Data checking and verification Randomization and coding

A guide to PCP, an approach towards classification of NP problems

P versus NP

Have you heard about P vs NP problem?

- Millenium prize problems by Cray institute

  http://www.claymath.org/millennium/

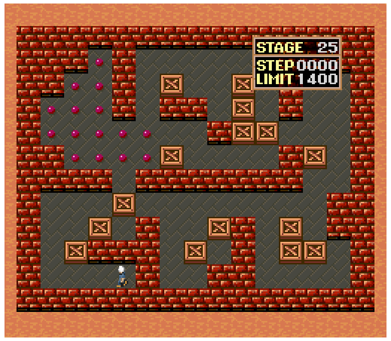
- P versus NP is not only million dollar problem, Its value is trillion dollars.

Class P and class NP

- Consider a computational problem M of input size n
- P: class of polynomial time solvable problems in usual model of computation
  - M is in P if there is an algorithm to solve it in polynomial time
    - Example: Sorting, Searching, Graph connectivity
    - Shortest path, text pattern matching
- NP: class of polynomial time solvable problems in nondeterministic model of computation
  - M is in NP if there is a nondeterministic algorithm to solve it in polynomial time (difficult to understand, isn’t it?)
- P is a subclass of NP

More intuitive description

- Verification is often easier than solution
  - Solving an equation is more difficult than verification
- NP: If we know the solution, we can show it such that verification is done in polynomial time
  - 1: Hamilton path
  - 2: Graph coloring,
  - 3: Does this SUDOK puzzle have a solution?
- A problem (probably) not in NP
  - 1: Will black win this Chess game ?
  - 2: SOKOBAN game
  - 3: #3SAT (number 3SAT)



- Intuitively, a problem that we are convinced with correctness of solution easily if it is shown.

NP and co-NP

- It is notable that negation of an NP problem might not be in NP
  - Is a given graph G is colored by using k colors?
    - We give the actual coloring to veryfy. Thus NP problem
  - Is a given graph G has no coloring with k colors?
    - If so, how to convince it?
- Co-NP problem: negation of an NP problem
- It is believed NP and co-NP are different
- It might be true that NP $\cap$ coNP = P

P vs NP

- Is it true that verification is easier than solution ?
  - Consider an equation $f(x) = x^5 - 3x^4 + 3x^2 + 4 = 0$
  - Ask " is this equation has an integer solution less than n (say, n=100)".
  - Verification: $f(2) = 32-48+12+4 = 0$
  - Solution???
- P vs NP: Is there any problem where verification is easy (P-time) but solution is difficult (not in P-time)?
- Philosophical question
  - Is it easy to learn than solve by yourself?
    - Is it true that a difficult problem remains to be difficult even if you are suggested a solution (you must make sure it is true).
  - In real life, we must solve many NP problems(or even more difficult problems). Human can solve them by training (like SUDOK) for most of instances. Why??

Implication of P=NP

- If P=NP, we can solve many problems
  - Many-body problem in physics
  - Protein folding problem in biology
  - Optimal scheduling in manufacturing
  - Optimal traffic control
  - Many problems in computational chemistry
- If P=NP, we have serious inconvenience
  - Current cryptology assumes P is not NP
  - Information security system is destroyed
- Most of researchers believes P is not NP, and the above situation only occurs in Scientific Fiction
  - But no one knows the truth
- The biggest mathematical challenge in 21st century

Several approaches towards P vs NP

http://people.cs.uchicago.edu/~fortnow/papers/pnp-cacm.pdf
Lance Fortnow's article

- NP- complete theory (S.Cook and R.Karp)
  - Almost all NP problems are either in P or NP-complete(one of the most difficult problems in NP)
- Circuit complexity
  - Show NP needs exponential size of circuits
- Algebraic/group theoretic method
  - Relation to generalized Riemann hypothesis (Mulmuley)
- From mathematical logic
- Relation to randomness
- **Interactive Proof and Checkable Proof**

Textbooks/papers

- Randomized Algorithms (Motwani-Raghavan)
- Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems
  (M. Sudan, ACM Distinguished Thesis, 1995)
- Proof Verification and the Hardness of Approximation Problems (S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy), J. ACM, Vol 45(3), 1998, pp. 501-555

Rajeev Motwani (from wikipedia)

- Motwani joined Stanford soon after U.C. Berkeley. Motwani was one of the co-authors (with Larry Page and Sergey Brin, and Terry Winograd) of an influential early paper on the PageRank algorithm, the basis for Google's search techniques. He also co-authored another seminal search paper *What Can You Do With A Web In Your Pocket* with those same authors.[3]
- He was also an author of two widely-used theoretical computer science textbooks, *Randomized Algorithms* (Cambridge University Press 1995, ISBN 978-0521474658, with Prabhakar Raghavan) and *Introduction to Automata Theory, Languages, and Computation* (2nd ed., Addison-Wesley, 2000, with John Hopcroft and Jeffrey Ullman).
- Prior to his involvement with Google, Motwani founded the Mining Data at Stanford project (MIDAS), an umbrella organization for several groups looking into new and innovative data management concepts. His research included data privacy, web search, robotics, and computational drug design.
- He was an avid angel investor and had funded a number of successful startups to emerge from Stanford. He sat on the boards of Google, Kaboodle, Mimosa Systems, Adchemy, Baynote, Vuclip, NeoPath Networks (acquired by Cisco Systems in 2007), Tapulous and Stanford Student Enterprises among others. He was also active in the Business Association of Stanford Entrepeneurial Students (BASES).[4][5][6]
- He was a winner of the Gödel Prize in 2001 for his work on the PCP theorem and its applications to hardness of approximation.[7][8]

Verification

Algebraic methods to verify information

Verification in real life

- Given two data A and B, we want to verify A=B
  - We want to do it without reading A and B explicitly
- If A and B are “persons”
  - How we can “read” data of persons completely?
    - Impossible!
  - Name, blood type, color of hair/eye,……
  - ID numbers, secret keywords,
  - Finger print, DNA identification
    - If A and B are twins………

Fingerprinting

- We recall the data checking problem we discussed some weeks ago
  - Given a set of n data {a(1),a(2),…,a(n)}, and we suspect one (or more) data is modified.
  - How should we check efficiently?
    - The sum of all data
    - Use hash function, and sum of h(a(i))
  - This is an example of “fingerprint”

Fingerprinting for matrix multiplication

- Consider a prime p
- Given n by n matrices A, B, and C, we want to verify AB = C mod p
  - Computation in the field GF(p) (or $\mathbf{Z_p}$)
- Can you do it in $O(n^2)$ time?

Pattern matching via verification

- Given a text T of length n (bit string)
- For each query pattern P of length m, we want to find location of occurrence of P in T
- Both m and n are long (say, m = 100000, n = 10000000)
- KMP algorithm, BM algorithm: optimal O(n+m), but not much practical
- Can we apply verification idea?

Verifying an identity

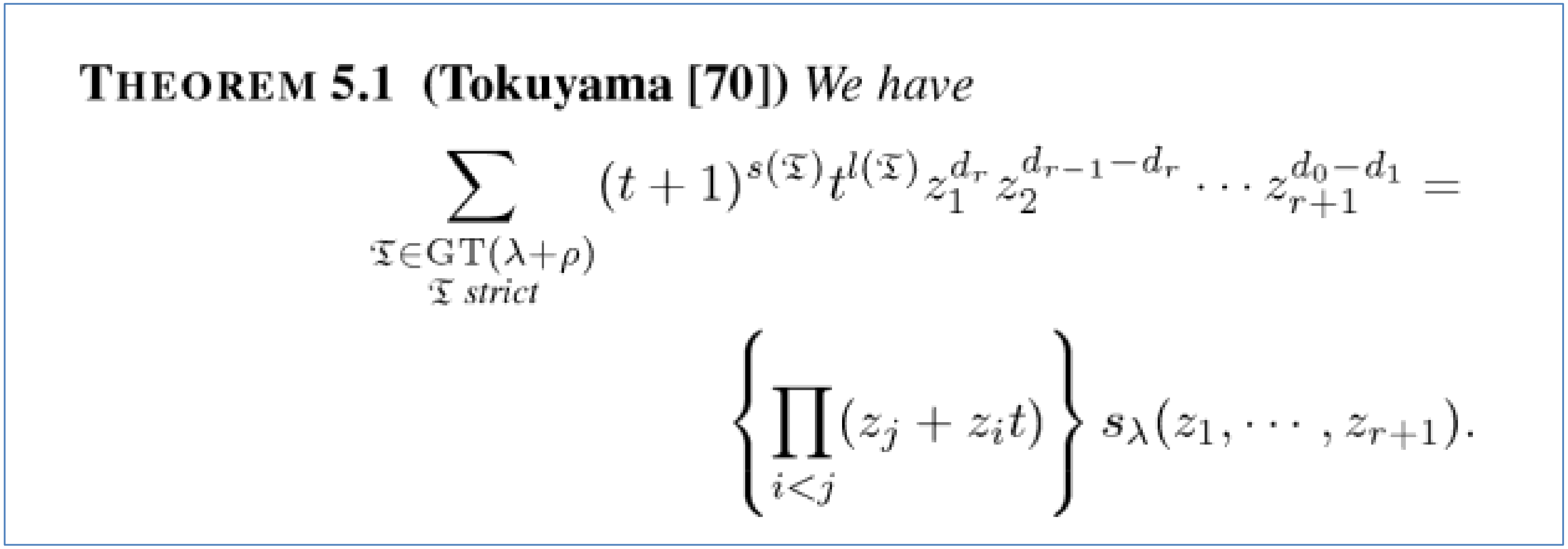
- $X = (x_1, x_2, .., x_d)$
- We want to verify a polynomial identity $F_1(X)=F_2(X)$ of degree n

$$\Delta(x_1, \ldots, x_n) \equiv \begin{vmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{vmatrix}$$

$$= \prod_{i,j} (x_i - x_j)$$

Verifying an identity

- $X = (x_1, x_2, .., x_d)$
- We want to verify a polynomial identity $F_1(X)=F_2(X)$ of degree n
  - Or more cruel identities



Theorem 5.1 **(Tokuyama [70])** *We have*

$$\sum_{\substack{\mathfrak{T}\in \mathrm{GT}(\lambda+\rho)\\ \mathfrak{T}\ \textit{strict}}} (t+1)^{s(\mathfrak{T})} t^{l(\mathfrak{T})} z_1^{d_r} z_2^{d_{r-1}-d_r} \cdots z_{r+1}^{d_0-d_1} =$$

$$\left\{ \prod_{i<j} (z_j + z_i t) \right\} s_\lambda(z_1, \cdots, z_{r+1}).$$

Prover and Proof

Prover is stronger than proof, since we can ask questions instead of reading the proof

Don't tell me you climbed **all the way up here** before trying **Google??**

Interactive Proof

- Suppose that you can ask a god (or a powerful supercomputer) to solve a problem
- If the answer is “YES” (or “NO”), do you believe it blindly?
  - In real life, we blindly believe the weather forecast which a computer software reports
  - In ancient Greek, people believed “oracle of Apollo”.
  - If you are given a program/software , how you believe it?
    - Or how you write a program convincing others its correctness?
  - Even in a university, most of students (and professors) believe Wikipedia blindly ????
- We want to request a proof or an evidence.
  - For a NP problem, we can ask for a proof if the answer is Yes
  - But if answer is NO, can we do something?

Graph non-isomorphism

- You have two graphs G=(V,E) and G'=(V',E')
- You suspect that they are isomorphic
  - There is a one-to-one map f of vertices of G such that (x,y) is an edge of E if and only if (f(x), f(y)) is in E'
- You ask your professor who says he is always honest and can solve the problem for any pair of graphs.
  - If he answers "yes", you can ask him to show the map f.
  - Can you believe him if he says "NO"?

An interactive proof

- You (verifier) have G and G’, and ask your professor (prover) whether G=G’
- Professor answers “NO”, but you suspect that he tells a lie.
- You ask some more questions to the professor to reveal whether he is honest
  - You can flip a coin, and the random choice is not known to the professor

How IP is strong

- I will show that #3SAT is solved by using interactive proof system
  - #3SAT: Find the number of solutions of a logical equation (given in a certain form)
- This implies #P is in IP, and co-NP is in IP
  - do not worry about such terminology
- A. Shamir showed that IP = PSPACE
  - PSPACE is considered to be larger than NP
  - 2 player’s game like GO and Chess are in PSPACE

3SAT and #3SAT

- 3SAT: Is a logic equation F(X(1),X(2),..,X(n)) = 1 in 3-CNF formula has a solution?
- #3SAT: How many solutions F(X)=1 has?
- 3SAT is an NP-complete problem
  - Any NP problem can be transformed into a 3SAT problem in polynomial time.
  - If 3SAT is in P, then NP = P
  - To show 3SAT is NP-complete
    - SAT is NP-complete (Cook's theorem)
    - SAT is transformed into 3SAT
- #3SAT is more difficult (called #P complete problem)
  - Toda's theory (1989, Goedel award)

From SAT to 3SAT

- Show that SAT is P-time soluble if 3SAT is.
- Given an instance (U,**C**) of SAT, we show a transformation of it into (U', **C**') of 3SAT
  - U: set of variables, C:set of clauses
    - U={X(1),X(2),..,X(n)}, **C**={C(1),C(2),..,C(m)}
    - $c(j) = l_{1,j} \vee l_{2,j} \vee ... \vee l_{k,j} = \{l_{1,j}, l_{2,j}, .., l_{k,j}\}$
- For each clause C of **C** of length k, we consider k-3 new variables, and transform it into a set of clauses each of which has length 3
- Thus, we transform into 3SAT input with at most nm variables and mn clauses

Transforming a clause

- C = {z(1) , z(2) ,.., z(k)}
  - z(i) is either X(i) or its negation
- We define new variables y(1),y(2), y(k-3)
  - These variables are only used to transform C
- The clause c is transformed into
  - S(c) = {{z(1),z(2),y(1)}, {~~y(1~~) , z(3), y(2)}, {~~y(2)~~,z(4), y(3)},..,{~~y(k-3)~~,z(k-1),z(k)}}
  - c is satisfied if and only if all clauses in S(c) are satisfied

Intaractive proof for #3SAT
Step 1: Arithmetization

- F(X(1),X(2),…X(n)) = a logical function in 3CNF
  - C(i) = L(i,1) $\vee$ L(i,2) $\vee$ L(i,3)
  - F = C(1) $\wedge$ C(2) $\wedge$ .. $\wedge$ C(m)
- Transfrom F into a real function f
- For each literal L = L(i , j), we define l(i,j) = 1-x(k) if L=X(k) and l(i,j) = x(k) if L=~~X(k)~~
- c(i) = 1 - l(i,1)l(i,2)l(i,3)
- Observation: C(i) is satisfied if and only if c(i)=1
- F = c(1)c(2)…c(m): a polynomial in x(1),..x(n)

Number of solutions

- If F= (X(1) ∨X(2)∨ ~~X(3)~~)∧(~~X(1)~~∨~~X(3)~~∨X(4)),
- f= (1-(1-x(1))(1-x(2))x(3)) (1-x(1)x(3)(1-x(4)))
- Number of solutions of F=1 is 8 + 6 = 14.
- Define $$\# f = \sum_{x(1)=0}^{1} \sum_{x(2)=0}^{1} \dots \sum_{x(n)=0}^{1} f(x(1), x(2), .., x(n))$$
- Then, # of solutions of F=1 equals #f
- Computation is difficult, but your professor who says he can compute #f for any f.
- Ask the professor "What is #f", and he answers that "its value is s ". (say, "its value is 2487000")
- Verify whether he tells a truth.

Key idea

$$\# f = \sum_{x(1)=0}^{1} \sum_{x(2)=0}^{1} \cdots \sum_{x(n)=0}^{1} f(x(1), x(2), .., x(n))$$

$$f_i(x(1), x(2), .., x(i)) = \sum_{x(i+1)=0}^{1} \sum_{x(i+2)=0}^{1} \cdots \sum_{x(n)=0}^{1} f(x(1), x(2), .., x(n))$$

Lemma

- $f_0 = \#f$
- $$f_1(x(1)) = \sum_{x(2)=0}^{1} \sum_{x(3)=0}^{1} \cdots \sum_{x(n)=0}^{1} f(x(1), x(2), .., x(n))$$
- $f_n(x(1),..x(n)) = f(x(1),x(2),..,x(n))$
- $f_{j-1}(x(1),\ldots,x(j-1)) = f_j(x(1),..,x(j-1), 0) + f_j(x(1),..,x(j-1), 1)$

Basic (failing) strategy 1

- Question 1: What is the value of f_0?
  - Professor answers : 247800
- Question 2: What are $f_1(0)$ and $f_1(1)$
  - Professor answers: 4000 and 243800
  - You check $f_0 = f_1(0) + f_1(1)$
  - If check fails, professor tells a lie: The end.
  - Else, guess which of two values is wrong…..
- Question 3: What are $f_2(0,1)$ and $f_2(0,0)$?

Basic (failing) strategy 2

- Question 1: What is the value of f_0?
  - Professor answers : 247800
- Question 2: What is the function $f_1(z)$?
  - Professor answers: $222800z^6 + 12000\,z^5 + 4000$.
  - You check $f_0 = f_1(0) + f_1(1)$
  - If fail, professor tells a lie: The end. Else, continue
- Question 3: What is the function $f_2\,(x(1),x(2))$?
- Proceed this process

Successful strategy

- Question 1: What is the value of f_0?
  - Professor answers : 247800
- Question 2: What is the function $f_1(z)$?
  - Professor answers: $g_1(z) = 222800z^6 + 12000\ z^5 + 4000$.
  - You check $f_0 = g_1(0) + g_1(1)$
  - If fail, professor tells a lie: The end. Else, continue
- Select a random value r, and compute $g_1(r)$
  - say, $r = 367$
- Question 3: What is the function $f_2(367, z)$
  - Professor answers: $g_2\ (z) = 34800\ z^5 + 34900\ z^2 + 403000$
  - You check $g_1(r) = g_2(0) + g_2(1)$
- Next, select another random value r', and compute $g_2(r)$
- CONTINUE

Analysis

- Al l functions are considered in GF(p) for a prime $p > 2^n$
- What is the probability that $g_i(z)$ is not $f_i(z)$ but $g_i(r) = f_i(r)$?
  - In other words, algorithm does not detect the lie in the i-th step
- Error probability is at most 3nm/p, VERY SMALL
- So if the professor tells a lie, the system detects it with high probability.

PCP

PCP (probabilistic checkable proof)

- Instead of god, we give a written proof.
- For a NP problem, we have a proof of length n
  - But, verifier wants to save time to verify
  - You prepare a proof such that verifier can easily verify the correctness
- This is just like database query!
- Like a database, we prepare the proof in a nice structure.
  - We need help of randomness and error correcting code

A puzzle

Captain cook hided a great treasure, but he need to hide for a long time to prevent from arrested.

He will send letters to his 20 pirates to inform the location of the treasure, but they are only reliable if they watch each other.

So, he want to encode the secret key so that it is revealed if and only if 11 or more meet.

How he should do?

Popular error correcting codes

- Reed solomon code
  - Your CD is encoded by using it
  - Use a polynomial on a field F
    - F= GF(2^q) in practical implementation
    - Here, we use GF(p) for a prime p
- Hadamard code
  - Use randomness
  - We can correct very large error.

Reed Solomon code

- a(1)a(2),..,a(k) :the key we want to send
  - Each a(0) is a member of F, thus a large number.
- Let $F(x) = a(k)x^k + a(k-1)x^{k-1} + .. + a(1)x + a(0)$
- We randomly select m>k values x(i) and let y(i)=F(x(i)).
- As for captain Cook, k=10, m=20
- Send (x(i), y(i)) to the i-th pirates for i=1,2,..,20