Genetic Algorithms As a Random Passwords Generator

Ali Shaker Mahmood
Al-Mustansiriyah University
College of Education
Computer Science Department

Abstract

Genetic algorithms has become a suitable searching or optimization tool for solving many complex problems comparing with the traditional search techniques, Genetic algorithms contains many manipulations to speed up and improve the genetic algorithms performance; these manipulations are selection, crossover and mutation.

In this work, the genetic algorithms are used as a random password generator, where the basic process of genetic algorithms and three statistical tests are used to ensure the outcome sequences of passwords are acceptable.

The genetic algorithms proved highly successful in generating a random password with good statistical properties and a high linear complexity.

المستخلص

تعتبر الخوارزمية الجينية أداة مناسبة من طرق البحث أو طرق تحقيق الأمثليه لحلّ العديد من المشاكل المعقدة في الوقت الحاضر مقارنة بخوارزميات البحث التقليدية، تحتوي الخوارزمية الجينية على العديد من العمليات لتحسين وتسريع أداءها، وهذه العمليات هي عملية الاختيار (Selection)، عملية التزاوج (Crossover) وعملية التهجين (Mutation).

تم في هذا البحث استخدام الخوارزميات الجينية كمولد لكلمات المرور العشوائية حيث تم استخدام العمليات الأساسية الخاصة بالخوارزمية الجينية إضافة إلى استخدام ثلاثة اختبارات

إحصائية معروفة للتأكد من جودة السلاسل الناتجة وقوة النظام المستخدم، أثبتت هذه الطريقة نجاحها وكفاءتها العالية في توليد كلمات المرور العشوائية.

1. Introduction

Genetic algorithms are general search algorithms based upon the principles of evolution observed in nature.

The solution of the problem is a collection of genes, which are simply the parameters to be optimized. A genetic algorithms creates an initial population; evaluates this population according to some criteria (fitness function), and then selects the individual according to the selection schemes, and mate (recombine) to form a new population **[1]**.

To make the genetic algorithms work well, the user must specify the number of parameters such as the population size, selection pressure, crossover rate and mutation rate **[2]**.

The genetic sequence is repeated until one or more of the following conditions are reached: proper solution is found, specific number of generations is reached and individuals in population are the same or no improvement is done on the population **[3]**.

The proposed system is used for data base and sites administrators to produce a sequence of different passwords that may be combined into sequences or blocks of random numbers to be used.

2. Elements of Genetic Algorithms

The genetic algorithms contains many basic elements these elements which are:

2.1 Encoding scheme [4]

First step needed before applying genetic algorithms is to create a coding scheme. A coding scheme is a method for expressing a solution in a string. There is no mechanical technique for creating one.

2.1.1 Value Encoding

Direct value encoding can be used in problems, where some complicated values, are used.

In value encoding, every chromosome is a string of some values. Values can be anything connects to the problem like real numbers, characters and some complicated objects.

Chromosome A	A & 2 D e b E * G I 9 F p B 6
Chromosome B	1.2324 5.3243 0.4556
Chromosome C	(back), (right), (left), (forward)

The type of the coding scheme depends on the problem, but the thing that must be keep in mind is that the genetic machinery will manipulate a finite representation of the solution, not the solutions themselves.

2.2 Fitness Function [5]

To solve a problem, some means or procedures must be used to discriminate good solution from bad solution. A fitness function returns a single numerical fitness value, which is proportional to the ability, of the individual represented by that chromosome and better chromosomes are assigned higher fitness function values.

The problem is used equation (1) as a fitness function.

$$\text{Fitness Function} = \frac{1}{\text{Number of (lowercase + Uppercase + Digits + Other)}} \quad \ldots (1)$$

The above equation computing the sum of occurrence of lowercase, uppercase, digits and other character then divided the sum by one and consider this number as fitness value.

2.3 Selection [6]

During this phase of genetic algorithms, individuals are selected from the population, according to their fitness values, to produce offspring, which will make up the next generation. Good individuals

will probably be selected several times in a generation; poor ones may not be selected at all. The goal of any selection method is to favor the reproduction of good individuals in the population.

2.3.1 Rank Selection

This version proposed by Baker (1985), the individuals in the population was ranked according to fitness, and the expected value of each individual depends on its rank rather than on its absolute fitness.

In ranking method, ignoring the actual object function values, instead, it uses a ranking of chromosomes to determine survival probability. The idea is straightforward: sort the population from best to worst (The best individual receives rank 1; the second best receives 2 and so on) and assign the selection probability of each chromosome according to the ranking but not its raw fitness.

There is a drawback in this selection scheme the genetic algorithms in the same cases be slower in finding highly fit individuals.

The most popular rank based selection scheme is linear rank, after ranking individuals in the population from 1 to population size applied the equation (2).

$$F_{new} = \text{Maxfit} - (\text{Maxfit} - \text{Minfit}) * \frac{\text{Rank (i)} - 1}{N - 1} \quad \text{... (2)}$$

where:

N: population size

Maxfit: expected value of individual with rank 1.

Minfit: expected value of individual with rank N.

There is another ranking method like inverse rank and exponential rank.

2.4 Crossover [7]

It is a recombinant operator that takes two individuals and combines them to form two new solutions (offspring). Crossover is not necessarily applied to all pairs of individuals selected for mating. A choice is made, depending on a probability specified by the user. If crossover is not applied; the offspring are simply duplications of the parents.

2.4.1 Two Point Crossover

It seems like a single point crossover with two exceptions, two crossover points are selected randomly at the same time, all the cells are swapped between the two strings, and the swapped operation generates two new strings.

Chromosome A	A & 2 D e b E * G I 9 F p B 6
Chromosome B	B 6 p F 9 T H = W l C s 7 # Y
Child A	A & 2 D 9 T H = W l C s p B 6
Child B	B 6 p F e b E * G I 9 F 7 # Y

2.5 Mutation [8]

Mutation is a genetic operator that alters one or more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithms may be able to arrive at better solution than was previously possible.

Mutation is an important part of the genetic search process as it helps to prevent the population from stagnating at any local optima. Mutation occurs according to user-definable probability.

2.5.1 Order Changing Mutation

Select two numbers randomly and then exchange between them,

Chromosome A	J 4 d N e b E * G I 9 F 7 # Y
Chromosome A'	J 4 d 9 e b E * G I N F 7 # Y

3. Passwords Standard Tests [9], [10] and [11]

Before starting with the actual system, various tests are executed on the obtained chain of the passwords to ensure that is excellent to use, these tests includes a letter frequency analysis, character type analysis and length distribution analysis.

3.1 Letter Frequency Test

The work of this test is to knowing the frequency of each letter that is give us the ability to define a finer grained metric for measuring password strength. By grading the chance of occurrence of each individual character rather than grading each letter a flat twenty sixths chance of occurrence.

This test allow to measure the degree in which the passwords conform to actual words. This helps us to better understand if the chain of the passwords follows a language and what language this might be.

Figure (1) shows the letters frequency of the English language, this table differs slightly from others, where produced this table after measuring 40,000 words, that is clearly the passwords chain must be great similarity to the English letters frequency.

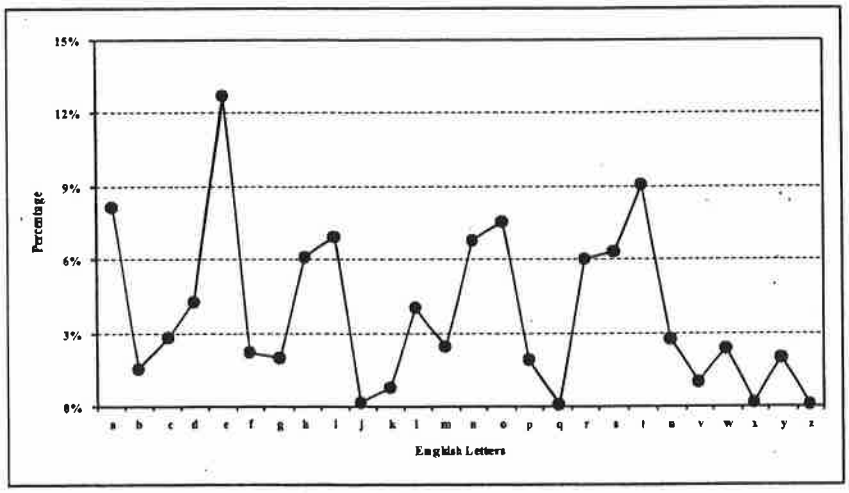

Figure (1) English Letters Frequency

3.2 Character Type Test

Looks at every password, what kinds of characters make up the password, which can be distinguish between the following character types:

- Lowercase, contains the standard lowercase letters of the alphabet.
- Uppercase, contains the standard uppercase letters of the alphabet.
- Digits, which are the digits from zero to nine.
- Symbols, which are any characters found in the non-extended ASCII set that do not belong to the above categories. Most of these can be found on a standard keyboard.
- Unicode, which are any characters that do not belong to the above categories.

Figure (2) shows the standard character type analysis. The largest category is passwords that consist solely out of lowercase characters. This is troublesome, considering that most passwords are only 6 to 8 characters long and the letters found in the passwords conform to that of a language.

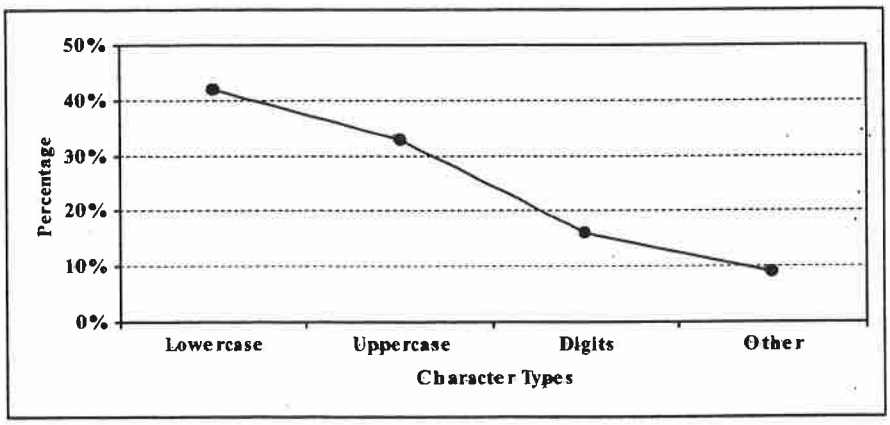

Figure (2) Character Type Analysis

3.3 Length Distribution Test

The common lengths of user chosen passwords are illustrated in Figure (3).

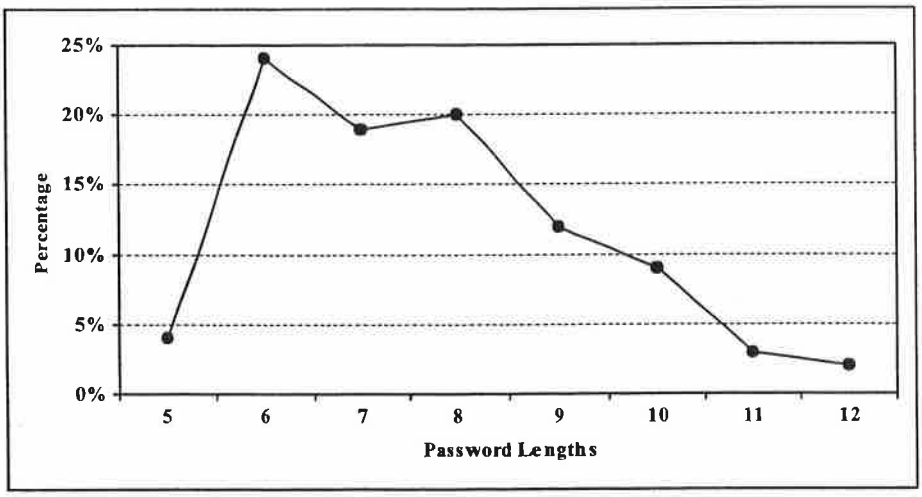

Figure (3) Common Password Lengths

As obvious in Figure (3), do not show a normal form distribution, but rather a truncated form. We argue that this is a result of minimum password length requirements. Strangely, the passwords file contained entries as short as one character.

4. The Proposed System

From functional point of view, the designed system mainly use a genetic algorithms as a tool to generate a chain of random passwords, in the fitness part there are two tests are used to improve the population quality, See Figure (4)

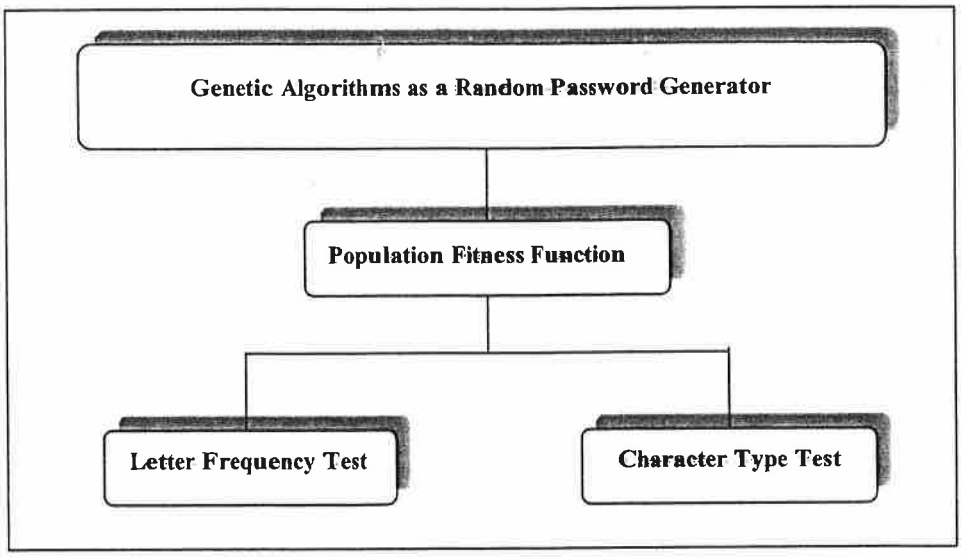

Figure (4) Proposed System

The following are standard steps used in genetic algorithms

Step 1: Creating an Initial Population

Create Initial population randomly, which used a sets of (a...z, A...Z, 1...9 and special characters), where the number of chromosome is 100 and each one have 10 characters.

Step 2: Compute Fitness Value

For each individual in the population compute fitness value, this is done by computing the value of equation (1), Other fitness value are computed for all population by using two standard tests of password as a population fitness function, When the population pass from these two tests is defined as an acceptable and save in file.

Step 3: Checking the Stopping Condition

If generation number reached to 500 iteration, then stop else go to next step.

Step 4: Perform Selection Operation

Perform the operation of selection scheme where the Rank Selection Algorithm is used.

Step 5: Perform Crossover Operation

There are many methods to do crossover operation; Two Point Crossover is used, with crossover rate 0.85.

Step 6: Perform Mutation Operation

The used method to do mutation operation is Order Changing Mutation, with mutation rate 0.01. After performing mutation operation the new population is ready for the next generations therefore that increments the generation number by one and goes to step 2.

All previous steps are illustrated in Figure (5).

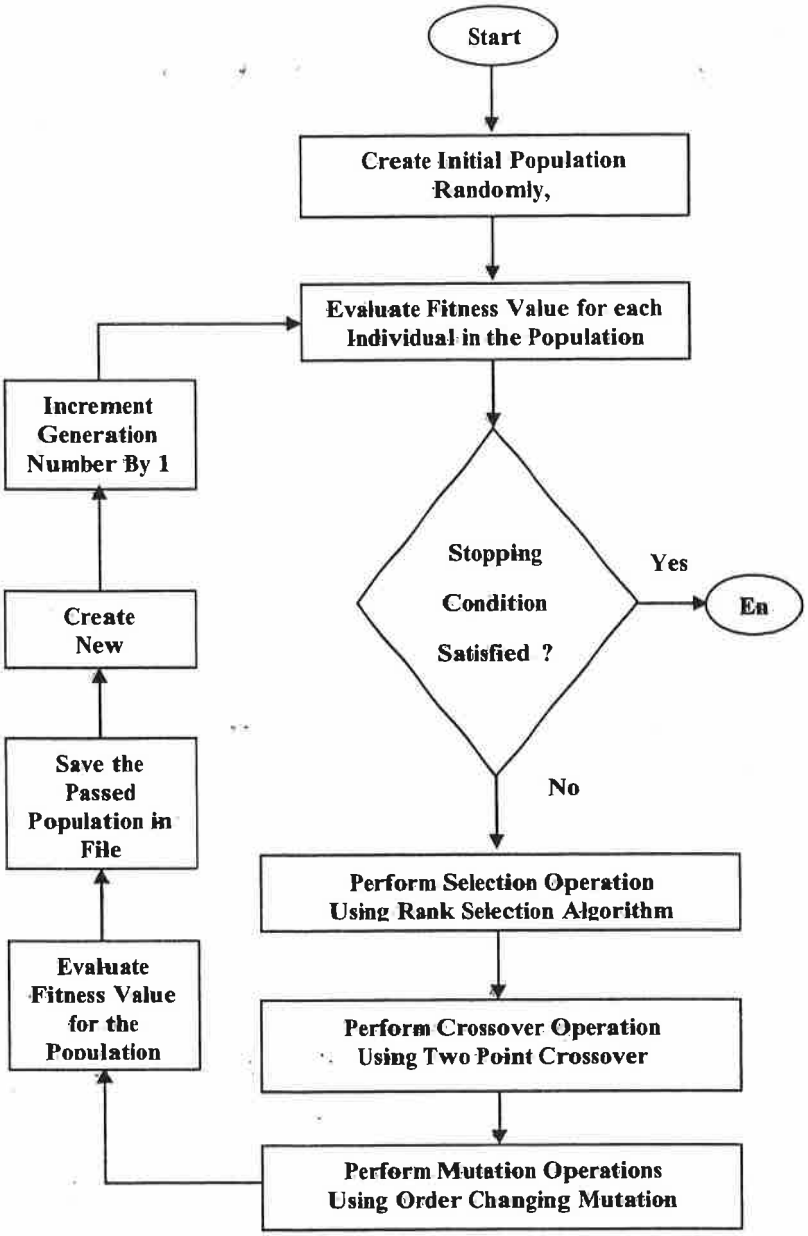

Figure (5) Flowchart of Genetic Algorithms as a Random Password Generator

5. System Results

The stopping condition is satisfied when the number of generations reached to 500 generation. The passed population is storing in file, which is mean, the system can generate approximately less than 50,000 random characters for each run.

5.1 Letter Frequency Test

This test count the occurrence of each letter in the saved file gained from the system, Figure (6) shows the letters frequency of the English language and the letter frequency of the system passwords chain that is clearly the passwords chain frequency is similar to the English letters frequency.

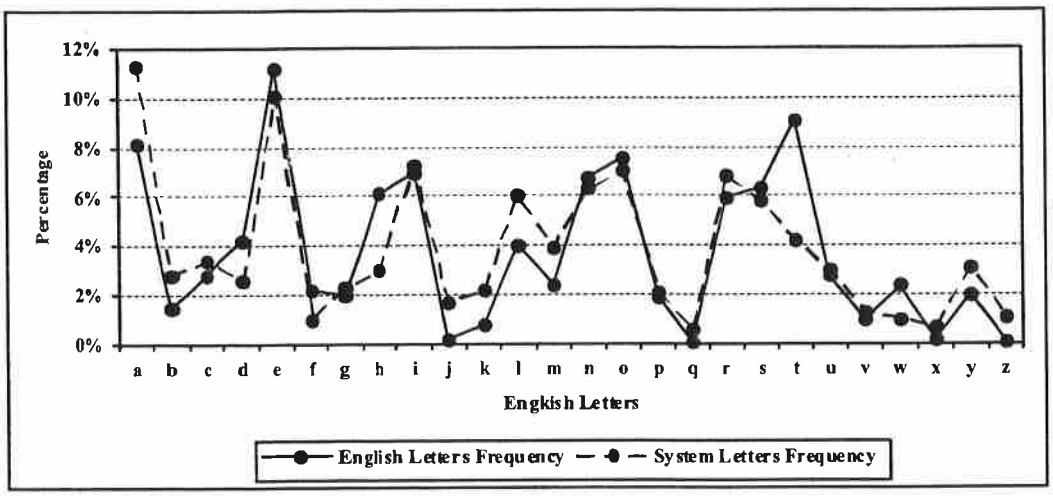

Figure (6) English and System Letter Frequency Test

5.2 Character Type Test

The character type test put the save file in four known category (lowercase, uppercase, digits and other), Figure (7) shows the standard character type and system character type. The largest categories in both are solely out of lowercase characters.

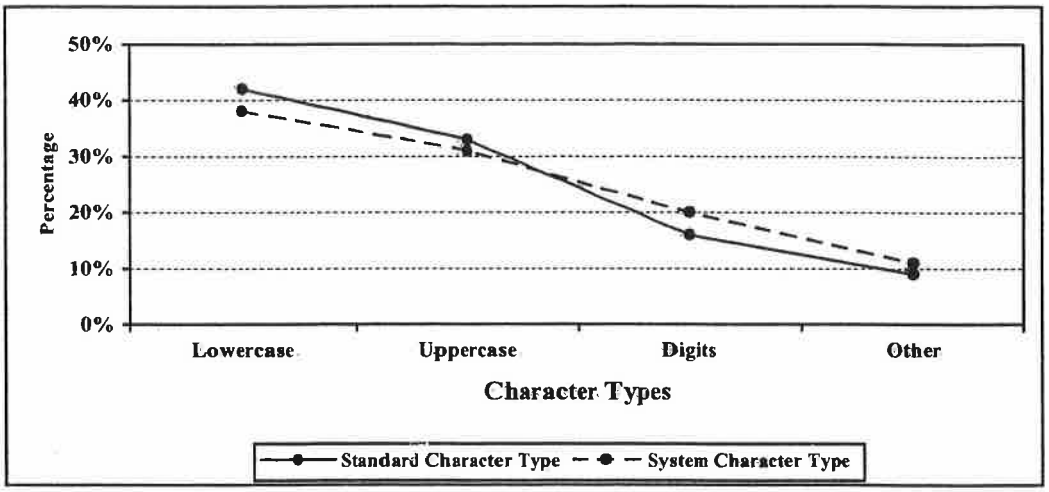

Figure (7) English and System Character Type Test

5.3 Length Distribution Test

The passwords generated by the system are not having certain length because it chain of character, the user can cut this chain and take the desire length of passwords character, but the common passwords length are between 6 to 8 characters.

6. Conclusion and Future Work

1. This paper has developed and tested the use of genetic algorithms as a random passwords generator, this was a achieved by two steps:
    a. Using of genetic algorithms operators which are selection, crossover and mutation methods.

b. Using of the known three standard tests for check the quality of produced chain of passwords.

2. The goal was achieved where the genetic algorithms proved highly successful in generating random passwords with good statistical properties.
3. The proposed system used English letters, digits plus special characters. In future other languages can be also used.
4. Optimization of control parameters that are used in genetic algorithms has been made.
5. This work suggests that a new approach is needed and tested in addition to genetic algorithms such as Neural Network, to be a standby method for generating random passwords.

7. References

1. Goldberg D., "*Genetic Algorithm in Search, Optimization and Machine Learning*", 1989, Addison Wesley Longman, USA.
2. Whilley L. and Vase M., "*Foundations of Genetic Algorithms*", 1995, Morgan Kaufmann Publishers, USA.
3. Winter G., Periaux J. and Galan M., "*Genetic Algorithms in Engineering and Computer Science*", 1995, John Wiley and Sons, USA.
4. Mitchell M., "*An Introduction to Genetic Algorithms*", 1996, MIT Press, England.
5. Koza J., "*Genetic Programming: On the Programming of Computers by Means of Natural Selection*", 1993, MIT Press, England.
6. Michalewicz Z., "*Genetic Algorithms + Data Structures = Evaluation Programs*", 1996, Springer-Verlag, Germany.
7. De Jong K. and Spears W., "*A Formal Analysis of the Role of Multi-point Crossover in Genetic Algorithms*", Annals of Mathematics and

Artificial Intelligence Journal, 1992, Scientific Publishing Company, Switzerland.

8. Martin S. and James S., "*Self Adaptation of Mutation Operator and Probability for Permutation Representations in Genetic Algorithms*", Technical report, P/491-514, Massachusetts Institute of Technology, England, 2010.
9. Elaine B. and Kelsey J., "*Random Number Generation Using Deterministic Random Bit Generators*", 2007, National Institute of Standards and Technology Special Publication, USA.
10. Dinei H., "*A Large Scale Study of Web Password Habits*", 2007, Proceedings of ACM Conference on Computer and Communications Security Identification and Protection Switzerland.

Maurer U., "*Universal Statistical Tests for Random Bit Generators*", 1998, Springer-Verlag, Switzerland.