


Arquitectura de Computadores

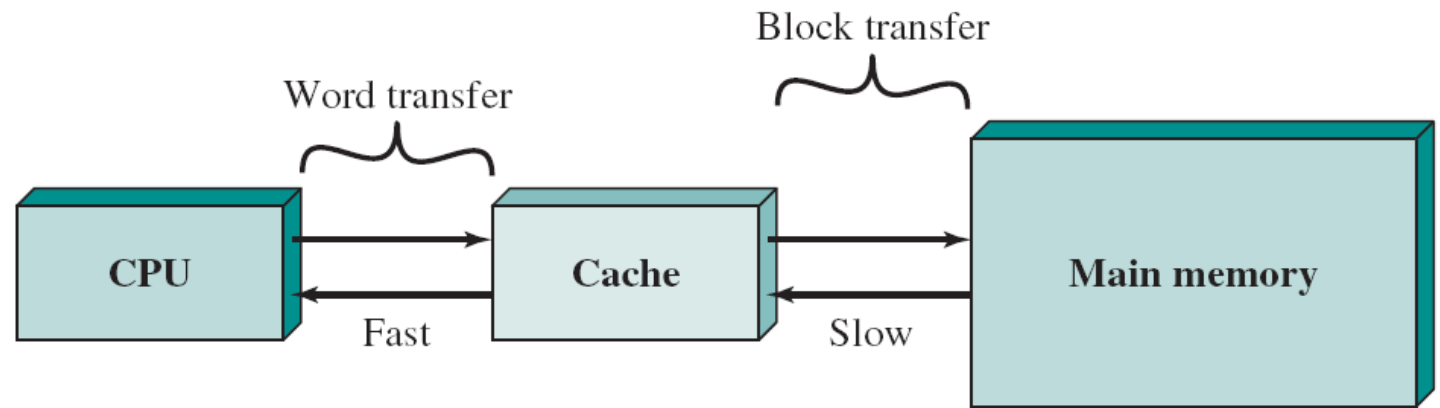
The background of the slide is a collage of digital and technological motifs. On the left, a blue silhouette of a human head is filled with a circuit board pattern, with binary code (0s and 1s) floating around it. The right side features a dark blue background with glowing white circuit traces and bright light flares. In the top right corner, there is a faint, stylized neural network diagram.

2021 A

Andrés Larco

6.3 Elementos de diseño de la cache

MEMORIA CACHE



(a) Single cache

Diseño de la Cache

Direccionamiento

Tamaño

Función de
correspondencia

Algoritmo de
reemplazo

Política de
escritura

Tamaño de
bloque

Numero de
caches

Elementos del diseño de la cache

Tamaño de caché

Función de correspondencia

Directa

Asociativa

Asociativa por conjuntos

Algoritmo de sustitución

Utilizado menos recientemente (LRU)

Primero en entrar-primero en salir (FIFO)

Utilizado menos frecuentemente (LFU)

Aleatorio

Política de escritura

Escritura inmediata

Postescritura

Escritura única

Tamaño de línea

Número de cachés

Uno o dos niveles

Unificada o partida

Direccionamiento de la Cache

- Donde esta ubicada la cache?
 - Entre el procesador y la unidad de administración de la memoria virtual.
 - Entre la unidad de gestión de memoria y la memoria principal.
- Cache lógica (cache virtual) almacena datos utilizando direccionamiento virtual.
 - El procesador accede a la cache directamente, no a través del cache físico
 - Cache accede rápidamente, antes que la dirección de la unidad de gestión de memoria
 - Direccionamiento virtual usa el mismo espacio de dirección para distintas aplicaciones
 - Debe vaciarse la cache luego de cada utilización de cierto contenido
- La cache física almacena datos utilizando las direcciones físicas de la memoria principal

El tamaño no importa

Precio

- Más cache es más caro

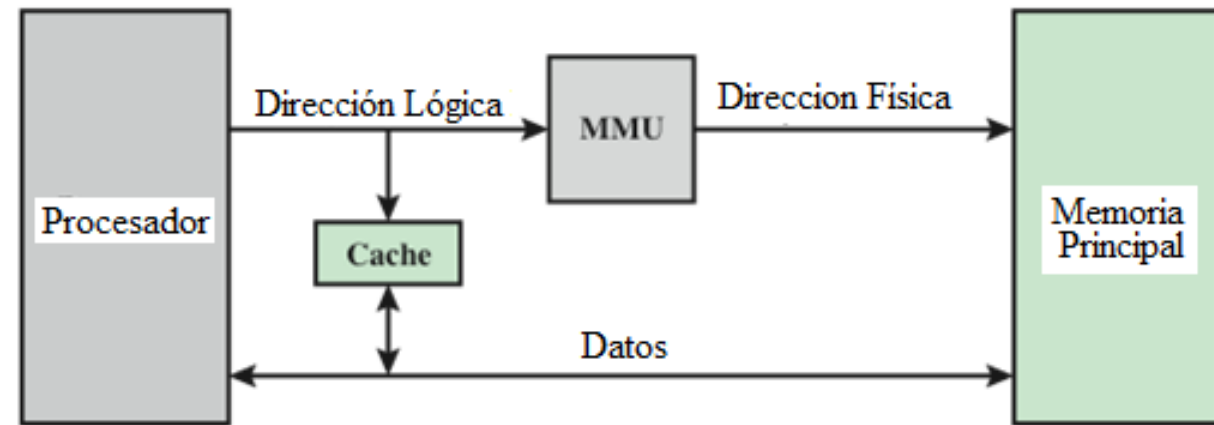
Velocidad

- Más cache es más rápido (por encima de un punto)
- Revisar la cache en busca de datos toma tiempo

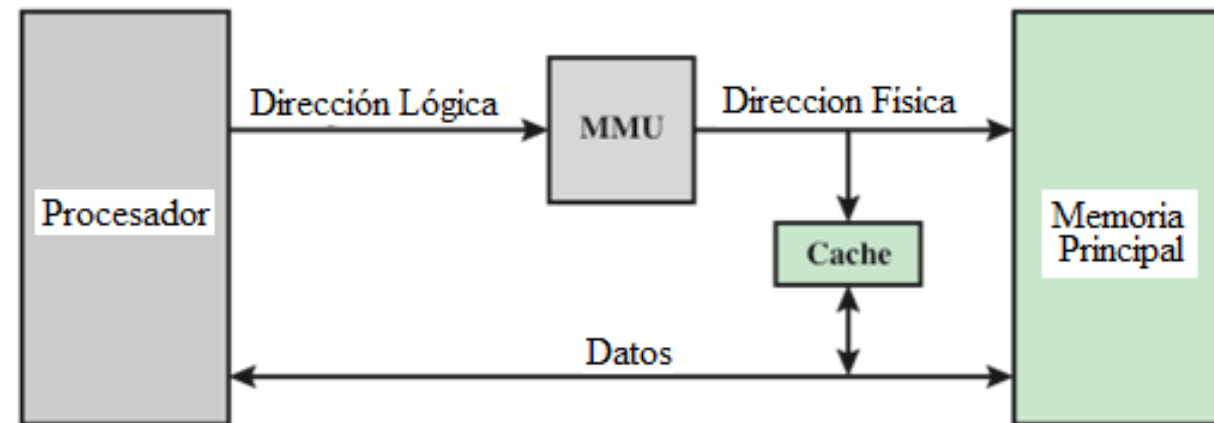
Memoria virtual

- Permite a los programas una dirección de memoria desde un punto de vista lógico, sin tener en cuenta la cantidad de memoria principal física disponible.
- Cuando son utilizados, los campos de dirección de instrucciones de una maquina contienen direcciones virtuales.
- Para la lectura y escritura desde la memoria principal, el hardware de la unidad de gestión de memoria transfiere cada dirección virtual en una memoria física dentro de la memoria principal.

Cache Física y Lógica



(a) Cache Lógica



(b) Cache Física

Función de correspondencia

Se utilizan 3 técnicas:

Directa

- La técnica más simple
- Corresponde cada bloque de memoria principal en solo una línea posible de cache

Asociativa

- Permite que cada bloque de memoria principal sea cargado en cualquier línea de cache.

Asociativa por conjuntos

- Presenta las fortalezas de las dos anteriores, mientras que reduce sus desventajas.



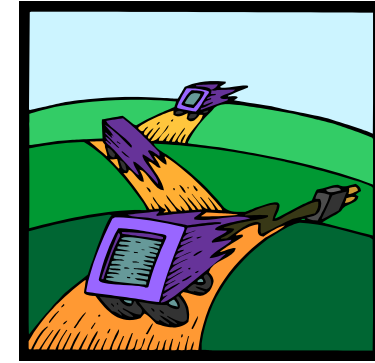
Victim Cache



- Originalmente propuesta con el propósito de reducir las zonas de conflicto de la correspondencia directa, sin afectar el tiempo de acceso.
- Cache totalmente asociativa
- El tamaño típico es de 4 a 16 líneas de cache.
- Ubicada entre la correspondencia directa de la cache L1 y en el siguiente nivel de memoria.



Algoritmos de sustitución



- Una vez que la cache se llena, y un nuevo bloque es traído dentro de la cache, un bloque existente debe ser remplazado.
- Para la correspondencia directa hay una solo línea posible para un bloque particular y no hay ninguna otra opción posible.
- Para la correspondencia asociativa y por conjuntos se requiere un algoritmo de remplazo.
- Para lograr una alta velocidad, un algoritmo debe ser implementado en el hardware.



Los 4 algoritmos de remplazo más comunes son:



Menos utilizado recientemente - Least recently used (LRU)

- Mas efectivo
- Remplaza aquel bloque del conjunto que haya estado por mas tiempo sin haber sido referenciado.
- Debido a su simpleza en la implementación, LRU es el algoritmo de remplazo mas popular.

Primero que entra, primero que sale - First-in-first-out (FIFO)

- Remplaza aquel bloque del conjunto que ha estado en la cache por más tiempo.
- Es fácilmente implementado con una técnica cíclica.

Menos utilizado recientemente - Least frequently used (LFU)

- Remplaza aquel bloque en el que un conjunto ha experimentado menos referencias.
- Puede ser implementado asociando un contador con cada línea

Política de escritura

Cuando un bloque que se encuentra en la cache va a ser remplazado, hay 2 casos para considerar:



Si el bloque en la cache no ha sido alterado entonces puede ser sobrescrito con un numero bloque sin escribir fuera el bloque de la cache.



Si al menos una operación de escritura a sido realizada en una palabra en una línea de la cache entonces la memoria principal debe ser actualizada antes de traer un nuevo bloque.

Hay dos problemas con los que lidiar:



Uno o mas dispositivos pueden tener acceso a la memoria principal.



Un problema mas complejo ocurre cuando varios procesadores son añadidos al mismo bus y cada proceso tiene su propia cache, si una palabra es alterada en una cache puede que se invalide en otras caches.



Escritura inmediata

- Técnica mas simple
- Todas las operaciones de escritura son hechas en la memoria principal así como la cache
- La principal desventaja de esta técnica es que genera un tráfico considerable en la memoria y crea un cuello de botella



Post escritura

- Minimiza las escrituras en la memoria
- Las actualizaciones son hechas solo en la cache
- Partes de la memoria principal son inválidos y por lo tanto los accesos de los módulos de I/O puede ser permitidos solo a través de la cache
- Se requiere una circuitería compleja y podría generarse un cuello de botella.

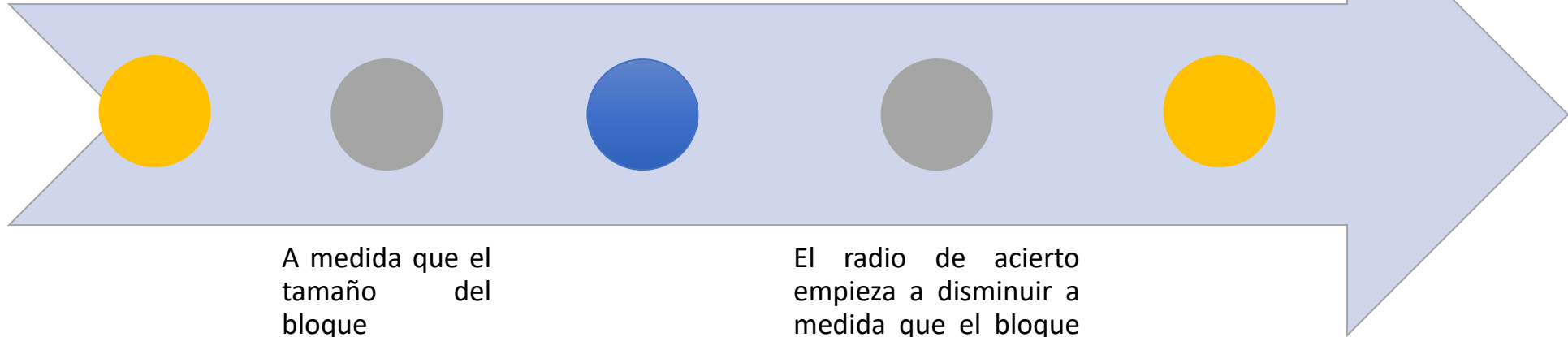
Tamaño de línea

Cuando un bloque de datos es recuperado y puesto en la cache no solo se obtiene la palabra que se deseada si no también algunas palabras que estén cerca también.

A medida que el tamaño del bloque incrementa mas datos útiles son traídos dentro de la cache.

Se puede llegar a dos conclusiones:

- Bloques mas grandes reducen el numero de bloques que se necesitan en la cache
- A medida que el bloque sea mas grande cada palabra adicional esta mas lejos de la palabra requerida

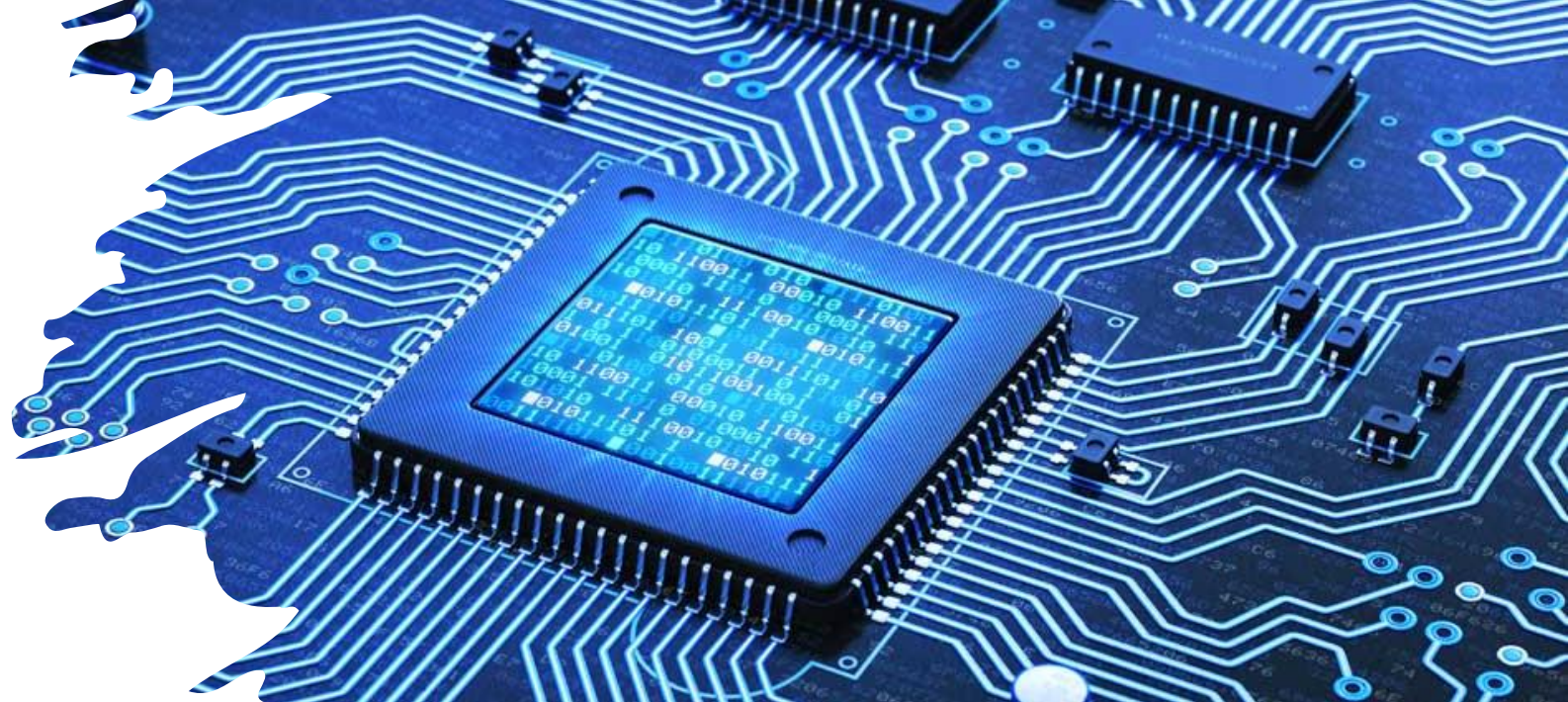


A medida que el tamaño del bloque incrementa el radio de acierto se incrementa debido al principio de localidad

El radio de acierto empieza a disminuir a medida que el bloque se vuelva mas grande y la probabilidad de utilizar nueva información se vuelve menos probable que utilizar información que ha sido remplazada.

Referencias:

- William Stallings, Computer Organization and Architecture: Designing for Performance, 9th Edition, Prentice Hall, 2010, ISBN-13: 978-0-13-607373-4



Gracias !!!

