Reconsidering Swapping

File-backed pages corresponde al segmento de archivo sobre un disco. Si no contiene archivos que hayan sido escritos recientemente y no hayan regresado al almacenamiento persistente, estos se recuperarían de manera sencilla.

Anonymous pages, estos mantienen los datos generados en run-time y usados por un proceso, pedir una página anónima, requiere reescribir el contenido Swap

Como regla general, pedir anonymous pages es mas caro que pedir file-backed pages

File-backed pages puede realizar operaciones de lectura y escritura desde el almacenamiento persistente a corto o largo plazo, y son almacenadas secuencialmente

Costo de Rotaciones: en cada LRU (Last-recent used) el MM (Memory manager hace lo posible por no reclamar páginas que estén en uso activo, esto lo hace pasando ocasionalmente por la lista y limpiando los bits de referencia de cada página, así las páginas que vuelvan a ser usadas se les volverá a asignar un bit y las que no se les asignen estarán en el fin de la lista. Por lo tanto estas paginas son las primeras en ser reclamadas las paginas que fueron referenciadas de nuevo se añadirán a la cabeza de la lista y pasará un tiempo antes de que se reclamen de nuevo

Anonymous pages tienden a dispersarse a lo largo del área swap, por lo tanto para un disco que necesite rotar para leer datos, buscar anonymous pages resultaría demasiado cara ya que estas están dispersas aleatoriamente

¿Cuando usar swap?

Con esto se denota el swapiness que va de 0 a 100 donde 0 no reclama swap en absoluto, si realmente no es necesario y 100 que reclama de forma igualitaria file-backed pages y anonymous pages. Actualmente Johaness incrementa a 200 el swapiness debido a las altas velocidades que existen actualmente

LLEGADA DE SSDs. con esto el swap vuelve a ser interesante debido a las velocidades de acceso aleatorio alcanzadas por estos dispositivos. Aquí el swap vuelve a ser interesante para manejar el overflow y además como una extensión de memoria para optimizar el balance entre las páginas cache y anonymous hasta una carga moderada.

Johannes cambió el sistema de desición para saber si se elegiría entre una anonymous page o file-backed pages con lo cual se introduce el concepto de costo por reclamar una página. la cual se elige de acuerdo al que tenga el menor costo

Making swapping Sociable

Swapping es cada vez más atractivo, así como los dispositivos de almacenamiento, en especial los SSDs. Anteriormente mover una página, o remover una página del dispositivo de almacenamiento era una tarea sumamente lenta, pero ahora la memoria persistente ha alcanzado niveles de velocidad cercanos a los de acceso a memoria directo en particular para el cómputo en la nube el overcommiting memory sería insignificante, permitiendo un mejor manejo del sistema entero

En 2013 el código de swap fue renovado para un mejor desempeño en SSDs ya que existían dos problemas

No hay demora de búsqueda en SSDs

En Kernels Actuales el dispositivo swap es representado por "swap_info_struct" y dentro de este se encuentra "swap_map" donde se apunta al byte del arreglo donde contiene el contador de referencia para cada página almacenada en swap

wear-leveling mejora esparciendo los datos al rededor del disco, a diferencia de los discos rotacionales que debían mantener cerca de él para una lectura rapida

En el mismo actualización de 2013 cuando el sistema detectaba que estaba trabajando con un SSDs, este se dividía en clusters.

Un apuntador "per-cpu-cluster" apunta a diferentes clusters de cada CPU. con esto, cada CPU puede alojar páginas de swap en su cluster, con el resultado de que estas alojaciones son esparcidas al rededor del dispositivo. (Aunque en la actualidad no se ha alcanzado el potencial esperado).

El problema es que algunas veces se tiene que lidiar con el bloqueo ya que los CPU no tiene acceso exclusivo a cualquier cluster. así que deben adquirir el "lock" de spinlock en el "swap_info_struct" antes de cualquier cambio hecho. En sistemas típicos solo hay un dispositivo swap, por lo que si hay demasiado swapping el spinlock es muy llamado. Y la contención del spinlock no es el camino a la escalabilidad

La contención no es necesaria cada cluster es independiente y puede ser adquirido sin tocar a los demás así que no es necesario un bloque global. Para esto se agrega un nuevo bloqueo a una entrada al array de "cluster_info" un solo bit de bloqueo es usado para minimizar el consumo de memoria agregada. Así, cualquier cluster puede asignar páginas (o páginas libres) del cluster sin pelear con las demás

Aún así hay sobrecarga adquiriendo el bloqueo y puede haber contención en cache-line cuando se esta accediendo al bloqueo en cualquier otro CPU cluster. Para minimizar el costo se agrego una nueva interfaz para asignar o liberar paginas en conjuntos. cuando un CPU tiene un conjunto de paginas swap puede usarlas sin tomar el bloqueo local del cluster. Paginas liberadas son acumuladas en un cache separado y son regresado en lotes.

Existe aún otro problema con el radix-tree el cual nos lleva a otro global-clock el cual hace que vuelvan a pelear por el CPU. Para solucionarlo el "address_space" es dividido cada 64MB de swap asignando un reloj a cada porción. Esto reduce la contención por un reloj individual