Integer partitions and their mapping to hash functions - Updated Draft

$SrinivasanKannan(alias)Ka.Shrinivaasan(alias)ShrinivasKannan$
$IndependentOpenSourceDeveloper, ResearcherandConsultant$
$Ph: 9789346927, 9003082186, 9791165980$
$KrishnaiResearchOpenSourceProducts: http://sourceforge.net/users/ka\_shrinivaasan,$
$https://www.ohloh.net/accounts/ka\_shrinivaasan$
$ResearchWebsite: https://sites.google.com/site/kuja27/$
$(ka.shrinivaasan@gmail.com, shrinivas.kannan@gmail.com, kashrinivaasan@live.com)$

April 17, 2014

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:

$$(1 + x + x^2 + x^3.....)(1 + x^2 + x^4 + x^6.....)(1 + x^3 + x^6.....)(1 + x^4 + x^8.....)... \quad (1)$$

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 mP_x ways for each partition of n elements(permutations instead of combinations for order of 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 =

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

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 Restricted Partitions, Compositions and Hash Collision Chaining

1. Riemann sums (discrete approximation of Riemann integral) of all the functions corresponding to the hash functions are same. Thus all such functions form an equivalence class.(Assuming each partition created by the hash functions as a function plot)
2. Hardy-Ramanujan asymptotic bound for partition function p(n) is $O(e^{pi*sqrt(0.66*n)}/(4 * 1.732*n)$ which places a bound on number of hash functions also: ($http://en.wikipedia.org/wiki/Partition_($
3. If m-sized subsets of the above $O(m! * e^{sqrt(n)}/n)$ number of hash functions are considered as a (k, u)-universal or (k, u)-independent family of functions - ($Pr(f(x1) = y1...) < u/m^k$,then follwing the notation above, this m-sized subset family of hash functions follow the $Pr(f(x1) = y1...) < u/m^n$ where n is number of keys and m is the number of values. ($m!$ is for summation over $(m, lamda(i))$ for all partitions)

4. Thus deriving a bound for number of possible hash functions in terms of number of keys and values could have bearing on almost all hashes including MD5 and SHA.

5. Birthday problem and Balls and Bins problem - Since randomly populating m bins with n balls and probability of people in a congregation to have same birthday are a variant of Integer partitioning and thus hash table bucket chaining, bounds for birthday problem and Chernoff bounds derived for balls and bins could be used for Hash tables also: ($http : //en.wikipedia.org/wiki/Birthday_problem, http : //www.cs.ubc.ca/\ nickhar/W12/Lecture3Notes.pdf$)

6. Restricted partitions which is the special case of integer partitions has some problems which are NP-complete. Money changing problem which is finding number of ways of partitioning a given amount of money with fixed denominations(Frobenius number) is NP-complete ($http : //citeseer.uark.edu : 8080/citeseerx/showciting;jsessionid = 92CBF53F1D9823C47F64AAC119D30FC3509754, NaokiAbe1987$). Number of partitions with distinct and non-repeating parts follow Roger-Ramanujan identities (2 kinds of generating functions).

7. The special of case of majority voting which involves integer partitions described in: $https : //sites.google.com/site/kuja27/IndepthAnalysisOfVariantOfMajorityVotingwithZFAOC_2014.pdf$ requires a hash function that non-uniformly distributes the keys into hashes so that no two chains are equal in size (to simulate voting patterns without ties between candidates). This is the special case of restricted partitions with distinct and non-repeating parts of which money changing is the special case and finding a single solution is itself NP-complete.

8. Thus Majority voting can be shown to be NP-complete in 2 ways: a) by Democracy circuit (Majority with SAT) in
$http : //sourceforge.net/projects/acadpdrafts/files/ImplicationGraphsPGoodEquationAndPNotEqualToNPQuestion_excerpts.pdf/download$ and
$https : //sites.google.com/site/kuja27/PhilosophicalAnalysisOfDemocracyCircuitAndPRGChoice_2014-03-26.pdf$ b) by reduction from an NP-hard instance of Restricted Partition problem like Money changing problem (MCP) for Majority voting with constituencies described in $https : //sites.google.com/site/kuja27/IndepthAnalysisOfVariantOfMajorityVotingwithZFAOC_2014.pdf$

9. Infact the above two ways occur in two places in the process of democratic voting: The democracy circuit is needed when a single candidate is elected while the restricted partition in second point is needed in a multi-partisan voting where multiple candidates are voted for.

10. Point 8b above requires a restricted partition with distinct non-repeating parts. There are many results on this like Roger-Ramanujan identities, Glaisher theorem and its special case Euler's theorem which equate number of partitions with parts divisible by a constant and distinctiveness of the parts (odd, differing by some constant etc.,). Such a restricted partition is needed for a tiebreaker and hence correspond bijectively to hash collision chaining.

11. An interesting manifestation of point 10 is that nothing in real-life voting precludes a tie and enforces a restricted partition, with no two candidates getting equal votes, where all voters take decisions independent of one another(voter independence is questionable to some extent if swayed by phenomena like "votebank","herd mentality" etc.,) thereby theoretically invalidating the whole electoral process.

12. Counting Number of such restricted partitions is a *SharpP-complete* problem - $https : //www.math.ucdavis.edu/\ deloera/TALKS/denumerant.pdf$

13. If a Hash table is recursive i.e the chains themselves are hashtables and so on... then this bijectively corresponds to a recurrence relation for partition function (expressing a partition of a higher integer in terms of lower integer).

14. If the hash table chains are alternatively viewed as Compositions of an integer (ordered partitions) then there are 2^{n-1} maximum possible compositions.
($http://en.wikipedia.org/wiki/Composition_(number_theory)$)

15. In the summation over all parts of partitions derived in
$https://sites.google.com/site/kuja27/IntegerPartitionAndHashFunctions.pdf$
if $m == n$ then it is the composition in point 14 above and thus summation over all parts of partitions is greater than or equal to 2^{n-1} as some permutations might get repeated across partitions. Thus the summation expresses generalized restricted composition :

$$\sum_{i=1}^{p(n)} nP_{lamda(i)} >= 2^{n-1}(where \quad m = n) \tag{4}$$

16. For large n the above summation asymptotically equals $e * n!$. Logarithm of above summation then is greater than or equal to $(n-1)$ and thus can be equated to any partition of n. Thus any partition can be written as a series which is the combinatorial function of parts in all individual partitions.

17. As a special case when all the collision chains are of equal size or have greatest common divisor, a bijection to factors of n and hash function collision chains is obtained.Number of such Hash functions is a function of number of non-trivial factors of n.

18. Generating function for number of distinct partitions is given by $\prod_{k=1}^{\infty}(1+x^k) = \sum_{n=1}^{\infty} q(n) * x^n$

19. Thus number of valid multipartisan majority voting patterns = number of distinct partitions of n(q(n)) = number of hash functions with distinct collision hash chain sizes (without permutations)

20. If the denominations are fixed as $1, 2, 3, 4, 5, ..., n$ then the denumerants to be found are from the diophantine equation: $a1*1+a2*2+a3*3+a4*4+a5*5+...+an*n$ ($with\ ai = 0\ or\ 1$). GCD of all $ai(s)$ is 1. Thus Schur's theorem for MCP or Coin Problem applies. Finding one such distinct partition and hence a majority voting pattern without tie is NP-complete by this reduction. This can also be proved to be NP-complete considering the above diophantine as a 0-1 Integer Linear Programming(ILP) NP-complete instance. This partition is guaranteed to have a biggest part always and resolves tie if any.

21. Maximizing Collision in hashes is necessary in finding similarity (e.g DNA, voice and data) and is hence a *pattern matching algorithm* where as minimizing collision results in a special case of *one way function with collision resistance.*

22. Restricted partitions can be classified into two sets as follows when mapped to hash function collision chains:

    (a) Set of partitions in which no part exceeds a constant size (more Collision Free)

    (b) Set of partitions in which no part is less than a constant size (more Collision Prone)

23. Recurrence relation for restricted partitions in which no part exceeds N and each partition is of size M is given by: $p(N, M; n) = p(N, M-1; n) + p(N-1, M; n-M)$

24. Above recurrence has to be subtracted from partition function to get restricted partitions in which no part is less than N

25. Above points on Schur's theorem are also described in handwritten notes at: $https://sites.google.com/site/kuja27/SchurTheoremMCPAndDistinctPartitions\_2014-04-17.pdf$

5 Acknowledgement

I dedicate this article to God.

6 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)

[4] Various website resources for Restricted Partitions

[5] http://sourceforge.net/p/asfer/code/HEAD/tree/AstroInferDesign.txt

[6] Majority voting related drafts and handwritten notes in http://sites.google.com/site/kuja27

[7] Generating function - $http://math.berkeley.edu/mhaiman/math172-spring10/partitions.pdf$

[8] Schur's theorem for asymptotic bound for number of denumerants - $http://en.wikipedia.org/wiki/Schur's_theorem$

[9] Frobenius problem - $http://www.math.univ-montp2.fr/ramirez/Tenerif3.pdf$

[10] Hash collision chains and Schur Theorem - $https://sites.google.com/site/kuja27/SchurTheoremMCPAndDistinctPartitions\_2014-04-17.pdf$

[11] Standard NP-Completeness reductions - $http://www.cs.cmu.edu/afs/cs/academic/class/15451-s10/www/recitations/rec0408.txt$