Un Estudio Experimental de Tablas Hash

F. González

June 7, 2006

Abstract

Este reporte estudia el comportamiento de una tabla hash en términos del tiempo de ejecución al variar el factor de carga.

1 Introduction

An instance of Hashtable has two parameters that affect its performance: initial capacity and load factor. The capacity is the number of buckets in the hash table, and the initial capacity is simply the capacity at the time the hash table is created. Note that the hash table is open: in the case of a ”hash collision”, a single bucket stores multiple entries, which must be searched sequentially. The load factor is a measure of how full the hash table is allowed to get before its capacity is automatically increased. When the number of entries in the hashtable exceeds the product of the load factor and the current capacity, the capacity is increased by calling the rehash method.

Generally, the default load factor (.75) offers a good tradeoff between time and space costs. Higher values decrease the space overhead but increase the time cost to look up an entry (which is reflected in most Hashtable operations, including get and put).

The initial capacity controls a tradeoff between wasted space and the need for rehash operations, which are time-consuming. No rehash operations will ever occur if the initial capacity is greater than the maximum number of entries the Hashtable will contain divided by its load factor. However, setting the initial capacity too high can waste space.

If many entries are to be made into a Hashtable, creating it with a sufficiently large capacity may allow the entries to be inserted more efficiently than letting it perform automatic rehashing as needed to grow the table.

This example creates a hashtable of numbers. It uses the names of the numbers as keys:

```
Hashtable numbers = new Hashtable();
numbers.put("one", new Integer(1));
numbers.put("two", new Integer(2));
numbers.put("three", new Integer(3));
```

To retrieve a number, use the following code:

```
Integer n = (Integer)numbers.get("two");
if (n != null) {
   System.out.println("two = " + n);
}
```

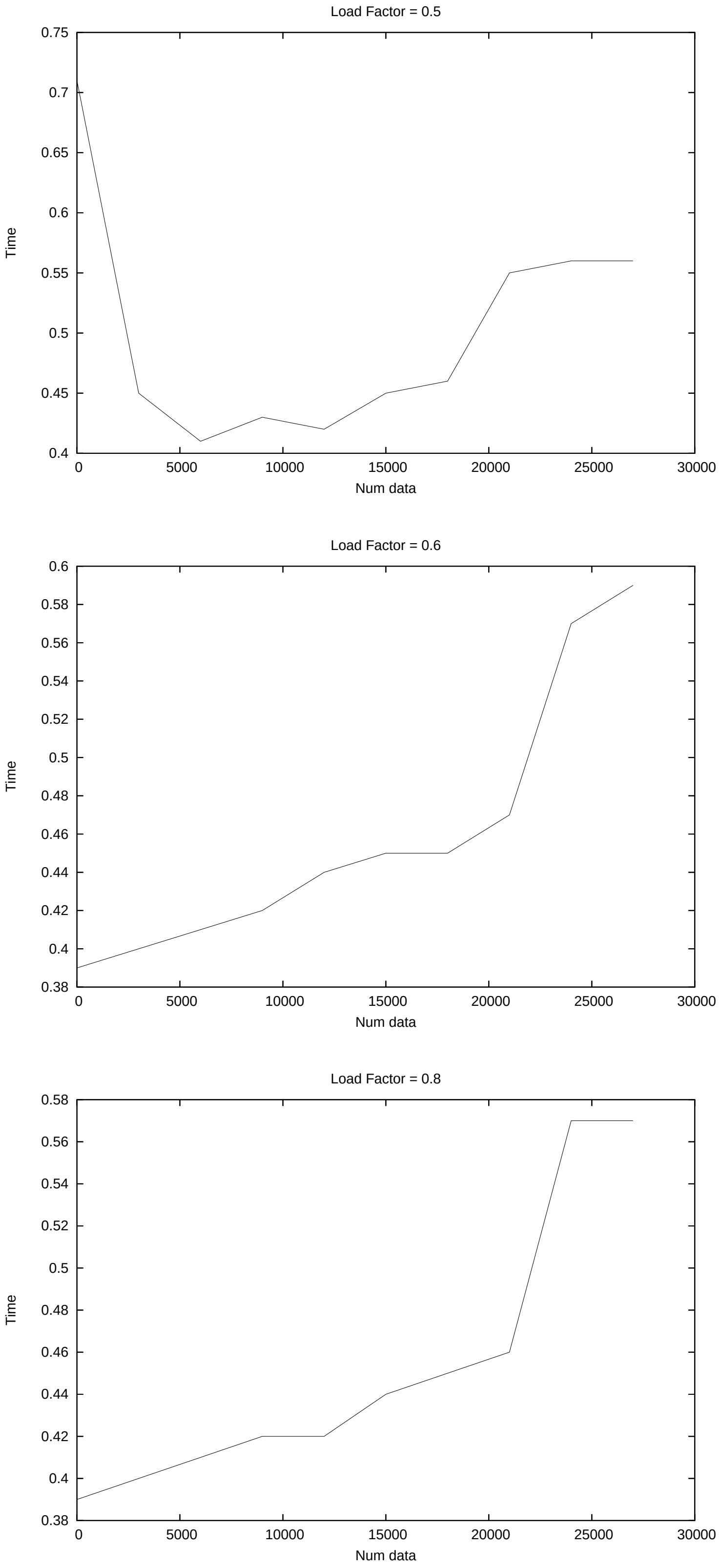

Figure 1: Gráficas número de datos vs. tiempo.

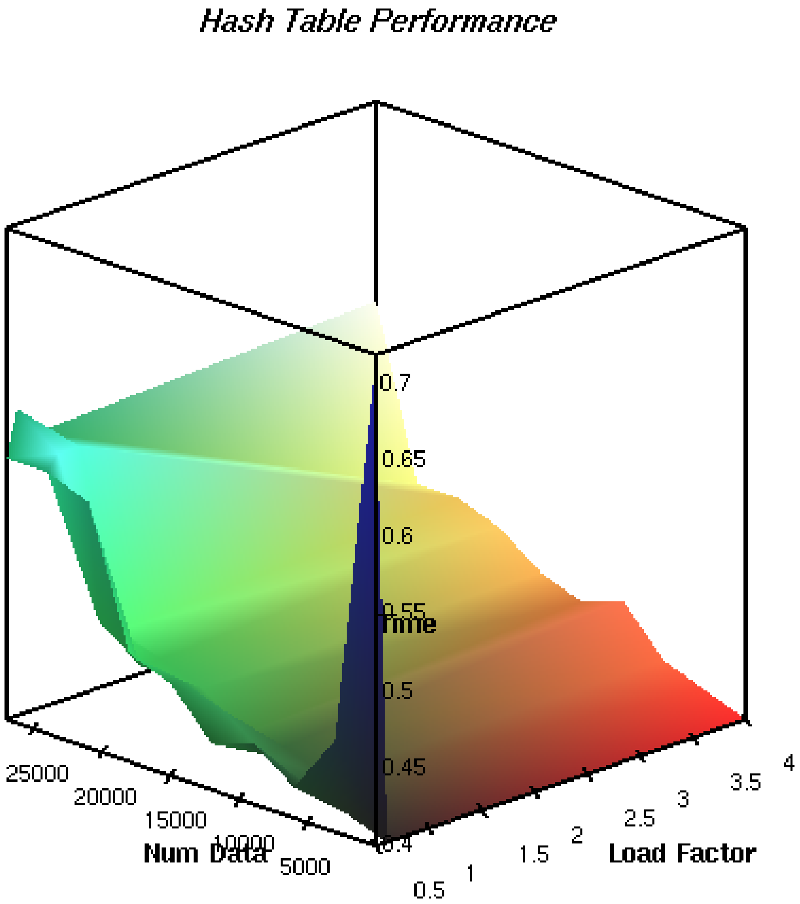

Figure 2: 3D Plot