What Lothar Collatz Thinks of the CMU Computer Science Curriculum

Gabriel Chuang (gtchuang@andrew.cmu.edu), Brandon Wu (bjwu@andrew.cmu.edu)
Carnegie Mellon University

***Abstract*— Judging a course by its five-digit course code (of the form 12-345) is a very difficult task; much effort is expended in classifying courses in various pseudo-mathematical ways. In this work, we introduce a *truly* mathematically-founded, rigorous method for classifying Computer Science courses, based on some ideas of Lothar Collatz, and discuss what exactly Collatz is attempting to tell us from beyond the grave about the Computer Science curriculum at CMU.**

I. Introduction

Courses at Carnegie Mellon University are classified in a variety of mathematically-adjacent ways [1]. However, many of these classifications are ill-defined, leading to much ambiguity and debate. Is graphics a systems course? Is it accurate to say that 15-251 is "Concepts 2.0" ? Should 15-281 and 15-259 be re-promoted to 300-level courses? Should Interp be a prereq for literally every course?

In this work, we propose a classification system to provide clarity on these fronts. First, we will discuss prior work on classification of computer science courses. We will then introduce some relevant mathematical background, before introducing our proposed equivalence-class-based classification system. Finally, we will discuss some implications of our system, and suggest some administrative changes to be made to the requirements of the CS curriculum at Carnegie Mellon.

II. Prior Work

Several mathematically-grounded classification methods already exist for classifying courses. For instance, the subject matter of a course is often determined by evaluating

$$n = \lfloor \text{course number}/1000 \rfloor \tag{1}$$

where a mental mapping is kept that associates numbers n with subjects, such as "$n = 15$ means CS" or "$n = 21$ is math" or "if it's anything else, it's a gened and I don't remember." [2]

Another common evaluation criterion is of the form

$$\text{difficulty} = \lfloor \text{course number}/100 \rfloor \mod {}^{1}10 \tag{2}$$

This style of evaluating course difficulty is often used to make administrative decisions such as barring freshmen from taking more than one of $\{15251, 15213, 15210\}$ ("We will be reviewing schedules post registration and will drop students from classes if [freshmen] sign up for more than one from: 15251, 15213, 15210." [3]).

However, these existing notions leave much to be desired, both in precision and clarity.

[1]Note that we abuse notation here to mean the "programmer's view of mod", that is, "take the remainder when you divide by 10."

III. Background: The Collatz Conjecture

The Collatz sequence dates back to 1937, and was originally proposed by Lothar Collatz [4]. It is also variously known as the hailstone sequence, the $3n + 1$ sequence, and the *dear-god-please-stop-writing-out-the-expansion-for-871* sequence[2].

Consider the following operation:

$$f(x) = \begin{cases} \frac{x}{2} & \text{if } x \text{ is even} \\ 3x + 1 & \text{if } x \text{ is odd} \end{cases} \tag{3}$$

Collatz's conjecture is as follows: Starting with any number x, repeatedly apply f to it; you will always (eventually) reach 1. It is currently proven to be true[3,4] [5].

For example, consider the number 150. The corresponding *Collatz sequence* would be:

$$150, 75, 226, 113, 340, 170, 85, 256, 128, 64, 32, 16, 8, 4, 2, 1$$

Note that there are 16 terms in this sequence. We will call the number of terms in a number's Collatz sequence its *Collatz number*. So, the *Collatz number* of 150 is 16. We will abbreviate this as $C_n(\cdot)$, i.e. $C_n(150) = 16$.

IV. Proposal and Methods

We define an equivalence relation $\sim$ on CMU course numbers as follows:

$$c_1 \sim c_2 \triangleq C_n(c_1) = C_n(c_2) \tag{4}$$

That is, two courses are related if their course numbers have the same Collatz number.

The proof that this is an equivalence relation is trivial and left as an exercise to the Concepts students.

We computed the Collatz number for a range of Computer Science (15-xxx) courses using the following SML function:

```
fun coll 1 = 1
  | coll n = 1 + (if n mod 2 = 0
                  then coll (n div 2)
                  else coll (3*n+1))
```

[5]

The results, sorted by course number, are displayed in Table I.

There are a few notable equivalence classes of $\sim$, which are displayed in Figure 1.

[2]Only a few people call it this.

[3]Give me a counterexample. Can't find one? Then it must be true. Obviously. Proof by lack of counterexample. □.

[4]I don't want to *actually* put any untruths in this paper, so it's actually currently unproven.

[5]Hey 150 students: Prove totality of `coll`.

course #	colloquial name	$C_n(\cdot)$
15110		134
15112		85
15122		85
15150		85
15151	Concepts	85
15210		33
15213		85
15251		85
15252		33
15259	PnC	85
15260	SnC	178
15281	AI	33
15300		41
15312	PL	59
15317	Clogic	59
15330	Security	59
15351		178
15354	CDM	134
15356	Crypto	116
15410	OS	147
15411	Compilers	147
15414	Bug Catching	72
15417	HOT Comp	147
15418	Parallel	54
15440	Distributed	28
15445	Databases	28
15451		59
15455	UCT	90
15459	Quantum	147
15462	Graphics	147
15751	Toolkit	147

TABLE I: C No., CN, and C_n for a range of CS courses. Note that most of the CS core has $C_n(\cdot) = 85$.

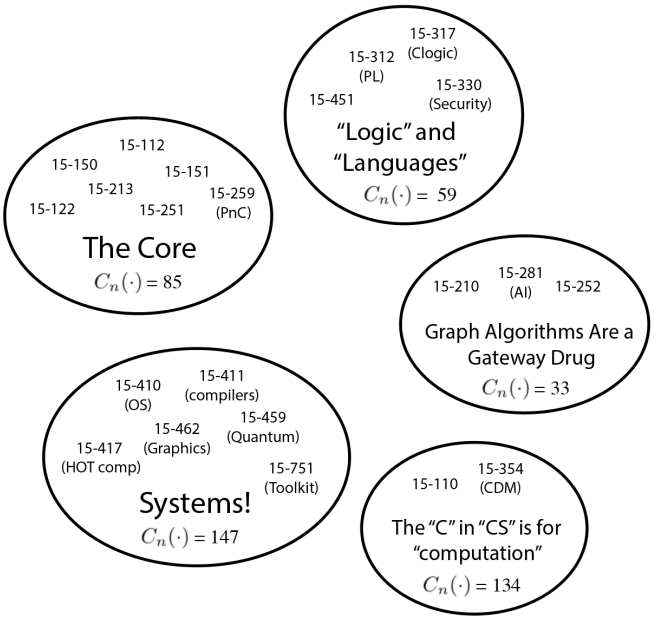

Fig. 1: The non-singleton equivalence classes under $\sim$. Notice that A Theorist's Toolkit, contrary to its name, is in fact a systems course.

V. Results and Discussion

There are two large equivalence classes under $\sim$, and a handful of smaller ones.

A. The Core

Astonishingly enough, every important core of the CS curriculum, other than 451, has a Collatz number of 85. Specifically, 15-112, 122, 150, 151, 213, and 251 all converge to 1 in 85 steps, joined by 15-259 (PnC).

The probability of this occuring by random chance is astronomically low[6]. Therefore, we can safely conclude that one of the following is true:

(1) Collatz was a precognitive psychic with a very specific interest in Carnegie Mellon's CS course numbers and a dislike for algorithms classes.
(2) CMU SCS admin is a bunch of nerds with a penchant for choosing course numbers to satisfy arbitrary mathematical properties.

Clearly, (2) is absurd, and so we must accept (1) as fact. One can also note that the Collatz number of the CS core curriculum is 85, which is also the prefix for the psychology department at CMU, only further suggesting that Collatz was a psychic.

Correspondingly, we propose that the CS curriculum at CMU be amended to require PnC to be taken by all students, just as the other courses in this equivalence class are. We also recommend that, since it is clearly not meant to be part of *The Core* by Collatz, 15-210 is removed from the core or otherwise suitably renumbered so as to comply with Collatz's categorizations.

B. Systems!

By virtue of Operating Systems (15-410) and Compiler Design (15-411) being in the same equivalence class, the class with $C_n(\cdot) = 147$ is obviously the class of systems courses. This is further supported by the membership of HOT Compilation (15-417), which, since it has "Compile" in its name, is clearly a systems course [7], and Computer Graphics (15-462), thus settling that Graphics is in fact a systems course.

Some may balk at the fact that 15-459, Quantum Computation, is in the Systems group. This suggests an immediate need to align Quantum's curriculum to focus more heavily on techniques for programming on (all 0) existing state-of-the-art quantum computers [8], and focus less on the theoretical foundations for quantum computing. After all, it is a well-established fact that all CS theory exists only to be applied to real-world systems[7] [9].

Perhaps the most controversial member of the $C_n = 147$ group is 15-751 (A Theorist's Toolkit), given the centuries-long Great Theory-Systems Schism[8]. However, the authors

[6]Proof: C_ns are kinda independent and kinda uniform. So the probability that a given $C_n(\cdot) = 85$ is like, $\frac{1}{n}$. Here, we had six of them, so the odds are $\frac{1}{n}^6$ which is basically 0.

[7]I type this from my laptop, which runs on Church's λ-Calculus.

[8]"Living in a world of mathematical abstraction is infinitely nicer than dealing with the limitations of real-world systems. Who *likes* dealing with underflow and overflow?" - St. Thomas Aquinas, *Summa Theologica*, 1265

have yet to take the course; a brief glance at the topic list shows topics such as "Analysis of Boolean Functions", which is tantamount to electrical engineering [10]. Thus, 15-751 is solidly placed into the Systems category.

C. "Logic" and "Languages"

Given that the two most popular logic and languages electives, 15-312 and 15-317, are in this group, it is clear that Collatz suggests a revamping of the logic and languages elective category to include two new courses, each discussed below.

15-451, Algorithms, is clearly a good fit for the "Languages" portion of "Logic and Languages", given the vast breadth of languages they permit and encourage. Since PL theory is merely the discussion of the merits of various programming languages [11], 451's proliferation of languages is a helpful step towards students' understanding of PL.

15-330, Computer Security, needs only slight reworking to fit under the new Logic and Languages category. We recommend that encryption be taught under the "exceptions-are-secrets" paradigm [12], and all systems-level real-world applications of the course be summarily purged. This would be only a small modification, and we expect that it can be be easily implemented.

D. Graph Algorithms Are a Gateway Drug

Every self-respecting CS student learns graph algorithms at least six times over the course of their career [13]. Collatz's classification makes graph algorithms into their own elective category, since such a central set of algorithms must be given its rightful place in the curriculum. As such, we recommend that students be required to take 15-210, 15-281, or 15-252, three courses that focus extensively on graph algorithms.

Notably, this removes 15-210 from being a core class. The authors decline to comment further on this choice of Collatz's.

E. The "C" in "CS" is for "Computation"

As the authors have taken neither of the courses with $C_n(\cdot) = 134$, we can only speculate at the connection that Collatz's function suggests. Based on a thorough and in-depth researching of both courses' names, we conclude that both courses center on "computation" (after all, they're named "Principles of Computing" and "Computational Discrete Math") which is surely related to the "C" in "CS" [14]

VI. Conclusion and Future Work

Frankly, the authors are astonished at the fact that Collatz had the foresight to choose a function that so neatly categorizes Carnegie Mellon SCS course numbers. Putting OS with compilers, Clogic with PL, and 150/151/112/122/251/213 together cannot merely be a coincidence; we strongly urge the administration and faculty to seriously consider the reorganizational proposals presented in the paper to more closely adhere to the prescriptions of Collatz.

It is unlikely that Collatz's precognitive interest in computer science was limited to a single institution in a country he never lived in. Given sufficient grant money, the authors would be willing to conduct a similar study on Collatz's classifications for other departments at Carnegie Mellon or at other universities entirely.

Given Collatz's precognition, it may also be worth trying to find other patterns in the Collatz Sequence. As it may be difficult to recognize patterns corresponding to events that have not occurred yet, the authors recommend searching for instances corresponding to famous or highly profitable past events, such as the explosion of Bitcoin in 2013 or the 2020-21 COVID-19 pandemic.

VII. References

[1] G. Chuang, B. Wu, "What Lothar Collatz Thinks of the CMU Computer Science Curriculum," SIGBOVIK 2021. Pittsburgh, PA, USA. 2021.[9]
[2] We don't actually
[3] have any other references;
[4] we just put some
[5] bracketed numbers
[6] wherever we made a claim
[7] that might seem like
[8] it could need
[9] a source.
[10] Hopefully the reviewers
[11] don't notice.
[12] Nobody looks at
[13] the references anyway
[14] right?

[9]This was the only credible source we could find for several of the claims in this paper.