Taller de Hash

Implementación de diccionarios con Tablas de Hash

Leticia Rodriguez

Algoritmos y Estructuras de Datos II
DC-FCEyN-UBA

21 de octubre de 2015

DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Recordemos

Temas vistos en la teórica:

- ¿Qué es una tabla de Hash?
- Para qué sirve
- Hashing perfecto
- Colisiones
- Hashing cerrado - abierto
- Factor de carga
- Funciones de Hash

Aplicaciones de hashing en la vida real

Aplicaciones: el lenguaje de programación Java

Implementación de hashing en Java

- Veamos cómo implementa un lenguaje de programación las herramientas que vimos, en este caso **Java**.

Aplicaciones: el lenguaje de programación Java

Implementación de hashing en Java

- Veamos cómo implementa un lenguaje de programación las herramientas que vimos, en este caso **Java**.
- La idea es que a cualquier objeto de una clase (similar a C++) puede pedírsele una función llamada **hashCode**.

Aplicaciones: el lenguaje de programación Java

Implementación de hashing en Java

- Veamos cómo implementa un lenguaje de programación las herramientas que vimos, en este caso **Java**.
- La idea es que a cualquier objeto de una clase (similar a C++) puede pedírsele una función llamada **hashCode**.
- La declaración es int hashCode(), es decir dado un objeto cualquiera, devuelve un entero. ¿Y qué hacemos con ese entero?

Aplicaciones: el lenguaje de programación Java

Implementación de hashing en Java

- Veamos cómo implementa un lenguaje de programación las herramientas que vimos, en este caso **Java**.
- La idea es que a cualquier objeto de una clase (similar a C++) puede pedírsele una función llamada **hashCode**.
- La declaración es int hashCode(), es decir dado un objeto cualquiera, devuelve un entero. ¿Y qué hacemos con ese entero?
- Podemos aplicarle alguna función adicional que nos mapee el entero a la tabla, ejemplo f(x) = x mód. k, con k el tamaño de la tabla.

Java (cont.)

Implementando nuestros hashCode

- En los tipos provistos por el lenguaje, ya viene implementado.
- Si nosotros no lo implementamos en nuestros tipos, por defecto se utiliza la dirección de memoria. ¿Les parece correcto?

Java (cont.)

Implementando nuestros hashCode

- En los tipos provistos por el lenguaje, ya viene implementado.
- Si nosotros no lo implementamos en nuestros tipos, por defecto se utiliza la dirección de memoria. ¿Les parece correcto?
- Como vimos al principio, dos variables observacionalmente iguales podrían estar repetidas en un conjunto. Ok, pero entonces ¿qué es $=_{obs}$ en Java?

Java (cont.)

Implementando nuestros hashCode

- En los tipos provistos por el lenguaje, ya viene implementado.
- Si nosotros no lo implementamos en nuestros tipos, por defecto se utiliza la dirección de memoria. ¿Les parece correcto?
- Como vimos al principio, dos variables observacionalmente iguales podrían estar repetidas en un conjunto. Ok, pero entonces ¿qué es $=_{obs}$ en Java?
- Como el == de C++, podemos implementar una función llamada **equals**.

Java (cont.)

Implementando nuestros hashCode

- En los tipos provistos por el lenguaje, ya viene implementado.
- Si nosotros no lo implementamos en nuestros tipos, por defecto se utiliza la dirección de memoria. ¿Les parece correcto?
- Como vimos al principio, dos variables observacionalmente iguales podrían estar repetidas en un conjunto. Ok, pero entonces ¿qué es $=_{obs}$ en Java?
- Como el == de C++, podemos implementar una función llamada **equals**.
- ¿Se acuerdan de $x_1 =_{obs} x_2 \Rightarrow h(x_1) = h(x_2)$? ¿y la vuelta?

Java (cont.)

La documentación oficial[1] dice

- Si dos objetos son iguales de acuerdo al método equals, entonces llamando a hashCode en cada uno de ellos debe producir el mismo entero.
- No es requerido que si dos objetos son distintos de acuerdo al método equals, entonces llamando a hashCode en cada uno dé diferentes enteros.
- Sin embargo, el programador debería ser consciente de que producir enteros distintos pueda mejorar la *performance* de las tablas de hash.

[1] `http://docs.oracle.com/javase/7/docs/api/java/lang/Object.html#hashCode()`

Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?

Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?
- A priori la complejidad dependería del tamaño de la tabla, que con un factor de carga bajo puede ser muy malo. ¿Lo podemos mejorar?

Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?
- A priori la complejidad dependería del tamaño de la tabla, que con un factor de carga bajo puede ser muy malo. ¿Lo podemos mejorar?
- Podemos tener una lista enlazada de todos los elementos y en la tabla guardar sólo un iterador a esa lista. ¿Cómo queda la complejidad?

Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?
- A priori la complejidad dependería del tamaño de la tabla, que con un factor de carga bajo puede ser muy malo. ¿Lo podemos mejorar?
- Podemos tener una lista enlazada de todos los elementos y en la tabla guardar sólo un iterador a esa lista. ¿Cómo queda la complejidad?

Iteradores ordenados

- Nos ponemos más exigentes y queremos iterar los elementos en orden. ¿Nos sirve para algo la tabla de hash?

Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?
- A priori la complejidad dependería del tamaño de la tabla, que con un factor de carga bajo puede ser muy malo. ¿Lo podemos mejorar?
- Podemos tener una lista enlazada de todos los elementos y en la tabla guardar sólo un iterador a esa lista. ¿Cómo queda la complejidad?

Iteradores ordenados

- Nos ponemos más exigentes y queremos iterar los elementos en orden. ¿Nos sirve para algo la tabla de hash?
- Bueno, en este caso la tabla de hash no es invencible y no nos sirve más que tener una lista enlazada con los elementos.

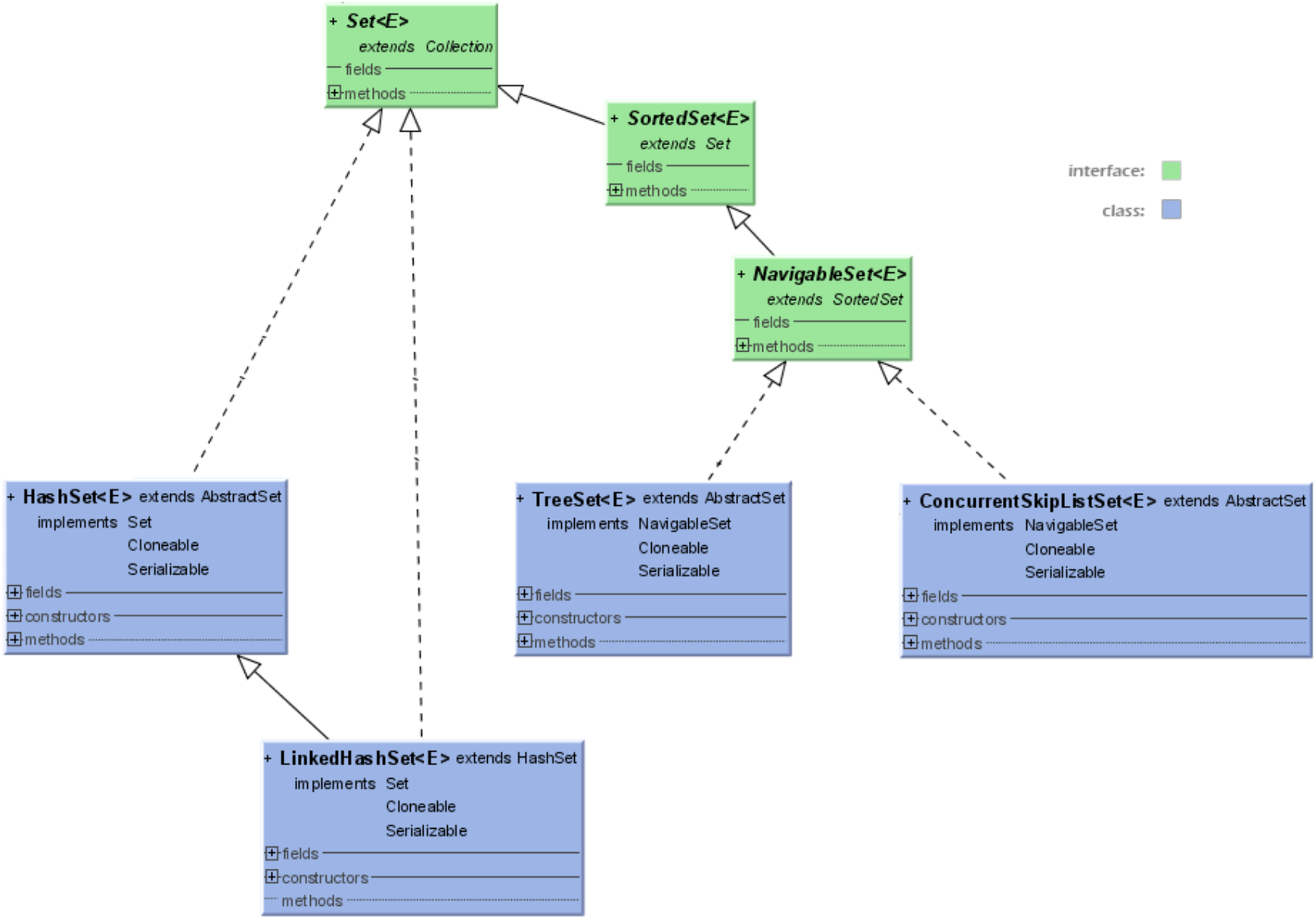
Entonces... ¡las tablas de hash son la posta! ¡son invencibles!

Iteradores

- ¿Qué pasa si queremos iterar los elementos? ¿Qué complejidad tiene?
- A priori la complejidad dependería del tamaño de la tabla, que con un factor de carga bajo puede ser muy malo. ¿Lo podemos mejorar?
- Podemos tener una lista enlazada de todos los elementos y en la tabla guardar sólo un iterador a esa lista. ¿Cómo queda la complejidad?

Iteradores ordenados

- Nos ponemos más exigentes y queremos iterar los elementos en orden. ¿Nos sirve para algo la tabla de hash?
- Bueno, en este caso la tabla de hash no es invencible y no nos sirve más que tener una lista enlazada con los elementos.
- SI necesitamos los elementos ordenados quizás nos convenga más tener un arreglo ordenado o un árbol.


+ Set<E>
extends Collection
fields
methods
+ SortedSet<E>
extends Set
fields
methods
+ NavigableSet<E>
extends SortedSet
fields
methods
interface:
class:
+ HashSet<E> extends AbstractSet
implements Set
Cloneable
Serializable
fields
constructors
methods
+ TreeSet<E> extends AbstractSet
implements NavigableSet
Cloneable
Serializable
fields
constructors
methods
+ ConcurrentSkipListSet<E> extends AbstractSet
implements NavigableSet
Cloneable
Serializable
fields
constructors
methods
+ LinkedHashSet<E> extends HashSet
implements Set
Cloneable
Serializable
fields
constructors
methods

Detección de herrores

Integridad

- Quiero enviar un mensaje/archivo entre dos computadoras y asegurarme de que llegue correctamente.

Detección de herrores

Integridad

- Quiero enviar un mensaje/archivo entre dos computadoras y asegurarme de que llegue correctamente.
- ¿Cómo puede ayudarnos una función de hash?

Detección de herrores

Integridad

- Quiero enviar un mensaje/archivo entre dos computadoras y asegurarme de que llegue correctamente.
- ¿Cómo puede ayudarnos una función de hash?

Checksum

- Puedo aplicar una función de hash a la cadena de caracteres o bytes y comunicar cuál es el valor h(x) para el mensaje.

Detección de herrores

Integridad

- Quiero enviar un mensaje/archivo entre dos computadoras y asegurarme de que llegue correctamente.
- ¿Cómo puede ayudarnos una función de hash?

Checksum

- Puedo aplicar una función de hash a la cadena de caracteres o bytes y comunicar cuál es el valor h(x) para el mensaje.
- Este procedimiento se llama usualmente *checksum* y deben haberlo visto más de una vez junto a un link de descarga de archivos.

Aplicaciones: criptografía

A codear se ha dicho!!

1. Bajar de la web los archivos con los fuentes del taller y los datos para probar: **taller_hash.zip** y **data.zip**
2. Completar la implementación de **DiccHashCerrado.cpp**
3. Correr los test provistos **test_redimensionado** y **test_colisiones** para evaluar la calidad de la función de hash
4. Proponer una mejor función de hash, tamaño de tabla y umbral para el factor de carga que reduzca al mínimo la cantidad de colisiones y las veces que se redimensiona la tabla (sin desperdiciar mucha memoria). Usar los archivos provistos en **data.zip**

Preguntas

?

??