Universidad de Carabobo

Facultad de Ciencias y Tecnología

Proyecto de arquitectura del computador

Profesor: Jose Canache

Autor:
Juan Zamora CI:20313463

Enero 20-2021

Marco Teorico

Implementación de c++ en el cual con el desarrollo se obtiene la caracteristica de los datos de la memoria el tamaño total y derivados se pasa una lista con las direcciones y numeros de cache con gran asociatividad (generalmente ¿4 vías), el coste de implementación de LRU resulta prohibitivo. En muchos cachés de CPUs, una política que casi siempre desplace uno de los registros menos usados recientemente resulta suficiente.

Política de Reemplazo LFU (Least Frequently Used) La función de reemplazo LFU, utiliza el mismo proceso que LRU, con unos Ajuste: Se utiliza un tipo de dato definidos como par, donde matendra las variables de diferentes tipos este proceso, se define un par donde se almacena una dirección y el número de veces en la que este fue replicado. Se definen las listas a usar y las mismas variables que el procedimiento anterior para este: Una lista que es el emulador de la memoria cache, otra lista auxiliar de pares para saber el orden que fueron accedidos los datos de la cache; el número de bits de índice y de etiqueta y unas variables booleanas que actúan como banderas asi poder utilizarlo como false . se abre el ciclo que donde un iterador pasa por todos los elementos dentro de la lista de direcciones. Para cada iteración se define el valor binario.

Pasos del codigo

ListaCache inicializada

Inicio de la lectura de Trazas.in Mientras (fin de archivo de Trazas.in) hacer DatoAux $\longleftarrow$ $Leer(Trazas.in)$

Se busca si el DatoAux leido esta dentro del cache. bandera $\longleftarrow$ $ListaCache.esta(DatoAux)$

Si ocurre un fallo (el dato no esta en la lista. Si (bandera = falso) entonces

Si la lista no esta llena Si (ListaCache.size $\longleftarrow$ $maxSize)ListaCache.insertarFin(DatoAux)$

maxSize++

Sino Metodo de reemplazo Fsi

Imprimir el resultado del cambio

Sino Se busca el dato que obtuvo el Hit DatoAux $\longleftarrow$ $ListaCache.buscar(DatoAux)$

Quitarlo de la lista y agregarlo de nuevo al Final ListaCache.insertarFin(DatoAux)

Imprimir el resultado del cambio

Fsi

Fmientras

• Si la memoria cache se encuentra de modo que no está en su capacidad, se aplica asi el par generado dentro de la lista auxiliar y la dirección actual dentro de la lista cache de la cual habiamos precedido antes.

• Si la memoria cache está en su capacidad nos encontramos con el otro proceso, En el cual el reemplazo de datos. Se define como otro dato tipo par donde, mediante el llamado de un procedimiento, obtiene el par que conforma el valor menos llamado dentro de la cache mediante comparaciones de los pares de la lista auxiliar.

Analisis

Criterio LRU (Least Recently Used) Es justamente el criterio contrario al MRU. En LRU elegimos la data que más tiempo lleve sin ser accedida. Se implementa mediante una FIFO MEMORIA FIFO:(First in-firts out), primero en entrar - primero en salir, es decir, es lo que se llama una fila de espera. que mantiene el orden de acceso a los. datos Es decir, cada vez que se accede a una página, en caso de estar ya en la FIFO, se retira de la cola y se añade al final.

LRU, se utiliza dos listas e interadores para cada una; donde una de las listas será nuestro emulador de la memoria cache y la otra es utilizada como auxiliar para conocer el orden de uso de un elemento de la cache, donde el primero de la lista es el menos usado mientras que el ultimo es el más reciente.

Se tiene un bit R y un contador por cada página. Cuando una página es cargada a un marco, se carga con su bit R a 1 y su contador inicial a 0. Cuando pasa un período determinado de tiempo (viene dado en el enunciado) se itera sobre toda la lista de páginas, y pueden ocurrir dos cosas:

Si su bit R está a 1: Se ponen su bit R a cero y su contador se incrementa. Si su bit R está a 0: No se hace nada. Cuando hay que seleccionar una página víctima, se escoge aquella cuyo contador sea más pequeño. En caso de empate, habrá que establecer un criterio de desempate.

Datos de LRU

Aunque LRU es realizable teóricamente, su implantación presenta problemas. Sería necesario mantener una lista enlazada de todas las espacios en la memoria, en donde la dato de uso más reciente esté al principio de la lista y la de uso menos reciente al final. La dificultad estriba en que la lista debe actualizarse en cada referencia a la memoria. La búsqueda de una data en la lista, su eliminación y posterior traslado al frente de la misma puede ser una operación que consuma mucho tiempo. Es preciso un hardware especial (caro), o bien determinar una solución aproximada mediante software.

La búsqueda y manipulación de una lista enlazada en cada instrucción es prohibitiva por su lentitud, aún en hardware. Sin embargo, existen otras formas de implantar LRU con un hardware especial. Consideremos primero el caso más sencillo. Este método requiere un contador especial de 64 bits, C, en hardware, que se incrementa de forma automática después de cada instrucción. Además, cada entrada de la tabla de almacenamiento debe tener un campo de tamaño adecuado como para contener al contador. Después de cada referencia a la memoria, el valor actual de C se almacena en la entrada de la tabla de espacio en la lista correspondiente a la espacio a la que se hizo referencia. Al ocurrir un fallo de página, el sistema operativo examina todos los contadores de la tabla de datos y elige el mínimo. Esa data es la utilizada menos recientemente.

Analicemos ahora un segundo algoritmo LRU en hardware. En una máquina con n marcos, el hardware LRU puede utilizar una matriz de nxn bits, cuyos datos iniciales son todos cero. En una referencia al marco k, el hardware primero activa todos los bits del renglón k y desactiva después todos los bits de la columna k. En cualquier instante, la fila cuyo valor en binario es mínimo es el marco utilizado menos recientemente, la fila con el siguiente valor más pequeño es el segundo marco utilizado menos recientemente, etc.

Conclusiòn

Proyecto en el cual se trabajo las habilidades del c++ se uso librerias binarias como de listas y se profundizo el tema de arquitectura del computador tambien se dio a entender el tema de LFU su forma de procesar la memoria en que respalda la cache su capacidad su remplazo su llamado y comparacion y compresion en la logica tambien se utilizo latex para el informe y el proyecto sera respaldado en Git Hub para mejorar herramientas