ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 345E
LOGIC & COMPUTABILITY

Question 2:
Provide a detailed proof sketch of how the PCP Theorem implies the hardness of approximating the Max-3-SAT problem to within a factor better than 7/8.

150210097 : Ali Emre Kaya

FALL 2024

Contents

1 Handwriten Report

Question 2

The PCP (Probabilistic Checkable Proofs) theorem states that it is possible to verify the correctness of solution to problems in NP class using a constant number of checks. Theorem can be expressed as:

$$NP = PCP(O(\log n), O(1))$$

This means that the correctness of an NP problem's solution can be verified with a logarithmic number of queries in input size and a constant number of verification steps.

The goal of MAX-3-SAT problem is maximizing the number of satisfied clauses in a 3-CNF formula (each clause exactly has 3 variable).

In MAX-3-SAT problem, the probability of each a clause being satisfied in a random assesment is calculated as follows:

- to make a clause unsatisfied, all 3 variable should be false:

$$\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8} \text{ (unsatisfied)} \longrightarrow 1 - \frac{1}{8} = \frac{7}{8} \text{ (satisfied)}$$

If we approach this problem with PCP theorem, a gap emerges among the satisfiable clauses. If "yes", all clauses are satisfied; If "no", maximum ratio of satisfied clauses are $\frac{7}{8}$. The gap is $\frac{7}{8} < (GAP) < 1$. The critical question is if that problem solvable in $\frac{7}{8} + \epsilon$ $(\epsilon > 0)$; according to PCP theorem this is NP-hard, it is only solvable when $NP = P$. The PCP theorem implies that MAX-3-SAT cannot be approximated to within a factor better than $7/8$ unless $NP = P$.

Ali Emre Kaya
150210097

2 Additional Report

2.1 Introduction

In this report, I tried to explain what are the PCP (Probabilistically Checkable Proofs) Theorem and the MAX-3-SAT problem, and in order to demonstrate how the PCP Theorem implies that the Max-3-SAT problem is difficult to approximate to a factor better than 7/8, I attempted to present a thorough proof sketch.

2.2 PCP Theorem and MAX-3-SAT Problem

2.2.1 Polynomial Complexity Classes

In logic and computability theory, problems are classified into categories such as P, NP, NP-complete, and NP-hard. P consists of problems that can be solved efficiently in polynomial time. NP includes problems in P and those for which solutions can be verified in polynomial time, even if finding the solution may not be efficient. NP-complete problems are a subset of NP that are the hardest within this class; solving any NP-complete problem efficiently would mean all NP problems could also be solved efficiently. NP-hard problems are at least as difficult as NP-complete problems but are not necessarily in NP, meaning they might not have verifiable solutions or even solutions at all.

2.2.2 PCP Theorem

PCP (Probabilistic Checkable Proofs) Theorem is a theorem which enable to control a solution is valid or invalid with only looking a part of solution. The validation of a solution of any NP problem can be found by a fixed number of random queries, with a small margin of error. Relation between NP and PCP may show like that:

$$NP = PCP(O(log(n)), O(1))$$

O(log(n)) denotes the number of bits which selected randomly when validation operation run, it is logarithmic. O(1) denotes required number of bits for answer control, it is constant, that mean it only look the little part of the answer. In the classical NP to validate the solution, all solution should controlled, but in PCP controlling only the part of solution is enough to say this solution is true with a small possible error.

Assume L $\in$ NP and x is a solution for that problem:

- If x $\in$ L, there exists a proof such that the verifier always accepts
- If x $\notin$ L, any proof will be rejected with high probability

PCP approximation is used to prove the hardness of approximating the solution to certain problems.

2.2.3 SAT, 3-SAT, MAX-3-SAT

MAX-3-SAT is an optimization problem which derived from 3-SAT. 3-SAT is a special case of SAT (Boolean Satisfiability Problem) problem. SAT is a method to validate there is a solution for a logical expression like: $X_1 \wedge X_2$, in there if both values are true then the expression is true. A more complex example in SAT is:

$$(X_1 \vee X_2) \wedge (X_1 \vee X_3 \vee X_4 \vee X_5) \wedge (X_2)$$

SAT systematically explores all possibilities to find a satisfying assignment, which makes the problem verifiable, not necessarily solvable in polynomial time. 3-SAT is a special case of SAT where each clause must contain exactly three literals (variables or their negations), such as:

$$(X_1 \vee X_2 \vee X_3) \wedge (X_1 \vee X_3 \vee X_4)$$

Formule must be writen in CNF (Conjuctive Normal Form). CNF is prefered for SAT dolvers because of the efficiency with algorithm like DPLL (Davis-Putnam-Logemann-Loveland) and CDCL (Conflict-Driven Clause Learning).

DPLL is a method for solving SAT problems that tries to find a satisfying solution for a logical expression using prediction (variable assignments) and backtracking when conflicts arise. CDCL is an advanced extension of DPLL. It also uses backtracking, but when it encounters a conflict, it learns from the conflict by adding a new clause to the formula that represents the negation of the conflicting condition. This learned clause prevents the solver from revisiting the same conflict, making the process more efficient.

The MAX-3-SAT problem is a special case of the 3-SAT problem. It seeks to find the maximum number of clauses in a 3-SAT formula (in CNF) that can be satisfied simultaneously. MAX-3-SAT is an NP-hard problem. Works using the PCP theorem has shown that the MAX-3-SAT problem can be approximated with a ratio of 7/8, meaning that there exists a polynomial-time algorithm guaranteeing at least 7/8 of the clauses are satisfied. Some theorems to approach MAX-3-SAT problem are those; Asano-Williamson 2000, Karloff-Zwick 1997, Håstad 1997. Some algorithms to solve MAX-3-SAT are those; Johnson's algorithm, Monte Carlo algorithm, Las Vegas algorithm.

2.3 Detailed proof sketch of how the PCP Theorem implies the hardness of approximating the Max-3-SAT problem to within a factor better than 7/8

The time complexity of brute force solution for MAX-3-SAT problem is $O(m2^n)$. m is the number of clauses and 2^n is the all possible inputs. Bunch of methods can use to show how the PCP Theorem implies the hardness of approximating the Max-3-SAT problem to within a factor better than 7/8.

2.3.1 Randomized

If Z_j is random variable and the C_j is the clause, that can be writable:

$$Z_j = \begin{cases} 1 & \text{if clause } C_j \text{is satisfied} \\ 0 & \text{if clause } C_j \text{is not satisfied} \end{cases}$$

That equality in the above is obvious:

$$\text{Pr}[C_j \text{ is satisfied}] = 1 \text{ - } \text{Pr}[C_j \text{ is not satisfied}]$$

C_j is not satifsies when all the literals in the clause are false. Because of we study in clauses with 3 literals the probability of all literals are false in the clause is $1/2^3$ which is 1/8.

$$\text{Pr}[C_j \text{ is not satisfied}] = 1/8$$
$$\text{Pr}[C_j \text{ is satisfied}] = 1 \text{ - } 1/8 = 7/8$$

Formula can be generalized for multiple clauses with using summation, E is the expected value:

$$\text{E}[Z_1] = 7/8$$
$$\text{E}[Z_j] = \textstyle\sum_{i=1}^{m} 7/8 = 7\text{m}/8$$

2.3.2 PCP

If we approach MAX-3-SAT problem with PCP theorem, a gap emerges among the satisfiable clauses.

- If 'YES' : all clauses are satisfied
- If 'NO' : maximum ratio of satisfied clauses are 7/8

The gap is that:

$$7/8 < \textbf{GAP} < 1$$

The critical question is that if that problem solvable in **7/8 + e** where (e>0), according to PCP theorem this is NP-hard, it is only solvable when P = NP.

2.4 Conclusion

The PCP theorem implies that MAX-3-SAT cannot be approximated to within a factor better than 7/8 unless P = NP.

REFERENCES

[1] A. Imdad. Random max-3-sat and derandomization. https://web.lums.edu.pk/ imdad/pdfs/CS510$_{S}lides/CS510 - slides - 26 - 07 - Random - Max - 3 - SAT - Derandomization.pdf$, 2024. $Accessed :$ 2024 − 12 − 13.

[2] S. Palash. Algorithm design and analysis: Max-3sat. https://cse.iitkgp.ac.in/ palash/2018AlgoDesignAnalysis/MAX3SAT$_{S}LIDES.pdf$, 2018. $Access$ 2024 − 12 − 13.

[3] Howard Karloff and Uri Zwick. A 7=8-approximation algorithm for max 3sat. https://www.cs.umd.edu/ gasarch/BLO/GPAPERS/max3satu.pdf, 1998. Accessed: 2024-12-13.