Práctica 7. Tablas Hash

Sesiones de prácticas: 2

Objetivos

Implementación y optimización de tablas de dispersión cerrada.

Descripción de la EEDD

En la práctica 6 han aumentado considerablemente el número de operaciones de búsqueda sobre los diccionarios general y específicos. Si a eso le añadimos la posibilidad de que pudiera haber varios objetos `TextoPredictivo` para diferentes idiomas, donde hubiera múltiples usuarios utilizándolo para realizar operaciones, el número total de operaciones de manipulación podría suponer una carga importante para el sistema a pesar de la eficiencia de dichas operaciones. Esta carga podría reducirse adecuadamente utilizando para los diccionarios una estructura de datos que permitiera realizar las operaciones de inserción y consulta de forma más eficiente.

Las tablas de dispersión están pensadas para hacer búsquedas sobre conjuntos de datos poco modificables, siendo nuestro problema de la búsqueda de palabras un buen ejemplo, puesto que una vez cargadas las palabras del diccionario, el número de nuevas inserciones es relativamente pequeño. En esta práctica se pide sustituir la representación del mapa de palabras del diccionario, tal y como aparece en la práctica 5, mediante una implementación de una tabla de dispersión, `THashCerrada<T>` instanciada a la clase Palabra. Se deben considerar los siguientes aspectos:

- La generación de claves numéricas a partir del término de la palabra se realizará con el algoritmo `djb2` (Lección 14, transparencia 12). La tabla de dispersión deberá recibir como clave el valor numérico (ya que será implementada como una plantilla) por lo que la implementación de este algoritmo no debe formar parte de la tabla. La entrada del algoritmo será el término de la palabra.
- La implementación del cálculo de la secuencia de exploración se hará de dos formas posibles durante el proceso de entrenamiento, tal y como se ha visto en la lección 15: exploración cuadrática y exploración doble por división.
- El tamaño de la tabla deberá ser adecuado para albergar la información con un número máximo de colisiones en torno a 10 (hacer pruebas con distintos tamaños).

- El factor de carga debe ser: λ >= 0.6

Programa de prueba: Control de Versiones (continuación)

En esta práctica se sustituirá el Map de palabras del diccionario por la tabla de dispersión. Se implementarán los siguientes métodos propios de la estructura así como aquellos para comprobar el estado de la tabla de dispersión:

- `THashCerrada<T>::THashCerrada(tamTabla)`
- `bool THashCerrada<T>::insertar(long int clave, const T& dato)`, que inserte un dato nuevo en la tabla a partir de una clave numérica.
- `bool THashCerrada<T>::borrar(long int clave, const T& resultado)`, que elimine de la tabla el dato.
- `T* THashCerrada<T>::buscar(long int clave, const T &resultado)`, que busque un dato a partir de su clave y lo devuelva (o 0 sin no se encuentra)
- `unsigned int THashCerrada<T>::tamaTabla()`, que devuelva el tamaño de la tabla de dispersión.
- `unsigned int THashCerrada<T>::numElementos()`, que devuelva el número de elementos insertados en la tabla.
- `unsigned int THashCerrada<T>::maxColisiones()`, que devuelva el número máximo de colisiones que se han producido en cualquier operación de inserción realizada sobre la tabla.
- `unsigned int THashCerrada<T>::promedioColisiones()`, que devuelva el promedio de colisiones por operación de inserción realizada sobre la tabla.
- `float THashCerrada<T>::factorCarga()`, que devuelva el factor de carga de la tabla de dispersión.

Hay que tener en cuenta que al tratarse de una tabla de dispersión cerrada, en las funciones borrado y búsqueda es necesario pasar como parámetro también el dato para compararlo, ya que pasando únicamente la clave sólo podremos obtener la posición inicial de la exploración cuadrática. Por lo tanto deberemos modificar la clase `Palabra` incluyendo el operador de comparación "=="

Programas de prueba 1:

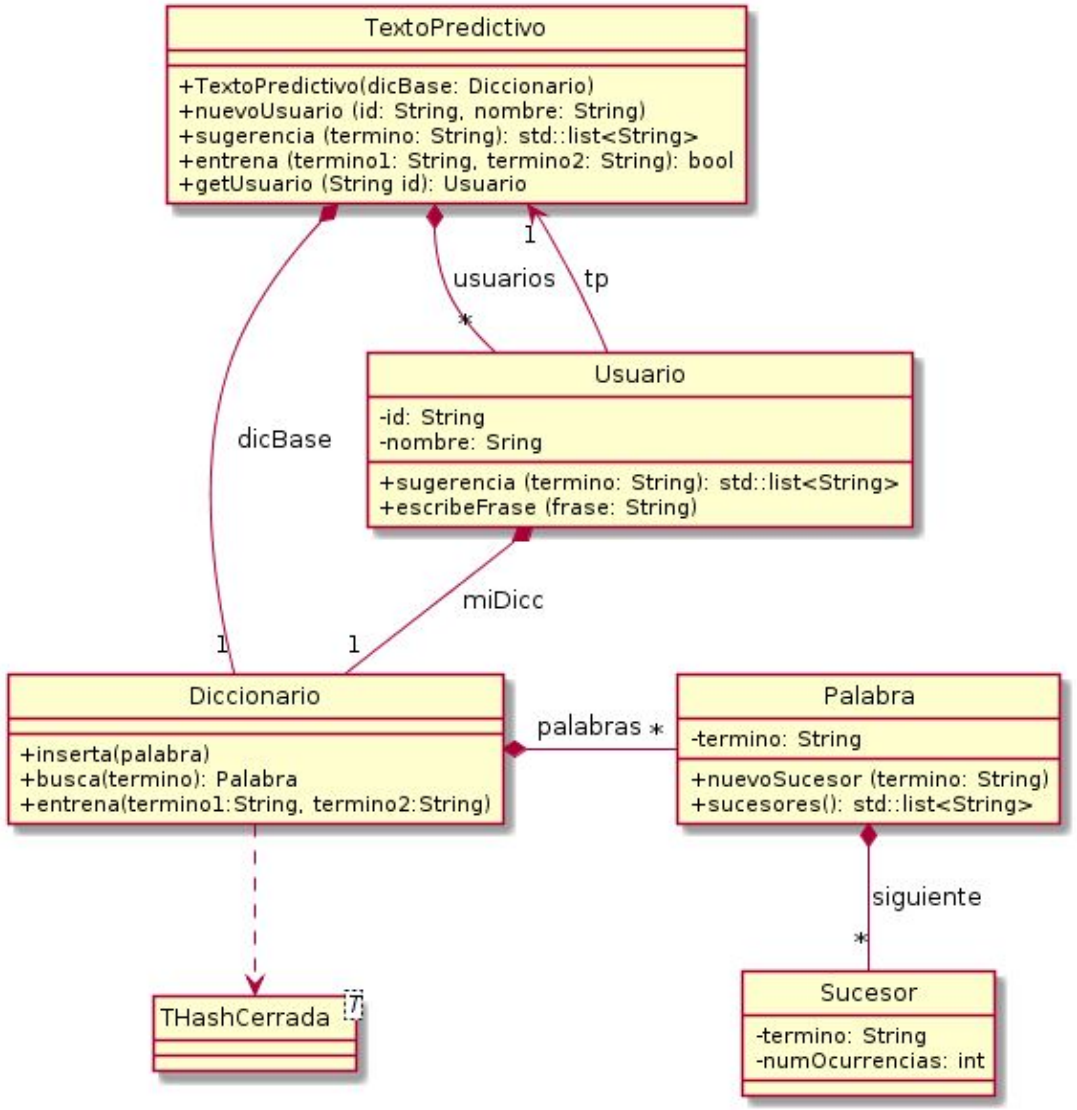
En primer lugar se realizará un programa de prueba que permita ajustar el tamaño de la tabla del diccionario atendiendo a las restricciones indicadas. Para ello, se realizará un estudio definiendo un Diccionario con su correspondiente `THashCerrada<Palabra>`, cargando las palabras del fichero de referencia (sin entrenamiento de corpus) y contabilizando el máximo de colisiones para tres posibles tamaños de tabla, junto con las dos funciones de

dispersión elegidas (3x2 valores recogidos)[1]. Elegir el mejor de los resultados para el siguiente apartado.

Programa de prueba 2:

Una vez que se conozca el tamaño idóneo de la tabla, obtenido del apartado anterior, se implementará un segundo programa de prueba cuya funcionalidad será similar al del enunciado de la práctica 5.

Medir de nuevo los tiempos como en las prácticas anteriores y ver el resultado con el AVL/mapa.



[1] Implementar el cálculo de los métodos de exploración como métodos internos utilizando uno u otro según el experimento que se realice. Estos métodos pueden implementarse `inline` para que el compilador trate de optimizar sus ejecuciones. Finalmente, utilizar únicamente el que haya sido seleccionado, dejando el otro sin utilizar para futuros usos.

Estilo y requerimientos del código:

1. El código debe ser claro, tener un estilo definido y estar perfectamente indentado, para ello se pueden seguir algunos de los estilos preestablecidos para el lenguaje C++ (http://geosoft.no/development/cppstyle.html).

2. Deben comprobarse todas los posibles errores y situaciones de riesgo que puedan ocurrir (desbordamientos de memoria, parámetros con valores no válidos, etc.) y lanzar las excepciones correspondientes, siempre que tenga sentido. Leer el tutorial de excepciones disponible en el repositorio de la asignatura en docencia virtual.

3. Se valorará positivamente la calidad general del código: claridad, estilo, ausencia de redundancias, etc.