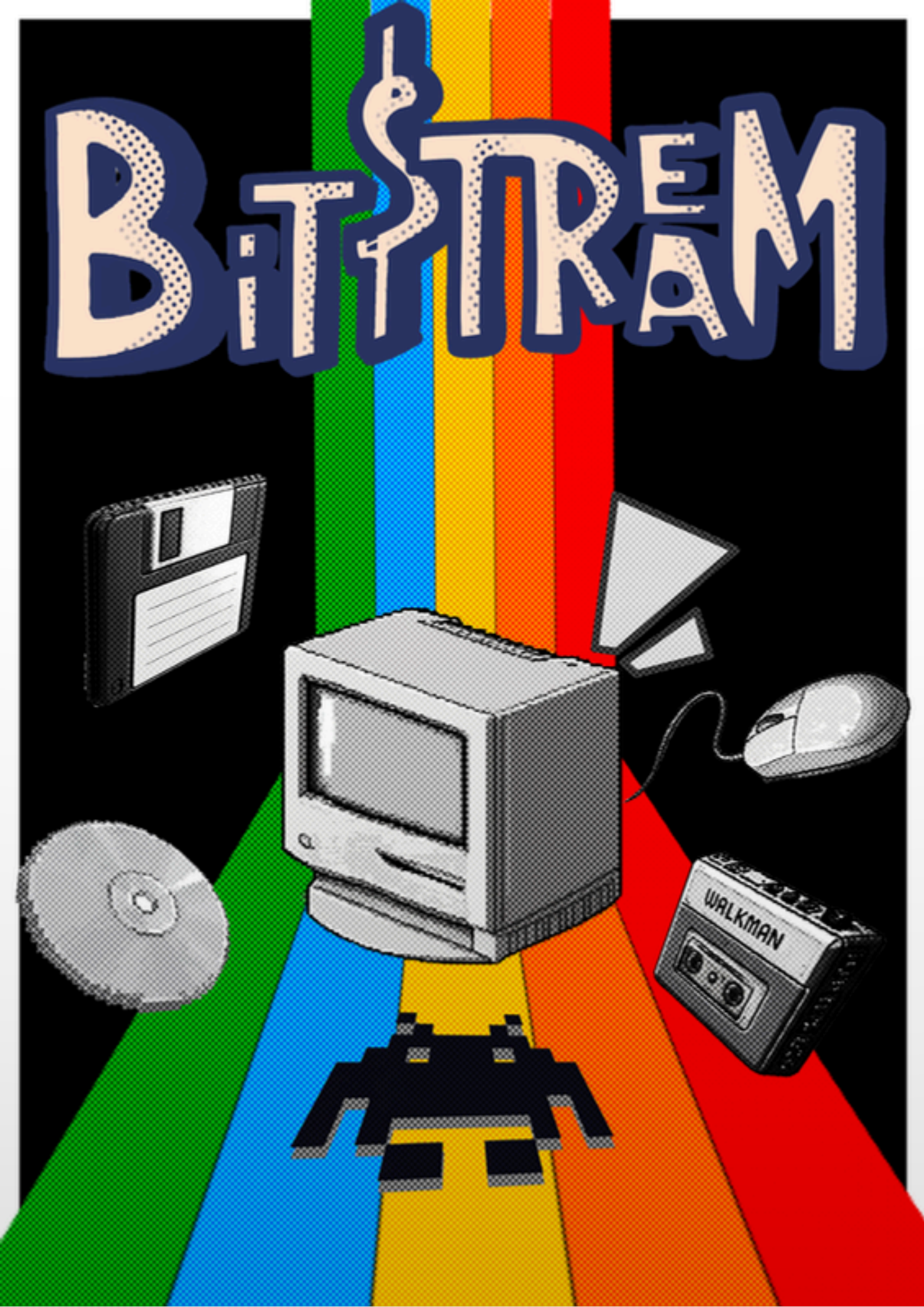

BiTSTREAM
WALKMAN



CONTENTS

SCOREBOARDS AND SOCIALS 04

THE THIRD WAY 06

CP101 08

STUDENT RESEARCH SHOWCASE 10

HEAR FROM ALUMNI: MADHAV GOEL (BTECH 19) 12

HOW'S THE STACK DOIN? 14

WHAT DOES A QUANT DO? 18

MAINTAINING CONNECT WITH A LARGE CLASS 20

LLMS AND THE WAY FORWARD 22

CSE BINGO 24

BAPTISING INTO THE DEPARTMENT: SQUARE ONE 26

IN MEMORIAM 29

EDITOR'S NOTE 30

MEET THE TEAM 31

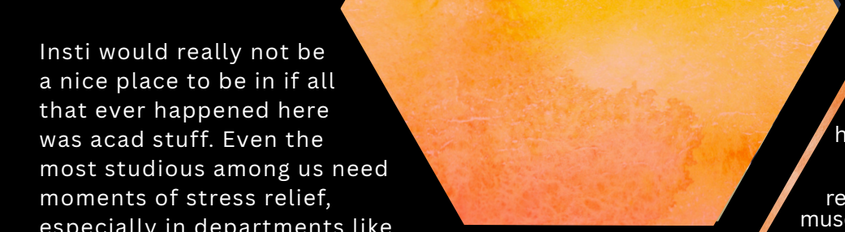
TURN THE PAGE FOR SOME AWESOME CONTENT

Scoreboards and S(#)CIALS

credits to :

CSEA

Insti would really not be a nice place to be in if all that ever happened here was acad stuff. Even the most studious among us need moments of stress relief, especially in departments like CSE, where the academic load matches that of other notoriously demanding branches such as Electrical and Mechanical. Freshers must have already gotten a taste of this reality (cough cough CS105).



Freshers' Welcome being only the beginning of some great things to happen (insert dramatic music), multiple sports tournaments were also organized for us to realize how we lost touch with our muscles after JEE (just kidding). Events were organized for various sports/games including Table Tennis, Badminton, Chess and Kho Kho. With impressive participation across batches, the events saw intense matches and friendly rivalries emerge and being settled on the field.



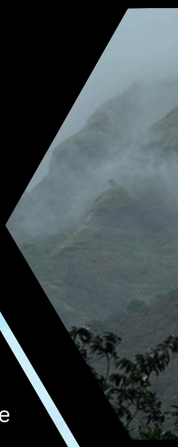


CSEA, the beloved student council of our department, makes it a point to organize regular and fun events which somewhat soften the blow of heavy workload. As soon as the blackout period is over, first years are invited to the Freshers' Welcome, serving as their first proper interaction with their 'very nice' seniors.









This year on 25th October, stage was set on fire with some electrifying performances, including multiple dance groups, a beautiful musical band performance and a magic show. The event also featured the council introduction, results of various google form responses from freshers collected by seniors and a special party dedicated to our own very popular Sports Secy, Dhruvin Doshi.



Between debugging code, surviving deadly courses, and convincing ourselves that sleep is optional, these events gave us something invaluable, that is, balance. Thanks to CSEA, we were reminded that insti life isn't just about grades and grind, but also about laughter, teamwork, and the occasional sprint on a Kho Kho ground. Here's to more such breaks from reality, more memories, and hopefully, fewer all- nighters (though we know how that usually goes).









Following this train of thought, CSEA organized a trek to Ratangad Fort. However, due to storms in the area, the destination was changed to Kalsubai Peak, highest peak in Maharashtra. For many, it may have started as just another trip. But the lush green trails, awesome weather and scenic views are bound have left a mark on each traveller.



Thinking about problem sets and errors, one is forced to remember a terrible entity called exams. And the first major instance of these in the semester are the mid semester exams. People losing their sanity and brains cooking as if in a pressure cooker. Exams ending is fantastic, sure, but to take it up a notch, trekking with friends really has no other alternative.



Stealing the spotlight among these was the Interdepartmental Kho Kho Tournament between CSE and EE departments. The high octane gameplay was enough to excite each amd every one among the audience. Everyone was on their feet as they cheered on for their respective teams. These tournaments truly showed that life exists beyond problem sets and debugging errors





the third way

~ the road less travelled by

There's a specific moment when it hits you. You've cracked the interview everyone wanted. The offer letter has more zeroes than you ever expected. Your parents are already planning the celebration. But somewhere between the congratulatory texts and the LinkedIn post you're supposed to write, a different thought creeps in: What if there's something out there I could build 10x better? Or what if the idea I keep thinking about... just doesn't exist yet?

Some people graduate and take incredible jobs. Some go for research fellowships and PhD programs. And then there are them- the seniors you find at 3 AM in labs, not debugging assignment code, but deploying their own. They're not there because they couldn't crack the interviews- many of them had offers they simply chose to decline. They're gripped by this one thought they can't stop chasing, this stubborn belief that they've either spotted a problem no one's solving right, or seen a solution that's being executed badly. A thought that nags at them- and once it takes root, it doesn't let go.

They form the Third Way. Either coming up with a solution from scratch or executing something better than anyone else has. Not because the other paths aren't incredible- they definitely are. But because they can't unsee the problem. But here's the thing: they aren't some separate breed. They had the same lectures, same assignments, same options as you and me. What set them apart was the relentless pursuit of that one problem they couldn't let go of- and they decided to build the solution themselves. They chose to chase it, to build it, to see it through; no matter how messy or uncertain the path looked.

And the precedent has been set -

Beerud Sheth (B.Tech CSE 1991) built GupShup into a messaging infrastructure giant. **Ritesh Arora** and **Nakul Aggarwal** (both B.Tech CSE 2006) bootstrapped BrowserStack from these very corridors into a billion-dollar tool that half the internet relies on. **Bhavish Aggarwal** (B.Tech CSE 2008) took Ola from zero to a mobility empire that moved cities- literally.

The trend holds for the latest founders too. Composio by **Soham Ganatra & Karan Vaidya** (both B.Tech CSE 2017) and GreyLabs AI by **Aman Goel** (B.Tech CSE 2017), who already sold his first venture Cogno AI in a multi-million dollar deal, are building the tooling and infrastructure that makes AI actually usable.



Here's what's interesting: most of these weren't 'eureka' moments or brand-new inventions. They saw existing solutions, existing markets, existing problems; and believed they could execute better. That's the pattern that keeps repeating. It's rarely about being first. It's about being better.

And here's where we stand in the bigger picture -

Compared to other top engineering colleges, IIT Bombay has consistently been one of the leading sources of entrepreneurs. Though IIT Delhi has more unicorn founders on their roster, when it comes to the number of people actually choosing to build, IIT Bombay's at the top.

Part of it is the environment. The technical rigor here doesn't just teach you to code, it teaches you to think like a builder. Or maybe it's just that when you're surrounded by people solving hard problems, building your own solution starts feeling less crazy and more... possible.



SINE (the Society for Innovation and Entrepreneurship), IITB's startup incubator, has been central to this. Over two decades, it has built an 80% startup survival rate, which is huge compared to the global average of around 20%.

These numbers and stats help, sure. But the real question is, why take that leap at all?

Not everyone takes the research fellowship or the 1.5 Cr quant offer. Some have the courage to bet on themselves instead. And here's the reality of that bet: not every startup becomes a unicorn. Some stall at the prototype stage. Some never find a market. Some lose momentum. The failure is rarely dramatic- it's usually silent. But in the IITB CSE world, those quiet endings aren't viewed as failures, they're part of the story. They're proof someone chose to build.



So when placement season rolls around and everyone's comparing offers, there's a question you should ask yourself: Which story do you actually want to live?

The department gives us the skills, the algorithms, the theory. But the seniors who looked at the safe path and said 'maybe later'- that's what gives us permission to even ask that question.

And maybe, just maybe, that is the best product our department has ever created.

CP 101

WHAT IS CP?

Competitive Programming (CP) is a sport of problem-solving where participants write code to solve algorithmic problems under strict time and memory constraints. Unlike regular programming, CP is less about building large, reusable systems and more about thinking efficiently, identifying patterns, and translating ideas into correct code quickly. Problems often combine mathematics, data structures, and logical reasoning, making CP a powerful way to sharpen analytical thinking. Chances are, you've already encountered problems of this style in CS101 - CP simply pushes that thinking a step further.

HOW DO I START ?

For beginners, getting started may feel intimidating, but the entry barrier is much lower than it appears. Once you're comfortable with the basics of a programming language, you're ready to start. Pick a past contest on Codeforces, open the easiest problem, and just try. That's genuinely enough.

CP is easy to jump into, but it does take some adjustment. The problem statements can feel unnecessarily long, and the required thinking may seem unfamiliar at first. That's completely normal. Like any new skill - be it a sport, an instrument, or a course - it takes time, patience, and repetition. The only real requirement is showing up and trying.

More than anything, CP trains you to think clearly under pressure, which is a skill that carries far beyond contests.

WHAT DO CONTESTS LOOK LIKE ?

CP contests are typically held on online platforms such as Codeforces, CodeChef, and AtCoder. A contest usually lasts 2-3 hours and consists of several problems arranged in increasing difficulty. You submit solutions that are automatically tested against multiple test cases, and your performance is ranked based on correctness, speed, and penalties for wrong submissions. Most of the time, there's no partial marking - you either pass all the test cases for a problem and get points, or none at all.

Even after your very first contest, you're assigned a rating. While it's far from perfect, it serves as a tangible way to track progress. Watching that number slowly increase over time can be surprisingly motivating.

HOW DO I GET GOOD ?

Getting good at competitive programming is not about talent or intelligence; it is about consistency. Almost everyone struggles in the beginning, including people who are highly rated today. Plateaus are common, ratings fluctuate, and bad contests are inevitable.

What truly matters is regular practice, solving problems that feel slightly uncomfortable, and learning from mistakes. Instead of obsessing over rank, focus on understanding ideas and patterns. Improvement in CP is often slow and invisible - until one day, problems that once felt impossible suddenly feel manageable.

Competitive programming is not mandatory, and it's perfectly fine if you decide it's not for you. But giving it an honest try, especially early on, can be deeply rewarding. Whether or not you pursue CP seriously, the mindset it develops can benefit you in academics, research, internships, and careers across technology and beyond.

CULTURE ?

IIT Bombay may not traditionally be known as a CP powerhous, but the community is growing rapidly. You'll find freshers, seniors, and alumni solving contests together, discussing problems on WhatsApp, and helping each other debug solutions late into the night. What makes this community special is that resources are open, learning is shared, and progress is collective. People genuinely want each other to get better.

Beyond contests, CP builds skills that stay with you: handling pressure, thinking both independently and collaboratively, writing clean and debuggable code, and learning to accept failure without losing motivation.

Competitive programming may start as a way to solve problems, but for many, it ends up shaping how they think.



STUDENT REASRCH SHOWCASE

Four students. Four frontiers. A glimpse into the research shaping the future of computing at IIT Bombay.

✦ A NOTE BEFORE WE BEGIN

A fundamental pillar of any academic programme at college or graduate level is the opportunity for research. The CSE department at IIT Bombay provides it abundantly — through seminar courses and one-on-one R&D with faculty. In this showcase, we present the work of four peers who are quietly shaping the future of computing.

01 SQUEEZING THAT LAST OUNCE OF PERFORMANCE

Hari Bhavsar · Final Year, CSE · CASPER Group

Hari is a final year student in CSE, currently affiliated with the CASPER group, which works primarily on security and performance aspects of computing. Hari's journey in Computer Science research started during his second summer here; he mentioned gravitating toward computing systems and architecture after doing a course (DLDCA) taught by Prof. Biswa, specifically architectural attacks and the security vulnerabilities that result from cache level optimizations.

Companies, racing to squeeze the last drop from hardware, accidentally create room for vulnerabilities — later fixed by software patches that almost always lead to losses.

Therefore it is imperative that we do not lose the forest in saving the tree. Hari's solution was to select the optimal minimal subset of fundamental changes in design that would ensure a certain level of protection. Hari's work was published (and received the Distinguished Artifact Award) in **USENIX in 2025.** Since then, he has migrated over to Performance optimizations in Caches. In particular, this next project stemmed from the case where one is dealing with efficient caching for graphs, and their associated data structures. Their next optimization leverages the fact that it is prudent to maintain data in cache that is well connected to other data in the graph, since the probability of such data-points being operated upon together is higher. Therefore, if the caching algorithm can identify what Hari termed as **"hot vertices"**, that reduces the likelihood of cache misses.

Finally, there are two components to the problem, identifying hot vertices in a given context, as well as deciding how long to keep these vertices in cache. The algorithm has **withstood the trials** of standard benchmarks such as Google's PageRank. Hari is an incoming SDE at D.E. Shaw, but plans to return to **micro-architecture research** at some point.



■ Highlight

"Do not lose the forest in saving the tree."

USENIX '25
Distinguished Artifact Award
CASPER
Security & Performance Group

02 FORAY INTO AI FRONTIERS

Anvay · 3rd Year PhD · Prof. Shivaram Kalyanakrishnan



Anvay is a 3rd year PHD student working with Prof. Shivaram Kalyanakrishnan. He works with Markov Decision Processes and Reinforcement Learning. He's an avid Chess and Scrabble player. That being said, it's only natural for him to find convergence in his hobbies and his work! Here is, in his own words, a description of his work:

"What got me interested was Deepmind's AlphaZero — an AI that learned superhuman Chess and Go from scratch, given only the rules."

"I'm working on designing algorithms for agents that can play **imperfect information games** - games like Poker, Scrabble and Reconnaissance Blind Chess.I love figuring optimal strategies of doing things and anything that involves building an AI that does better than humans. While AI does well in Chess and Go (perfect information games where the exact game state is known to both players), **games like Reconnaissance Blind Chess are far more challenging for AI** due to the fact that each player has private information that makes the exact game state hidden. In my recently published paper, we design algorithms that can learn approximate best response strategies against a fixed but unknown and stochastic opponent in an (perfect or imperfect information) extensive form game. My plans are to continue working on techniques to address the challenges that hidden information brings."

■ Research Focus

"Designing AI that thrives even when the full game state is hidden from view.

MDP + RL
Imperfect information games
↑ Human
Target: superhuman play in Recon Blind Chess

APPLIED ALGORITHMS

Mrinal Garg · Final Year Undergrad · Prof. Sujoy

03



Mrinal is a final year undergrad, working with Prof. Sujoy on Graph Matching, clustering and Submodular functions. Wow, that was a mouthful. Perhaps some demystification is in order. The concept of clustering is ubiquitous across computer science. The idea is simple, group together data that is similar in some way. The **"way"** is often described by the use-case, or is sometimes the standard point to point distance in n-dimensional space. Mrinal's work corresponds to **matching, and clustering** (which have parallels between them) in geometric spaces. There are notions of fairness in such matchings/clusterings.

A protocol that assigns jobs to servers should not overload one while leaving another idle. Fairness is not a luxury — it's a constraint.

A lot of Mrinal's work was understanding the behavior of such routines on different types of data, looking for optimizations wherever possible. Most algorithms that deal with the kind of problem described here are online algorithms. In such cases on often has to commit to decisions made in the past. There are options for recourse, but they come at an expense. Mrinal's work provides a state of the art solution for such scenarios that optimizes over task specific optimization metrics. There are many components involved in such research, and Mrinal component was **crystallization and implementation** of their own algorithm as well as many others, so as to draw comparisons and set benchmarks. He recently published his research in ICLR 2026.

■ Publication

State-of-the-art fair clustering in geometric spaces — benchmarked and published.

ICLR 2026
Published research
Online
Real-time decision algorithms

04 A THEORETICAL LENS FOR COMPLEXITY

Priyanshu Singh · MS Student · Profs. Chethan Kamath & Rajendra Kumar



When life gives you lemons, make lemonade. **When life gives you logically sophisticated problems, you start working because not everything is a metaphor.** Getting results in Computer Science theory is not easy. It involves thinking. A LOT OF THINKING. And working out problems, often by hand. Good research takes time, and a lot of effort. There are levels of abstraction so convoluted, intricate, and fragile, that it sometimes takes a month understanding the problem statement. To illustrate the Complexity of NP, here's a glimpse into the work of Priyanshu Singh, an MS student in the department, in his own - if slightly complicated - words!

"Can we design systems where raw computational power alone cannot undermine fairness?



■ The Core Question

Building functions where step-by-step computation cannot be shortcut — even with unlimited parallel hardware.

Crypto
Lattice-based inspiration
Fairness
Blockchain security applications

"In certain **cryptographic protocols**, we require guarantees that remain secure - there are many notions of security though - even when an adversary can deploy large amounts of parallel compute. For instance in blockchain protocols where participants or stakeholders with enormous computational power might try to gain an unfair advantage by influencing outcomes at the last moment or by exploiting massive parallel computation. One way to prevent is through a special function called **Sequential function**. Such a function is designed so that it must be computed step by step. Even if someone has thousands of processors, the task cannot be meaningfully sped up by dividing it into parallel pieces. The delay is built into the structure of the computation itself.

Together with Prof. Chethan Kamath and Prof. Rajendra Kumar, I am working on constructing such functions with strong - again, this is very context dependent - security guarantees. The goal is to begin from a problem that is believed to be hard in some sense, implying that no efficient algorithm can solve every instance of it. From this foundation, we aim to design a new function whose randomly generated instances are hard to compute for any "efficient adversary with polynomially bounded parallelism". This requires converting **"hard in the worst case" into "hard on average,"** a transformation that is well known in **lattice-based cryptography**. We are exploring whether similar ideas can be adapted to certain structured search problems that naturally capture step by step computation. The fundamental question the problem asks is : Can we design systems where raw computational power alone cannot undermine fairness?"

Computer Science & Engineering

IITB

These four stories are just a slice of the research happening across the department. Every paper starts with a student who stayed curious long enough to find something worth building.

• Compiled by Ishaan Chandra



MADHAV GOEL

HEAR FROM ALUMNI

CAREER OPTIONS | MBA

" HEAR FROM MADHAV GOEL (CLASS OF 2019), CURRENTLY AN MBA STUDENT AT HARVARD BUSINESS SCHOOL. WITH PROFESSIONAL EXPERIENCE AT BAIN & COMPANY AND INTERNSHIPS AT ADOBE AND EBAY, MADHAV WILL PROVIDE A DEEP DIVE INTO CONSULTING INDUSTRY - A FIELD RARELY EXPLORED BY OUR GRADUATES BUT FULL OF IMMENSE POTENTIAL "

Harvard Business School, MBA: Expectations vs Reality



"I expected an academic MBA, but the biggest surprise was everything beyond that. At HBS, learning is case-based and discussion-driven; no one teaches top-down. Everyone brings 5-6 years of real work experience, so conversations are grounded in reality. Grades don't really matter, which frees you to explore. More than technical skills, it's about personal growth, leadership, and discovering career paths: consulting, VC, startups, that I never even considered as an undergrad."

Who is this path for ?

"I think this path works well for someone who enjoys problem-solving, but not in a purely technical way. You're often working with incomplete data and very ambiguous problems, and the focus is on speed, breadth, and telling a convincing story rather than depth or perfect answers. You also have to like working with people: clients, teams, and stakeholders, because you can't avoid that part. If ambiguity gives you energy and you enjoy thinking about strategy, customers, and leadership more than day-to-day coding, it's probably a good fit."

Breaking MBA & Consulting Myths

"One big misconception is the glamor around MBAs and consulting - the travel, hotels, perks - without fully understanding what it takes behind the scenes. Those perks come with long, intense weeks and very demanding work. Another myth is that consulting isn't rigorous, or that you're just telling clients what they already know. That's not true. The real work is structuring messy problems, bringing different stakeholders together, and actually driving change, especially now with more implementation-heavy projects. And finally, people think client skills are innate. They're not. Consulting is extremely coaching-driven; you learn communication, Excel & leadership on the job. You don't need to be born with it."

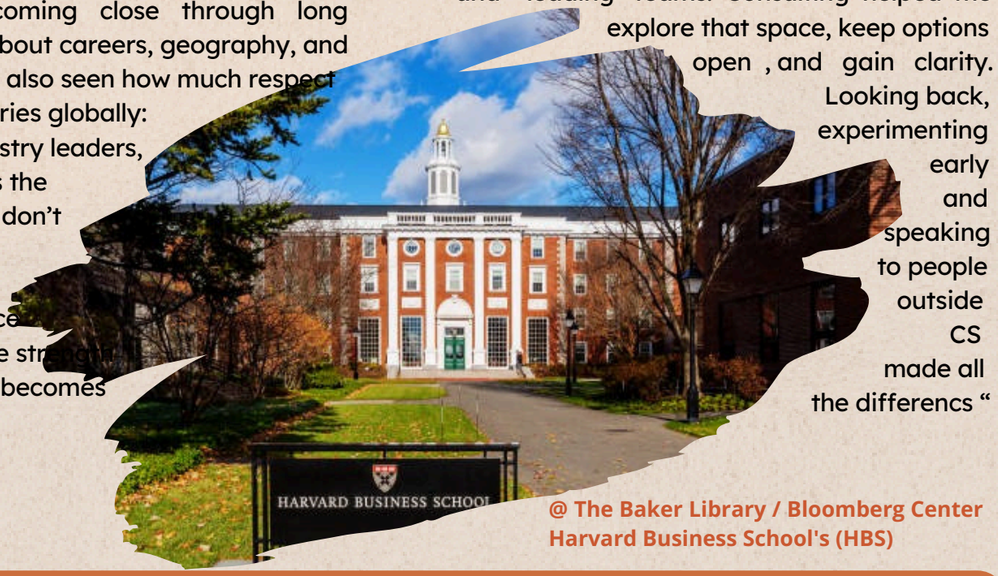


The IIT B's Alumni Network

"My strongest alumni connections came very organically, mostly through DebSoc. That group is still my closest circle: friends, mentors, people I go to for career advice, and honestly, people I party and grow up with. Even years later, we keep reconnecting across cities and countries. What surprised me was how paths collide again, especially in the Bay Area. People I barely spoke to in insti ended up becoming close through long conversations about careers, geography, and life choices. I've also seen how much respect IIT Bombay carries globally: professors, industry leaders, everyone knows the ecosystem. You don't always feel it when you're on campus, but once you step out, the str of that network becomes very real."

Why MBA insted of Tech ?

"The MBA decision came much later. I genuinely explored the tech route: research projects, a BTP, and an internship at Adobe; before realizing that while I enjoyed tech, I was more energized by product, strategy, and customer-facing discussions. I discovered I wasn't most differentiated as an engineer, but in translating tech into products and leading teams. Consulting helped me explore that space, keep options open , and gain clarity. Looking back, experimenting early and speaking to people outside CS made all the differencs "



@ The Baker Library / Bloomberg Center
Harvard Business School's (HBS)

The Experience That Changed Everything

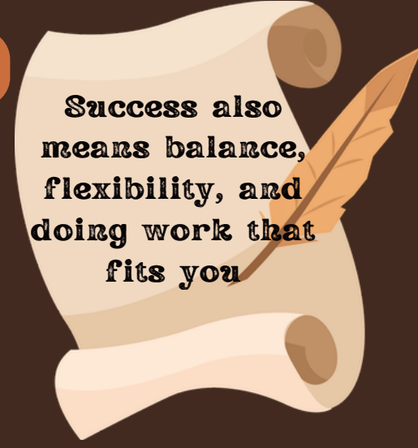
"One experience that really shaped me was doing a semester exchange at NUS (National University of Singapore). Just stepping out of insti and studying abroad changed my perspective a lot. Academics were still competitive, but more practical and less exam-driven, which gave me time to travel and explore life beyond coursework. It was my first time outside India, and it really opened my eyes to what life could look like beyond just academics and careers. I'd strongly recommend international exposure: exchange or internships, while you're at insti."

Learning at IIT B vs HBS

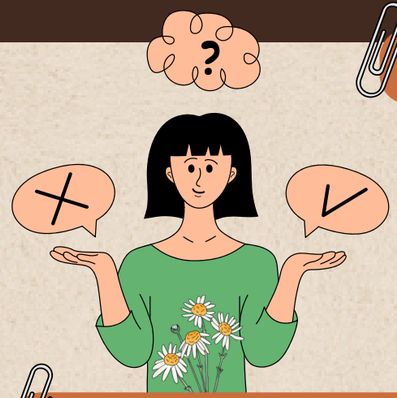
"I think the biggest difference is the point of view from which you learn. At IIT Bombay, you're trained to think like an executor: you go deep into the nuts and bolts, build things, write code, and really understand how systems work. That foundation is super important, and it gives you credibility later when you work with engineers. At HBS, you're taught to think like the protagonist or the leader. Classes are case-based, very discussion-driven, and you're constantly asked to take a position and defend it. You're not expected to remember every detail, but to synthesize information, deal with ambiguity, and communicate clearly, often on the spot. Communication and higher-order thinking matter a lot more. IIT gave me the depth; HBS pushes breadth, judgment, and leadership. Both mindsets are valuable, just very different."

Redefining Success

"At IIT, I don't think I really owned my definition of success. It felt borrowed — from seniors, from expectations — and it was mostly about going all in and sacrificing everything else for career payoff later. I didn't think about health, family, or balance at all. After working and being at business school, that's changed. I still care about ambition, but now success also means balance, flexibility, and doing work that fits who I am — not just what my résumé says I should do."



India vs Global Careers



"I think the big trade-off is timing and risk. If your goal is to settle in the US, going right after undergrad for a funded master's is easier and more structured, with much less financial risk. Going later, after working, means loans and delayed earnings, but you arrive more secure, with work experience that helps you learn more and think more clearly. It's really a trade-off between early stability and delayed but deeper payoff."

Beyond the Classroom

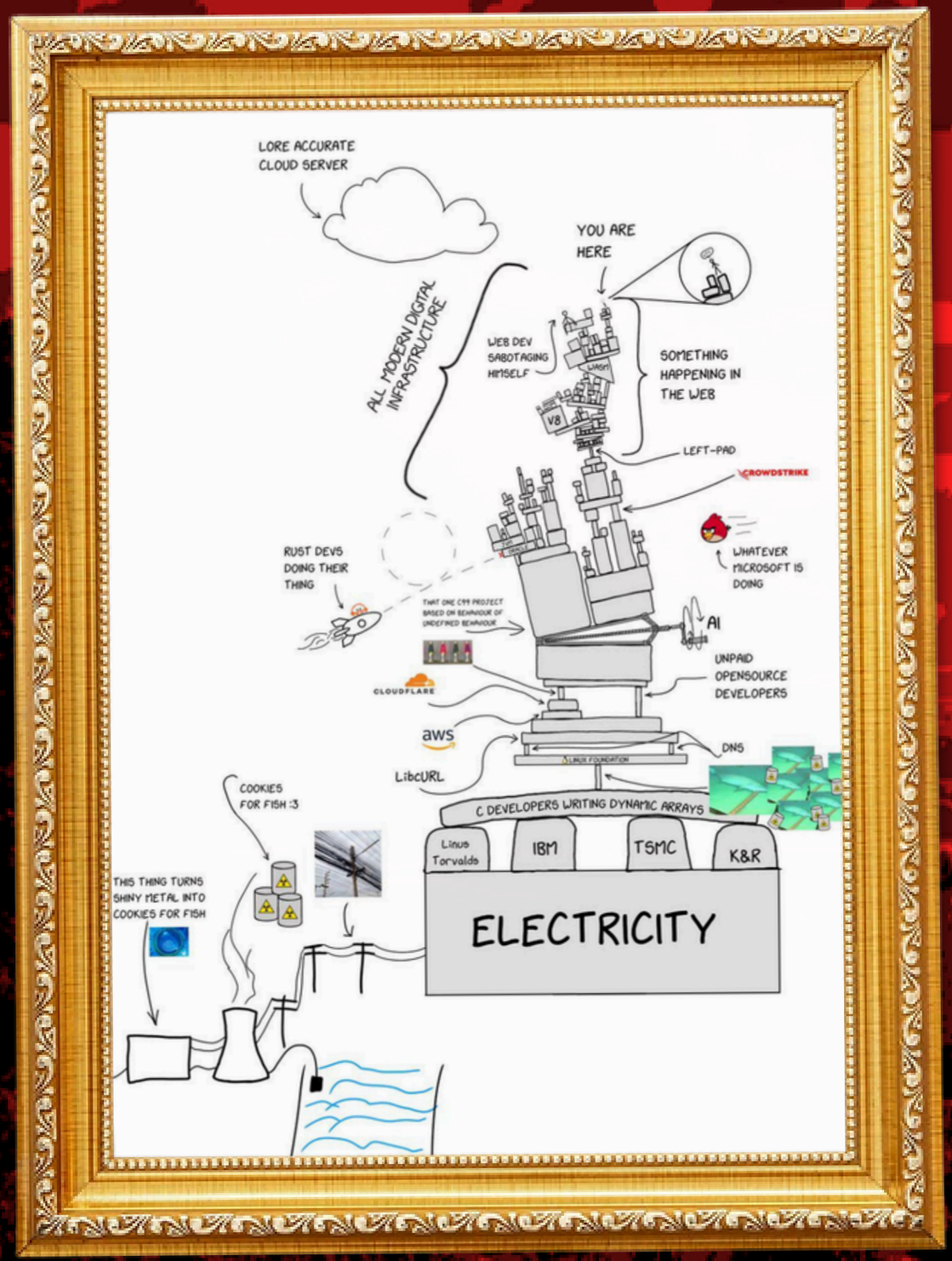
"I think it's extremely important to do something outside academics, but what that something is really up to you. It doesn't have to be debating or consulting or any one thing. Anything you genuinely care about — sports, arts, writing, even just building a routine — helps create balance. I didn't realize this early enough. For me, debating gave me community outside CS and made big decisions feel less isolating. Having interests and people beyond your department really grounds you, and honestly, academics alone can't do that."



---------- Final Note ----------

I don't think I'm at a place to give grand advice, but one thought that really stays with me is:

> "STOP PERFORMING FOR AN AUDIENCE THAT ISN'T WATCHING YOU ANYWAY. VALUE THE IITB CS HALO — IT'S REAL, AND YOU'LL FEEL IT MORE ONCE YOU LEAVE — BUT DON'T LET IT DEFINE YOU". I WISH I'D SPOKEN MORE TO PROFESSORS AND TAPPED INTO THAT WISDOM EARLIER
> AND DON'T BE SCARED OF UNCERTAINTY — IT'S OFTEN WHERE THE MOST MEANINGFUL GROWTH HAPPENS."



How the Stack doin'?

WHEN WORKING ON A SOFTWARE PRODUCT, WE ARE OFTEN ASKED WHAT OUR TECH "STACK" IS, AND USUALLY THIS QUESTION FINDS AN ANSWER IN WHATEVER AI-ENABLED FULL-STACK FRAMEWORK IS TURNING HEADS AT THE TIME. HOWEVER, IF WE STOP TO PONDER FOR A MOMENT AND INVESTIGATE THE "DEPENDENCIES" OF OUR FRAMEWORK, AND THEN THE DEPENDENCIES OF THOSE DEPENDENCIES, WE ARE FACED WITH AN UNCOMFORTABLE TRUTH: WE HAVE NO CLUE WHAT IS HOLDING THIS THING TOGETHER! TO DISAMBIGUATE, WE PRESENT TO YOU THE SUPER OPTIMIZED, ULTRA CLEAN TECH STACK OF ALL APPS ON THE PLANET. ADJACENT IS A POPULAR MEME THAT WAS MAKING ROUNDS ON SOCIAL MEDIA LAST YEAR. AT FIRST GLANCE THIS LOOKS LIKE IT HAS THE CABLE MANAGEMENT OF A 1970S COMPUTER, BUT BELIEVE US, THERE ALREADY IS A LOT OF ABSTRACTION!

BABBAGE'S DIFFERENCE ENGINE USED A HAND POWERED CRANK. THE ENTIRE TECHNOLOGICAL INFRASTRUCTURE TODAY REQUIRES ABOUT A BILLION HAND CRANKS RUNNING CONTINUOUSLY (~ 945 TWH PROJECTED ANNUAL CONSUMPTION, IEA). TO SAVE EACH OF US 3 HOURS OF DAILY LABOR, WE RELY ON STEAM TURBINES POWERED BY FOSSIL FUELS AND, OCCASIONALLY, NUCLEAR FISSION, THE WASTE FROM WHICH GOES TO THE SEA-WATER WE LATER USE FOR COOLING THE DATACENTERS WE JUST POWERED. ALSO, FISH EAT THIS; SUSHI AFICIONADOS MIGHT WANNA SOURCE THEIR FISH CAREFULLY.

NEXT, WE HAVE THE FOUR PILLARS OF HARDWARE AND LOW LEVEL SOFTWARE (MONOPOLIES, MIND YOU, EXCEPT MAYBE LINUS – HE'S A NICE GUY) AND THEIR C DEVELOPERS, WHICH SUPPORT THE INTERNET INFRA, HAVING BUILT LIBCURL (YOU USE CURL IN UBUNTU) AND OTHER THINGS ON THE KERNEL. MEANWHILE, OUR RADIOACTIVE FISH FRIENDS HAVE EVOLVED TO FIND SUSTENANCE ON UNDERSEA INTERNET CABLES NOW, SO DNS RESOLUTION MIGHT FAIL UNTIL AN ALTERNATIVE IS FOUND.

FINALLY, AI HAS BEEN MAKING HEADWAY IN MYRIAD APPLICATIONS, USUALLY WRITING CODE FOR LAZY DEVELOPERS WHO PROBABLY NEVER WATCHED TERMINATOR. OR AVENGERS. OR, EX MACHINA. IF AWS US-EAST 1 IS ONLINE, ASK PERPLEXITY ABOUT THE OCTOBER 2025 OUTAGE.

FOR THOSE WHO DON'T KNOW RUST, ITS AN EXCELLENT, ULTRA-EFFICIENT, SUPER-SAFE PROGRAMMING LANGUAGE. THERE WOULD NEVER BE A MEMORY LEAK IN THE APPLICATION(THE DEVELOPER WOULD NEVER FINISH). SPEAKING OF SAFETY, MICROSOFT MIGHT ROLL OUT ANOTHER WINDOWS UPDATE TO CRASH CROWDSTRIKE AND AIRLINE OPERATIONS SOMEWHERE. FINALLY, IF WE MAKE IT TO THE APPLICATION LAYER, WE CAN FIND EFFICIENT WEB BROWSER ENGINES SUCH AS V8 (BUY RAM TODAY, ITS GONNA GET EXPENSIVE SOON) AND WEB DEVELOPERS TRYING TO WRITE A WEB APP IN RUST SO THEY CAN INJECT BEHAVIOR USING WASM – ITS ALWAYS WASMTIME – WHICH RUNS NATIVELY, AND MIGHT PROVIDE PERFORMANCE IMPROVEMENTS. SO THE NEXT TIME YOU DOOM-SCROLL ON INSTAGRAM, BE GRATEFUL TO THE UNIVERSE FOR NOT SENDING A STRAY COSMIC RAY IN THE WRONG DIRECTION ; DON'T BE SURPRISED IF THERE IS A CORE UTILITY WITH NO ERROR CORRECTION WHATSOEVER!

HERE ARE SOME WORDS DISCONNECTED FROM THE REST OF THE ARTICLE IN SPIRIT OF THE CLOUD SERVER ALWAYS DISCONNECTED FROM THE REST OF THE STACK.

But what does a QUANT CEO?

When people first come across the word quant, regardless of context, the mind immediately goes to a six-figure salary (in dollars of course), and a high performing individual in the midst of 10 computers calculating whether to stake a small country's GDP against a metric that just observed a tiny change.

OUR MENTAL PICTURE SWITCHES CONSTANTLY BETWEEN A HYPER-CAFFEINATED TECH GEEK WRITING CODE AND AN IMMACULATELY DRESSED, CIGAR-SMOKING GIGACHAD FROM AMERICAN PSYCHO.

Market Makers provide liquidity at exchanges, and the primary task in such a setup is to ascertain the optimal price of stocks/commodities/derivatives.
These quant firms have a large amount of compute that constantly analyzes data and decides quotes for option prices. The core challenge lies in managing inventory risk, and pricing these instruments in real-time, and here spontaneity is key.

HYBRID TRADING MODELS

Systematic hedge fund asset managers run portfolios driven by models, with explicit risk budgets and long-horizon evaluation of performance. On the other extreme we have High frequency trading firms, who specialize in short (very) trades and intraday trading – an example would be arbitrage, more on this later – and leverage the firm's deep pockets to guarantee low risk in their trading windows. High Frequency Trading is more of an engineering challenge, since it requires extremely robust and low latency pipelines.

To demystify quants, it helps to understand two things:

- what kind of firm they work at (quant firms come in many flavors), and
- which role they play inside it.

For most of the prominent firms, there is no one business model and their operations are usually a hybrid of many business profiles. Beyond the trading style of the firm, is its distribution of researchers, traders and developers. These three groups work closely to ensure rapid development and deployment, along with back-testing of trading strategies. For example, an HFT firm that is also a Market Maker, could profit from the slight difference in price of the same instrument/commodity in two different markets that the firm operates in. And since there is a lot of liquidity provided by the market maker itself, the transaction, if legal, is practically guaranteed!

THE 3 PROFILES IN QUANT



A quantitative researcher works on improving existing models – most Quant firms already have working models of the market – so they reflect the recent behavior of the market more accurately, and so that any trading strategy that builds on top of this might stand a chance of turning a profit. Thus, quant researchers must be at the top of their game in financial modeling and statistics. Researchers are constantly being bombarded with new data, and their pipelines are always undergoing change. Whatever works at the moment, is immediately made available to the development and trading teams.



Once models and predictors are designed by researchers, they enter the implementation phase, where Quantitative developers quickly stitch together frameworks that ensure research prototypes can work with the incoming slew of data at the desired rates . Often the driving factor behind system design for quant developers is speed. High frequency operations require low-latency algorithms. Sometimes specialized compute – FPGAs – is sanctioned to achieve the tight timing requirements of the market. Once the developer team signs off on the gadget at their end, the traders have the tools of the trade at their disposal.



Departing from this RnD pipeline, we finally have Quant Traders. The nature of a trader's job is defined by the firm, as well as the market. Traders are generally required to make decision calls based on the paraphernalia provided by researchers. Moreover, these tools and technologies might not work out of the box, as market conditions vary over time and trading windows. These models need to be fine-tuned, either manually, or using optimization techniques. Therefore, working knowledge of the principles that these tools are founded upon is very much required in a trader. Couple that with immense back of the envelope math ability, and voila! you have yourself a candidate quant trader.

FAQS

What are the various roles available at a quant firm?

While the most coveted roles (besides management) are quant trader/researcher, there are many equally lucrative opportunities at a quant firm. For instance, developers who work on implementation of algorithms and/or establishing low-latency frameworks (HPC) are also in demand in the quant market. Moreover, they are always on the lookout for data scientists and AI engineers to augment their teams.

What does a day look like for a trader?

The daily work of traders largely involved monitoring signals, tools and models. Quite often, traders in the same vertical come and make decisions together based on their independent analyses. The division of labour also depends on the outcome of such meetings. There are also frequent meetings with the research and development liasons for ideation and feedback purposes.

Maintaining connect in LARGE classrooms

A distinctive characteristic of teaching at IIT Bombay, compared to most other IITs, is the size of our classes. Worldwide, the ideal class size is considered to be between 30 and 50 students – large enough for variety of discussion, yet small enough for in-person connection. However, soon after joining CSE IITB as a Systems faculty member, I realized that the number 25 that I had in my first course offering was the closest I would ever get to the ideal number!

In classes of 100-250 students, the biggest challenge is not classroom delivery. IITB has high quality classrooms that allow your material to reach easily to all the students. The real challenge is to ensure you are able to reach your students. This is what I refer to as the CONNECT.



As instructors, we want students to take meaningful learning from a course, and we want to ourselves remain motivated and steady throughout the semester. I have come to realize that the success of a course is not measured by any numbers, but in whether the students and the teaching team would like to relive its experience. In an era where access to information is no longer scarce, what truly remain with students are the memories of the content, the delivery, and the interaction – that is, the human aspects of a course.

TRUST-BASED CONNECTION



Scaling that human experience to large, diverse classes is not easy, and things go wrong very quickly. A looming paper deadline, a conference travel, a struggling mentee, a TA being unavailable, a spell of viral fever – any of these can cause students to disengage and leave the instructor counting the days until the semester ends.

So what has helped me establish, and maintain, a good connect in large classes?

EXPRESSING MYSELF MORE THAN I EARLIER THOUGHT WAS ESSENTIAL.

Whatever we may feel about 0.5-mark cries or post-evaluation friction, our students look up to us and learn from us. If we clearly articulate the philosophy and logistics of the course at the beginning, including communication channels and expected response times; if we allow students to see modifications in our course plan; if we ensure transparency and fairness in marks and feedback; and of course if we try our best to teach well (no shortcuts to preparation), we lay the foundation for "trust". That trust is the strongest reinforcement of connection. I know it exists when my students reach out to me, even if it is to complain.



A GLIMPSE AT THE SHEER VASTNESS OF LAS

PROF. MANAS THAKUR



AS AN INSTRUCTOR, I HAVE ACCEPTED THAT NOT EVERYONE WILL ALWAYS BE HAPPY. BUT IF EVERYONE IS UNHAPPY, THAT IS A SIGNAL. IT CALLS FOR CONVERSATION — WITH STUDENTS, WITH TAS, AND WITH COLLEAGUES. MOREOVER, IT CALLS FOR UTILIZING THE CONNECT TO FIND SOLUTIONS.

When such a trust-based connection exists, I feel free to explain my reasoning during difficult situations – whether they are academically essential decisions or circumstances beyond my control. Sometimes, these moments even become learning-plus-bonding opportunities (for instance, students enjoy hearing the experiences from a conference). More often than not, students reciprocate with cooperation – and occasionally, even with help.

In a Nutshell

To summarize, the challenge of teaching large classes is not about broadcasting content; it is about creating a frequency that everyone can tune into. When resonance is established through a trust-enforced connection, the semester stops feeling like a solo climb and becomes a collaborative exercise. The best courses aren't taught to students; they are built with them.

LARGE LANGUAGE MODELS

Large Language Models (LLMs) have taken the world by storm. ChatGPT, for instance, became the fastest application ever to reach 100 million users and today sees nearly 700 million weekly active users. This rapid adoption is mirrored in investment trends, with global funding in the AI space exceeding $100 billion in 2024 alone, a remarkable signal of the technology's growing importance.



TECHNOLOGY THAT MOVES YOU FORWARD

What began as fascinating but deeply flawed systems has quickly evolved into highly capable and intelligent assistants. Modern LLMs display great versatility: they can answer everyday questions, draft emails, assist with coding, and even solve complex mathematical problems. Using them to complete homework assignments may be efficient- if not entirely noble- but it underscores just how powerful these tools have become in a remarkably short time.

Despite these advances, AI-generated content can still feel generic, inauthentic, or noticeably inferior to expert-crafted work. This limitation highlights an important point : **LLMs are most effective when used as productivity enhancers rather than as replacements** for human expertise. When deployed thoughtfully, they combine speed and scale with human judgment, creativity, and domain knowledge, resulting in outcomes that are stronger than either could achieve alone.

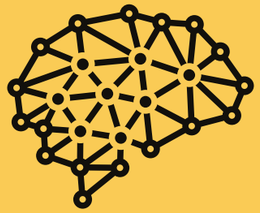
IS IT A BOON OR BANE ?

A particularly significant development in this space is the rise of agentic AI, built on top of foundational models such as ChatGPT, Gemini, and Claude. This has become a large field in its own right, with numerous startups using these models to automate narrowly defined workflows. Such agents can drastically reduce both time and cost, but they also contribute to substantial disruption across the tech industry. Large-scale layoffs, though unfortunate, may be an inevitable consequence when a single prompt can generate a functional website in minutes, a task that previously demanded hours of skilled labor.

AND THE WAY FORWARD

THE CORE OF LLM'S

At their core, however, LLMs remain neural networks (essentially massive matrix multiplications with activation functions) trained to predict the next word in a sequence. Through back-propagation (basically differentiation), these models tune hundreds of billions or even trillions of parameters to capture patterns and structure in natural language.



The key breakthrough that made this possible was the landmark paper **"Attention Is All You Need"**, which introduced self-attention. This mechanism enabled models to reason over long-range context far more effectively, laying the foundation for the transformer architectures that dominate today's AI landscape.

LIVE THE CHANGE

Large Language Models are no longer just an exciting research idea; they are quickly becoming a core part of how work gets done. From writing and coding to analysis and decision-making, their impact is already visible across industries. **While these systems are powerful, they are not perfect, and they works best when paired with human judgment rather than used in isolation...**



LLM'S & INDUSTRIES

The AI boom has also benefited supporting industries, particularly hardware. NVIDIA is a famous example, having become the first company in the world to surpass a $5 trillion valuation. Another interesting set of beneficiaries includes companies like Mercor, which hires lawyers, analysts, bankers, journalists, and other professionals to train AI systems. One of the largest bottlenecks in building advanced AI models today is the **lack of high-quality data** needed to train the model to perform better in niche tasks, and such firms help bridge this gap by injecting expert knowledge into training pipelines for companies like OpenAI and Google.

> AI WON'T REPLACE PEOPLE, BUT PEOPLE WHO USE AI WILL REPLACE THOSE WHO DON'T.



Sam Altman CEO of OpenAI

CSE BINGO

STRIKE!

GET A SEQUENCE



Copied friend's code as just changed the variable names	Crammed the course the night before exams	abused chatgpt for not being able to help u.	forgot to submit assignment	had over 3 Deadlines at once
Safe did not work for you.	Didn't bathe for a week	Debugged for hours to find a stupid bug	Ate maggie for 10 days in a row	Skipped atleast 40 Breakfast in a sem
Woke up at 8pm instead of 8am	Wore the same clothes for 5 days straight	Answered incorrectly with full confidence	Did a chemistry expt completely incorrectly	Added fake/incomplete projects to Resumé
slept and didn't wake up for an interview	slept on the floor during the class	0 cold mailing response	"Yaar agle sem se pakka padhunga"	"abse roz saari classes attend karunga"
Contributed nothing meaningful in a group project	Submitted an incorrect file or dir structure	Submitted an assignment over 3 weeks late.	Missed a grade by less than 1 mark.	"Just 1 more episode" and sleeping at 7am.

Baptising *into the department : Square One*

Right as freshmen get into gear with life at IIT, they are blind-sided with the mandatory Computer Programming and Utilization course+lab. If you're in the lucky (lol) half, you get to do this in the first semester of your programme. Then the students are split into two, those who bring their own systems, and those who are too lazy to lug around the behemoth of a gaming laptop that they convinced their parents they were gonna use for "machine learning". If you are in the latter group, you will be provided a system by the institute in a cold, musty lab in the Computing Complex. This system runs Linux, with a profile "labuser" which is your sandbox for the next 6 months. As far as the course is concerned, navigating it through linux is not that different from doing it on windows/mac, so many people are motivated to install Ubuntu on their own systems.

Using Linux can be very liberating, coming from Windows (except maybe with games). You get an incredible amount of control on whatever processes are running on your OS, and can write quick bash scripts that can substitute for full blown programming frameworks in many cases. You can write code that is incredibly efficient (the entire source is exposed on github for those who know their stuff). Linux is quite power efficient too. First users might recall gaining an hour or two of battery life when running Ubuntu (compared to windows).

Actual representation of us in lab



Why is ubuntu *so powerful?*

The reason Ubuntu is so power efficient is that it does not run on battery, it runs on hate, and the thing powering the computer is YOU! Visually/Graphically, its not that difficult to figure things out on a Ubuntu Desktop. The trouble comes much before that, i.e. during installation. Money usually buys comfort and convenience, Not so with Linux! Your super expensive GPU makes it impossible to get a working system at the first boot. This marks the beginning of a day long struggle to log into your user. Almost half the problems you'll encounter will in some way be related to Nvidia Drivers. Nvidia, unfortunately, is always catching up with the relatively fast paced development of kernels in the open source Linux. The proprietary nature of Nvidia drivers have caused them to end up on Linus Torvald's "Naughty List". Don't believe us? Just google Linus Nvidia and go to images.

After this, a lot of things are plug and play, such as installing VsCode, conda environments and Jupyter notebooks. But someday, you gotta circle back to the steaming Nvidia pile, when you want to run ML models – this requires CUDA – and then you have to work at the system level again. There are a lot of dependencies everywhere, in everything you install. The sooner youunderstand this dependency structure, the better, since relying entirely on ChatGPT commands is VERY likely to break your OS. That said, Ubuntu does provide a lot of liberty when developing open source, working with security and hacking (ahem, ethical hacking) and opens possibilities for IoT research. Using a proprietary operating system (like Windows) has its cons, you usually don't have kernel level control, for security reasons. When working with open source projects, especially systems software, you typically need direct access to these deeper layers. Furthermore, a lot of tools (system traces and function traces)



CSE peeps after learning 55 languages, 55 different algos for the same binary search question, 55 different code editors, fixing 100 lines of codes filled with error, doing 155 sudo apt install to fix the dependencies on a random tuesday.

Some Insider *code for you*

Finally, once you manage to get most of the drivers working on Ubuntu, you can go ahead and download your favorite IDE. There are some people who prefer to code in Vim, or Gedit, but they are a rare breed. Just install VSCode, and things will sort themselves out. For the majority of your college journey, you'll be learning and prototyping, and the language of choice for these tasks is Python. Ubuntu usually ships with the latest version of Python3 (there are separate python executables with different levels of system permissions though). Often, a lot of system level utilities are dependent on the python executable that comes pre-packaged with Ubuntu, as well as the library binaries that are associated with it. When you work on your projects, you'll likely require new libraries, which might have very specific "dependencies". If you go ahead and install these to the system-level Python3, you might have dependency conflicts (different programs looking for different versions). So its best to work in isolation. There are many ways to achieve this isolation. One way is containerization at the OS level (docker), which runs the core kernels of the OS virtually on top of another OS. This is perhaps overkill for most tasks you'll be doing, and so plan B is to use a Python Virtual Environment, or venv. This provides a separate space for storing the python interpreter (Python executable), the library binaries associated with the interpreter (site-packages), and some other useful utilities. Essentially, this is a local workspace with all the programs necessary to run your code in isolation. Think of it as the "Game install folder" from when you used to pirate games using utorrent. You can install whatever you want here, and it won't break anything else in the system probably). Its considered a good practice to have a separate venv for every large project one works on.

VSCode provides a "terminal", using which you can enable the venv, and make general "system calls", which is sufficient for most tasks. If you wish to run JS code, the executable command changes, and there are similar ways of isolating all dependencies to within the project directory. With C++, the compiler and linker (again, these ship with Ubuntu) need invoking, which would simply build your code into an executable.

Besides the Development Environment, you'll have things like the .bashrc (think of this as a terminal preset config if you will) and other tools you can install. The most apt tool for installing utilities on Linux is, well, APT. But it takes a magic word to get it working, and thankfully, no one needs to say please (you run Linux COMMANDS, not linux REQUESTS). Prefixing every Linux command with sudo – stands for Superuser Do – will allow you to run with administrator privileges. So the next time something refuses to do its job, just prepend sudo at the beginning of the command. Then there are networking tools like net-tools (Duh!) and wireshark. With these you would not believe how easy it is to listen to unprotected traffic. So, kids, always use protection HTTPS (TLS)!

Since we have mentioned Linus twice already, we might as well introduce GIT. Yes, this is the same as the git in GitHub. Before LLMs were around, this is where people used to go for "borrowing" code for their projects. Git is a version management system. Long story short, it keeps track of what changed when in a project. When working in groups, this can be especially useful – if not for having working rollbacks, for assigning blame – and you can have this version control system installed both

Final Words :)

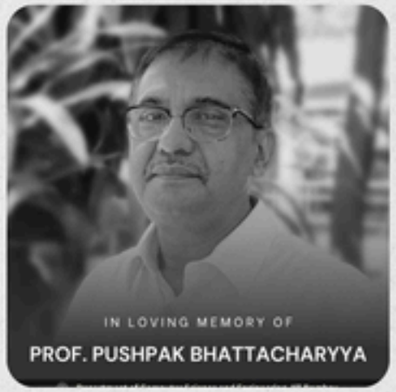
Finally, at some point or the other, you'll work with open source tools, which are often built from source. This requires installing, re-installing and configuring a lot of stuff. This is all done using Linux commands. Often with sudo access. Building things on Linux is where you stop being the "person installing software". Building is hardly ever a one-step process. Or even a 10 step process. The moment you run "sudo make install", you'll (likely) be hit by a dependency error which looks like a cat walked all over the keyboard (missing libFrom30YearsAgo-dev). On the off-chance that you screw up somewhere, the system might enter a state from where its very hard to undo the damage. For instance, there is no coming back from running "sudo rm -rf /*" instead of "sudo rm -rf ./*". So using github to save your work files online is always a good idea.
If you've managed to read to the end of this section, run "apt moo" :=)



Henceforth, I baptize you in the name of the Father (Alan Turing), the Son (Charles Babbage) and of the holy spirit (Linux).

Linus Torvald - the creator of LINUX

In Memoriam



PROF. PUSHPAK BHATTACHARYYA (1962 - 2025)

PROF. PUSHPAK BHATTACHARYYA WAS A TITAN OF ARTIFICIAL INTELLIGENCE AND A VISIONARY LEADER WHO SHAPED THE LANDSCAPE OF INDIAN ACADEMIA. BEYOND HIS GLOBAL INFLUENCE AND SCHOLARLY BRILLIANCE, HE WAS A DEDICATED MENTOR WHOSE GENEROSITY OF SPIRIT TOUCHED COUNTLESS LIVES. HIS PROFOUND DEDICATION TO LANGUAGE AND TECHNOLOGY LEAVES AN ENDURING LEGACY. HE REMAINS DEEPLY CHERISHED BY HIS FAMILY, STUDENTS, AND COLLEAGUES.



PROF. AJIT A. DIWAN (1962 - 2025)

PROF. AJIT A. DIWAN WAS A CORNERSTONE OF THE IIT BOMBAY COMMUNITY FOR OVER FOUR DECADES. A FABLED INTELLECT IN THEORETICAL COMPUTER SCIENCE, HIS MASTERY OF GRAPH THEORY WAS MATCHED ONLY BY HIS GIFT FOR TEACHING. HE GUIDED STUDENTS WITH PROFOUND CLARITY AND WISDOM, SERVING THE DEPARTMENT WITH UNERRING PRECISION. HE REMAINS A DEEPLY BELOVED MENTOR, COLLEAGUE, AND FRIEND.

BOTH PROFESSORS REPRESENTED THE VERY HEART OF OUR DEPARTMENT, BLENDING TOWERING INTELLECTUAL BRILLIANCE WITH A DEEP, SELFLESS COMMITMENT TO THEIR STUDENTS. THEIR COMBINED DECADES OF WISDOM AND MENTORSHIP HAVE LEFT AN INDELIBLE MARK ON IIT BOMBAY, AND THEIR PRESENCE WILL CONTINUE TO BE FELT THROUGH THE GENERATIONS THEY INSPIRED.

Editor's Note



Dearest Reader,

BITSTREAM IS BACK!! After a small pause in our publication, we return with an exciting new edition. A lot of thought and effort has been put in by the entire team, who've done their best in publishing this edition.

We've brought you articles spanning many avenues, and an interview with an alumnus. Do attempt the CSE Bingo, and post it on social media (tag @csea.iitb for shoutouts). We'd love to see how people fare in it. The making of this edition began with curating what articles we might include. The thought process for that alone made us understand the dynamics of BitStream even deeper, and the purpose it serves.

Being published at the end of the year, this edition serves as a flashback to the activities throughout the term- celebrating milestones and acknowledging those who've raised the bar high for others to follow. It explores uncharted territories, inviting students to look toward the horizon and imagine what lies ahead. It speaks not only to the current students, but also welcomes the freshies arriving next term. It reflects the shared identity that binds us as a community and engages with emerging technical trends. And it strives to bridge the gap between students and faculty.

On top of all this, it must keep evolving, as we all should. The editors have worked hard to write articles that meet the standards set by these functions and the designers have done a great job in putting their thoughts into striking visuals.

I conclude by thanking you, the reader, for giving us your time and reaching here. We hope we were able to deliver. **Do come back next year!**

With Love,

Ansh Garg

Meet the team

Editors

Ansh Garg
Chief Editor

Ishaan Chandra

Yashniel Laddhar

Nuupur Deshmukh

Designers

Gowri K Balakrishnan

Akepati Sriya

Prasoon Badjatya

Tamanna Kumari

Special Thanks

Prof. Manas Thakur
Rishab Rajprakash
Dhruvin Doshi

