Journal of Universal Computer Science, vol. 10, no. 9 (2004), 1239-1249
submitted: 1/9/04, accepted: 21/9/04, appeared: 28/9/04 © J.UCS

Full Hash Table Search using Primitive Roots of the Prime Residue Group *Z/p*

Joerg R. Muehlbacher
(FIM, Johannes Kepler University of Linz, Austria
muehlbacher@fim.uni-linz.ac.at)

Abstract: After a brief introduction to hash-coding (scatter storage) and discussion of methods described in the literature, it is shown that for hash tables of length p >2, prime, the primitive roots r of the cyclic group ***Z/p*** of prime residues mod p can be used for a simple collision strategy $q(p,i) = r^i \bmod p$ for $f_i(k) = f_0(k) + q(p,i) \bmod p$. It is similar to the strategy which uses quadratic residues $q(p,i) = i^2 \bmod p$ in avoiding secondary clustering, but reaches all table positions for probing. A table of n primes for typical table lengths and their primitive roots is added. In cases where $r = 2^j$ is such a primitive root, the collision strategy can be implemented simply by repeated shifts to the left (by j places in all).
To make the paper self-contained and easy to read, the relevant definitions and the theorems used from the Theory of Numbers are included in the paper.

Key Words: Hash tables; Full table scatter storage techniques; Collision strategy; Cyclic group mod p; Primitive roots of the prime residue group mod p

Category: E.2, G.4

1 Introduction

Methods of scatter storage (hashcoding) to store and search for data, particularly in tables, are discussed in detail in the literature and clearly described in [Maurer, Lewis 75], for instance.

We start from a key set $K \subseteq \boldsymbol{N}$ and an address space A in the form of a table T of size n with $A = \{0, 1, 2, ..., n-1\} \subset \boldsymbol{N}_0$ and a *hash function* $f_0: K \rightarrow A$, which implies $T[a] \in K$ für $k \in K$. In general $|K| >> n$. Although only a small subset $K' \subset K$ with $|K'| \leq n$ is stored in A for locating, it must be assumed that f_0 is not injective, i.e. it is possible for two different keys $k_1, k_2 \in K$ to be assigned the same home address: $\exists\ k_i, k_j \in K$ with $a_0 = f_0(k_i) = f_0(k_j)$. In this case a *primary collision* occurs, and a *collision strategy* q is needed to find a substitute for an already occupied address a_0. Thus $a_0 = f_0(k)$ does not imply $T[a_0] = k$.

Among the procedures discussed in the literature, we consider so-called *open hash-coding*. Here, if a primary collision occurs, a search is carried out in the table itself for an as yet unoccupied address $a_i \neq a_0$. It must of course be possible to reconstruct this collision address during a search. The drawback of open hash-coding is that *secondary collisions* can occur:

Assuming that, for $k_1 \neq k_2$, $f_0(k_1) = f_0(k_2) = a_0$, and that owing to this primary collision k_2 has been stored in a_3, in line with the collision strategy – if we now need to enter a key k_3 with $k_3 \neq k_1$, $k_3 \neq k_2$ and $f_0(k_3) = a_3$ in table T, its home address a_3 is

already occupied, a secondary collision occurs, and k_3 must be stored elsewhere, say in a_4. But this will provoke a collision for a further key k_4 with $f_0(k_4) = a_4$.

In general, therefore, the first try made for k is at its home address, and if this is occupied a sequence of addresses a_1, a_2, ... a_i is worked through until the first empty table item a_i has been found as a substitute address (collision address). Note that no inferences can as yet be drawn from the address sequence laid down in the collision strategy, i.e. i>j does not imply $a_i > a_j$ mod n.

The way in which collision strategy q identifies a_i depends on table length n = | A |, from i and in certain procedures from k itself, too.

We use the following notation:

$$f_i(k) = f_0(k) + q_i(k,n,i) \bmod n \text{ with } q_0 = 0$$

where i is the smallest $0 \leq i < n$,

(i) for which an as yet unoccupied slot a_i for a new key is found by means of $f_i(k)$ If no such i can be found, the table is described as full. To keep things simple we shall always assume that with t slots in the table occupied the load factor f = t/n < 1, i.e. that at least one slot is empty. It is perfectly possible for $q_i(k,n,i)$ not to reach all slots, and thus to terminate the procedure, even though empty slots are available.

(ii) such that in the search for a key k this is found by means of q_i (k,n,i) or a empty slot is reached without success.

The main aim is thus to determine the simplest possible procedures $f_i(k)$ that

(i) reach all addresses 0, 1, ... n-1

(ii) avoid secondary collisions, and the resulting clusters of already occupied collision addresses, as far as possible.

In what follows we are going to concentrate on q(k,n,i); we can thus assume $f_0(k)=0$ w.l.o.g. If the distribution of the key set K' $\subset$ K actually encountered is unknown, one usually employs $f_0(k)$ = k mod p with p > 2 prim , so that A corresponds to the (smallest representative of the) residue classes { $\overline{0}$, $\overline{1}$,..., $\overline{p-1}$ }. In this paper it is assumed that $f_0(k)$ = k mod p.

2 Classification of Collision Strategies

2.1 Linear Collision Strategy

$f_i(k) = f_0(k) + c * i \bmod n$ with c relatively prime to n.

So that q(k,n,i) = q(n,i) = c * i mod n can address the entire table in any case, module n and coefficient c must be relatively prime to each other: gcd(n,c)= 1.

This follows from the following

Theorem 2.1: If $\{r_0, r_1, \ldots\ldots r_{n-1}\}$ is a complete residue class system mod n and gcd (n,c) =1, $\{c* r_0, c*r_1, \ldots\ldots c* r_{n-1}\}$ is one as well.

The linear collision strategy is a prey to cluster formation, since, even if $f_0(k_1) \neq f(k_2)$, once $q_i(k_1) = q_j(k_2)$ for $i \neq j$, the consequence will always be that $f_{i+m}(k_1) = f_{j+m}(k_2)$ für $m \geq 0$. Accordingly the same addresses must be checked for k_2 that have been tested unsuccessfully to eliminate a collision in the case of k_1, say.

2.2 Collision Strategy Using a Pseudorandom Number Generator

The aim of this procedure is to avoid cluster formation resulting from collisions. The approach is to determine collision addresses by means of a pseudorandom number generator. Here the quality of such a generator in the statistical sense is less important than the requirement that its period is long enough for the entire table to be worked through. In the original paper [Morris 68] specified the generator

$$q(n,i) = (5^i \bmod 2^{j+2})/4 = (5^i \bmod 4*n)/4$$

for tables of length $n = 2^j$, without explanation. We shall return to this in our analysis.

2.3 Collision Strategy Using Quadratic Residues

Quadratic collision strategies also attempt to avoid cluster formation; they were introduced by W.D.Maurer [Maurer, Lewis 75].

The following approach is adopted (assuming a prime number p > 2 for table length):

$$f_i(k) = f_0(k) + c * i + b*i^2 \bmod p$$

and since c>0, b>1 yield no advantage, the strategy boils down to

$$q(p,i) = i^2 \bmod p, \text{ i.e. } f_i(k) = f_0(k) + i^2 \bmod p$$

with the following

Definition 2.1: the number a is called a *quadratic residue* QR(p) of p, p prime, if an x exists such that $x^2 \equiv a \bmod p$.
Otherwise a counts as a *quadratic non-residue* NR(p).

One difficulty with this procedure is that there are only (p-1)/2 quadratic residues, so only half the table is available for collision addresses, if no further assumptions are made about p.

The sequence of quadratic residues mod p can be generated by means of 1^2, 2^2, ….., $((p-1)/2)^2$, for it can be demonstrated that these numbers are all pairwise incongruent mod p. It is also easy to compute them by addition with diff = 1; while (diff < p) {…; diff = diff+2}. C.E.Radke [Radke 70] and in modified form C.Day [Day 70] have specified a procedure with which, for primes of the form p = 4*j +3, the entire table can be accessed. A supplement for tables of length p^j is due to A.F. Ackermann [Ackermann 74].

Using the relation NR(p) = QR(p)*NR(p), J.R.Muehlbacher [Muehlbacher 81] has shown that it is possible to search the table for quadratic residues and quadratic non-residues in parallel, and thus to cover the entire table. This is also true for prime powers p^j, p>2, for if a is a QR(p), then a is also a QR(p^j), p>2.

In the case of primes of the form p= 8*j +3 or p = 8*j –3 the number 2 can be used as NR(p), since C.F.Gauss has proved that 2 is a NR(p) for such primes. It is thus possible to generate the addresses complementary to 1^2,2^2, ….., $((p-1)/2)^2$ by means of $\{i^2 * 2 \mid i=1,2,..,(p-1)/2\}$, i.e. by a simple shift left on every quadratic residue generated.

2.4 Further Procedures

All the procedures discussed so far are of the form q(n,i,k) = q(n,i), that is, they are independent of the key itself. In any attempt to avoid clusters developing, it makes sense to introduce the actual key (in a suitable form) when computing the addresses to be tested. The double hash procedure [Luccion 72] [Knuth 73] and its numerous variations are particularly significant representatives of this approach.

As an example we cite the linear quotient method [Bell 70], which is a generalization of the linear strategy:

$$f_i(k) = f_0(k) + i * c(k) \bmod p, \quad p \text{ prim}$$

If the representation k = p*c + a is used for $f_0(k)$, then c is dependent on k: c = c(k).

Since p is prime, gcd(c,p) = 1 holds. However, the exceptional case $c \equiv 0 \bmod p$ must be dealt with explicitly by means of c = 1.

3 Period Length of Random Number Generators

What follows below has actually been clear from the Theory of Numbers for many years; but this brief summary is meant to explain to the reader why (for instance) in Morris' procedure [Morris 68] the number 5 and $5^i \bmod 2^{n+2}$ are selected, and that the entire table can be covered as a result. The explanations also provide the basis for the extension to ***Z/p,*** the prime residue class group mod p, p>2 in section 4.

The class of the multiplicative random number generators is characterized by $x_{i+1} = x_i + c \bmod m$, where $c \geq 0$ and the initial value x_0 need to be chosen suitably. Since only a finite number of values is possible for the sequence $\{x_i\}$, every such generator becomes periodic. Usually c = 0 is selected.

In the example mentioned, $x_{i+1} = (5^i \bmod 2^{j+2})/4$, the claim is made that the period is maximal, i.e. the addresses $\{0,1,2,...,n= 2^j -1\}$ are worked through without repetition.

In principle we have a multiplicative generator of the form $x_{i+1} = 5^i \bmod m$, generalized as $x_{i+1} = a^i \bmod m$ and gcd(a,m) = 1

Here one might be surprised at first that $(5^i \bmod 2^{j+2})$ is computed to start with and subsequently divided by 4, so as to remain in the address region [0, n-1].

Definition 3.1: If a, m are relatively prime, then the smallest number

m = ord(a) for which $a^{\mu} \equiv 1 \bmod m$ is true is called the *order* of a.

We are now interested in numbers a for which μ is maximal, so that the longest possible period can be achieved.

Definition 3.2: A number a mod m (relatively prime to m) is called a *primitive element modulo m* if its order μ is maximal.

As long ago as 1801 C.F. Gauss proved the following theorem:

Theorem 3.1: In a series $x_{i+1} = a^* x_i \bmod m$, a maximal period μ is achieved if
(i) x_o is relatively prime to m
(ii) a is a primitive element modulo m.
(iii) In the special case $m = n = 2^j$, $\mu(2) = 1$, $\mu(4) = 2$ and $\mu(2^j) = 2^{j-2}$ for $j \geq 3$.

In condition (iii) we have referred only to the special case $m = 2^j$; in actual fact Gauss provided a proof to $\mu(j)$ for any j. Much the same applies to the investigation of what conditions must be satisfied for a to be a primitive element modulo p^j, p prime [Knuth 98].
For $m = 2^j$ we also obtain:

Theorem 3.2: If $m = 2^j$ with $j \geq 4$, then a is a primitive element modulo m if and only if either $a \bmod 8 = 3$ oder $a \bmod 8 = 5$.

The two foregoing theorems provide the explanation for Morris' procedure.
Because $n = 2^j$ and $\mu(2^j) = 2^{j-2}$ and $j \geq 4$, mod 2^{j+2} is computed and $a \bmod 8 = 5$ holds.
In addition, $x_o = 5^0$ is trivially relatively prime to n.

4 Primitive Roots of the Cyclic Residue Class Group mod p

As hash function we again employ $f_0(k) = k \bmod p$, with $p > 2$ prime, and can assume $f_0(k) = 0$ w.l.o.g. This leaves the task of working through all addresses $A\backslash\{0\}=\{1,2,\ldots\ldots,p-1\}$ in such a way that cluster formation is avoided as far as possible. The starting-point is the multiplicative prime residue class group mod n. To keep the presentation self-contained, we begin by repeating various definitions from the Theory on Numbers. To make the mathematical background intelligible, we also list the requisite theorems. The interested reader is referred to [Hasse 64], for instance.

Definition 4.1: The Euler φ function $\varphi(n)$ is the number of elements relatively prime to n from $\{1,2,\ldots,n\}$.

For prime numbers p we have $\varphi(p) = p-1$.

Theorem 4.1: The numbers relatively prime to module n form by multiplication a group ***Z/n***, which contains $\varphi(n)$ elements and is called a *prime residue class group mod n*.

If we choose n = p, then ***Z/p*** coincides with the set ***Z/n*** = $\{\overline{1},\ldots,\overline{p-1}\}$ = $\{1,2,\ldots,p-1\}$, since everything is computed mod p.
Because $f_0(k) = 0$, $0 \cup$ ***Z/p*** describes the entire table T by A.
Since p is prime, gcd(a,p) =1 is true for all $a \in$ ***Z/p***, and because the number of elements in ***Z/p*** is finite the order μ = ord(a) for which $a^{\mu} \equiv 1$ is explained. In the general case the set $[a] = \{a^n | n = 1,2,..\}$ defines a subgroup ***Z/p***, i.e. a subset of A**.** We must therefore search for an a such that [a] = ***Z/p***.

The aim is to generate the entire group ***Z/p*** with the aid of a single element a ∈ ***Z/p*** , for we obtain all addresses in the hash table in conjunction with the home address 0!
Here we add the following

Definition 4.2: A group *G* is called *cyclic* if an element x ∈ *G* exists such that [x] = *G*. In that case x is called an element generating *G*.

For the prime residue class groups mod m with m = p the following theorem is valid:

Theorem 4.2: For every prime number p the prime residue class group ***Z/p*** is cyclic.

Definition 4.3: A natural number a is called a *primitive root* if the residue class of a mod n generates the residue class group ***Z/n***.

Conclusion 4.1: From φ(p) = p-1 it follows directly that a number a (residue class) for which gcd(a,p) = 1 is a primitive root if and only if a as generating element has the order φ(p) = p-1. The smallest residues of the powers $a,a^2,a^3,\ldots,a^{p-1}$ are then all different from each other and generate the required address set A\{0}= {1,2,…,p-1}.

We also note that primitive roots exist for every prime number, and that more than one primitive root may exist for a given prime number; thus the generating element ***Z/p*** is not uniquely defined a priori.
The following theorem is valid:

Theorem 4.3: For every divisor d ∈ **N** of p-1 exactly φ(d) prime residue classes mod p of the order d exist. These arise from such an a mod p in the form

a^b mod p with gcd(b,d) = 1.

From this it follows with d = p-1 that exactly φ(p-1) primitive roots a mod p exist and that, for any known a, they can be generated by means of a^b mod p with gcd(b,p-1) = 1.

For the application the following conclusion is also important:

Conclusion 4.2: If 2 is not a primitive root mod p, p > 2, then 2^i, i ∈ ***N***, is not a primitive root mod p either.

Example:
Module p = 7

Residue class:	2	3	4	5	6
Primitive root:	n	j	n	j	n

$a,a^2,a^3,\ldots,a^{p-1}$ with a = 3 yields 3, 2, 6,4,5,1 order φ(p) = p-1 = 6
$a,a^2,a^3,\ldots,a^{p-1}$ with a = 5 yields 5,4,6,2,3,1
$a,a^2,a^3,\ldots,a^{p-1}$ with a = 6 yields 6,1,6,1,6,1 order 2
Here we are briefly concerned with assertions in the Theory of Numbers and algebra regarding the cyclicity of ***Z/n*** for any n whatever, or for powers of p and assertions

connected with this, to the extent necessary to give reasons why the more general case is of no particular practical value for hash-coding.

The general theorem proved by Gauss holds:

Theorem 4.4: Let $n \geq 2$ and $p \neq 2$ prime. The prime residue class group ***Z/n*** is cyclic if and only if $n = 2$ or $n = 4$ or $n = p^j$ or $n = 2\,p^j$ with $j \in \boldsymbol{N}$.

For the application, though, we need a cyclic group of the order n-1 to go with the module n, so as to be able to generate the addresses A in conjunction with the home address $f_0(k) = 0$. The Euler function $\varphi(n)$ is multiplicative, i.e. $\varphi(a) * \varphi(b) = \varphi(ab)$, if $\gcd(a,b) = 1$. Since $\varphi(2) = 1$ and $\varphi(p^j) = p^j - p^j - 1$, a residue class module of the form $2\,p^j$ is no use.

The case $m = 2^j$ is covered by Morris' random number generator, but has the disadvantage that $k \bmod 2^j$ cannot be used as a hash function. In the binary representation of k this corresponds to a shift right by j places, and it is obvious that the hash function $f_0(k)$ would tend a priori to result in clusters as a result.

In the same way, unanswered questions about the existence of primitive roots are of no further importance here for practical reasons, except that it is known that at least one primitive root exists for every prime number. But determining primitive roots w to a module p systematically, for instance the smallest, does involve a slight problem.

No general constructive procedure exists for this, so we are forced to have recourse to a table, which can be computed for $1 \leq w \leq p-1$ by means of the necessary and sufficient relation

$w^{p-1} = 1 \bmod p$ with $\varphi(p) = p-1$ and $w^j \neq 1 \bmod p$ for $0 < j < p-1$

Various software packages (such as *Mathematica*) provide functions to compute all primitive roots of a given prime number p in their program libraries.

For hash tables of length p, the values of p of special interest as table lengths – for reasons of efficiency – are those for which 2 is a primitive root and $j \in \boldsymbol{N}$, so that 2^j is also a primitive root. For it is then possible to handle raising to a power iteratively by a shift left by j places!

5 Algorithmic Solution

We can implement the results so far in the following collision strategy:

Since A and A\{0} together form a group, we simply obtain, with $f_i(k) = f_0(k) + q(i,p) \bmod p$ and a primitive root w mod p and $q(0,p) = 0$,

$$f_0(k) = k \bmod p$$

$$f_i(k) = f_0(k) + w^i \bmod p \quad i=1,2,...,p-1$$

The programs below implement this strategy. A table listing primitive roots w for typical values of p is appended.

```
/*
        let  p be a prime number suitable for the length of the hash table hashtable
        let k!=0 be the key
        T[a] == 0 : place is empty;
        let w be a primitive root mod p
```

home address home = k mod p // $f_o(k)$;
It is assumed that at least one slot in the table is empty.

```
*/
boolean find_Key(int k, int p, int w) {
  int home=k mod p;
  int tmp=w;
  int a=home;

  while ((t[a]!=k) && (t[a]!=0)) {
    a=(home+tmp) mod p;
    tmp=(tmp*w) mod p;
  }
  return (t[a]==k);
}


insert_Key(int k, int p, int w) {
  int home=k mod p;
  int tmp=w;
  int a=home;

  while ((t[a]!=k) && (t[a]!=0)) {
    a=(home+tmp) mod p;
    tmp=(tmp*w) mod p;
  }
  t[a]=k;
}
```

6 Generalization with a Factor Group

Definition 6.1: If G is an abelian group, $U \subseteq G$ a subgroup and $g \in G$, then the coset gU = Ug is called a normal divisor of G.

A normal divisor U of G generates a decomposition of G into cosets
$G = U \cup U_{g1} \cup U_{g2} \ldots \cup U_{gi}$ and the following theorem is valid:

Theorem 6.1: If G is a group, $g \in G$ and U is a subgroup of G, then

(1) $g \in U$ if and only if g U = U
(2) the various cosets with respect to U form a partition of G and every g lies in a uniquely defined coset
(3) all cosets with respect to U have the same cardinality, i.e. the same number of elements.

The cosets for the normal divisor U form a group, the factor group G/U.
In addition, ord(G/U) * ord(U) = ord(G).

Before we discuss the general case, let us start with a decomposition G = U ∪ Ug, i.e. we want to partition the address set A \ {0} achieved with the collision strategy into two subsets A = U ∪ V of equal size.
Starting from theorem 4.3, we select d = (p-1)/2 as divisor and determine a prime residue class r in ***Z/p*** such that $r^d \equiv 1 \bmod p$ and $r^i \neq 1 \bmod p$ for i < d.
Because d < p-1, this is not a primitive root, but it generates a cyclic subgroup U for which ord(U) = ord (***Z/p***)/2 = (p-1)/2 in ***Z/p*** with [r] = {r,r^2,...,r^{d-1},1}, and U is a normal divisor.

If we select r with this property, then U = [r] supplies exactly half the addresses. The set V = G \ U complementary to U has the same number of elements, in line with theorem 6.1, and with rU = Ur = U we obtain ***Z/p*** = U ∪ V with |U| = |V| = (p-1)/2.
To determine V algorithmically we need a g ∈ ***Z/p*** with g ∉ U ⊆ ***Z/p***. Therefore $g^d \not\equiv 1 \bmod p$ with d < p-1 must hold. A primitive root w ∈ ***Z/p*** does the job, because $w^{p-1} \equiv 1 \bmod p$ and $w^i \not\equiv 1 \bmod p$ for 0 < i < p-1.

The normal divisor U and the coset V can be generated step by step once r and g have been selected:
U: $\{r^i\}$ i = 1,2,...,(p-1)/2
V: $\{r^i * g\}$ i = 1,2,...,(p-1)/2

Example: p = 13, r = 4 with $r^6 \equiv 1 \bmod 13$, primitive root w=2
yields the sequence 4,8,3,6,12,11,9,5,10,7,1,2.
For a primitive root w and a generating element r with
ord(r) = (p-1)/2 we can thus formally write:

$f_i(k) = k \bmod p + q_i(p,i) \bmod p$ mit $q_0 = 0$;

$$q_i = \left\{ \begin{array}{ll} i \text{ odd} & v^{\frac{i+1}{2}} \\ i \text{ even} & v^{\frac{i}{2}} * w \end{array} \right\} \text{ for i>0}$$

This strategy can be formally transferred to any normal divisor U of ***Z/p*** whatever, and in the general case, once U has been selected, one obtains a partition of the address set A which is determined by the factor group ***Z/p*** in line with U. However, it is no longer possible (as it was in the case ***Z/p*** = U ∪ V) to use a primitive root to raise the elements r ∈ [r] to a power.

Example: p = 13, r = 5 with $r^4 \equiv 1 \bmod 13$, generating elements 6 and 3
yields the following decomposition:
{5,12,8,1}∪{4,7,9,6}∪{2,10,11,3}

If U is a normal divisor, i.e. a cyclic subgroup of ***Z/p*** with ord (U) = u, then one obtains a decomposition into cosets with (p-1)/u classes and these form a partition of the address set. The individual partitions correspond to the factor group ***Z/p*** with respect to U.

References

[Ackerman 74] F. Ackerman: Quadratic search for hash tables of sizes p^n. CACM, Volume 17, Issue 3, p.164 (March 1974)

[Batagelj 75] V. Batagelj: The quadratic hash method when the table size is not a prime number. CACM, Volume 18, Issue 4, p.216-217, April 1975

[Bays 73] C. Bays: The reallocation of hash-coded tables. CACM, Volume 16, Issue 1, p.11-14, Jan 1973

[Bell 70] J. R. Bell: The quadratic quotient method: a hash code eliminating secondary clustering. CACM, Volume 13, Issue 2, p.107-109, Feb 1970

[Bell, Kaman 70] J. R. Bell, C. H. Kaman: The linear quotient Hashcode. CACM 13, 675-677, 1970

[Brent 73] R. P. Brent: Reducing the retrieval time of scatter storage techniques. CACM, Volume 16, Issue 2, p.105-109, Feb 1973

[Burkhard 73] W. A. Burkhard: Full table quadratic quotient scatter table searching. Proc.6^{th} Hawaii Int. Conf on Systems Sc., 81-82, 1973

[Day 70] A. C. Day: Full table quadratic searching for scatter storage. CACM, Volume 13, Issue 8, p.481-482, Aug. 1970

[Gauss 1871] C. F. Gauss: Disquisitiones Arithmeticae. 1871

[Hasse 64] H. Hasse: Vorlesungen über Zahlentheorie, 1964, Springer

[Knuth 73] D. E. Knuth: The Art of Computer Programming. Vol. 3: Searching and Sorting, Addison Wesley 1^{rd} Edition 1973

[Knuth 98] D. E. Knuth: The Art of Computer Programming. Vol. 2: Seminumerical Algorithms,3^{rd} Edition, p 10- 21, Addison Wesley 1998

[Lamport 70] L. Lamport: Comment on Bell's quadratic quotient method for hash coded searching. CACM, Volume 13, Issue 9, p.573-574, Sept. 1970

[Larson 88] P.-A. Larson: Dynamic Hash Tables. CACM, Volume 31, Issue 4, p.446-457, 1988

[Luccion 72] F. Luccion: Weighted increment linear search for scatter Storage. CACM Volume 15, 1045-1047, 1972

[Maurer 68] W. D. Maurer: An improved hash code for scatter storage. CACM, Volume 11, Issue 1, p.35-38, 1968

[Maurer, Lewis 75] W. D. Maurer, T.G. Lewis: Hash Table Methods. ACM Computing Surveys (CSUR), Volume 7, Issue 1, p. 5-19, 1975

[Morris 68] R. Morris: Scatter Storage Techniques. CACM 11(1): p.38-44 (1968)

[Muehlbacher 81] J. R. Muehlbacher: Full Table Scatter Storage Parallel Searching. Computing, Volume 26, p. 9-18, 1981

[Radke 70] C. E. Radke: The use of quadratic residue search. CACM Volume 13, Issue 2, p. 103- 105, Feb 1970

[Ullmann 72] J. D. Ullmann: A Note on the Efficiency of Hashing Functions. Journal of the ACM (JACM), Volume 19, Issue 3, p.569-575, July 1972

A Appendix

Table of various prime numbers with selected primitive roots

Otherwise the order ord is listed. N.B.: ord is a divisor of p-1.
For every prime number there are exactly φ(p-1) primitive roots, which are incongruent mod p.

p \	2	3	4	5	7	8	16	32	φ(p-1)
127	7	**Y**	7	42	126	7	7	7	36
227	**Y**	113	113	**Y**	113	**Y**	113	**Y**	112
211	**Y**	**Y**	105	35	**Y**	70	105	42	48
239	119	119	119	119	**Y**	119	119	119	96
241	24	120	12	40	**Y**	8	6	24	64
509	**Y**	**Y**	254	254	**Y**	**Y**	127	**Y**	252
523	**Y**	58	261	**Y**	261	174	261	**Y**	168
1019	**Y**	509	509	509	**Y**	**Y**	509	**Y**	508
2029	**Y**	169	1014	1014	676	676	507	**Y**	624
4021	**Y**	1005	2010	1005	20	1340	1005	804	1056
8093	**Y**	**Y**	4046	1156	2023	**Y**	2023	**Y**	3264
16381	**Y**	1170	8190	4095	630	5460	4095	3276	3456
32749	**Y**	16374	16374	2729	**Y**	10916	8187	**Y**	10912
65357	**Y**	**Y**	32678	**Y**	**Y**	**Y**	16339	**Y**	32676