

# OLIVER ILIFFE

Email | +44 7944 727309 | [GitHub](#) | [Stack Overflow](#) | [LinkedIn](#) | [Website](#)

## EDUCATION

- Imperial College London** – Advanced Computing MSc (Oct 2024 – Present)  
Dissertation: *Improving the Throughput of a Distributed Programming Runtime for Hierarchical Memory Models*  
Research Project: *Distributed Shared Memory in the era of CXL*
- King's College London** – Computer Science BSc (First-class Honours) (Sep 2021 – Jun 2024)  
Dissertation: *Retrofitting heap-tracing for high-level languages*

## EXPERIENCE

- Software Engineer Intern @ Optiver ;)** (Jul 2025 – Sep 2025)  
**Algorithms Research Intern** (Haskell, Scala & Isabelle/HOL) (Jun 2024 – Sep 2024)  
Supervised undergraduate research position at King's College London
- A team of three, lead by a professor with research interests in theorem provers and compilers
  - Research novel matching algorithms for regular expressions
  - Build a testing framework for verifying the correctness of extended regular expression matchers
- Graduate Teaching Assistant** (C++, Java & Scala) (Sep 2023 – Present)  
TA for various modules at KCL. Most involved in the **Operating Systems & Concurrency** module.
- Taught in a classroom setting and occasionally during lectures.
  - I create additional content for the students, both module-specific (e.g. revision sessions) and related content.
  - For example, an in-depth look at **Linux's scheduler** including a [visual demonstration](#) of the impact of nice values.
  - Some feedback: *"very engaging, and you could tell that he has a thorough and in-depth knowledge of the content."*

## PROJECTS

- Mr. Boc – Parallel Runtime for Deeply Tiered Memory Systems** (Rust, C) Repository is Currently Internal  
A highly granular concurrent runtime for tiered memory systems (e.g. **CXL-based** or **NUMA**).
- Mr. Boc hides the latency of memory accesses by relaxing program order.
  - Schedules many granular bits of work on all cores of the system dynamically based on core load.
  - Keeps all working sets mutually exclusive! while maintaining great performance.
  - I roll several concurrent interfaces from scratch specifically for x86-64 and beat the SoA every time. (1) A batch lock (5x faster than **xchg**-based). (2) A SPSC channel (4x better throughput than **crossbeam** across CPUs) (3) Cache-to-cache copies of large, hot buffers (~2x faster than ignorant).
- Speed up ML Classifier for Acute Kidney Injury** (C) [view repository](#)  
ML Classifier for Acute Kidney Injury (AKI) with tight performance constraints.
- Implement (from **absolute scratch**) a random-forest classifier for detecting AKI in C. (**libc** is used).
  - Model achieves an  $F_3$ -score of 0.985, correctly classifying 1180/1200 people with AKI.
  - Performs inference (including data parsing) in  $< 2\mu s$  and trains in  $< 150ms$  on 13,000 data entries.
  - Use novel methods to compress training data. An article is available on my website [here](#) (easy read).
- Associative Array Research** (C++) [view repository](#)  
Researched a variety of associative array implementations, in an effort to produce a very fast one.
- Create a testing and benchmark suite for hashtables for any language that supports the C ABI.
  - Implement many different hashtable designs (robin-hood hashing, quadratic-probing, chaining etc. **SIMD-lookup**).
  - Current best effort shows up to a 500% improvement on **std::unordered\_map**.
  - Formally describe the complexity of probing hashtable lookups (as well as other [articles](#) about my research).
- libhopeful – Tracing Heap Allocations** (Rust & C) [view repository](#)  
Build inspectable graphs of the allocations active Unix processes. 'Tracing' is used here in the 'tracing GC' sense.
- Consume DWARF debug info, such that we can attempt to link any T to a representation.
  - Considerable investigation into the operational semantics surrounding allocation in Rust (ask me about it!).

## SKILLS

tokio; axum; wgpu; wgsl; bevy; Linux; Windows; C; C++; C#; .NET; Unity; Lua; Python; Django; TypeScript; JavaScript; axios; React; HTML; CSS; Java; Scala; Redux; Prisma; REST API; OOP; Functional Programming; Web-Dev; TCP/IP; Serialization; gdb; Haskell;