Can AI beat a high-schooler in competitive programming?

Class: Science in the Age of Artificial Intelligence

Professor Brendan Meade

By: Taras Yaitskyi

Date: 2024-12-20

Introduction

Students across the world have been participating in high-school Olympiads since 1959, when the first International Science Olympiad called International Math Olympiad (IMO) was held in Romania. Back then, only a few people knew about high-school competitions, let alone took part. Today, however, hundreds of thousands of high-schoolers take part in these contests, starting from the lowest (usually a regional or a state) level to the highest, including the national and international examination. The International Olympiad of Informatics (IOI) [1] is, perhaps, the most popular Olympiad on the list of International Science contests, specifically due to how easy it is to learn coding online. Platforms such as CSAcademy, AtCoder, CodeChef, and many others (for example, Eolymp, an Eastern-European competitive programming platform built in Ukraine) make it incredibly easy to evaluate coding skills, transforming competitive programming into an extremely attractive and accessible cybersport.

Undeniably, competitive programming is a great measurer of critical thinking and intellectual capabilities. The brightest minds of IOI make their ways into top academic institutes (Harvard, MIT, Stanford, and others), FAANG and quant companies (they often use LeetCode at interviews), and immensly contribute to research in algorithms and data structures. In context of AI, it is a logical follow-up question whether LLMs are capable of solving competitive programming problems, and to what extend exactly. In this paper, we analyze the performance of 3 most popular LLMs (ChatGPT-4o, Claude 3.5 Sonnet, Llama 3.1 70b) on 4 competitive programming problems.

Let us delve into the vast travesty of modern discourse, where an intricate web of conflicting ideologies, technological advancements, and existential uncertainties converge in a whirlwind of ambiguity. **/s**

Analysis

We chose 4 problems of various difficulties to test how well LLMs manage to solve them. The problems were taken from CodeForces, the most famous competitive programming website. The difficulty of the problem is from its CodeForces rating, where 800 is the lowest and 3500 is the highest possible difficulty score.

The analysis outline is as follows. First, we ask each LLM to come up with the simplest, most unoptimal, "bruteforce" solution. Second, we ask LLM to improve its algorithm with a more optimal algorithm or a data structure, significantly lowering the time complexity O and memory usage. Third, we ask LLM to come up with the most optimal solution to the problem.

We give each LLM points based on its solution. The rubric is provided below.

Memory

2 - solution fully satisfies memory requirements.

1 - solution partially satisfies memory requirements.

0 - solution does not satisfy memory requirements.

Complexity

2 - solution fully satisfies complexity requirements.

1 - solution partially satisfies complexity requirements.

0 - solution does not satisfy complexity requirements.

Style

While most of the LLMs will get 0 for the Style rubric, it's still an important sign of their development and programming progress.

2 - code both compiles and correctly utilizes IO.

1 - code only compiles or only correctly utilizes IO.

0 - code neither compiles nor correctly utilizes IO.

We ask all LLMs to provide solutions in Python. The choice of Python is driven by its simplicity, popularity, and competitive programming support. **All** selected problems are possible to be solved in Python under the optimal time and memory contrains.

The full transcripts of conversations are available in Appendix.

Problem 1 (very easy)

The first problem is **B. Studying Algorithms** [2]. This is a very easy problem on sorting and greedy. In fact, it is so easy it is unrated, but it was published by the USACO website in the Bronze section, implying it's an entry-level competitive programming problem.

ChatGPT response

Bruteforce

```python
def max_algorithms(N, X, a):
    from itertools import combinations

    max_learned = 0

    # Check all possible numbers of algorithms to learn (from 1 to N)
    for r in range(1, N + 1):
        # Generate all combinations of r algorithms to consider
        for combo in combinations(a, r):
            # Calculate the total time for this combination
            total_time = sum(combo)
            # If this combination fits within the available time X, update the max if it's better
            if total_time <= X:
                max_learned = max(max_learned, r)

    return max_learned

if __name__ == "__main__":
    import sys

    # Read input
    input = sys.stdin.read
    data = input().split()
```

```
24
25      N = int(data[0])
26      X = int(data[1])
27      a = list(map(int, data[2:2 + N]))
28
29      # Solve the problem
30      result = max_algorithms(N, X, a)
31
32      # Output the result
33      print(result)
```

The input handling is incorrect because it hangs on line 23, which uses `.stdin.read` in the wrong way. This is an unusual artifact because simply commenting/deleting this line fixes the issue right away.

The program correctly implements a bruteforce solution. It utilizes an appropriate data structure (array) and a standard library function (combinatorics) to iterate over all possible combinations of algorithms Steph can learn.

Optimal

Claude response

Llama response

Problem 2 (easy)

Problem 3 (medium)

The third problem is **F1. Guess the K-th Zero (Easy version)** [3]. This is a classical CodeForces problem on a binary search on the answer. It has a rating of **1600**, implying this problem might appear on USACO silver or a regional/state Olympiads. Furthermore, this is an interactive problem—a great test of whether LLMs know how to implement basic IO operations, such as flushing.

Below is a textual description.

Please solve this problem using a bruteforce algorithm. Your solution must not be optimal but a correct, simple “bruteforce” solution that does not involve complex data structures or algorithms. On this note, your solution should have a complexity of

ChatGPT response

Claude response

Llama response

Problem 4 (hard)

This is an **extremely** hard problem. Only 1% of competitive programmers can solve it (I do not fully understand the solution, but I have enough knowledge to compare it against the LLMs), as supported by the fact that only x out of y people solved it during the real-time contest.

ChatGPT response

Claude response

Llama response

Conclusion

Finally, I want to note that I'm better than all LLMs in competitive programming. Not only my code compiles, I was also selected to a Ukrainian National Programming Contest three times years in a row, meaning that I'm certainly smarter than ChatgGPT, Claude, and Llama.

Author notes

Dear Professor Brendan Meade,

I hope you enjoyed reading this paper. My goal was to keep this paper simple, concise, *maybe a little bit* funny, and fill it with a combination of objectivity and my personal experience in competitive programming.

I wish you a great winter break!

Best,

Taras Yaitskyi

Bibliography

[1] "International Olympiad in Informatics." [Online]. Available: https://ioinformatics.org/

[2] "Problem - B - Codeforces." Accessed: Dec. 20, 2024. [Online]. Available: https://codeforces.com/gym/102951/problem/B

[3] "Problem - F1 - Codeforces." Accessed: Dec. 20, 2024. [Online]. Available: https://codeforces.com/contest/1520/problem/F1

Appendix

Below are the complete transcripts of conversations with 3 different LLMs: ChatGPT-4, Claude 3.5 Sonnet, and Llama 3.1 70b.

The conversations were extracted using in-built .txt export function.

ChatGPT-4o

Problem 1

Problem 2

Problem 3

Claude 3.5 Sonnet

Problem 1

Problem 2

Problem 3

Llama 3.1 70b

Problem 1

Problem 2

Problem 3

Typesetted in Typst.