Tarea 1

Hashing Lineal

Profesores: Benjamín Bustos y Gonzalo Navarro
Auxiliares: Máximo Flores y Sergio Rojas

Fecha de entrega: jueves 3 de octubre a las 23:59

1. Contexto

El hashing lineal tiene una estructura que requiere sólo $O(1)$ datos en memoria principal, por lo que puede manejar conjuntos arbitrariamente grandes de datos. También permite controlar el promedio de búsqueda para intentar disminuirlo. En esta tarea utilizaremos hashing lineal para ver su efectividad cuando variamos sus hiperparámetros.

2. Inserción del hashing lineal

El archivo contiene p páginas, con $2^t \leq p < 2^{t+1}$, con 2^{t+1} la cantidad de páginas disponibles en ese instante. Inicialmente se tiene $p = 1$ y $t = 0$. Las páginas $0 \leq i < p - 2^t$ ya fueron expandidas y repartidas entre las páginas i e $i+2^t$, mientras que las páginas $p-2^t \leq i < 2^t$ aún no lo han sido.

Para insertar un elemento y, se calcula $k \leftarrow h(y) \mod 2^{t+1}$. Si $k < p$, se inserta en la página k (o en una nueva página si la actual se rebalsa). Si $k \geq p$, la página k aún no ha sido creada, por lo cual se debe insertar en la página $k - 2^t$.

Cuando se cumple una condición (que para efectos de esta tarea será superar el número máximo de accesos promedio permitido al realizar búsquedas) se realiza una expansión de la *siguiente página*, la cual siempre será la página $p - 2^t$. Se recorre esta página (junto a las de desborde) y los elementos y se insertan en la página $h(y) \mod 2^{t+1}$, es decir, se reparten los elementos en la página $p - 2^t$ y la p que se agrega al final del archivo. Después de este proceso se debe compactar los elementos que quedaron en la página $p - 2^t$ y eliminar las páginas de rebalse que no sean necesarias.

Al finalizar la expansión, se hace $p \leftarrow p + 1$ y si resulta que $p = 2^{t+1}$, se realiza $t \leftarrow t + 1$.

Ilustraciones de esto se pueden ver en el apunte del curso, página 48.

3. Descripción de la tarea

Para esta tarea el trabajo será analizar, implementar y evaluar experimentalmente el costo real de inserción del hashing lineal realizando variaciones en el máximo costo promedio permitido para

esta operación. Nótese que este último parámetro es controlado en el experimento, mientras el parámetro empírico sólo se puede obtener a partir de una ejecución de sus programas. Deberán realizar comparaciones entre sus resultados controlados y prácticos, y analizar según el caso que se presente durante su experimentación.

Se simulará el hashing lineal en memoria principal, y consideraremos como una I/O cada vez que accedemos, escribimos o borramos una página. Para insertar un elemento y en la tabla de hash, la metodología **obligatoria** es recorrer toda la lista para verificar previamente si está o no. En el caso de que no esté, se inserta siguiendo la metodología descrita. Se deberá entregar un informe que describa el proceso y los resultados del análisis.

4. Estructuras a utilizar

- La función de hashing $h(y)$ debe devolver un valor aleatorio entre 0 y $2^{64} - 1$ para cualquier elemento y. Limítese a asumir que esta función cumple con todas las propiedades que necesitamos. No necesita preocuparse de la consistencia (es decir, cuál es el output de $h(y)$ para distintas aplicaciones de h), puesto que en esta tarea sólo nos importan las inserciones.
- Los elementos a almacenar serán del tipo **long long** de C/C++, es decir, enteros de 64 bits (si utilizan otro lenguaje permitido deben asegurarse de que cumplan con ese tamaño).
- Una página debe ser una lista de elementos que tiene como espacio máximo 1024 bytes.
- La tabla de hashing será una lista indexada por el valor entregado por la función de hash. Sus elementos serán listas de páginas, simulando de esta forma las listas de rebalse.
- Definan las variables globales y otras estructuras que necesiten.

5. Objetivos

Para esta tarea se deberá:

1. Implementar el funcionamiento en construcción del hashing lineal.
2. Experimentar variando como hiperparámetro el máximo costo promedio permitido al hacer inserciones para analizar y comparar con el costo promedio real al hacer esta operación.
3. Analizar la relación entre el porcentaje de llenado de las páginas y el costo promedio real de realizar las inserciones.

6. Experimentación

Se debe generar una secuencia N de números de 64 bits, con $|N| \in \{2^{10}, 2^{11}, 2^{12}, \ldots, 2^{24}\}$ e insertar los elementos en una estructura basada en el hashing lineal explicado en secciones anteriores, generando variaciones en el costo promedio máximo $c_{\text{máx}}$ permitido al realizar inserciones. Se debe analizar cómo varía el costo promedio real de inserción cuando cambia la variable controlada $c_{\text{máx}}$.

Se esperará que analicen con sus propias palabras por qué se pueden producir diferencias, en el caso de haberlas, entre los costos reales y controlados. Además, deben realizar un gráfico que demuestre de manera clara la relación entre estas dos métricas.

Por otro lado, también deben graficar la relación entre el porcentaje de llenado de las páginas y el costo promedio real de realizar las inserciones. Este gráfico debe ir acompañado de un análisis donde se exponga claramente por qué creen que se cumplen las relaciones satisfechas.

Esta experimentación (con respecto a las métricas estadísticas, suposiciones, etc.) se puede realizar de forma libre siempre y cuando siga la metodología obligatoria, pero procure entregar información sobre todo el proceso que realizó en la etapa de construcción e inserciones. Recuerde que en modelos de memoria secundaria, medimos la cantidad de I/Os para mostrar eficacia de los algoritmos.

¡IMPORTANTE! Para efectos de esta tarea, recuerde asumir que un bloque en disco (página) es de 1024 bytes, de lo contrario, sus resultados serán inconsistentes con la experimentación deseada.

7. Entregables

Se deberá entregar el código y un informe donde se explique el experimento en estudio. Con esto se obtendrá una nota de código ($NCod$) y nota de informe ($NInf$). La nota de la tarea será:

$$NT_1 = 0.5 \cdot NCod + 0.5 \cdot NInf$$

7.1. Código

La entrega de código debe ser hecha en C, C++ o Java. Tiene que contener:

- (**0.5 pts**) README: Archivo con las instrucciones para ejecutar el código, debe ser lo suficientemente explicativo para que cualquier persona solo leyendo el README pueda ejecutar la totalidad de su código (incluyendo las librerías no entregadas por nosotros que potencialmente se deban instalar).
- (**0.2 pts**) Experimento: Creación de la lista N variando lo señalado en este informe.
- (**0.3 pts**) Estructuras: Las estructuras permiten una buena realización del trabajo.
- (**1.5 pts**) Implementación de la inserción en hashing lineal.
- (**3 pts**) Obtención de resultados: La forma en el que se obtienen los resultados es correcta y es suficiente para poder obtener conclusiones no triviales.
- (**0.5 pts**) Main: Un archivo o parte del código (función main) que permita ejecutar la construcción y experimentos.

7.2. Informe

El informe debe ser claro y conciso. Se recomienda hacerlo en LaTeX o Typst. Debe contener:

- (**0.8 pts**) Introducción: Presentación del tema en estudio, resumir lo que se dirá en el informe y presentar una hipótesis.

- (**0.8 pts**) Desarrollo: Presentación de algoritmos, estructuras de datos y cómo funcionan y por qué. Recordar que los métodos ya son conocidos por el equipo docente, lo que importa son sus propias implementaciones.
- (**2.4 pts**) Resultados: Especificación de los datos que se utilizaron para los experimentos, la cantidad de veces que se realizaron los tests, con qué inputs, que tamaño, etc. Se debe mencionar en que sistema operativo y los tamaños de sus cachés y RAM con los que se ejecutaron los experimentos. Se deben mostrar gráficos/tablas y mencionar solo lo que se puede observar de estos, se deben mostrar los valores y parámetros que se están usando.
- (**1.2 pts**) Análisis: Comentar y concluir sus resultados. Se hacen las inferencias de sus resultados.
- (**0.8 pts**) Conclusión: Recapitulación de lo que se hizo, se concluye lo que se puede decir con respecto a sus resultados. También ven si su hipótesis se cumplió o no y analizan la razón. Por último, se menciona qué se podría mejorar en su desarrollo en una versión futura, qué falta en su documento, qué no se ha resuelto y cómo se podrían extender.

Todo lo mencionado debe estar en sus informes en las secciones en las que se señalan, la falta de algún aspecto o la presencia de algún aspecto en una sección equivocada hará que no se tenga la totalidad del puntaje.