ივანე ჯავახიშვილის სახელობის თბილისის
სახელმწიფო უნივერსიტეტი

დავით ჯიქია

მონაცემთა წარმოდგენის მეთოდები საცავებში (ქვანტური კომპიუტერის მოდელზე)

სამაგისტრო ნაშრომი შესრულებულია საინფორმაციო ტექნოლოგიების მაგისტრის აკადემიური ხარისხის მოსაპოვებლად

ხელმძღვანელი: პაატა კერვალიშვილი

თბილისი

2013

ანოტაცია

ნაშრომში განხილულია ინფორმაციული ტექნოლოგიების ერთ-ერთი ყველაზე მნიშვნელოვანი საკითხი, რომელიც შეეხება მონაცემთა წარმოდგენის მეთოდებს საცავებში. წარმოდგენილია ფორმატები, რომლებითაც ხდება ინფორმაციის წარმოდგენა კომპიუტერში. აგრეთვე განხილულია მონაცემთა საცავები, მათი ტიპები, ფუნქციონალურობა და საცავების იერარქია.

უკანასკნელ წლებში განხორციელებული კვლევები ქვანტური კომპიუტერის არსებობას სულ უფრო რეალურს ქმნია, ამიტომ ნაშრომში განხილულია ქვანტური კომპიუტერი, ქვანტური სქემა, ქვანტური ინფორმაციის წარმოდგენა, ქვანტური ინფორმაციის ერთეული ქუბიტი, მოცემულია განსხვავებები ქვანტურ და კლასიკურ კომპიუტერს შორის. გაანალიზებულია ქვანტური ალგორითმები და არჩეულია მათ შორის ოპტიმალური და მის საფუძველზე ნაჩვენებია კლასიკური კომპიუტერში ინფომრაციის წარმოდგენის ქვანტური მოდელი, რომელიც შეიძლება საფუძვლად დაედოს მონაცემტა საცავებს ქვანტურ კომპიუტერებში.

Anotation

There is examined one of the most important issues of informational technologies in this work,which is about data representation methods in storage. There is examined formats,by which represents information in computer. There is also examined data storages, there types, how they function and hierarchy of storages.

Researches held in recent years make existence of quantum computers more real, that's why there is examined quantum computer in this work, quantum circuit, presentating of quantum information, unit of quantum information- qubit, there is given differences between quantum and classical computers. There is analyzed quantum algorithms and is chosen the most optimal and on basis of that there is shown quantum model of presenting information in classical computer, which can be the basis of data storages in quantum computers.

შინაარსი

შესავალი

მონაცემთა წარმოდგენა მიეკუთვნება მეთოდებს რომელიც გამოიყენება შიდა ინფორმაციების შეგროვებისთვის კომპიუტერში. კომპიუტერები ინახავს ბევრ განსხვავებული ტიპის ინფორმაციას.

- რიცხვებს
- ტექსტებს
- გრაფიკულ ნაირსახეობებს(ანიმაციებს, ვიდეოებს)
- ხმოვან ინფორმაციებს

აღნიშნული ინფორმაციები გვეჩვენება განსხვავებულად. თუმცა, ყველა ტიპის ინფორმაცია კომპიუტერში ინახება ერთი და იგივე მარტივი ფორმატით: ნულებისა და ერთიანების თანმიმდევრობით. როგორ შეიძლება წარმოვადგინოთ ნულებისა და ერთიანების თანმიმდევრობის სახით ფოტოსურათი, სიმღერა და ფილმი? ეს ყველაფერი დამოკიდებულია იმაზე თუ როგორ განვმარტავთ ჩვენ ამ ინფორმაციას. კომპიუტერი იყენებს რიცხვით კოდებს, მასსში შენახული ინფორმაციების წარმოსადგენად. ამ კოდების მეშვეობით ნებისმიერი დაწერილი შეტყობინება შეიძლება იყოს წარმოდგენილი რიცხობრივად. ეს კოდები დაფუძნებულია თვლის ორობით სისტემაზე ნაცვლად ათობითი სისტემისა. კომპიუტერები იყენებს ბევრ განსხვავებულ კოდს. ზოგიერთი მათგანი გამოიყენება რიცხვებისთვის, ზოგიერთი ტექსტისთვის, ხმებისთვისა და გრაფიკებისთვის.

მონაცემთა წარმოდგენის ხერხები და მეთოდები კლასიკურ კომპიუტერებში

ორობითი რიცხვები

ძირითადად ჩვენ ვწერთ რიცხვებს ნულიდან ცხრამდე ციფრების გამოყენებით. თუმცა ნებისმიერი დადებითი მთელი რიცხვი შეიძლება ადვილად წარმოადგინო ნულებისა და ერთიანების გამოყენებით. ამ ფორმის რიცხვებს ეწოდება ორობითი რიცხვები.

თვლის ორობითი სისტემა აგებულია პოზიციურ პრინციპზე 2-ის ფუძით. ამ სისტემაში იყენებენ მხოლოდ ორ ნიშანს - ციფრებს 0 და 1; აქაც, ისევე როგორც ყოველ პოზიციურ სისტემაში, ციფრის მნიშვნელობა დამატებით დამოკიდებულია მის მიერ დაკავებულ ადგილზე. რიცხვი 2 ითვლება მეორე თანრიგის ერთეულად და ჩაიწერება ასე: 10 (იკითხება: „ერთი, ნული“). შემდეგი თანრიგის ყოველი ერთული ორჯერ მეტია წინაზე, ე. ი. ეს ერთეულები ადგენენ რიცხვთა მიმდევრობას: 2, 4, 8, 16, ..., 2^n, ... იმისათვის, რომ ათობით სისტემაში ჩაწერილი რიცხვი ჩაიწეროს ორობით სისტემაში, მას მიმდევრობით ყოფენ 2-ზე და მიღებულ ნაშთებს (0 და 1) ჩაწერენ რიგით ბოლოდან პირველისაკენ. მაგ., 27=13·2+1; 13=6·2+1; 6=3·2+0; 3=1·2+1; 1=0·2+1; ამრიგად 27-ის ორობითი ჩაწერა იქნება 11011. ამ სისტემაში განსაკუთრებით მარტივად სრულდება ყველა არითმეტიკული მოქმედება: მაგ., გამრავლების ტაბულა დაიყვანება ერთ ტოლობამდე 1·1=1, მაგრამ რიცხვების ჩაწერა მოითხოვს ციფრების დიდ რაოდენობას. მაგ., რიცხვი 7000 იქნება 13-ნიშნა.

არითმეტიკა ორობით რიცხვებში

შეკრება

ეს ოპერაცია უმარტივესია ორობით არითმეტიკაში. იგი ეყრდნობა 4 ძირითად ტოლობას:

```
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10
```

მათი გათვალისწინებით შეგვიძლია ქვეშმიწერით შევკრიბოთ (ათობითი რიცხვების მსგავსად) ნებისმიერი ორობითი რიცხვები, მაგალითად, 01101_2 და 10111_2:

```
   1 1 1 1 1     (<- ვიმახსოვრებთ)
     0 1 1 0 1
+   1 0 1 1 1
------------------
  1 0 0 1 0 0 (<- შედეგი)
```

გამოკლება

ძირითადი ტოლობებია:

```
   0 − 0 = 0
   0 − 1 = 1 (ვიმახსოვრებთ (-1)-ს)
   1 − 0 = 1
   1 − 1 = 0
```

მათი გათვალისწინებით შეგვიძლია ქვეშმიწერით გამოვაკლოთ (ათობითი რიცხვების მსგავსად) ნებისმიერი ორობითი რიცხვები, მაგალითად, 1101110_2 და 10111_2:

```
       -1 -1 -1 -1   (<- ვიმახსოვრებთ)
    1 1 0 1 1 1 0
 -        1 0 1 1 1
-----------------------
     1 0 1 0 1 1 1
```

გამრავლება

ძირითადი ტოლობებია:

```
0 • 0 = 0
0 • 1 = 0
1 • 0 = 0
1 • 1 = 1
```

მათი გათვალისწინებით შეგვიძლია ქვეშმიწერით გავამრავლოთ (ათობითი რიცხვების მსგავსად) ნებისმიერი ორობითი რიცხვები, მაგალითად, 1011_2 და 1010_2:

```
             1 0 1 1
         ×   1 0 1 0
         ---------------
              0 0 0 0
  +       1 0 1 1
  +    0 0 0 0
  + 1 0 1 1
  ------------------
     1 1 0 1 1 1 0
```

გაყოფა

გაყოფა ორობით რიცხვებში ანალოგიურია გაყოფისა ათობით რიცხვებში.
მაგალითად, გავყოთ 11011_2 ($=27_{10}$) 101_2 ($=5_{10}$) -ზე:

```
   1 1 0 1 1 : 1 0 1 = 1 0 1
- 1 0 1
  --------
      0 1 1
 -    0 0 0
       -------
         1 1 1
  -    1 0 1
        -------
           1 0
```

ტექსტი

ტექსტი შეიძლება წარმოდგენილი იქნას მარტივად, ტექსტის ყოველი სიმბოლოსათვის უნიკალური რიცხვითი მნიშვნელობის მინიჭებით. მაგალითად ფართოდ გამოყენებადი ASCII code(American Standart Code For information Interchange) რომლის მიხედვითაც კომპიუტერში სიმბოლოები წარმოდგენილი არის რიცხვების მეშვეობით, ის განსაზღვრავს 128 სხვადასხვა სიმბოლოს და აღნიშნავს თითოეულ უნიკალურ რიცხვით კოდებს 0-დან 127 -ის ჩათვლით. ASCII კოდში “A” -ს შეესაბამება 65, “B” შეესაბამება 66, “a” - 97, “b” - 98 და ა.შ. ASCII ფორმატი იყენებს 1 ბაიტს ყოველი სიმბოლოსთვის, მაგრამ ერთი ბაიტი გვაძლევს 256(128 სტანდარტულ და 128 არასტანდარტულ) შესაძლებელ სიმბოლოებს.

გრაფიკა

გრაფიკა რომელიც გამოისახება კომპიუტერის ეკრანზე შედგება პიქსელებისგან: გრაფიკულ გამოსახულებას კომპიუტერის ეკრანზე ჰქმნის პატარა ფერადი წერტილების ერთობლიობა. კომპიუტერის ეკრანზე გამოსახულ მწკრივებს ორგანიზებაში მოჰყავს პიქსელი. გავრცელებული კონფიგურაციით თითო მწკრივი არის 640 პიქსელის სიგრძის, და მასში არის 480 ასეთი მწკრივი. კიდევ ერთი კონფიგურაციის მიხედვით მასში არის 600 მწკრივი და

თითო მწკრივი მოიცავს 800 პიქსელს, რომელიც იძლევა გაფართოებას 800x600. პიქსელს გააჩნია ორი თვისება: მდებარეობა ეკრანზე და ფერი.

გრაფიკული გამოსახულება შეიძლება წარმოდგენილი იქნეს პიქსელის სიის სახით. წარმოვიდგინოთ რომ ყველა პიქსელის მწკრივი ეკრანზე ასახავს ერთ მთლიან გრძელ მწკრივს. ეს გვაძლევს პიქსელის სიას და პიქსელის ადგილმდებარეობა სიაში შეესაბამება მის მდებარეობას ეკრანზე. პიქსელის ფერი გამოისახება ორობითი კოდით და შეიცავს ბიტების იმავე რაოდენობას. ერთფეროვან გამოსახულებას (შავ-თეთრს) მხოლოდ ერთი ბიტი სჭირდება ერთ პიქსელზე: 0 შავს და 1 თეთრს . თექვსმეტ ფერიან გამოსახულებას ესაჭიროება 4 ბიტი ერთ პიქსელზე . თანამედროვე ჩვენებას ესაჭიროება 24 ბიტი ერთ პიქსელზე რმელიც უზრუნველყოფს 16,7 მილიონ შესაძლო ფერს თითო პიქსელზე.

მონაცემების შეკუმშვა/ოპტიმიზაცია (Compression)

დღესდღეობით ფაილები ინფორმაციულად იმდენად მდიდარია რომ მისი ზომა საგრძნობლად გაიზარდა, განსაკუთრებით ხაზგასასმელია გრაფიკული ფაილები. უამრავი პატარა ნაწილაკს პიქსელში გრაფიკული ფაილი მყარ დისკზე იტევს. ფაილები რომლებიც შეიცავენ დიდი რაოდენობით კომპიუტერული პროგრამების აპლიკაციებს, ესაჭიროებათ 50 ან მეტი მეგაბაიტი! ეს იწვევს ორ პრობლემას: ფაილების განთავსება ძვირადღირებულია (ესაჭიროება ბევრი floppy disk ან გადაჭარბებული ტევადობა მყარ დრაივზე.) ასევე ძვირადღირებულია ტელეფონის ხაზებსა და ქსელებში მისი გადაცემა, რადგანაც ეს პროცესი დიდ დროსა და რესურსს საჭიროებს. სხვადასხვა ტიპის მონაცემების შესწავლისას, ჩქვენ გვაქვს შესაძლებლობა გადავხედოთ ტექნიკას რომელიც მონაცემთა შეკუმშვის(დაპატარავების) სახელით არის ცნობილი. მთავარი აზრი შეკუმშვისა ფაილის დაპატარავებაა მისი ნამატების (ნაწილაკების განმეორებული ნიმუშების) მოშორებით. ეს ფაილები რათქმაუნდა უნდა დაკომპრესდეს_ნამატები მოშორდეს_იმისათვის რომ შესაძლებელი იყოს მისი გამოყენება. თუმცა იგი შეიძლება განთავსდეს მისი შემოკლებული ფორმით, დროისა და ფულის დაზოგვის მიზნით.

კომპიუტერული მონაცემთა საცავი

საცავი ან მეხსიერება არის ტექნოლოგია რომელიც შეიცავს კომპიუტერულ კომპონენტებს და იძლევა ჩაწერის საშუალებას, აგრეთვე გამოიყენება ციფრული მონაცემების შესანახად. ეს არის ბირთვი, კომპიუტერის ძირითადი კომპონენტი. კომპიუტერის CPU(central processing unit) უზრუნველყოფს ძირითადი ლოგიკური და არითმეტიკული, ასევე შეტანა გამოტანის ოპერაციების შესრულებულას პროგრამული ინსტრუქციებს მიხედვით. პრაქტიკულად, ყველა კომპიუტერი იყენებს საცავების იერარქიის მეთოდს, მეხსიერების იერარქია გამოიყენება კომპიუტერის არქიტექტურაში მისი პროექტირების შესრულების პრობლემის განხილვისას. ხშირად სწრაფი და არასტაბილური ტექნოლოგიები (ინფორმაციის დაკარგვა გამორთვის შემთხვევაში) მოხსენიებულია, როგორც "მეხსიერება" ხოლო ნელი, მუდმივი ტექნოლოგიები მოხსენიებულია როგორც “საცავი”, მაგრამ ეს ტერმინები აგრეთვე არის ურთიერთ ჩანაცვლებადი. Von Neumann ის არქიტექტურაში, CPU შეიცავს ორ ძირითად ნაწილს: კონტროლის ერთეულს (UNIT) და არითმეიკულ ლოგიკურ ერთეულს (ALU). წინანდელი კონტროლი მონაცემებზე CPU- სა და მეხსიერებას შორის განისაზღვრება არითმეტიკული და ლოგიკური ოპერაციებით მონაცემებზე.

ფუქციონალურობა

დიდი ოდენობის მეხსიერების გარეშე, კომპიუტერი მხოლოდ შეძლებდა შეესრულებინა ფიქსირებული ოპერაციები და დაუყოვნებლივ მოეცა პროდუქციის შედეგი. მაგალითად ეს არის ხელმისაწვდომი მოწყობილობებისთვის როგორიცაა მაგიდის კალკულატორები, ციფრული სიგნალის პროცესორები და სხვა სპეციალური მოწყობილობები. Von Neumann ის მოწყობილობები განსხვავდება მეხსიერებით რომელიც იმარაგებს მოქმედ ინსტრუქციებს და მონაცემებს. ასეთი კომპიუტერები არის უფრო მრავალფეროვანი იმიტომ რომ მათ არ სჭირდებათ hardware ს რეკონფიგურაცია სათითაოდ ახალ პროგრამაზე, მაგრამ შესაძლებელია მარტივად რეპროგრამირება ახალი მეხსიერების ინსტრუქციებით. ისინი აგრეთვე მიმართავენ მარტივ და გასაგებ დიზაინსა და შედარებით მარტივ პროცესორს. თანამედროვე კომპიუტერები იყენებენ von Neumann ის მოწყობილობებს.

საცავების იერარქია

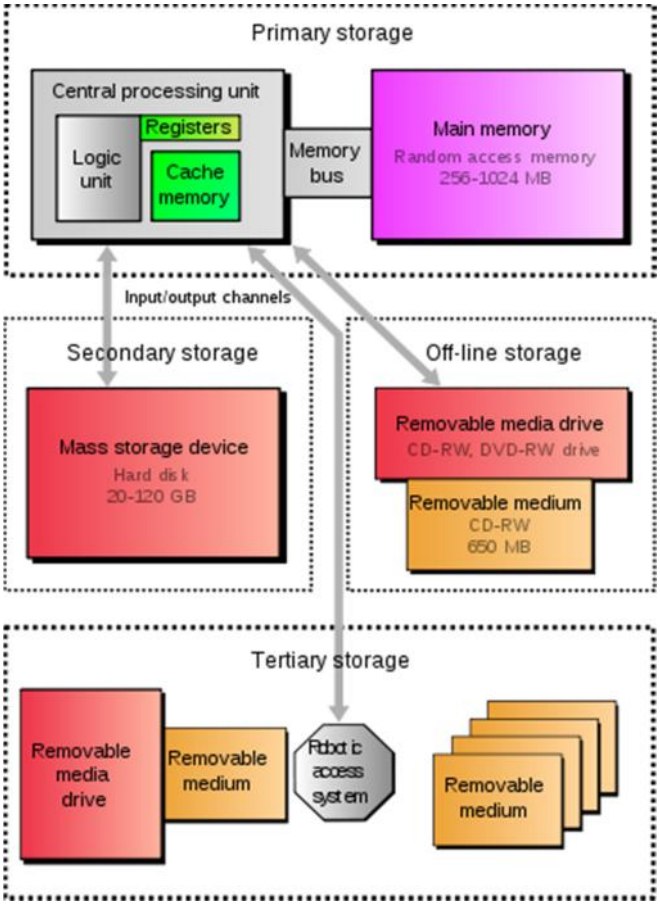
კომპიუტერების მეხსიერების და მისი ტევადობის ზღვარი, ცნობილია როგორც საცავების იერარქია. რომ გავიგოთ თუ რა არის ეს სიღრმისეულად შეგვიძლია ის დავყოთ 4 სეგმენტად. ზოგადად, რაც უფრო დაბლაა საცავი იერარქიაში, მით უფრო ნაკლებია მისი გამტარუნარიანობა და უფრო ხელმისაწვდომი ხდება პროცესორისთვის. საცავის ტრადიციული დაყოფა მთავარ, მეორეულ, მესამეულ და გარე საცავებად ხდება ბიტების მეშვეობით. დღევანდელ გამოყენებაში, მეხსიერებას ზოგადად უწოდებენ ნახევრადგამტარ საცავს, იგივე დინამიურ ოპერატიულ მეხსიერებას ან უბრალოდ სხვა ფორმის სწრაფ, მაგრამ დროებით საცავს. საცავი შედგება სხვადასხვა მოწყობილობისგან, რომელთა შემადგენლობა პროცესორისთვის პირდაპირ ხელმისაწვდომი არაა, (მეორეული ან მესამეული საცავი), ტიპიური მყარი დისკი, ოპტიკური დისკი, და სხვა შედარებით ნელი მოწყობილობები ვიდრე ოპერატიული მეხსიერება, მაგრამ ამავდროულად არა-ცვალებადი (ინარჩუნებს ყველაფერს ელექტრო-ენერგისს გათიშვის შემთხვევაშიც). ისტორიულად, მეხსიერებას ეძახდნენ ბირთვს, მთავარ მეხსიერებას, ნამდვილ საცავს ან შიგა მეხსიერებას. საცავის მოწყობილობებს კი აიგივებდნენ მეორეულ, გარე, დამხმარე საცავთან.

პირველადი საცავი არის უმაღლესი დონის და შედგება cpu ს რეგისტრებისაგან, მარაგისგან და მეხსიერებისგან, რომლებიც არიან ერთადერთი კომპონენტები და პირდაპირ ხელმისაწვდომია cpu-ს სისტემისათვის. CPU ს შეუწყვეტლად შეუძლია წაიკითხოს ამ მიდამოებში შენახული მონაცემები და შეასრულოს ყველა მოთხოვნილი ინსტრუქცია სწრაფად ერთგვაროვანი მანერით. მეორადი მარაგი განსხვავდება მთავარი მარაგისაგან იმით რომ ის არ არის პირდაპირ მისაწვდომი CPU სათვის. სისტემა იყენებს შემავალ/გამომავალ არხებს რომ დაუკავშირდეს მეორად მარაგს, რომელიც აკონტროლებს ინფორმაციის მიმოქცევას სისტემაში მისი მოთხოვნის დროს.

მეორადი საცავი არ კარგავს მონაცემებს როცა ის არის მიუწვდომელი, მაშასადამე თანამედროვე კომპიუტერულ სისტემებში უფრო განვითარებულია მეორადი მარაგები ვიდრე პირველადი, მთავარი. დღესდღეობით ყველა მეორადი მარაგი შეიცავს "hard disk drives"-ს ძირითადად განლაგებული RAID კონფიგურაციაში, თუმცა ძველი ინსტალაციები აგრეთვე მოიცავდნენ მოძრავ მედიას როგორიცაა magneto optical an MO.

მესამეული საცავი ძირითადად გამოყენებულია როგორც ცალკე შენახული ასლი და მონაცემების არქივი, ასევე დაფუძნებული ყველაზე ნელ მოწყობილობებზე რომლებიც შეიძლება მივაკუთვნოთ ყველაზე მნიშვნელოვან კომპონენტს მონაცემების დაცვაში მრავალი საშიშროებისგან, რომლებსაც შეუძლია გავლენა მოახდინოს IT ინფრასტუქტურაზე. ამ სეგმენტში მოწყობილობების უმრავლესობა იმართება ავტომატურად , რომ შეამციროს მენეჯმენტის დანახარჯები და რისკი ადამიანების შეცდომებისა,მესამეული საცავი ძირითადად შედგება ჩანაწერზე (disk &tape) დაფუძვნებული ასლის მოწყობილობებით.

მიუწვდომელი (Offline) საცავი არის ბოლო კატეგორია და საშუალებას აძლევს მომხმარებელს შეაგროვოს ინფორმაცია, რომელიც არ არის დაზიანებული ვირუსების მიერ ან პროგრამული შეფერხების შედეგად. Offline საცავი ყველაზე გავრცელებული მაგალითებია CD და DVD. Offline საცავი შეიძლება გამოყენებული იქნას სისტემებს შორის ინფორმაციის გასაცვლელად, მაგრამ აგრეთვე მონაცემებს ნებას რთავს რომ იყვნენ დაცული მუშაობით (საიტისგან მოშორებით) რათა კომპანიები დარწმუნებულნი იყვნენ რომ ყოველთვის აქვთ მნიშვნელოვანი მონაცემის სარეზერვო ასლი.


Primary storage
Central processing unit
Registers
Logic unit
Cache memory
Memory bus
Main memory
Random access memory
256-1024 MB
Input/output channels
Secondary storage
Mass storage device
Hard disk
20-120 GB
Off-line storage
Removable media drive
CD-RW, DVD-RW drive
Removable medium
CD-RW
650 MB
Tertiary storage
Removable media drive
Removable medium
Robotic access system
Removable medium

ქვანტური კომპიუტერი და ქვანტური მონაცემები

უკანასკნელ წლებში განხორციელებული კვლევების შედეგები ქვანტური კომპიუტერის არსებობას სულ უფრო რეალურს ქმნიან. შექმნილია ქვანტური კომპიუტერის მოდელი (ალგორითმული სახით). მიმდინარეობს კვლევები მისი სხვადასხვა ლოგიკური კომპონენტების მოდელის დასახვეწად. მათ შორის ინფორმაციის საცავის მოდელის შესაქმნელად, სადაც შესაძლებელი იქნება ქვანტურ მონაცემთა შენახვა.

ქვანტური კომპიუტერების მთავარი შემადგენელი ნაწილია ქვანტური ბიტები, ე.წ. **qubits**(**qu**antum **bits**). როგორც იცით დღევანდელ კომპიუტერებში და ზოგადად ციფრულ ტექნიკაში, გამოიყენება ორობითი კოდი. ამ ორობითი კოდის წარმომადგენელია ინფორმაციის უმცირესი განმსაზღვრელი ერთეული – ბიტი. მას შეიძლება ქონდეს მხოლოდ ორი მნიშვნელობა: 0 ან 1. განსხვავებით სტანდარტული ბიტისაგან, ქვანტურ ბიტებს შეუძლიათ მიიღონ მინიშვნელობა 0, 1 ან ორივე ერთად (ნულიც და ერთიც). ამ მოვლენას სუპერპოზიცია (**superposition**) ეწოდება და ქვანტური მექანიკის შემადგენელი ნაწილია, ამიტომ დეტალურად აღარ განვიხილავთ.

დღევანდელ კომპიუტერებში, ბიტები იქმნება წრედში ელექტრული დენის არსებობით; ანუ როდესაც წრედში არის დენი, ბიტის მნიშვნელობა განისაზღვრება როგორც "1", ხოლო როდესაც წრედში არ არის დენი, ბიტის მნიშვნელობა არის "0". რაც შეეხება ქვანტურ ბიტებს, აქ ინფორმაციის გადამტანად გამოიყენება ატომები, იონები, ფოტონები და/ან ელექტრონები. ამ შემთხვევაში 0-ის და 1-ის მინიჭება ბიტისათვის ხდება არა წრედში დენი არსებობა-არარსებობით, არამედ იონის მუხტის ან ელექტრონის სპინის (**spin**) მნიშვნელობით: ⬇-down ან ⬆-up, ანუ 0 ან 1.

ქვანტური ბიტების სუპერპოზიცირების უნარი, ასევე სხვა თვისებები, რომლებიც ახასიათებს ქვანტურ მექანიკას, იძლევა საშუალებას ქვანტური ბიტი ერთდროულად გამოყენებულ იქნას როგორც პროცესორის ერთეული და როგორც მეხსიერების ერთეული. ეს წარმოუდგენლად ზრდის დროის ერთეულში შესრულებულ კალკულაციათა რაოდენობასა და სისწრაფეს. შედარებისათვის, დღევანდელი პროცესორები მუშაობენ

ტრანზისტორებზე, რომლებიც ასრულებენ მილიარდობით ოპერაციას წამში, რაც გიგაფლოპებით (gigaflop) განისაზღვრება, ხოლო მაქსიმალური წარმადობა მულტიპროცესორულ სისტემებში მიიღწევა პროცესორების რაოდენობის გაზრდით; ქვანტური კომპიუტერების შემთხვევაში, წარმადობის გაზრდა დაკავშირებულია თითოეული ქვანტური ბიტის უნართან დროის მოცემულ ერთეულში დაამუშავოს მეტი მოცულობის ინფორმაცია, რაც საშუალებას იძლევა ქვანტური პროცესორების ოპერაციათა რიცხვი გაიზარდოს ტრილიონ კალკულაციამდე ერთ წამში, რასაც ტერაფლოპი (teraflop) ეწოდება.

ქვანტური კომპიუტერების შესაძლებლობები იმდენად დიდია რომ ადამიანს შეუძლია აღარ იდარდოს გამოთვლითი სისტემებისათვის მიწოდებული მონაცემების მოცულობასა და რაოდენობაზე. მაგალითად კრიპტოგრაფებს ხშირად აქვთ საქმე რთულ კოდებთან. როგორც მათ გენერაციასთან, ისე დეშიფრირებასთან. როგორც იცით თანამედროვე შიფრირებით გენერირებული კოდის გატეხვა საკმაოდ ძნელია და თუ გასაღების მორგებისათვის რაიმე ალგორითმი არ იქნა გამოყენებული, კოდის გატეხვას სიმბოლოების ბრმა შერჩევის მეთოდით წლები დაჭირდება. ამ საქმეში კი სწორედ ამ ალგორითმის შექმნაა ყველაზე ძნელი. ქვანტური კომპიუტერისათვის კი ეს ყველაფერი სულ რამდენიმე წუთის საქმეა. კონკრეტულად კი კოდის გასატეხად საჭირო დრო წარმოადგენს კვადრატულ ფესვს n-დან – √ n (სადაც n შესაძლო ვარიანტების რაოდენობაა). ასევე საპირისპიროდაც, თუ კოდის გენერირება ხდება ქვანტური პროტოკოლის გამოყენებით, მისი სირთულე ფაქტიურად შეუძლებელს ხდის ბინარული სისტემის გამოთვლითი ხელსაწყოებისათვის მის გატეხვას.

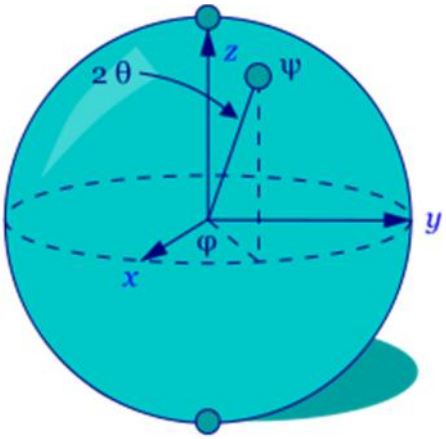
რაც შეეხება ქვანტური კომპიუტერების დღეს არსებულ მოდელებს, უნდა ითქვას რომ ისინი ჯერ კიდევ კონცეპტუალურ დონეზე არიან, ვიდრე ყოველდღიურობისათვის მისაღები სახით. ჯერჯერობით მათი გამოყენება ხდება სუფთა მათემატიკური ოპერაციების შესასრულებლად, ასევე მათი ტესტირებაც ამ კუთხით მიმდინარეობს, რადგანაც თავად ქვანტური პროცესორების ექსპერიმენტულობიდან გამომდინარე, ჯერ არ არის შექმნილი მუშა სახის სპეციალური პროგრამული უზრუნველყოფა ამ ტექნოლოგიების მულტიმედიურ ან სხვა სფეროში ინტეგრაციისათვის.

ქვანტური კომპიუტერი — ჰიპოთეტური გამომთვლელი მოწყობილობაა, რომელიც ქვანტური ალგორითმების შესრულების გზით მუშაობისას იყენებს ქვანტურ-მექანიკურ ეფექტებს, როგორიცაა ქვანტური პარალელიზმი.

მიუხედავად იმისა, რომ ქვანტური გამოთვლები ჯერ კიდევ ჩანასახის სტადიაშია, დღემდე ჩატარებულ ექსპერიმენტებში ქვანტური გამოთვლების ოპერაციები შესრულებულ იქნა ძალიან მცირე რაოდენობის ქუბიტზე (ქვანტური ბინარული ერთეულები). გაცხოველებული პრაქტიკული და თეორიული კვლევა ამ სფეროში ამჟამადაც მიმდინარეობს და მრავალი ეროვნული მთავრობა ხელს უწყობს ქვანტური გამოთვლების კვლევებს ქვანტური კომპიუტერების დამუშავებაში სამოქალაქო და ეროვნული უსაფრთხოების მიზნებისთვის (მაგ. კრიპტოანალიზისთვის).

თუ მსხვილმასშტაბიანი ქვანტური კომპიუტერების აწყობა შესაძლებელი გახდა, ისინი შეძლებენ გარკვეული პრობლემების გადაჭრას გაცილებით სწრაფად, ვიდრე ნებისმიერი სხვა კლასიკური კომპიუტერი (მაგ. შორის ალგორითმი). ქვანტური კომპიუტერები განსხვავდება სხვა კომპიუტერებისგან, როგორიცაა დნმ კომპიუტერები და ტრადიციული კომიუტერები ტრანზისტორებზე)

ბლოხის სფერო ქუბიტის განსახიერებაა, რაც ქვანტური კომპიუტერის ფუნდამენტური ნაწილია.



ქვანტური სქემები

ქვანტური ლოგიკური სქემების კომბინაცია შეიცავს ქვანტურ გეიტებს. ქვანტურ გეიტს აქვს შემავალი და გამომავალი ინფორმაციის ერთი და იგივე რაოდენობა. ქვანტური სქემა შეიძლება გავიგოთ, როგორც ქვანტური ლოგიკური იპერაციების თანმიმდევრობის წარმოდგენა ქვანტურ რეგისტრში. მოცემული გამოსახულება შეიცავს თორმეტ ერთ და ორ ქუბიტიან გეიტს, რომლებიც იყენებენ სამ ქუბიტიან რეგიტრს. თუ დავაკვირდებით შევამჩნევთ , რომ სამ ქუბიტიანი რეგისტრის მდგომარეობა არის აღწერილი H_3 (რვა ელემენტიანი სვეტი) ვექტორის გამოყენებით, მაშინ როდესაც ერთ და ორ ქუბიტიანი

გეიტები აღწერილია უნიტარული ოპერაციებით H_2 -ზე და H_1-ზე(მოცემულია 4x4 და 2x2 მატრიცით). იმისათვის რომ შევუთავსოთ სხვადასხვა მდგომარეობის ვექტორების და მატრიცების ზომები, ჩვენ უნდა განვიხილოთ ტენზორული ოპერაციები.

განვიხილოთ ℓ+m=n - ქუბიტიანი რეგისტრი. სადაც ℓ - ქუბიტიანი გეიტი V მოქმედებს ℓ ქუბიტზე. M ქუბიტიანი გეიტი w მოქმედებს დანარჩენზე. ფორმულის ძირითადი სახე წარმოდგენილია შემდეგში.

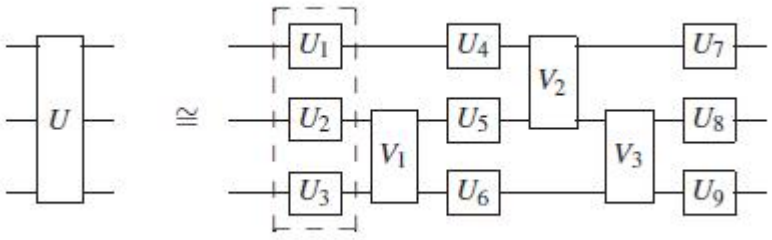
$$|\psi\rangle = \sum_{b\in\mathbb{B}^n} \alpha_b |b\rangle = \sum_{b\in\mathbb{B}^\ell, b'\in\mathbb{B}^m} \alpha_{b\cdot b'} |b\rangle |b'\rangle$$

რეგისტრზე შესრულებული ოპერაცია ავღნიშნოთ ***V*** Ă ***W***

$$V\otimes W|\psi\rangle = \sum_{b\in\mathbb{B}^\ell, b'\in\mathbb{B}^m} \alpha_{b\cdot b'} (V|b\rangle)(W|b'\rangle)$$

ამ განტოლებიდან შეგვიძლია დავასკვნათ ***V*** Ă ***W***-ის 2^nx2^n მატრიცა მოცემულია:

$$(V\otimes W)_{r\cdot r', c\cdot c'} = V_{r,c} W_{r',c'} \quad \text{for} \quad r,c\in\mathbb{B}^\ell, \quad r',c'\in\mathbb{B}^m$$



ტიპიური ქვანტური ლოგიკური სქემა . ინფორმაცია მიედინება მარცხნიდან მარჯვნივ. ამ სქემის მიხედვით შესრულებული ქვანტური ოპერაცია არის (*U*7Ă*U*8Ă*U*9)(*I*2 Ă*V*3)(*V*2 Ă*I*2)(*U*4 Ă*U*5 Ă*U*6)(*I*2 Ă*V*1)(*U*1 Ă*U*2 Ă*U*3) . ქვანტური სქემებით აღნიშნული ფორმულები იკითხება მარჯვნიდან მარცხნიც.

მონაცემთა წარმოდგენის ხერხები ქვანტური კომპიუტერისათვის

ჩვენი მიზანია შევქმნათ კომპიუტერში ინფორმაციის წარმოდგენის ქვანტური მოდელი.

ქვანტური კომპიუტერი მუშაობს ქვანტური ბიტებით. იგი არის ანალოგი კლასიკური ბიტის. თუმცა ქვანტურ ბიტს გააჩნია კლასიკური ბიტისაგან პრინციპულად განსხვავებული თვისებები: მას შეუძლია ყოფნა ბაზისური მდგომარეობების სუპერპოზიციაში.

$^{0}q^{0}=\sim^{0}0¡+\ ^{0}1¡$

სადაც ~ და კომპლექსური რიცხვებია და ეწოდება სისტემის ამპლიტუდა. ისინი აკმაყოფილებენ პირობას:

$\sim^{2}+\ ^{2}=1$

სისტემის მთლიანად აღწერისათვის საჭიროა თითოეული მდგომარეობისათვის განისაზღვროს ამპლიტუდა, რომელიც არის კომპლექსური რიცხვი.

შესაძლებელია განვიხილოთ ქუბიტის მდგომარეობა როგორც მდგომარეობა ვექტორის რომელიც მდებარეობს ორ განზომილებიან კომპლექსურ ვექტორულ სივრცეში ორთონორმირებული ბაზისით { |0⟩, |1⟩ }:

$$a|0\rangle + \mathrm{b}|1\rangle = \begin{bmatrix} a \\ b \end{bmatrix}$$

სადაც

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

ქვანტური ვენტილი

როგორც კლასიკურ, ასევე ქვანტურ კომპიუტერებში, ბიტები და ქუბიტები გაერთიანებულია ვენტილებად, ვენტილი არის ქვანტური კომპიუტერის საბაზისო ელემენტი, იგი ახდენს შემავალი ქუბიტების გარდაქმნას გარკვეული წესით გამომავალ ქუბიტებად.

ორ ქუბიტიანი ქვანტური რეგისტრი შესაძლებელია გამოვსახოთ ოთხ განზომილებიან კომპლექსურ ვექტორულ სივრცეში, ორთონორმირებული ბაზისით { |00⟩, |01⟩, |10⟩, |11⟩ } შემდეგნაირად:

$$a_0|00\rangle + a_1|01\rangle + a_2|10\rangle + a_3|11\rangle = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

n ქუბიტიანი ქვანტური რეგისტრისათვის რეზულტატური მდგომარეობა იქნება 2^n

ქვანტური მდგომარეობა არის ისეთი მდგომარეობა, როდესაც სისტემას შეუძლია ერთდროულად ყოფნა სხვადასხვა მდგომარეობაში გარკვეული ალბათობით. ამიტომ ქვანტური ალგორითმები ძირითადად ალბათური ალგორითმების ჯგუფს მიეკუთვნებიან. სისტემის ევოლუცია შეიძლება გამოისახოს მდგომარეობის ცვლილების მატრიცით.
კლასიკური ალგორითმების მსგავსად ქვანტური ალგორითმები ახასიათებენ სისტემის სხვადასხვა მდგომარეობას განსხვავებული ალბათობებით. შესაძლებელია ისეთი სუპერპოზიციის შექმნა, რომელსაც შეესაბამება 50% ⁰0¡ და 50% ⁰1¡

$\frac{|0\rangle+|1\rangle}{\sqrt{2}} \rightarrow 0$ ალბათობით $\frac{1}{2}$

$\frac{|0\rangle+|1\rangle}{\sqrt{2}} \rightarrow 1$ ალბათობით $\frac{1}{2}$

prob ⁰0¡+ prob ⁰1¡=1 ≫~²+ ²=1

განსხვავებით კლასიკურისაგან, სადაც n ნაწილაკიანი სისტემის მდგომარეობა შეიძლება გამოვსახოთ ცალკეულ მდგომარეობათა დეკარტული ნამრავლით, ქვანტურ სისტემებში მდგომარეობათა შეერთება ხდება მდგომარეობათა ტენზორული ნამრავლით.

ქვანტური გეიტები

ქვანტური კომპიუტერი, კლასიკურის მსგავსად აგებულია ქვანტური სქემებით. ქვანტური სქემა ეს არის გამოთვლითი მოდელი , რომელიც წარმოადგენს ქვანტური გეიტების სიმრავლეს. ქუბიტების ტრანსფორმაცია ხორციელდება ქვანტური გეიტებით.ზოგადად n ქუბიტიანი გეიტის მდგომარეობა შეიძლება გამოვსახოთ როგორც

$|\psi\rangle=\sum_{i=0}^{2^n-1} a_i\,|i\rangle$

ქვანტური ფიზიკის კანონებზე დაყრდნობით დადგენილია, რომ ყველა გარდაქმნა, რომელიც ხორციელდება ქვანტური პროცესებით, არის შექცევადი და უნიტარული. რაც ნიშნავს, რომ თუ გვაქვს ქვანტური სისტემა U, რომლის შესავალი მდგომარეობა არის $|\psi\rangle$, ხოლო გამოსავალზე გვაქვს განსხვავებული $|\psi'\rangle$მდგომარეობა ,მაშინ შეგვიძლია U ავღწეროთ, როგორც უნიტარული წრფივი ტრანსფორმაცია.

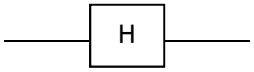
$$U\begin{pmatrix} a & b \\ c & d \end{pmatrix} = U^{+}\begin{pmatrix} a* & b* \\ c* & d* \end{pmatrix}$$

ოპერატორი, რომელიც ახორციელებს უნიტარული ვექტორის გარდაქმნას სხვა უნიტარულ ვექტორად, არის უნიტარული და აკმაყოფილებს პირობას: $U^{+}U=1$

ამ მხრივ მნიშვნელოვანია ჰადამარის გეიტი. ჰადამარის ოპერატორი არის

$$H=\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

სქემატურად იგი გამოისახება შემდეგნაირად:



მისი მოქმედების შედეგი ასეთია:

$$H : |0\rangle \rightarrow \frac{|0\rangle+|1\rangle}{\sqrt{2}}$$

$H : |1\rangle \rightarrow \frac{|0\rangle - |1\rangle}{\sqrt{2}}$

თუ გვაქვს n ქუბიტიანი რეგისტრი, მაშინ თუ გამოვიყენებთ ადამარის გარდაქმნას თითოეული ქუბიტისათვის მივიღებთ სუპერპოზიციას 2^nმდგომარეობების.

(H Ă H Ă . . . H) $|00\ \ldots 0\rangle = \frac{1}{\sqrt{2^n}}\ ((|0\rangle + |1\rangle)\otimes(|0\rangle + |1\rangle)\otimes\ \ldots\otimes(|0\rangle + |1\rangle)) = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle$

ეს არის უოლშ-ჰადამარის გარდაქმნაW, რომელიც შეიძლება განისაზღვროს რეკურსიულად შემდეგნაირად:

$W_1 = H$, $W_{n+1}=H\otimes W_n$

განვიხილოთ სუპერპოზიცია ყველა შემავალი შესაძლო მნიშვნელობების, როგორც ერთი მდგომარეობა. საწყისი მდგომარეობა იყოს $|00. \ . \ .0\rangle$. გამოვიყენოთ უოლშ-ჰადამარის გარდაქმნა სუპერპოზიციის $\frac{1}{\sqrt{2^n}}\ (|00\ \ldots 0\rangle + |00\ \ldots 1) + \ \ldots + (|11\ \ldots 1\rangle + |1\rangle)$მისაღებად , რომელიც შეიძლება განვიხილოთ, როგორც სუპერპოზიცია ყველა რიცხვის $0\leq x \leq 2^n-1$

U_f წრფივი გარდაქმნა შეიძლება გამოყენებულ იქნეს ყველა ბაზისური მდგომარეობისათვის სუპერპოზიციაში დამოუკიდებლად. ამიტომ თუ მას გამოვითვლით ერთხელ, მაშინ შესაძლებელია გამოთვლილი იქნეს ყველა 2^n მნიშვნელობა $f\ (0), \ldots, f(2^n-1)$,

$|\psi\rangle = U_f\ (\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle)|0\rangle = (\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle|f(x)\rangle$

ეს კი არის ქვანტური პარალელიზმის ეფექტი.

ნებისმიერი ქვანტური ალგორითმი მანიპულირებს ქვანტური პარალელიზმის ეფექტით, ისე, რომ შედეგი ყოველთვის მიიღება გაზომვის მაღალი ალბათობით.

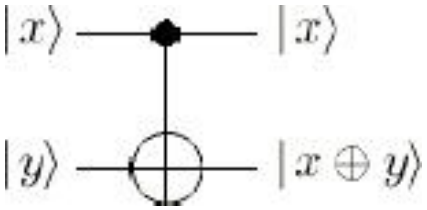
არაკონტროლირებადი გეიტი

Controlled not gate(CNOT) არის ქვანტური გეიტი, რომელიც ქვანტური კომპიუტერის მნიშვნელოვანი შემადგენელი ნაწილია. იგი მოქმედებს ორ ქუბიტზე და სრულებს NOT

ოპერაციას მეორე ქუბიტზე მხოლოდ მაშინ როცა პირველი ქუბიტის მნიშვნელობა არის |1⟩. სხვა შემთხვევაში ტოვებს მათ უცვლელად.

$$CNOT = \begin{cases} |00\rangle \to |00\rangle \\ |01\rangle \to |01\rangle \\ |10\rangle \to |11\rangle \\ |11\rangle \to |10\rangle \end{cases}$$

მისი შესაბამისი სქემა გამოისახება შემდეგნაირად:

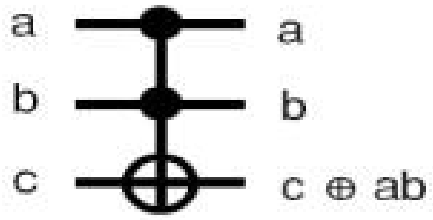


ტოფოლის გეიტი

ტოფოლის გეიტი არის უნივერსალური გეიტი კლასიკური გამოთვლებისათვის, თუმცა მას საკმაოდ მნიშვნელოვანი ადგილი უჭირავს ქვანტურ გამოთვლებშიც. მისი უნივერსალურობა განპირობებულია იმით, რომ იგი შესაძლებელია გამოყენებულ იქნეს ნებისმიერი გეიტის წარმოდგენისათვის. იგი არის სამ ქუბიტიანი გეიტი. შესავალზე იღებს სამ ქუბიტს a, b, c ხოლო გამოსავალზე იძლევა a', b', c'. პირველი ორი a და b ქუბიტები არის კონტროლირებადი, რაც ნიშნავს, რომ ისინი არ იცვლებიან გამოთვლების განმავლობაში, ხოლო მესამე იცვლება იმ შემთხვევაში, თუ ორივე a, b არის 1-ის ტოლი.

შემავალი			გამომავალი		
a	b	c	a’	b’	c’
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

მისი შესაბამისი სქემა გამოისახება შემდეგნაირად:



ყველა გამოთვლა, რომელიც სრულდება კლასიკურ კომპიუტერებზე, შესაძლებელია განხორციელდეს ქვანტურ კომპიუტერზეც. უპირატესობა არის ის, რომ ქვანტური გამოთვლები, რომლებიც ხორციელდება ქვანტური ალგორითმებით არის უფრო სწრაფი.

ყოველივე ეს კი განპირობებულია ქვანტური პარალელიზმით და ქვანტური აბურდვით, რომლებიც მიუღწეველია კლასიკურ კომპიუტერებში.

შორის ალგორითმი

შორის ალგორითმი არის ფაქტორიზაციის(რიცხვის მარტივ მამრავლებად დაშლა) ქვანტური ალგორითმი, რომელიც საშუალებას გვაძლევს რიცხვი M დავშალოთ მამრავლებად $O(\lg^3 M)$ დროში, $O(\lg M)$ ლოგიკური ქუბიტების გამოყენებით.

შორის ალგორითმი შემუშავებულ იქნა პიტერ შორის მიერ 1994 წელს. 2001 წელს მისი ეფექტურობა აჩვენეს IBM(International Business Machines) -ის სპეციალისტებმა. რიცხვი 15 დაშალეს მამრავლებად 3 და 5 ქვანტური კომპიუტერის დახმარებით 7 ქუბიტით.

ალგორითმის მნიშვნელოვნობა მდგომარეობს იმაში, რომ მისი დახმარებით ქვანტური კომპიუტერის გამოყენებით რამდენიმე ასეული ქუბიტით შესაძლებელი ხდება ღია გასაღებიანი კპრიპტოგრაფიული სისტემის გატეხვა. მაგალითად RSA იყენებს ღია გასაღებს M-ს . ერთ-ერთი საშუალება RSA-ს შიფრის გატეხვისა არის M-ის მამრავლები პოვნა. საკმარისად დიდი M-ისთვის ამის გაკეთება შეუძლებელია ცნობილი კლასიკური ალგორითმების გამოყენებით. ფაქტორიზაციის ცნობილი კლასიკური ალგორითმებიდან საუკეთესო მოითხოვს $M^{1/3}$ დროს. შორის ალგორითმს ქვანტური კომპიუტერის შესაძლებლობების გამოყენებით შეუძლია აწარმოოს რიცხვის ფაქტორიზაცია პოლინომურ დროში პრაქტიკულად იმაზე სწრაფად რა დროც სჭირდება თვითონ შიფრაციას.

შორის ალგორითმის საფუძველი არის ის რომ ქვანტური კომპიუტერის ინფორმაციის ერთეულს ქუბიტს შეუძლია მიიღოს ერთდროულად რამდენიმე მნიშვნელობა, ამიტომ ის საშუალება იძლევა ჩატარდეს გამოთვლა ქუბიტების ეკონომიური რაოდენობით.

M - რიცხვი, რომელიც არ წარმოადგენს კენტი რიცხვის ფესვს, რომელიც უნდა დავშალოთ მამრავლებდ.

N - მეხსიერების რეგისტრის ზომა.

$$M^2 < N = 2^n < 2M^2$$

t - შემთხვევითი პარამეტრი, ისეთი რომ $1 < t < M$ и $gcd(t, M) = 1$,

gcd - უდიდესისაერთო გამყოფი.

აღვნიშნოთ, რომ t,N,M -ფიქსირებულია.

კლასიკური ალგორითმი
მინიმალური r ისეთი, რომ $t^r = 1 \ mod \ M$

არის ფუნქციის პერიოდი $f(x) = t^x \ mod \ M$, სადაც $x = 0, 1, 2, ..., N - 1$

თუ შესაძლებელია ეფექტურად გამოვთვალოთ r როგორც t ფუნქცია, მაშინ შევძლებთ ვიპოვოთ გამყოფი M, განსაზღვრულ პოლინომურ დროში $log_2 M$ c -დან ალბათობით $\geqslant 1 - M^{-m}$ ნებისმიერი ფიქსირებული m -ისთვის.

ვივარაუდოთ, რომ t პერიოდისთვის r აკმაყოფილებს

$r \equiv 0 \ mod \ 2, t^{\frac{r}{2}} \neq -1 \ mod \ M$.

მაშინ

$gcd(t^{\frac{r}{2}} + 1, M)$ -საკუთარი გამყოფი M. ფუნქცია gcd გამოითვლება პოლინომურ დროში.

$\geqslant 1 - \frac{1}{2^{k-1}}$, ამ პირობის შესრულების ალბათობა, სადაც k — რიცხვი M-ის კენტი მარტივი გამყოფების, მაშასადამე $\geqslant \frac{1}{2}$ ამ შემთხვევაში. tc კარგი მნიშვნელობა $\geqslant 1 - M^{-m}$ ალბათობით მოიძებნება $O(\lg M)$ ცდაში. ყველაზე დიდი გამოთვლა ამ ცდაში არის $t^{\frac{r}{2}}$.

ქვანტური ალგორითმი

ამ ალგორითმის ქვანტური ნაწილის შესრულებისთვის გამომთვლელი სქემა წარმოადგენს ორ ქვანტურ რეგისტრს X და Y . თავდაპირველად ყოველი მათქგანი შედგება ქუბიტების ერთობლიობისაგან ნულოვან მდგომარეობაში $|0\rangle$.

რეგისტრი X გამოიყენება x არგუმენტის განთავსებისათვის $f(x)$ ფუნქციაში.

რეგისტრი Y (დამხმარე) გამოიყენება $f(x)$ ფუნქციის r პერიოდით განთავსებისათვის.

ქვანტური გამოთვლა შედგება 4 ბიჯისაგან:

პირველი ბიჯი:

პირველ ბიჯზე უოლშ ჰადამარის ოპერაციების დახმარებით X რეგისტრის თავდაპირველი მდგომარეობა $|0\rangle$ გადადის თანაბარალბათურ სუპერპოზიციაში N-ს ყველა მდგომარეობებისა. მეორე რეგისტრი Y რჩება $|0\rangle$ მდგომარეობაში. საბოლოოდ მიიღება შემდეგი მდგომარეობა ორი რეგისტრის სისტემისთვის

$$\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x,0\rangle$$

მეორე ბიჯი:

დავუშვათ U_f არის უნიტარული გარდამქნა, რომელიც გადადის $|x,0\rangle$ в $|x,f(x)\rangle$

მეორე ბიჯზე ხდება ორი რეგისტრის სისტემის გარდაქმნა. მიიღება სისტემის შემდეგი მდგომარეობა:

$$|x,0\rangle\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}\xrightarrow{U_f}\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x,t^x \ mod \ M\rangle$$

,

ორი რეგისტრის მდგომარეობებს შორის იქმნება განსაზღვრული კავშირი.

მესამე ბიჯი:

ფურიეს ქვანტური გარდაქმნა წარმოადგენს ქვანტური რეგისტრის მდგომარეობის გარდაქმნას, აღწერილს N განზომილებიანი ვექტორის მდგომარეობით შემდეგი სახით

$\sum_{x=0}^{N-1} f(x)|x\rangle$ შემდეგ მდგომარეობაში $\sum_{k=0}^{N-1} \tilde{f}(k)|k\rangle$

$QFT_N : \sum_{x=0}^{N-1} f(x)|x\rangle \Rightarrow \sum_{k=0}^{N-1} \tilde{f}(k)|k\rangle$ სადაც ფურიეს გარდაქმნის ამპლიტუდას

$f(x)$ აქვს სახე $\tilde{f}(k) = \frac{1}{N} \sum_{x=0}^{N-1} \exp(2\pi\ i\ kx/N) f(x)$

ორგანზომილებიანი x, k სიბრტყე ფურიეს გარდაქმნის შესაბამისად შემობრუნდება კოორდინატთა ღერძიდან 90° - ით. მესამე ბიჯზე პირველი რეგისტრის ზედა მდგომარეობაში იწარმოება ფურიეს გარდაქმნა და მიიღება

$$\frac{1}{N} \sum_{x=0}^{N-1} \sum_{k=0}^{N-1} \exp(2\pi\ i\ kx/N)|k, t^x\ mod\ M\rangle$$

მეოთხე ბიჯი:

მეოთხე ბიჯზე სრულდება პირველი X რეგისტრის გაზომვა

$|0,0\rangle \otimes I, |1,1\rangle \otimes I, |1,1\rangle \otimes I, \ldots, |N-1, N-1\rangle \otimes I,$ სადაც I არის იგივეობის ოპერატორი. შედეგად მიიღება $|k, t^k\ mod\ M\rangle$ ალბათობით

$$\left| \frac{1}{N} \sum_{x:\ t^x \equiv t^k\ mod\ M} \exp(2\pi\ i\ kx/N) \right|^2$$

დანარჩენ ნაწილზე მუშაობს კლასიკური კომპიუტერი.

საუკეთესო მიახლოება ქვემოდან არის $\frac{k}{N}$ მნიშვნელით $r' < M < \sqrt{N}$:

$$\left|\frac{k}{N} - \frac{d'}{r'}\right| < \frac{1}{2N}$$

ვცადოთ r' r-ის როლში:

თუ $r' \equiv 0 \ mod \ 2$, მაშინ გამოვთვალოთ $gcd(t^{\frac{r'}{2}} \pm 1, M)$

თუ r' კენტია ან ლუწი, მაგრამ გამყოფი M არ არის აღმოჩენილი, მაშინ მოხდება განემორებით წარმოება $O(\lg \lg M)$ იგივე t თი. უარყოფის შემთხვევაში შევცვალოთ t და დავიწყოთ დავიდან ალგორითმის შესრულება. რომ განვსაზღვროთ პერიოდი r არ არის საჭირო $f(x)$ - ის ყველა მნიშვნელობის გამოთვლას.

დავუშვათ F არის ფუნქცია უცნობი r პერიოდით

$$F : |x, 0\rangle \ \rightarrow |x, f(x)\rangle \ f : Z \rightarrow Z_{2^m} \ r < 2^n$$

რომ განვსაზღვროთ პერიოდი r საჭიროა ორი რეგისტრი ზომებით $2n$ და m ქუბიტი, რომლებიც თავდაპირველ მდგომარეობაში უნდა იყოს $|0, 0\rangle$, პირველ ეტაპზე სრულდება პირველი რეგისტრის ყველა საბაზისო ვექტორის ცალმხრივი სუპერპოზიცია U ოპერატორის გამოყენებით, რომელსაც აქვს შემდეგი სახე:

$$U|0, 0\rangle \ = \sum_{i=0}^{N-1} c_i |i, 0\rangle \quad , \text{где} \quad |c_i| = \frac{1}{\sqrt{N}} \text{ и } N = 2^{2n}$$

აქ გამოიყენება ჰადამარის ფსევდოგარდაქმნა H. F - ის გამოყენებით მიმდინარე მდგომარეობაში მიიღება:

$$|\psi\rangle = FH|0,0\rangle = F\frac{1}{2^n}\sum_{i=0}^{N-1}|i,0\rangle = \frac{1}{2^n}\sum_{i=0}^{N-1}|i,f(i)\rangle$$

მეორე რეგისტრის გაზომვის შედეგად $k = f(s)$, где $s < r$ მოვიღებთ მდგომარეობას:

$$|\psi'\rangle = \sum_{j=0}^{[N/r]-1} c'_j|rj+s,k\rangle$$ სადაც $$c'_j = \left[\frac{N}{r}\right]^{-1/2}$$

$|\psi'\rangle$ -ის მდგომარეობის გაზომვის შედეგად პირველი რეგისტრი შედგება მხოლოდა საბაზისო ვექტორებისგან $|rj+s\rangle$,ისეთებისგან რომ $f(rj+s) = f(s)$ ყველა j -სთვის.

ამიტომ მას აქვს ერთგვარი დისკრეტული სპექტრი. შეუძლებელია პირველი რეგისტრის გაზომვის შედეგად პირდაპირ მივიღოთ პერიოდი r , იმიტომ რომ s არის შემთხვევითი სიდიდე. აქ გამოიყენება ფურიეს დისკრეტული გარდაქმნა, რომელსაც აქვს შემდეგი სახე:

$$DFT : |x\rangle \to \frac{1}{\sqrt{N}}\sum_{y=0}^{N-1} e^{\frac{2\pi i}{N}xy}|y\rangle$$, გარდაქმნაში მოხვდება მხოლოდ ფაზები და არა ამპლიტუდის აბსოლუტური მნიშვნელობა.

$$|\psi''\rangle = DFT|\psi'\rangle = \sum_{i=0}^{N-1} c''_i|i,k\rangle$$

$$c''_i = \frac{\sqrt{r}}{N}\sum_{j=0}^{p-1} exp\left(\frac{2\pi i}{N}i(jr+s)\right) = \frac{\sqrt{r}}{N}e^{\phi_i}\sum_{j=0}^{p-1} exp\left(\frac{2\pi i}{N}ijr\right)$$

სადაც $\phi_i = 2\pi i\frac{is}{N}$ და $p = \left[\frac{N}{r}\right]$

თუ $N = 2^{2n}$ -ის ჯერადია r, მაშინ $c_i'' = \frac{e^{\phi_i}}{\sqrt{r}}$, თუ i ჯერადია $\frac{N}{r}$ და $c_i'' = 0$ წინააღმდეგ შემთხვევაში. თუ r არ არის 2-ის ხარისხი, მაშინ $|\psi''\rangle$ სპექტრი აჩვენებს ცალკეულ პიკებს $\frac{N}{r}$ პერიოდით, იმიტომ რომ

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} e^{2\pi i k\alpha} = \begin{cases} 1 & , \alpha \in Z \\ 0 & , \alpha \notin Z \end{cases}$$

პირველი რეგისტრისთვის გამოიყენება $2n$ ქუბიტი, როცა $r < 2^n$, იმიტომ რომ ეს იძლევა გარანტიას უკიდურესი 2^n ელემენტების ჯამში შეტანისა, ამგვარად პიკების სიგანე იქნება $O(1)$. თუ გამოვთვლით პირველ რეგისტრს, მაშინ მივიღებთ c მნიშვნელობას ახლოს

$\frac{\lambda N}{r}$ -თან, სადაც $\lambda \in Z_r$, ის შეიძლება ჩაწერილი იყოს როგორც $\frac{c}{N} = c \cdot 2^{-2n} \approx \frac{\lambda}{r}$

ეს დადის აპროქსიმაციის ძიებაზე $\frac{a}{b}$, სადაც $a, b < 2^n$ კონკრეტული რიცხვისთვის გაორებული წერტილებით $c \cdot 2^{-2n}$, ამ პრობლემის გადასწყვეტად გამოიყენება ჯაჭვური დაყოფა. რაციონალური რიცხვი არ არის ერთადერთი, რომ λ და r განისაზღვროს როგორც $\frac{a}{b} = \frac{\lambda}{r}$, თუ ალბათობა $gcd(\lambda, r) = 1$, მაშინ λ და r არიან მარტივები უფრო მეტად ვიდრე $\frac{1}{\ln r}$, ამგვარად მხოლოდ $O(n)$ ცდა არის საჭირო, რომ წარამტების ალბათობა რაც შეიძლება ახლოს იყოს 1-თან.

დასკვნა

ნაშრომის ფარგლებში ჩატარებული სამუშაოების შედეგად ანალიზის საფუძველზე, შემუშავდა ქვანტური მონაცემების წარმოდგენის მოდელი, რომელიც ეფუძნება ქუბიტის წარმოდგენას და მის ანალიტიკურ აღწერას. მოდელი გამოიყენება ქვნატურ კომპიუტერში ინფორმაციის საცავის მოდელის ერთერთ კომპონენტად. ამ მოდელის საფუძველზე შესაძლებელია შემუშავდეს პროგრამა, რომელიც მოახდენს მოდელის ალგორითმის რეალიზებას.

შედეგად მივიღებთ ქვანტური მონაცემთა საცავის პროგრამულ მოდელს, რომელიც შეიძლება წარმოვიდგინოთ, როგორც ქვანტური კომპიუტერის ლოგიკური სქემის ერთ-ერთი ნაწილი.

გამოყენებული ლიტერატურა

1. Database Manipulation on Quantum Computers Ahmed Younes (2008)
2. Elementary Quantum Gate Realizations forMultiple-Control Toffoli Gates D. Michael Miller, Robert Willeand Z. SasanianD. Michael Miller1, Robert Wille2 and Z. Sasanian1
3. L.K.Grover, Quantum Mechanics Help in Searching fora Needle in a Haystack, Phys. Rev. Letter 79,325-328,1997.
4. A fast quantum mechanical algorithm for database searchLov K. Grover
5. Nielsen, Michael and Chuang, Isaac (2000). *Quantum Computation and Quantum Information*. Cambridge: Cambridge University Press. ISBN 0-521-63503-9. OCLC 174527496
6. Derek Abbott, Charles R. Doering, Carlton M. Caves, Daniel M. Lidar, Howard E. Brandt, Alexander R. Hamilton, David K. Ferry, Julio Gea-Banacloche, Sergey M. Bezrukov, and Laszlo B. Kish (2003). "Dreams versus Reality: Plenary Debate Session on Quantum Computing". *Quantum Information Processing* **2** (6): 449–472.