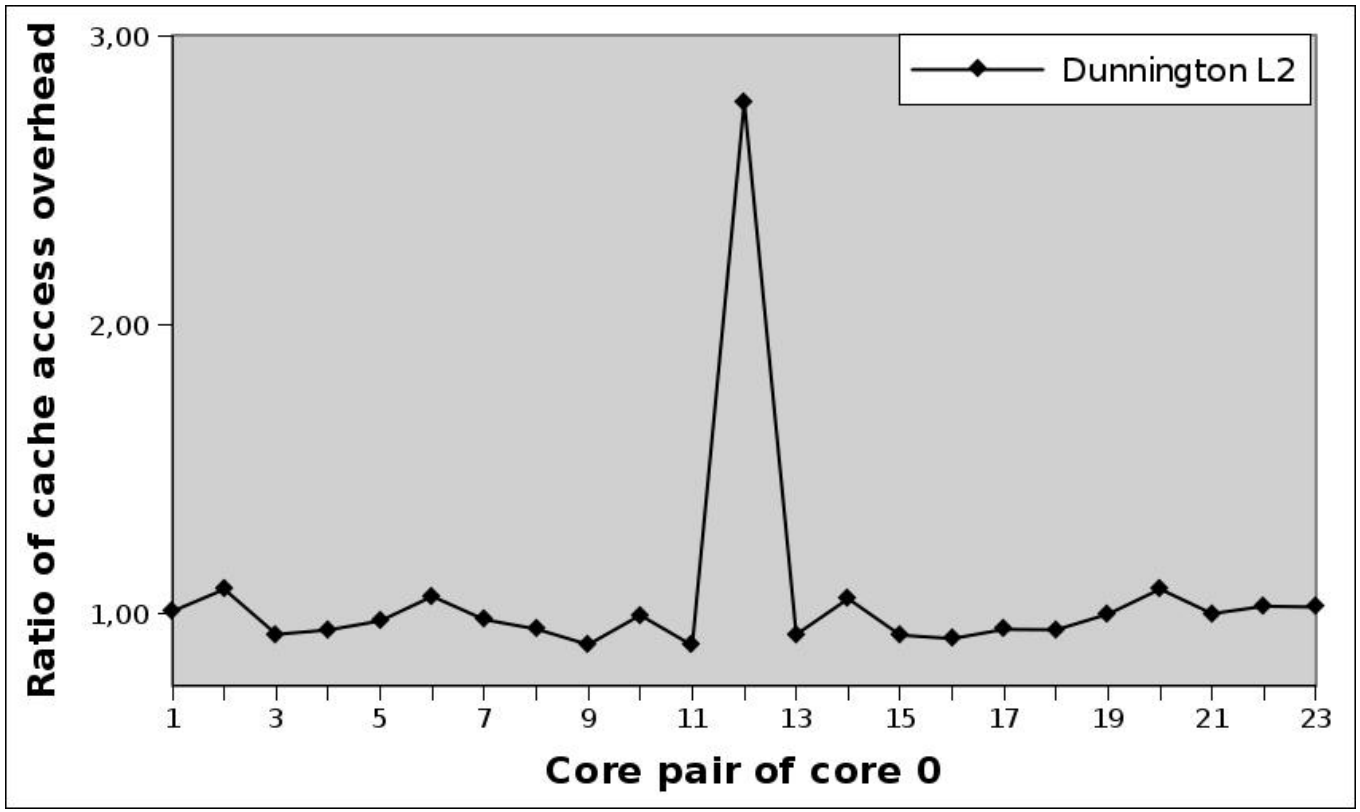


$L1 \rightarrow$ no compartida

Topología con L1

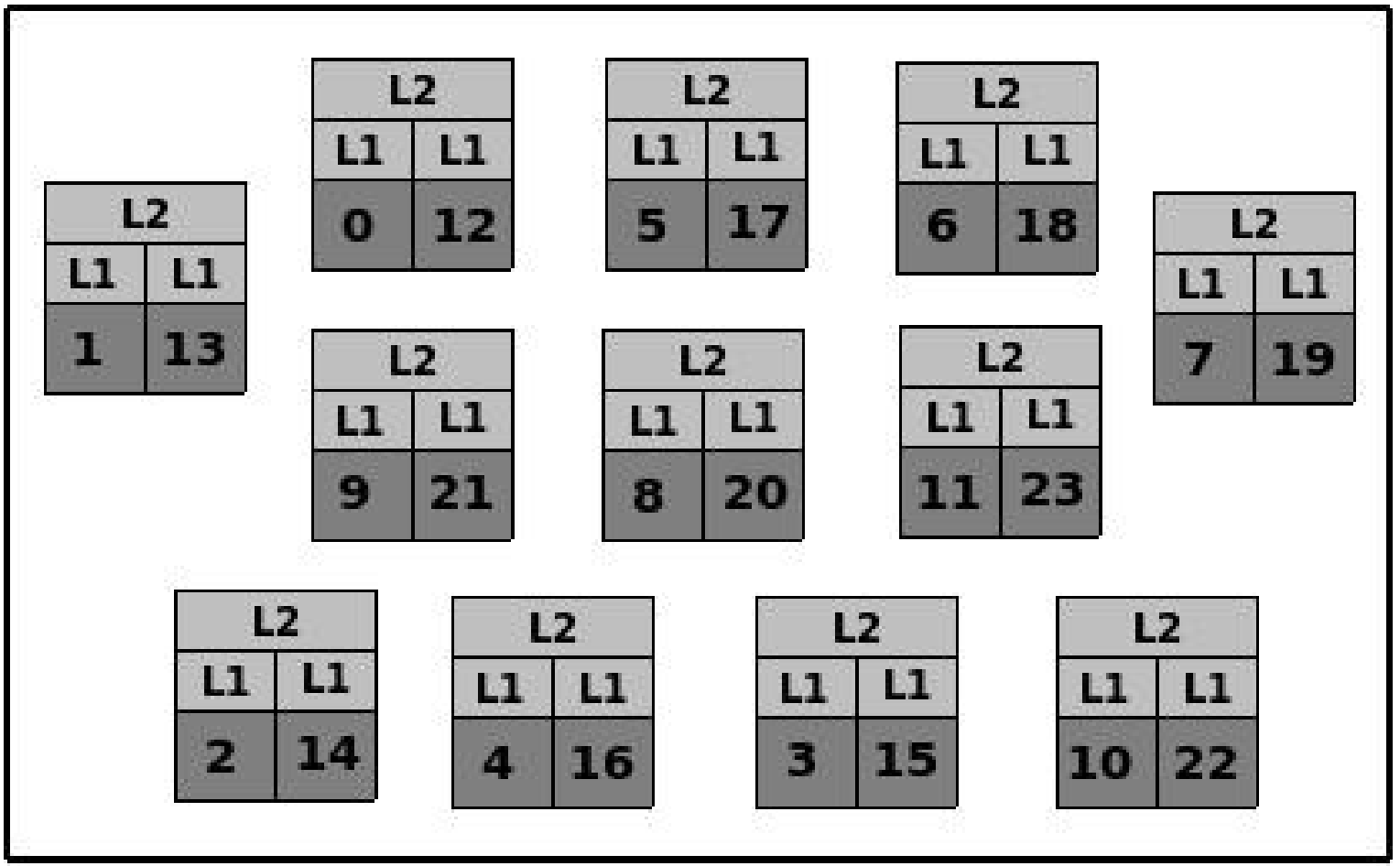


Ejemplo Dunnington: Resultados para L2

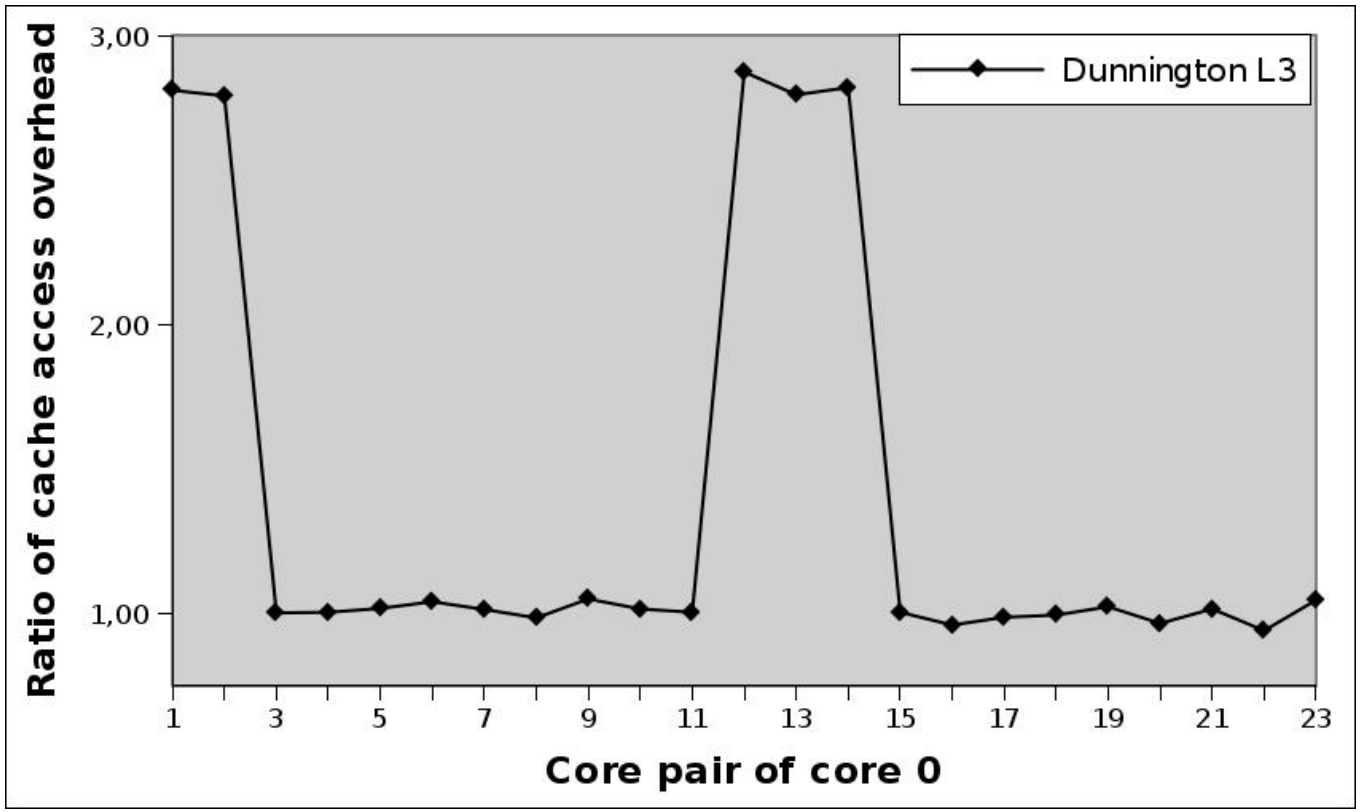


$L2 \rightarrow$ compartida por 0 y 12

Topología con L2

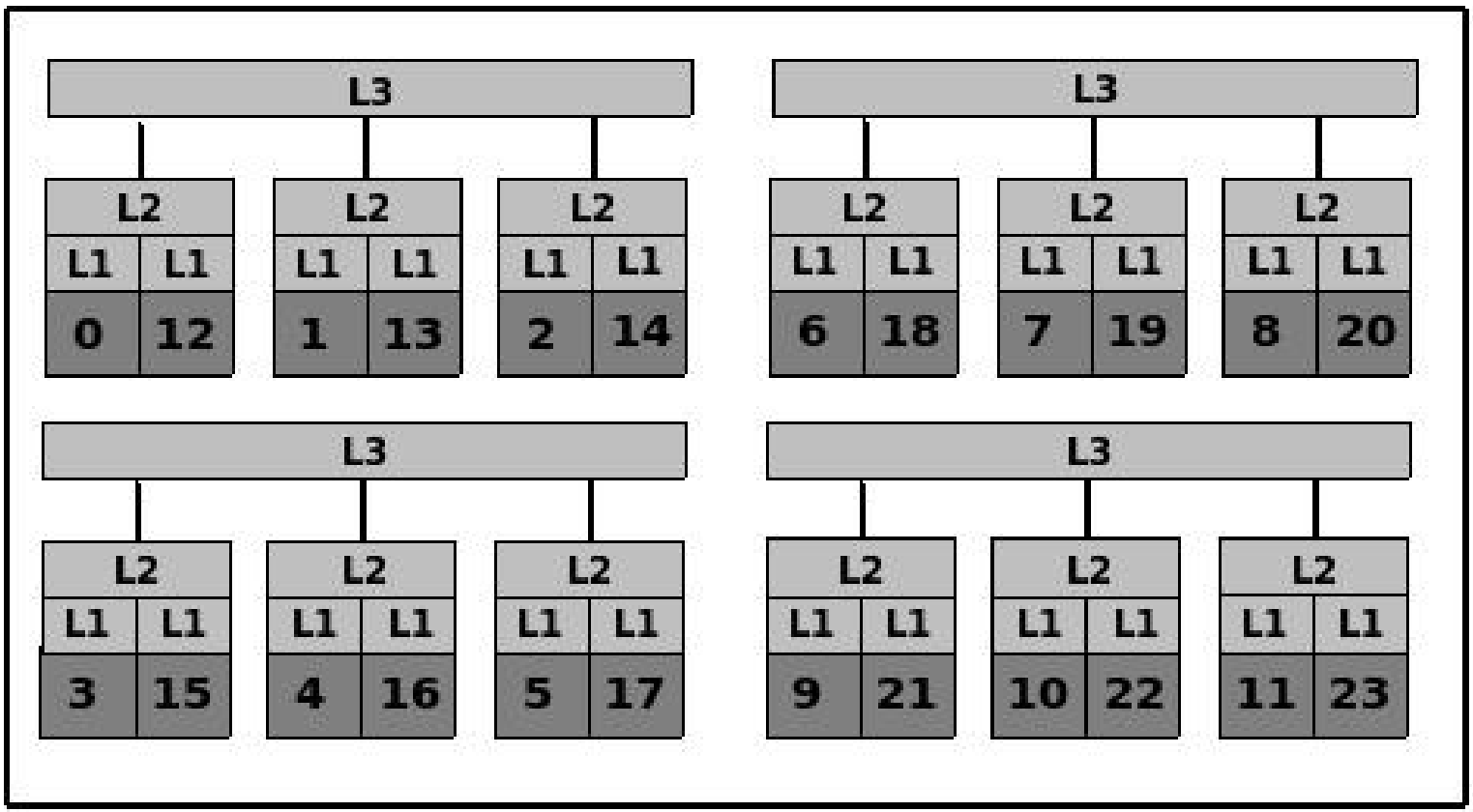


Ejemplo Dunnington: Resultados con L3



$L3 \rightarrow$ compartida por 0,1,2,12,13 y 14

Topología con L3



1 Introducción

2 Topología Caché

3 Caracterización de Sobrecargas de Acceso a Memoria

4 Determinación de los Costes de Comunicaciones

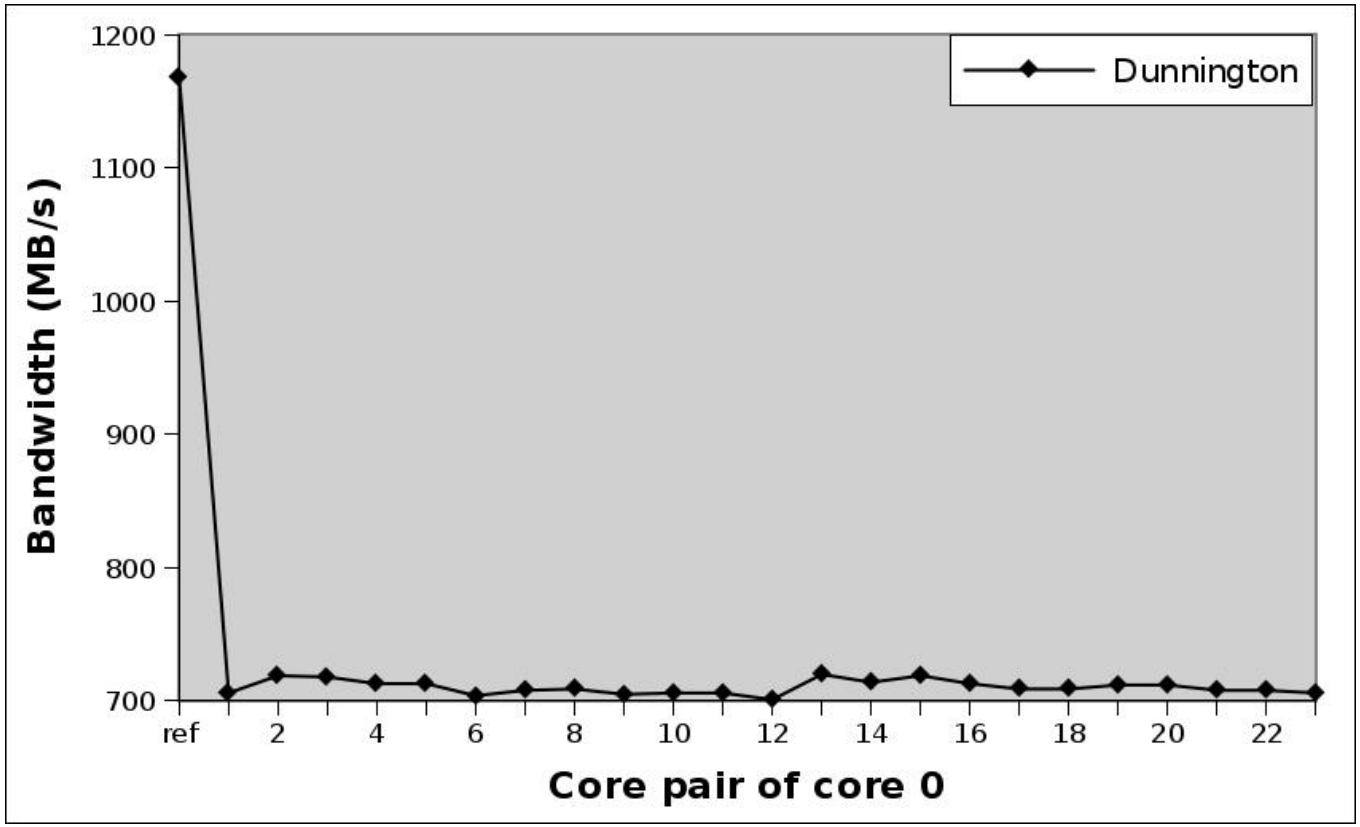
5 Conclusiones

Detección de Diferentes Sobrecargas de Acceso a Memoria

Algoritmo

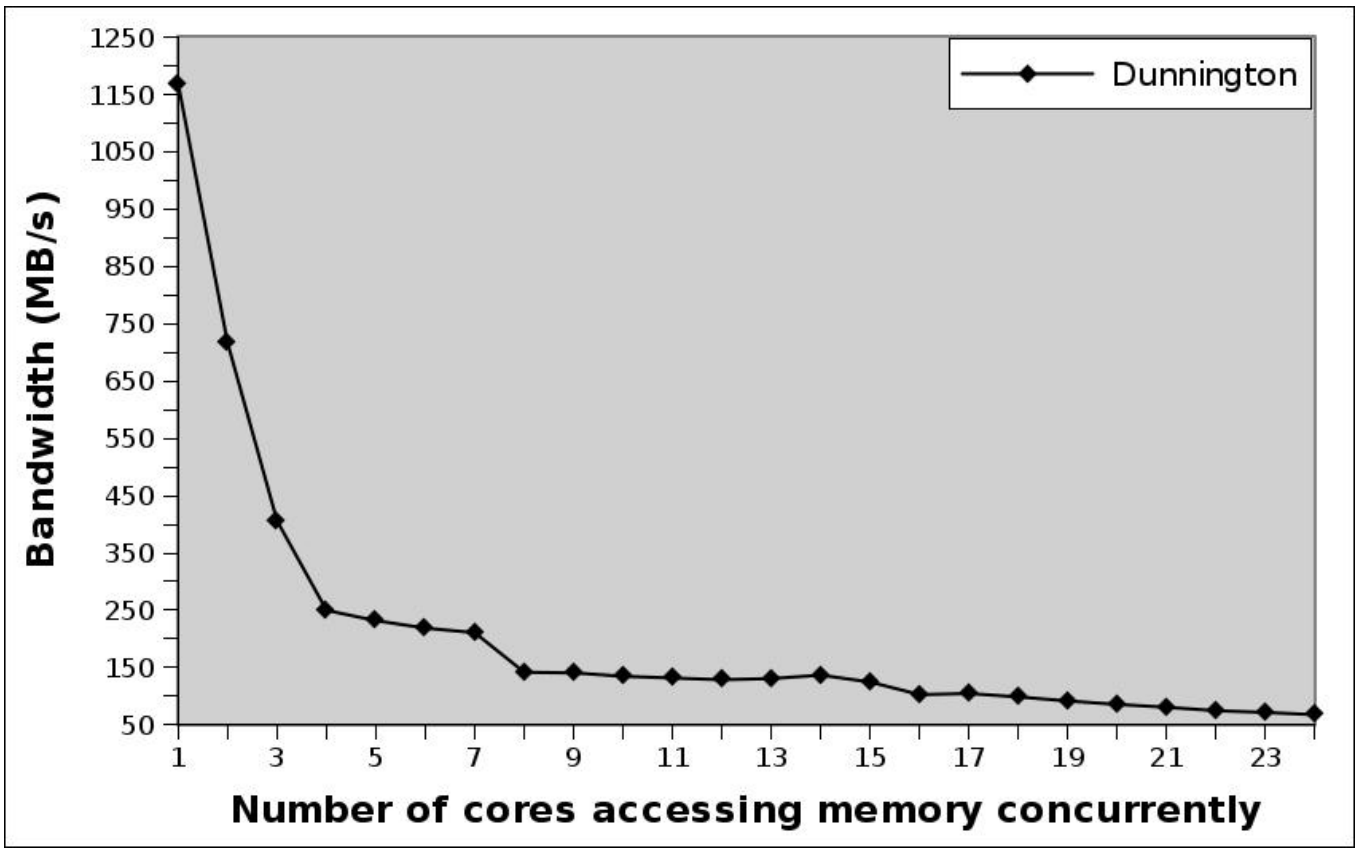
$n = 0$
ref = ancho de banda a memoria accediendo con un núcleo aislado
foreach(par de núcleos)
    b = ancho de banda de un núcleo al acceder ambos
    if($b < ref$)
        if(b similar a un $BW[i]$ obtenido previamente)
            Añadir el par a $P_m[i]$
        else
            $BW[n] = b$
            $P_m[n]$ = El par en estudio
            $n = n + 1$

Resultados Dunnington



60% del ancho de banda al acceder por parejas

Ancho de Banda a Memoria según la Sobrecarga



1 Introducción

2 Topología Caché

3 Caracterización de Sobrecargas de Acceso a Memoria

4 Determinación de los Costes de Comunicaciones

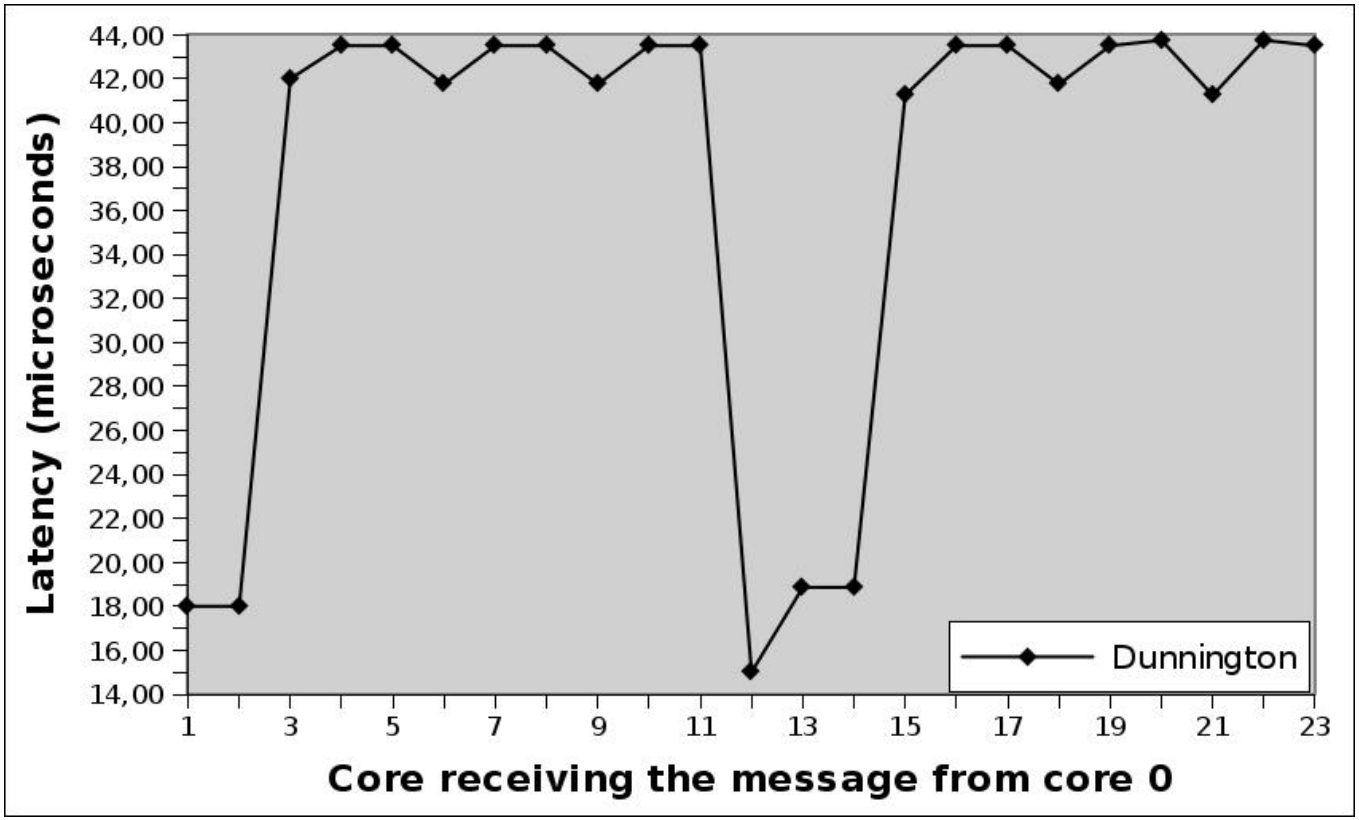
5 Conclusiones

Determinación de las Capas de Comunicaciones

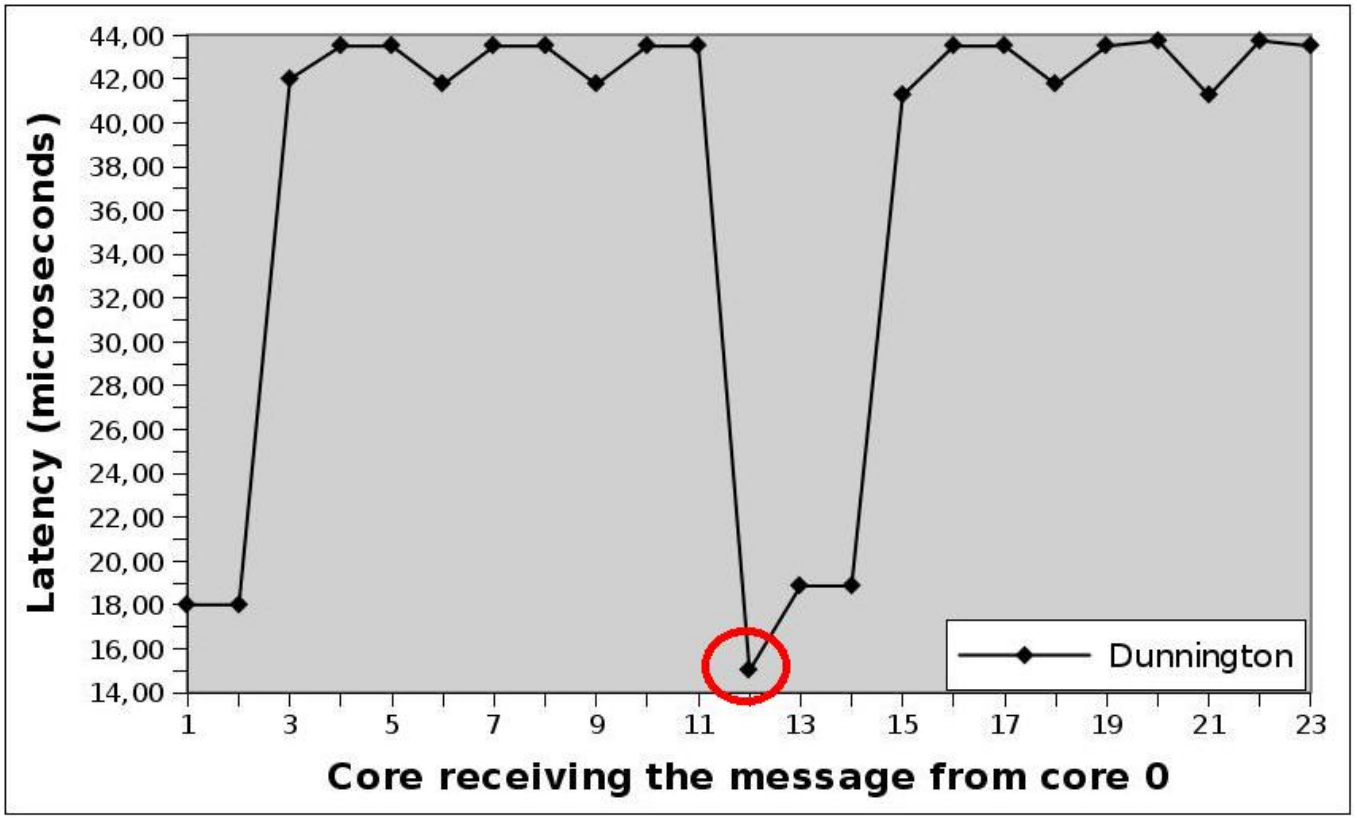
Algoritmo

$n = 0$
foreach(par de núcleos)
    $l =$ latencia de un mensaje de tamaño L1 entre los núcleos
    if(b similar a una $L[i]$ obtenida previamente)
        Añadir el par a $P_l[i]$
    else
        $L[n] = l$
        $P_l[n] =$ El par en estudio
        $n = n + 1$

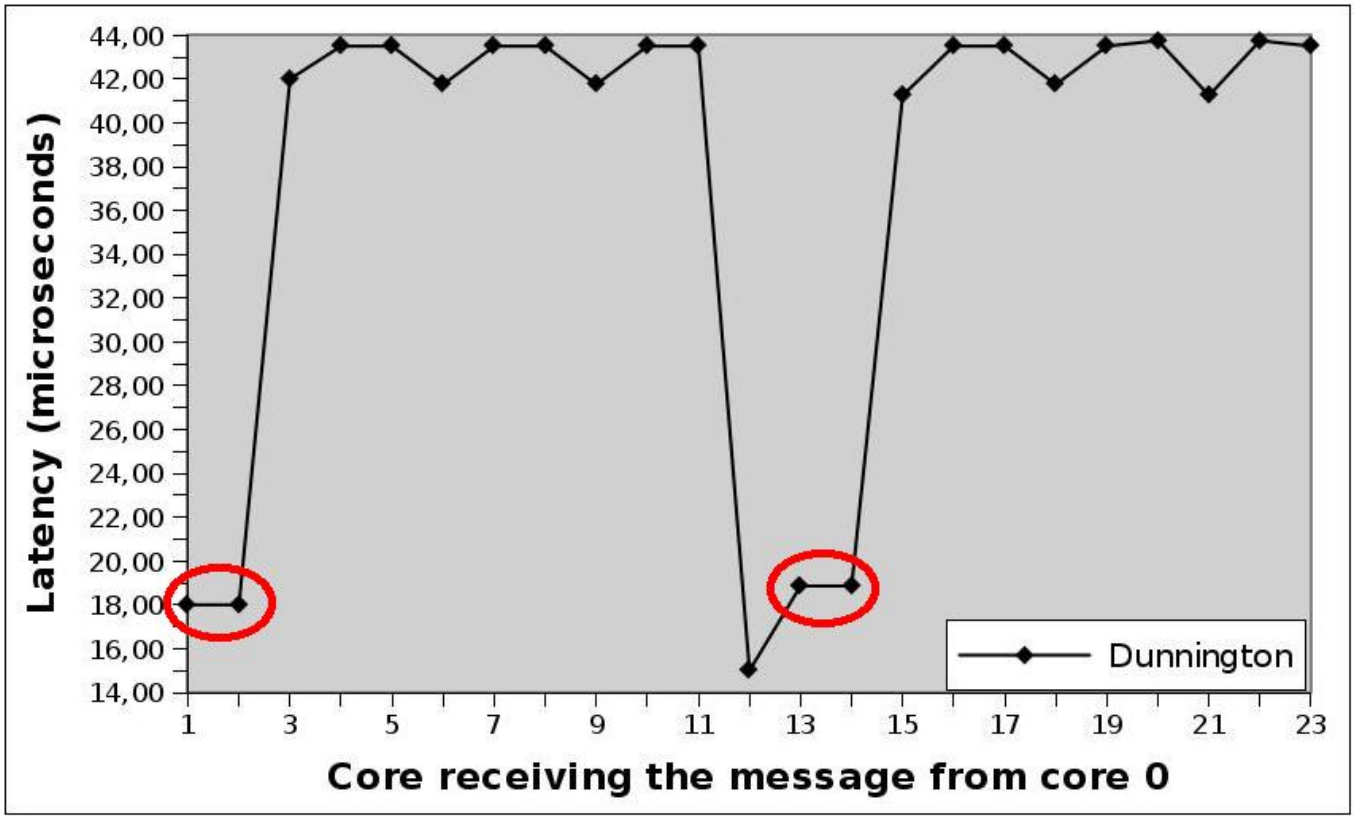
Resultados Dunnington (I)



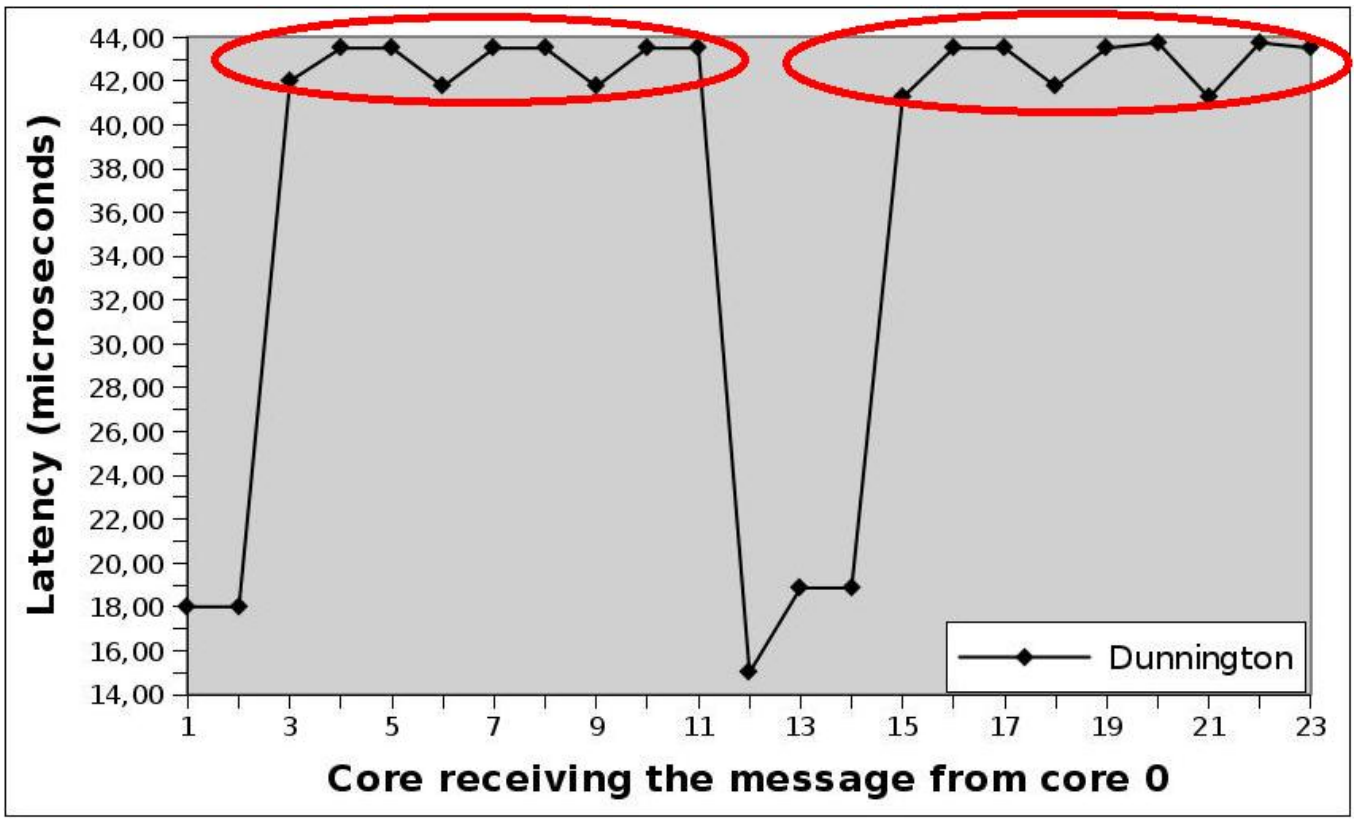
Resultados Dunnington (I)



Resultados Dunnington (I)



Resultados Dunnington (I)

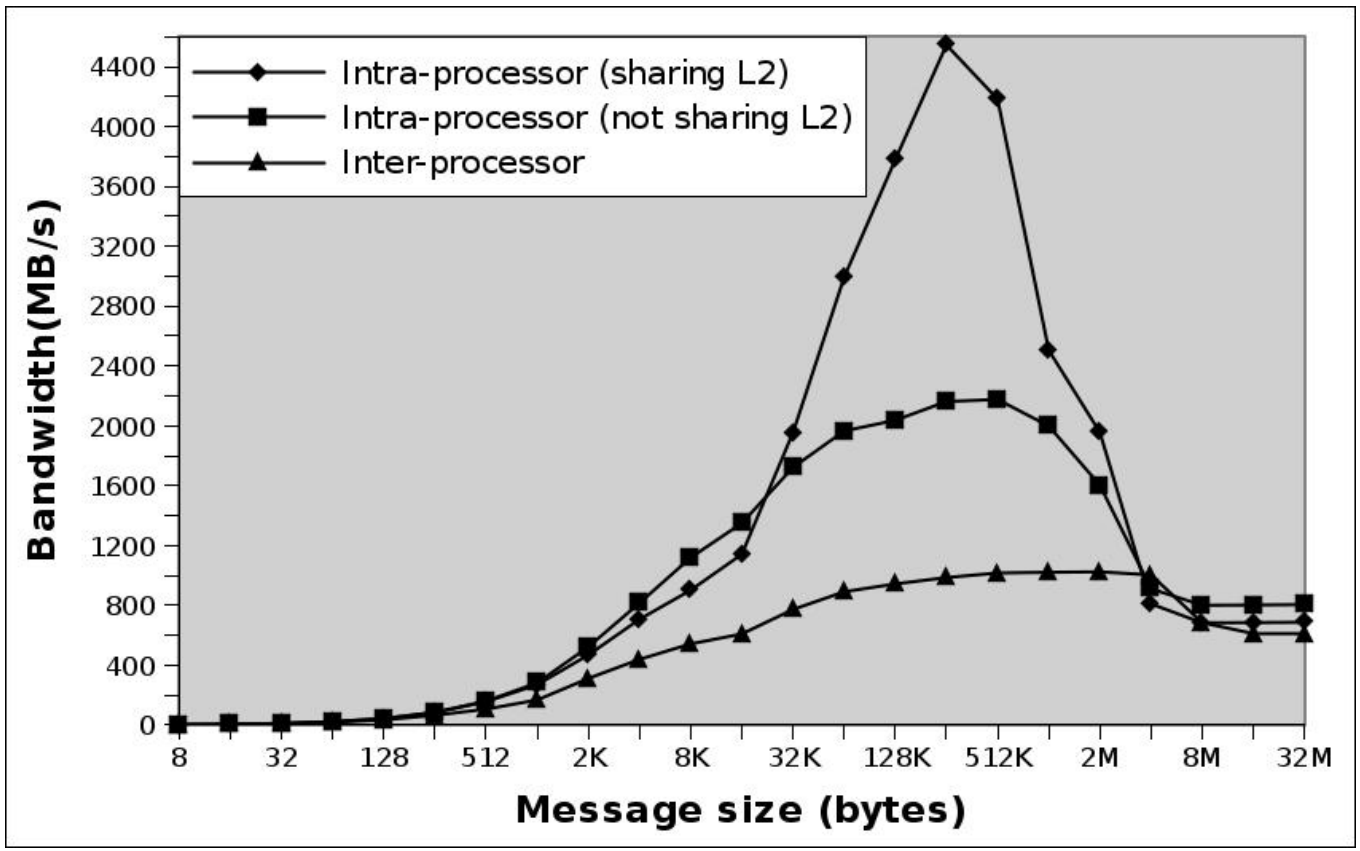


Resultados Dunnington (y II)

Dunnington (mensajes del tamaño de L1)

- Comunicaciones intra-procesador (compartiendo caché L2)
  - 2130 MBytes/s
- Comunicaciones intra-procesador(sin compartir caché L2)
  - 1780 MBytes/s
  - 83% del máximo ancho de banda
- Comunicaciones inter-procesador
  - 750 MBytes/s
  - Sólo el 35% del máximo ancho de banda

Caracterización de las Capas de Comunicaciones



1 Introducción

2 Topología Caché

3 Caracterización de Sobrecargas de Acceso a Memoria

4 Determinación de los Costes de Comunicaciones

5 Conclusiones

Conclusiones

Suite para detectar los parámetros hardware:

- Tamaños de las cachés
- Topología de cachés compartidas
- Sobrecargas de acceso a memoria
- Anchos de banda de comunicaciones

Características

- Portable
- Muy precisa
- Centrada en proporcionar soporte para el autotuning en clusters de sistemas multinúcleo
- Disponible públicamente: `http://servet.des.udc.es`

PRUEBA Y DISFRUTA!!!

`http://servet.des.udc.es`

Jorge González-Domínguez, Guillermo L. Taboada, Basilio B. Fraguela, María J. Martín, Juan Touriño, "Servet: A Benchmark Suite for Autotuning on Multicore Clusters". **24th IEEE International Parallel and Distributed Processing Symposium (IPDPS'10)**. Atlanta, GA, Estados Unidos, Abril 2010.

Contacto: Jorge González-Domínguez
jgonzalezd@udc.es
Grupo de Arquitectura de Computadores,
Dept. de Electrónica y Sistemas,
Universidad de A Coruña