# ALGORITHMS FOR PRIMALITY TESTING

(By Tomasz Adamski Warsaw University of Technology)

## 3.1 Fermat’s Test

The Fermat’s algorithm for primality testing (in short Fermat’s test) is based on the following theorem called the first (or little) Fermat’s theorem.

**Theorem 3.1.1** (Fermat’s little theorem)

If *p* is a prime number and *a* is an integer which is relatively prime to *p* then,

** (3.1)

**Proof**. The Fermat little theorem is a direct conclusion from the following Euler’s theorem.■

**Theorem 3.1.2** (Euler’s theorem)

If **and *a* is an integer which is relatively prime to *n* (i.e. **) then

** (3.2)

where ** is a the Euler’s function defined in the following way: **, and for every **: **.

**Proof**. The Euler’s theorem is a direct conclusion from the following simple theorem from group theory.■

**Theorem 3.1.3**

If *G* is a finite group then for every element  ** we have **.

**Proof.** If **denotes order of the element *a* then set of powers ** is a subgroup of *G* and **. Then from the Lagrange theorem from group theory we have that **divides **. Then **divides ** i.e. there is a integer **that **but

** then also **. ■

**Corollary 3.1.4** If  , , ,  and a congruence  is not fulfilled then *n* is not a prime (then it is a composite number).

In the above corollary the assumption  can be omitted, because if  then the corollary thesis is of course fulfilled. Hence we can formulate our corollary 3.1.4 in the following way.

**Corollary 3,1.5** If  , ,  and a congruence  is not fulfilled then *n* is not a prime (then it is a composite number).

Hence we have obtained a criterion on which can be based a primality test. Such an algorithm is called the Fermat primality test and is shown on the Fig.3.1.1.

Of course even numbers *n* greater than 2 are not primes then the we can assume that we test only odd numbers.

It is worth to notice, that if for an integer  we have  then of course we have , where  denotes the remaider after division by *n*. More strictly for every :  if and only if . Hence we can verify in the Fermat test only bases .

In the Fermat primality test, the congruence  have to be verified only for bases . Indeed, because  and tested natural number *n* is odd then we have always .

For the given odd *n*, we don’t need to test such a basis  that  because in this situation *n* is not a prime. It is easy to verify that in this situation the congruence  is not true. Then our criterion correctly detects the situation when .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithm:** Fermat’s primality test

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Input data:** tested odd natural number  and security parameter defining probability of error consisting in decision „tested number *n* is a prime”, when in fact it is a composite number.

**Output data:** answer „ *n* is a composite number” or „*n* is a prime” (a probable prime)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ; **for**  **to** *t* **do** **begin**
   1. choose at random a natural number ;
   2.  ;

(it can be computed using an algorithm of fast raising to a power modulo *n*)

* 1. **if**  **then** ;

**end**

2. **if**  **then** write( „*n* is a prime”) **else** ; write( „ *n* is a composite number”);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fig. 3.1.1 Fermat’s primality test

If the Fermat test returns the answer „*n* is a composite number” then for sure *n* is a composite number. If the Fermat test returns the answer „*n* is a prime number” then it can be a mistake. The tested number *n* can efficiently „pretend” a prime i.e. more strictly a odd composite number *n* can for fixed  fulfill a congruence . This remark leads to the following definition.

**Definition 3.1.1** (Fermat’s pseudoprime number to basis *a*)

If  is an odd composite natural number, fulfilling the congruence  for such a basis , that , then *n* is called a Fermat’s pseudoprime number to basis *a*. The basis *a* is called a Fermat liar to primality for the number *n*.

A Fermat pseudoprime is often called a pseudoprime, with the modifier ‘Fermat’ being understood. A Fermat pseudoprime fulfills the thesis of the little Fermat theorem as every prime.

Sometimes, in definition of the Fermat pseudoprime *n* to basis *a*, the condition  is omitted, because fulfilling the condition  implies .

**Definition 3.1.2** (a Fermat witness of compositeness)

If  is an odd composite natural number, which doesn’t fulfill the congruence  for such a basis , that , then *a* is called a Fermat witness to compositeness for the number *n*.

**Example 3.1.1.** An odd composite number is a Fermat pseudoprime to basis , because , *n* is an odd number and , but it is not a pseudoprime to basis 2, because . ■

**Example 3.1.2** An odd composite number  is a Fermat pseudoprime to basis 2 because . The least pseudoprime to basis 2 is just 341. ■

**Theorem 3.1.4.** For every basis there are infinite number of Fermat pseudoprimes

**Proof.** See for example Koblitz [xx]. ■

It is worth to notice that if a number *n* is a pseudoprime to basis  then is also pseudoprime to a basis . Then if we want to verify if a number *n* is a pseudoprime for every possibile basis  it is sufficient to verify if *n* isa pseudoprime for every .

Examples given above proves, that psedoprimes exist. But there are also pseudiprimes for all . It means that there are odd composite numbers *n* which can’t be detected at all by our criterion. These special numbers are called Carmichael numbers.

**Definition 3.1.3** (Carmichael numbers)

A Carmichael number is a composite number  fulfilling the condition:

for every , we have: 

**Example 3.1.3** A number  is a Carmichael number. It is the least Carmichael number. Consecutive Carmichael numbers are: 1105, 1729, 2465, 2861.■

**Theorem 3.1.7** (about basic properties of Carmichael numbers)

1.There are infinite number of Carmichael numbers.

2. Every Carmichael number is odd.

3. Every Carmichael number is a product at least 3 different primes.

4. Every Carmichael number is a square-free number i.e. is not divisible by a square of a prime.

5. If is a squarefree number then *n* is is a Carmichael number if and only if,  for every prime divisor *p* of the number *n*.

6. There is such a number  that for every  a number of Carmichael numbers in the interval  is greater than .

**Proof.** See for example Koblitz [xx], Shoup [xx]. ■

Carmichael numbers are composite numbers but are qualified by the Fermat primality test always as primes.

## 3.2 Miller-Rabin Primality Test

The Fermat primality test has an evident drawback. Every Carmichael number is assessed as a prime. But the Fermat primality test can be easily improved. Such a improved version of the Fermat primality test is called the Miller-Rabin primality test. In practice the most useful probabilistic primality test is the Miller-Rabin test which is also known as strong pseudoprime test.

Assume,  and *p* is a prime and . Then we have to have  or . It is a crucial fact for the Miller-Rabin algorithm. In the sequel it will be referenced as property #1. The second important element of the algorithm are so called strong pseudoprime numbers to a basis *a*.

The thesis of the following theorem give a simple criterion of primality.

**Theorem 3.2.1**

If a number  is an odd prime and , where ,  and  *d* is oddthen for every ,  there is  that we have  or .

**Proof.** 1. If there is  for which  then  and. Hence if *p* is a prime then we have  or  (property #1).

2.If *p* is a prime fulfilling assumptions of the above theorem then from the small Fermat theorem we have:  i.e. .

Using the point #1 (i.e. property #1) to a congruence: , we obtain, that  or . If  then the thesis of our theorem is fulfilled. If  and  then the thesis of our theorem is fulfilled.

If  and  then we can use again the point #1. Hence after at most *k* steps we obtain a congruence fulfilling the thesis of the theorem. ■

**Remark.** In particular the above theorem is fulfilled for every basis .

The above theorem motivates the following definition.

**Definition 3.2.1** (a strong pseudoprime number to basis *a*)

Assume  is an odd composite number and , where  and *d* is odd and. Assume additionally that the basis  fulfills the condition  and . A number *n* is called a strong pseudoprime to basis *a* , if  or there is  that: . The integer *a* is called a strong liar to primality for *n*.

**Remark.** Of course if *n* is a strong pseudoprime to basis *a*, then

 .

Assumption that *n* and *a* are coprime (i.e. ) in the above standard definition is in fact not necessary because if  then the congruence  is not fulfilled in this case and *n* cannot be a strong pseudoprime to basis *a*.

From definition, a prime is not a strong pseudoprime (from definition a strong pseudoprime is an odd composite number) but the theorem 3.2.1 says that every odd prime number fulfills the condition from definition of a strong pseudoprime number.

**Definition 3.2.2** (a strong witness to compositness for *n*)

Assume  is an odd composite number and , where  and *d* is odd and. Assume additionally that the basis  fulfills the condition . If  and for every  we have  then *a* is called a strong witness to compositeness for *n*.

**Example 3.2.1**  Consider a number . It is a composite integer. Because then *s*=1 and *r*=45. Since  then 91 is a strong pseudoprime to base 9. For 91 the set of all strong liars to primality is the following:

{1,9,10,12,16,17,22,20,38,53,62,69,74,75,79,81,82,90}

Note that there are exactly 18 strong liars (to primality) for 91 and we have  where  is the Euler function . ■

**Example** **3.2.2** The only strong liars to primality for a composite integer 

are 1 and 104. More generally, if  and *n* is a product of the first *k* odd primes then there are only two strong liars to primality for *n*, namely 1 and *n*-1. ■

**Theorem 3.3.2**

If a number  is an odd composite number then the number *n* is strong pseudoprime to basis *a* for at most ¼ all basis . If a number  is odd composite number and  then the number *n* is strong pseudoprime to basis *a*, where  for at most  numbers from the set , where  is the Euler function.

**Proof.** see N. Koblitz [xx] i V. Shoup [xx]. **■**

From the above theorem it follows that for strong pseudoprimes there is no equivalent of Carmichael numbers.

If we have a tested number  in common binary notation then we can easily assess if *n* is even or odd. It is sufficient to verify the LSB of *n* if *n*  is odd the LSB is equal to 1 if *n* is even the LSB is equal to 0,

The Miller–Rabin algorithm is given in the fig. 3.2.1. For simplicity reason we assume that the input data are odd numbers .

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithm:**  Miller–Rabin primality test ; the first version of the Miller-Rabin algorithm

Miller–Rabin (*n, t*) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Input data**: tested odd natural number  and security parameter defining probability of error consisting in decision „tested number *n* is a prime”, when in fact it is a composite number

**Output data**: answer „ *n* is a composite number” or „*n* is a prime” (a probable prime)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1.write down a number  in the form: , where  and *r* is an odd number.

2**.for**  **to** *t* **do** **begin**

* 1. choose at random a number ;
  2. ; (using an algorithm of fast raising to a power modulo *n*)
  3. **if**  and  **then**

**begin**



**while** and  **do**

**begin**

;

**if**  **then** return(„ *n* is a composite number”);

;

**end**

**if** **then** return( „ *n* is a composite number”)

**end**

**end**

3. return(„ *n* is a prime”)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fig.. 3.2.1 Miller-Rabin primality test (the first version, “bottom to up”)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithm:**  Miller–Rabin primality test ; the second version of the Miller-Rabin algorithm,

Miller–Rabin (*n, t*)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Input data**: tested odd natural number  and security parameter defining probability of error consisting in decision „tested number *n* is a prime”, when in fact it is a composite number

**Output data**: answer „ *n* is a composite number” or „*n* is a prime” (a probable prime)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1.write down a number  in the form: , where  and *r* is an odd number.

2. **for**  **to** *t* **do** **begin**

2,1 choose at random a number ;

* 1. 
  2. 

2.4 **while**  **begin**

2.5  (using an algorithm of fast raising to a power modulo *n*)

**if**  and  **then** return („*n* is a composite number”)

**if**   **then**  **else begin** ;  **end**

**end**

**end**

3. return(„ *n* is a prime”)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fig.. 3.2.2 Miller-Rabin primality test (the second version “up to bottom”)

**Correctness of the Miller-Rabin algorithm**

If the input number *n* of the algorithm is a prime and  then according to the property #1 it behaves always like a a strong pseudoprime number and it is independent of the chosen basis .

If the input number *n* of the algorithm is a composite odd number then according to the theorem 3.2.2 for at most ¼ of all basis  the number *n* is a strong pseudoprime.

If we assume that we choose a basis *a* with a uniform probability distributions then probability of the correct compositeness detection (in one experiment) is and the probability of failure is . But the experiment of choosing a basis  with a uniform probability distribution on  is repeated *t* times in independent way. Then the probability of compositeness detection failure after *t* experiments is .

For large *t* for example we are sure from practical point of view that the tested number

*n*  is a prime.

**Definition 3.2.3** Let denotes the first *t* primes. Then is defined to be the smallest positive composite integer which is strong pseudoprime to all bases .

The numbers can interrupted as follows: to determine the primality of any odd positive integer  it is sufficient to apply the Miller-Rabin algorithm to *n* with bases *a* being the first *t* prime numbers. With this choice of bases, answers returned by the Miller-Rabin algorithm are always correct. Table from the fig. 3.2.3 gives value of  for .

The essence of the algorithm consists on verification if for basis  the tested odd number *n* behaves like a strong pseudoprime. A prime behaves always (for consecutive *t*  independent experiments consisting on randomly choosing a basis ) as a strong pseudoprime. A composite number *n*  behaves as a strong pseudoprime number (for all independent *t* experiments of choosing basis  at random with uniform probability) with probabilisty less even than . Hence probability of „fault” (i.e. acceptance of a composite number as a prime) is for large *t*  negligible small.

|  |  |
| --- | --- |
| *t* |  |
| 1 | 2047 |
| 2 | 1373653 |
| 3 | 25326001 |
| 4 | 3215031751 |
| 5 | 2152302898747 |
| 6 | 3474749660383 |
| 7 | 341550071728321 |
| 8 | 341550071728321 |

## 

Fig. 3.2.3 Smallest strong pseudoprimes 

**Computational complexity of the Miller-Rabin algorithm**

If we assume that the multiplication modulo *n* is a basic operation in the algorithm then a computational complexity of the Miller-Rabin algorithm is equal to .

The Miller-Rabin algorithm is a classical example of the so called probabilistic algorithm. As a rule probabilistic algorithms are much faster than deterministic ones solving the same problem but in the case of probabilistic algorithms the answer can be with usually very small probability false. In the case of the Miller-Rabin algorithm the answer “*n* is a composite number” is quite correct but the answer "*n* is a prime" can be false with probability less even than  where  is a so called security parameter which is equal to number of experiments (repetitions of the algorithm).

**Deterministic version of the Miller-Rabin algorithm**

There is a deterministic version of the Miller-Rabin algorithm based on the Generalized Riemann Hypothesis. It is based on the following theorem.

**Theorem 3.2.3**

If GRH hypothesis is true and  is odd composite number then the least basis *a*  detecting compositeness in the Miller-Rabin algorithm is (i.e the least so called strong witness of compositeness) is less than .

**Corrollary 3.2.4**

It is possibile to verify primality of an odd number *n* executing the Miller-Rabin algorithm for every basis . It gives a deterministic version of the Miller-Rabin algorithm which is correct under assumption that the GRH is true.

It can be proved that the computational complexity of the deterministic version of the Miller-Rabin algorithm is equal to  (of bit operations). Hence the algorithm has polynomial complexity.

**3.3 Solovay-Strassen algorithm**

A Solovay-Strassen algorithm is a probabilistic algorithm for primality testing. Description of the algorithm is given in the Fig. 3.3.1. Correctness of the algorithm is a direct conclusion from the following six theorems 3.3.1-3.3.6.

**Theorem 3.3.1**

1.A set  with multiplication  taken from the set of integers is a group.

2. If  then a set  with multiplication modulo *n* is a group and  is a subgroup of a multiplicative group .

3. A map  defined by the formula: ,  is an isomorphism of groups  and .

**Proof.** Ad 1. To prove, that  is a group it is sufficient to verify if the set  is closed for defined multiplication and the group axioms are fulfilled.

Ad 2. In similar way we verify if the set is closed for multiplication modulo *n.* For  we can easily verify, that .



Group axioms can be verified like in the point 1. We have  and  and . Indeed, if it would be:  then there are such , , , that  and . Then we have, that but it is not possible. Therefore we come to the conclusion that  then we have . Then indeed .  is a subgroup of the multiplicative group  , it follows from the fact, that the set  is finite and closed under multiplication modulo *n*.

Ad 3.The fact that the map  defined by the formula: ,  is an isomorphism of the group  and  follows from simple verification:









■

**Theorem 3.3.2**

If is an odd natural number then

1. a map  is a homomorphism of the multiplicative group  into the multiplicative group . Raising to a power in  is done in the multiplicative group  (in equivalent way  can be written as , if raising to a power is done in the ring of integers *Z*).

2. a map , (where  denotes a Jacobi symbol), is a homomorphism of the multiplicative group  into a multiplicative group .

**Proof.** Ad1. We have to show, that for every  we have . Indeed the multiplicative group  is an Abelian one then:



Then  is an endomorphism of the group .

Ad 2. In general the Jacobi symbol can take values from the set . Because the map  has arguments *a* from the set  then we have  and values of the function  belong to the set . It follows from the theorem 1 that the set  is a group with the multiplication from the ring of integers. The Jacobi symbol has the following property: for every  we have , then



Hence  is a homomorphism of the group  into the group .

**Theorem 3.3.3**

Assume *G* is a finite group, *H* an arbitrary group and  and  are two homomorphisms. If there is an element *a* of the group *G*, that  then homomorphisms  and are different in at least  points.

**Remark.** Intuition behind the above theorem is the following. Two homomorphisms defined on the finite group can be the same homomorphism or be “very different” .

**Proof.** Let’s define a set . The set  is closed under multiplication. Indeed if we have elements ,  and  then , and therefore we obtain .

If a subset of the finite group is closed under multiplication then it is a subgroup of this group. Therefore  is a subgroup of the finite group *G*.

The group  is a proper subgroup of the group *G* because there is such a element *a* of the group *G* , that , then .

It follows from the Lagrange theorem (from group theory) that  is a divisor of . But  then  . It means, that two homomorphisms  and  are different in at least  points.

**Theorem 3.3.4**

If a number  is an odd natural number then:

a number *n* is prime if and only if, for every  we have

 (3.3.1)

where  is a Jacobi symbol. If  denotes raising to a power in the multiplicative group  then the congruence (\*) can be written as an equality: .

**Proof.** 1.Implication right it is so called the Euler criterion. To prove the above implication left it is sufficient to show, that if *n* is not a prime then mappings:

 (3.3.2)

and

 (3.3.3)

are different. It follows from the theorem 3.3.2 that these two mappings are group homomorphisms. To show, that (3.3.2) and (3.3.3) are different it is sufficient to find only one such , that



2.At first, consider a case number 1 when *n* is not a square-free. It means assume that *n* has such a prime factor *p* , that  divides *n*. If , where  are such primes, that  and  then the value of the Euler’s function is the following:. Therefore, if  then .

The multiplicative group  has  elements and because  then there is a cyclic subgroup of the order *p* of the multiplicative group  . Denote a generator of this subgroup by *g*. Hence we have . The value  cannot be equal to 1 or  (i.e. ), because in this case  and we obtain that  which gives contradiction (because we cannot have  and ). Therefore we can write: . On the other hand it follows from the theorem 3.2.2, that for  we have , i.e.  or .

Hence we have found such an element , that  i.e. the congruence  is not fulfilled.

3. Consider now the second case, when *n* is a composite square-free number i.e. assume, that , where  are such primes that:  and . Assume, that there is  (we can admit for example ) that we have such , that



i.e. the congruence  is not fulfilled. In this situation we can easily find such , that: . Indeed from Chinese remainder theorem it follows that there is a standard isomorphism of the ring  and a direct sum of rings  given by the formula:



(which is written in short as ) and additionally the map *h* constrained to  is a standard isomorphism of the multiplicative group  and a direct product of groups . Then we have . The value  for  is a number *x* written in Residue Number System (RNS notation) with moduli .

Choose such an argument , for which the isomorphism *h* has the value:  i.e. . We obtain now from definition of the Jacobi symbol:



Therefore there is such an element , that (*g* indeed belongs to , because  and for every  we have ). Because  and *h* is an isomorphism then using definition of multiplication in a direct sum of rings we obtain:

 (3.3.4)

Observe, that

 oraz  (3.3.5)

We have assumed, that the congruence  is not fulfilled or equivalently .

The value of the Jacobi symbol  is even to (see theorem 3.3.2) 1 or -1. We consider both cases.

If  then it follows from the inequality , that . From the equations (3.3.5) and (3.3.4) we have, that . Because  then . Hence we have found such , that the congruence  is not fulfilled.

If  then it follows from inequality: , that . Now from equality (3.3.5) and (3.3.4) we obtain, that . Because , then . Therefore we have found such , that the congruence:  is not fulfilled.

4. Consider now the last situation, when *n* is a square-free composite number i.e. *n* is such a number, that , where  are such primes that  and . Assume that for every  and every , we have



i.e. the congruence  is not fulfilled. Then we have no „start point” as previously. We can find now a quadratic residue  modulo  i.e. find such , that and a quadratic nonresidue modulo  i.e. find such , that . It follows from Chinese remainder theorem, that there is a unique element  such that:



Raising both sides of the above equality to the power  we obtain:



But because *h* is an isomorphismwe have in the sequel:



Now, it follows from (3.3.5), that  is not equal to 1 or  (i.e. ). We have found such , that:  or equivalently, we have found such , that the congruence  is not fulfilled.

5. Finally we can say that: if an odd number *n* is not a prime then in every considered case we find such an element , that  or equivalently the congruence  is not fulfilled. ■

**Theorem 3.3.5**

If  is an odd natural composite number then for at least a half of elements  from the set  is not fulfilled the following congruence (or equivalently ).

**Proof.** 1. From the theorem 2 it follows, that the mapping  is a homomorphism of the multiplicative group  into the group  and the mapping  is a homomorphism of the multiplicative group  into the group  .

2.It follows from the theorem3.3.4, that because *n* is a odd composite number then for at least one  we have .

3. From the theorem 3.3.3 and the point 1 it follows that for at least a half of elements  from the set  we have  i.e. for at least a half of elements *a* from the set  the congruence  is not fulfilled.

**Theorem 3.3.6**

If *n* is an odd natural number and a random variable  defined on a probabilistic space  and with values in the multiplicative  (more strictly in a measurable space ) has the uniform probabilisty distribution on  then



**Proof.** The above theorem is a direct conclusion from the theorem 3.3.5. ■

Hence probability of detection that an odd natural number *n* is composite by choosing at random a number  with the uniform probabilisty distribution and verification if the conguence  is not fulfilled is greater even ½.

Probabilistic algorithm of primality testing of a natural odd number  based on the above remark is called the Solovay-Strasssen algorithm. The Solovay-Strassen algorithm is the following:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithm:** Solovay-Strassen algorithm of primality testing

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Input data:** An odd natural number and  an arbitrary chosen parameter. The parameter *t* defines probability of the errorwhen we accept a composite number as a prime, . The algorithm answer „the number *n* is composite” is always true.

**Output data:** Answer if a number*n* is a composite one or a prime. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**for**  **to** *t* **do**

**begin**

1.we choose at random a number  with the uniform probability distribution on 

2. ; // we compute the value of the Jacobi symbol 

3. ; // we compute the value 

4. **if**   **then** **begin** *write* („the number *n* is composite”); **goto** finishing\_label **end** ;

// we verify if the congruence:  is fulfilled

**end;**

*write* („the number *n* is a prime”);

finishing\_label:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Fig. 3.3.1 Solovay-Strassen algorithm of primality testing