Can we achieve *ease of use* and *high performance*?

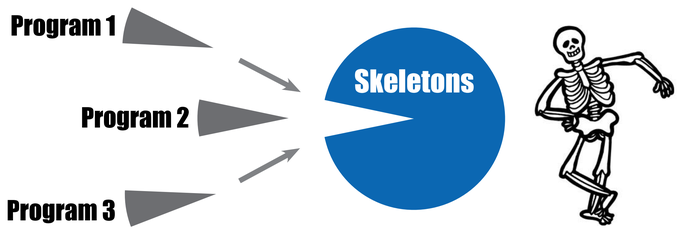
Developers are typically forced to choose between either ease of use or high performance. This project demonstrates how both can be achieved by combining autotuning with high level parallel programming constructs.

Chris Cummins. Supervisors: Hugh Leather, Pavlos Petoumenos, Richard Mayr.

9x Faster Programs

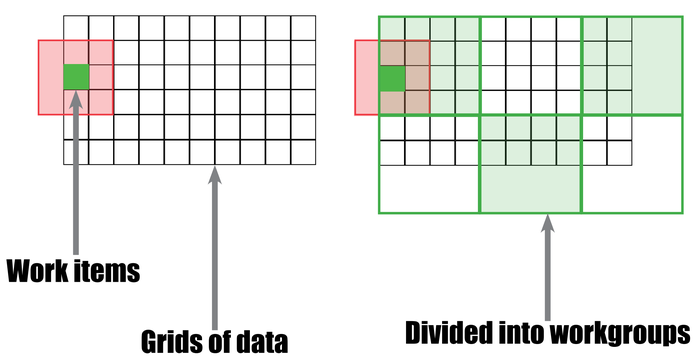
Why Algorithmic Skeletons?

- Simplified parallelism.
- Robust implementations.
- Code reuse.



What is this project?

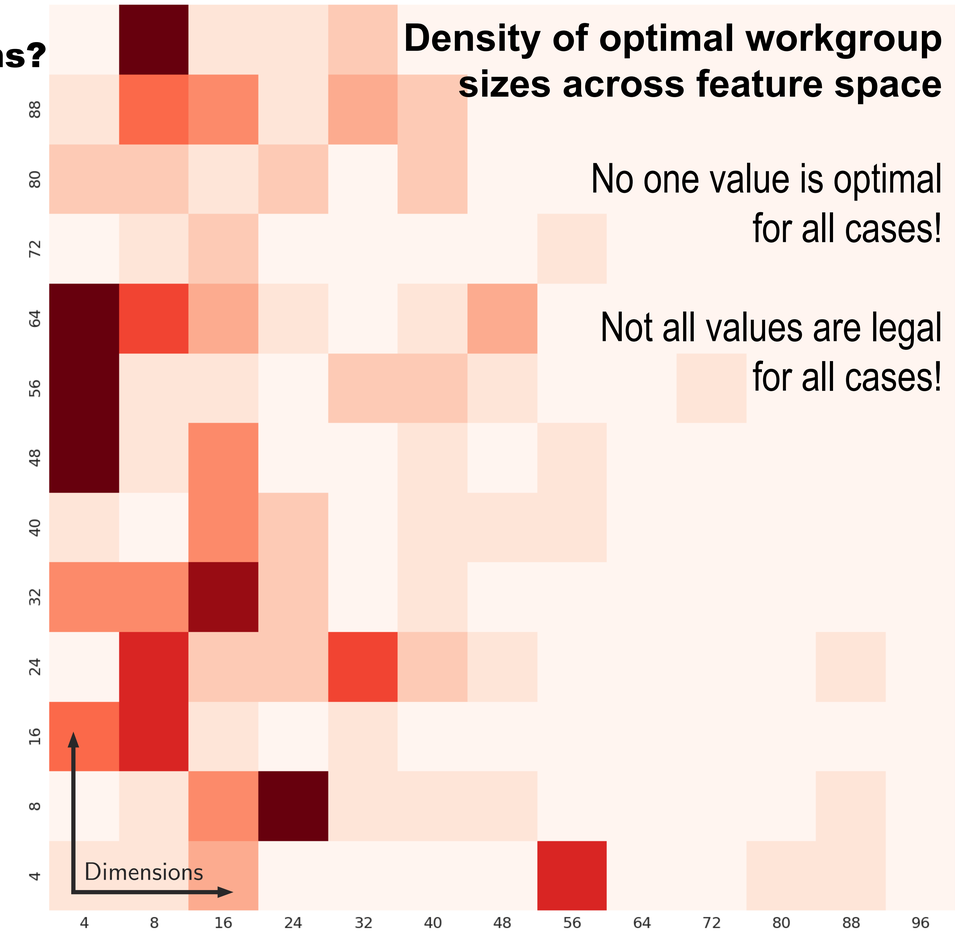
An autotuner to select optimal **workgroup** sizes for **OpenCL** stencil codes using SkelCL.



The **performance** of a workgroup size depends on:

Program	Shape of border region, instruction counts, control flow ...
Hardware	Local memory capacity, num processors, ...
Data	Size of dataset, data types, ...

Density of optimal workgroup sizes across feature space

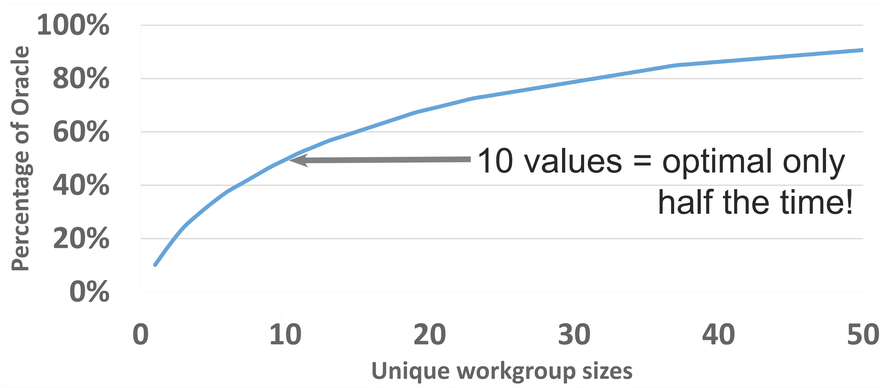


No one value is optimal for all cases!

Not all values are legal for all cases!

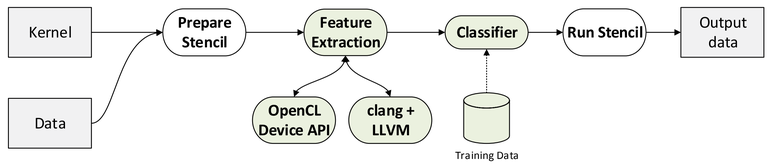
Why do we need Autotuning?

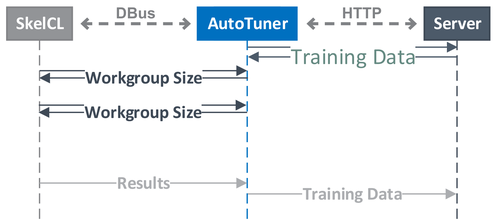
There is no silver bullet which works for all cases, and there are hard **constraints** which can only be satisfied at **runtime**.



How does it work?

At runtime, machine learning classifier **predicts** optimal workgroup size.

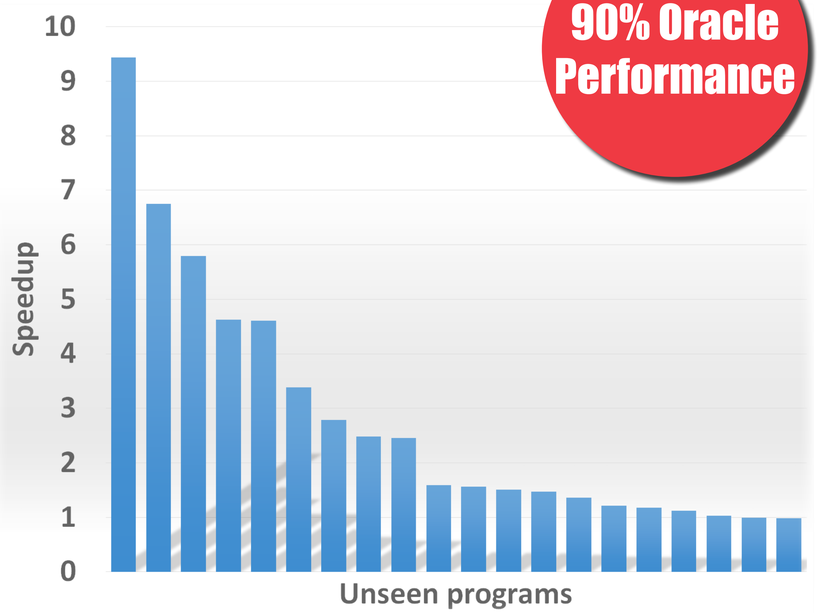




Results

Tested against **unseen** programs. Speedup is relative to the best **statically chosen** workgroup size.

90% Oracle Performance



THE UNIVERSITY of EDINBURGH informatics
EPSRC Centre for Doctoral Training in Pervasive Parallelism
EPSRC Engineering and Physical Sciences Research Council