Integer partitions and their mapping to hash functions

Ka.Shrinivaasan (ka.shrinivaasan@gmail.com)

June 17, 2012

Abstract

This article is a short observation of an interesting relation between Integer Partitions and Hash Functions and derives a number of possible hash functions based on this relation.

1 Introduction

A widely used notion of hash function maps a key to value as h(x) = y. If x1 and x2 are two key values and if h(x1) and h(x2) are equal then x1 and x2 are placed in same bucket. Thus a hash table partitions the set of keys to be hashed into sets of buckets of keys having same hash value.

2 Generating functions to represent Integer Partitions and Euler's theorem

First we study how integer partitions are represented and later their mapping to buckets of hash functions . One way is through generating function and applying Euler's theorem. Consider the product:

$$(1+x+x^2+x^3......)(1+x^2+x^4+x^6.......)(1+x^3+x^6.......)(1+x^4+x^8.......)... \tag{1}$$

To illustrate, consider the coefficient of x^3. By choosing x from the first parenthesis, x^2 from the second, and 1 from the remaining parentheses, we obtain a contribution of 1 to the coefficient of x^3. Let the monomial chosen from the i-th parenthesis $1+x^i+x^{2i}+x^{3i}$ in (1) represent the number of times the part i appears in the partition. In particular, if we choose the monomial x^{i*c_i} from the i-th parenthesis, then the value i will appear c_i times in the partition. Each selection of monomials makes one contribution to the coefficient of x^n and in general, each contribution must be of the form $x^{1*c_1}x^{2*c_2}x^{3*c_3}... = x^{1*c_1+2*c_2+3*c_3.....}$. Thus the coefficient of x^n is the number of ways of writing $n = c_1 + 2*c_2 + 3*c_3 + ...$ where each $c_i \geq 0$. Notice that this is just another way to represent an integer partition. As an example $5 = 1+2+2$ can be written as $5 = 1*1+2*2$. Above generating function is an infinite product of geometric series p(n) which is the Euler's partition theorem.

3 Number of possible hash functions

Having arrived at a way to express the integer partitions and parts in a partition, we analyse how integer partitions and hash functions are related. Each hash table partitions the hashed elements into sets of buckets. We can map each of these buckets to a part in an integer partition. Thus if there are x parts in a partition of n elements then there will be x non-empty buckets in the hash table where size of each bucket is equal to the value of the corresponding part in the partition and is thus a one-to-one and onto mapping. If there are m possible hash values then each of these x parts or buckets can be arranged in mC_x ways for each partition of n elements. If we aggregate it over all the partitions we get all possible ways of placing an element in a bucket which is nothing but all possible hash functions.

1. Let m be the number of possible values of hash function h(x)
2. Let n be the total number of elements which will be hashed and placed in buckets
3. Each hash entry would have a linked list of elements hashed on to a hash value for that entry
4. Let lamda(i) be the number of parts in partition i
5. Let p(n) be the partition function $[lamda(i) \leq m$ and $m \geq n]$
6. Then number of possible hash functions =

$$\sum_{i=1}^{p(n)} mC_{lamda(i)} \tag{2}$$

where lamda(i) which is the number of parts in partition i can be obtained from above generating function for integer partitions as sum of all c_i's,

$$\sum_{i=1}^{q} c_i \tag{3}$$

where the value i will appear c_i times in the partition and q is total number of distinct integer in the partition

4 Acknowledgement

I dedicate this article to God.

5 Bibliography

References

[1] Lectures on Integer Partitions by Herbert Wilf, University of Pennsylvania

[2] This idea was first mentioned by the author in some informal internal email communications at Sun Microsystems when the author worked at Sun Microsystems from 2000-2005(mailid: kannan.srinivasan@sun.com)

[3] Various lecture notes on Generating Functions(Combinatorics)