Scottish Enterprise

University of St Andrews

ParaFormance: An Advanced Refactoring Tool for Parallelising C++ Programs

Chris Brown

Craig Manson, Kenneth MacKenzie, Vladimir Janjic, Kevin Hammond

University of St Andrews, Scotland

@chrismarkbrown

@rephrase_eu

cmb21@st-andrews.ac.uk

C++ Meetup Edinburgh – November 2015

RePhrase

ParaFormance

Scottish Enterprise

University of St Andrews

RePhrase Project: Refactoring Parallel Heterogeneous Software – a Software Engineering Approach (ICT-644235), 2015-2018, €3.6M budget

8 Partners, 6 European countries
UK, Spain, Italy, Austria, Hungary, Israel

IBM

evopro it's possible!

PRQA Programming Research

scch software competence center hagenberg

UNIVERSITÀ DI PISA

UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

RePhrase

ParaFormance

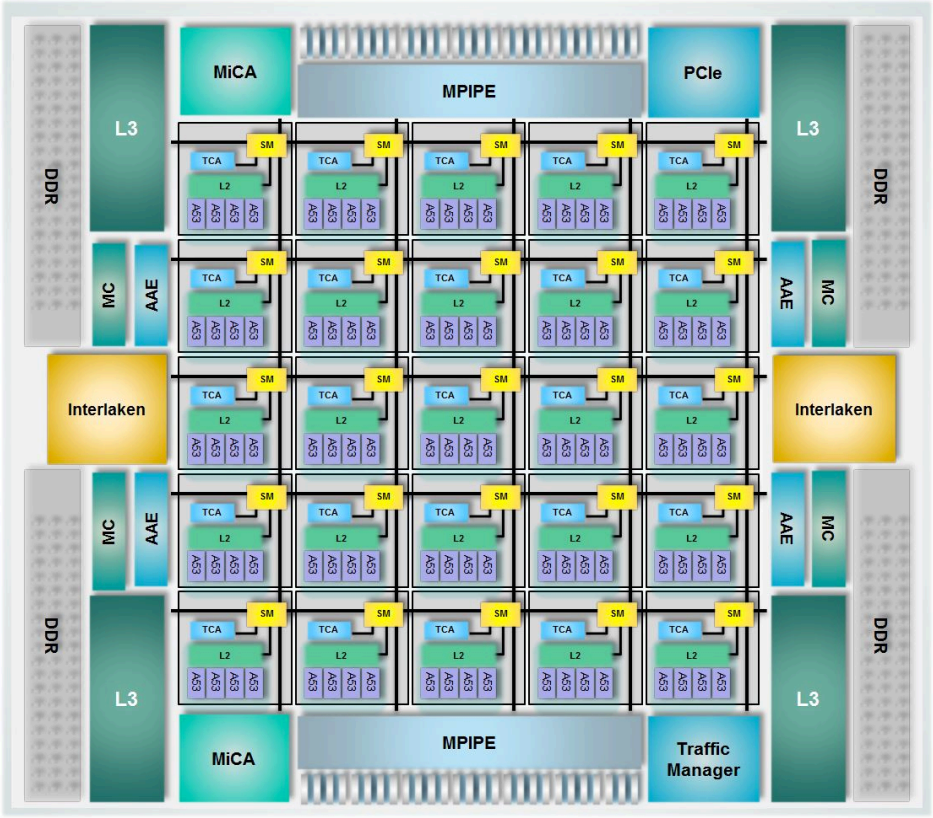
The ParaFormance Project

University of St Andrews

- Seed Funding from Scottish Enterprise
  - 1-3 years pre-commercialisation funding
  - Develop work from EU/UK publicly-funded projects
- Start Date
  - 1st June 2015
- Led by: Kevin Hammond and Chris Brown, University of St Andrews

The Dawn of a New Age

University of St Andrews



- EZCHIP – TILE-MX100
- 100 64-bit AMD x86 Cores
- 3-level cache with > 40 Mbytes on-chip cache
- Multitude of network accelerators
- Over 200Gbps integrated I/O including Ethernet,
- DDR supports up to 1 TB RAM

It's not just about large systems

University of St Andrews

- Even mobile phones are multicore
  - Samsung Exynos 5 Octa has 8 cores, 4 of which are "dark"
- Performance/energy tradeoffs mean systems will be increasingly parallel
- If we don't solve the multicore challenge, then no other advances will matter!

ALL Future Programming will be Parallel!

Programing Multicore Systems…

University of St Andrews

RePhrase

ParaFormance

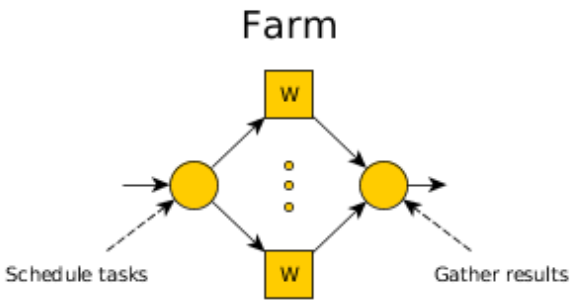
Thinking in Parallel

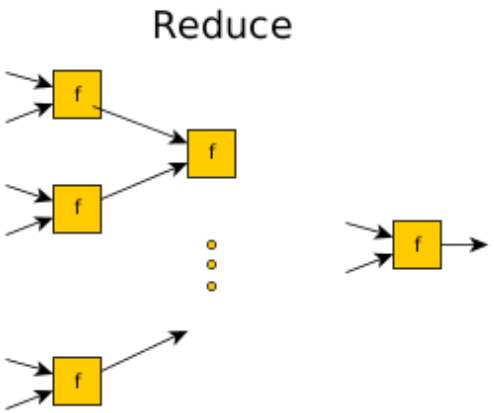
University of St Andrews

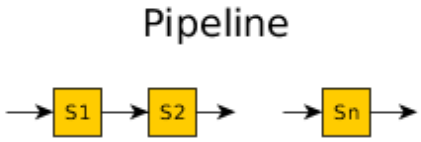
- Fundamentally, programmers must learn to ***"think parallel"***
  - this requires new *high-level* programming constructs
  - you cannot program effectively while worrying about deadlocks etc.
    - they must be eliminated from the design!
  - you cannot program effectively while fiddling with communication etc.
    - this needs to be packaged/abstracted!
  - you cannot program effectively without performance information
    - this needs to be included!
- **We use two key technologies:**
  - **Refactoring (changing the source code structure)**
  - **Parallel Patterns (high-level functions of parallel algorithms)**

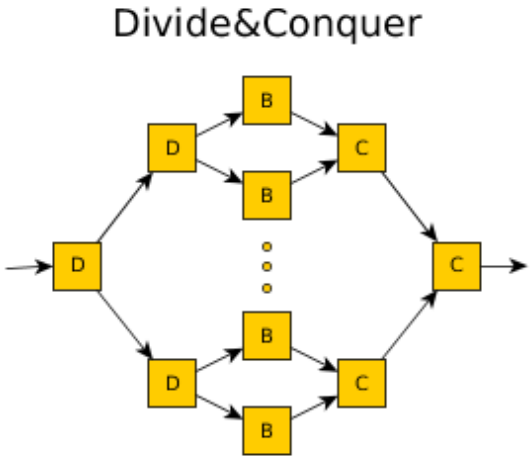
RePhrase ParaFormance

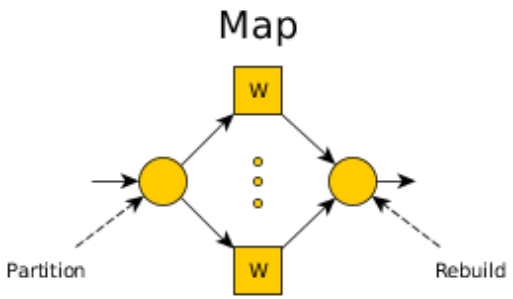
Parallel Patterns











Our Approach

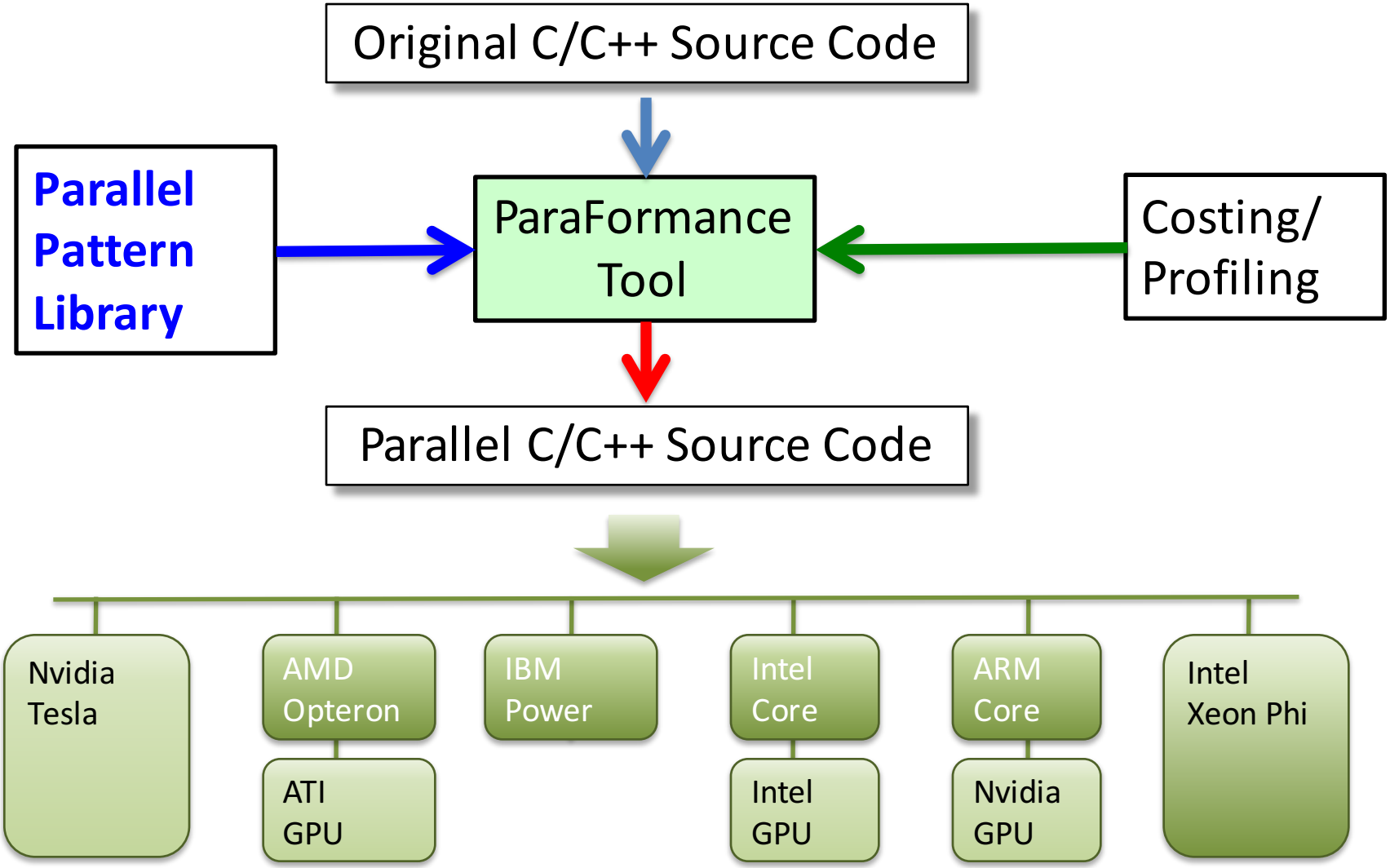
University of St Andrews

- Start bottom-up
  - Identify components (side-effect free functions that correspond to parallel computations)
  - *using semi-automated refactoring*
- Think about the **PATTERN** of parallelism
  - e.g. map(reduce), task farm, parallel search, parallel completion, …
- Structure the components into a parallel program
  - *turn the patterns into concrete (skeleton) code*
  - Take performance, energy etc. into account (multi-objective optimisation)
  - also using refactoring
- Restructure if necessary!
  - *also using refactoring*

RePhrase ParaFormance

General Technique

University of St Andrews



Sequential Refactoring

University of St Andrews

Semi-automated (user-driven)

1. Renaming
2. Inlining
3. Changing scope
4. Adding arguments
5. Generalising Definitions
6. Type Changes

Examples include refactoring Linux kernels using Cocinelle, refactoring Java/C++ in Eclipse, etc.

Refactoring = Condition + Transformation

University of St Andrews

Transformation

- Ensure change at all points needed.
- Ensure change at only those points needed.

Condition

- Is the refactoring applicable?
- Will it preserve the “semantics” of the program?
- The module? The File?

Both pre- and post- conditions

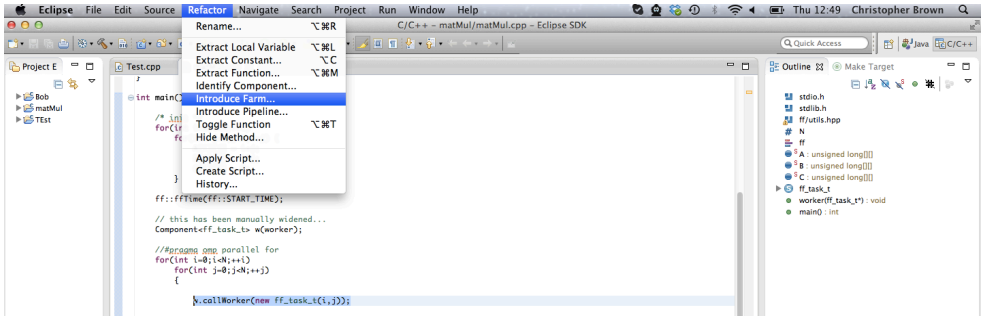
RePhrase

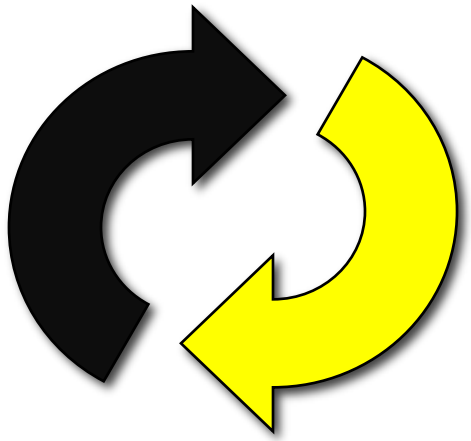
ParaFormance

Refactoring can help parallel thinking!

University of St Andrews

- can be used to introduce parallelism, and help choose the right abstraction
- parallel programs can be refactored for new parallel architectures
- can check the conditions for applying parallel skeletons
- performance information can be integrated
- Programmer 'in the loop'





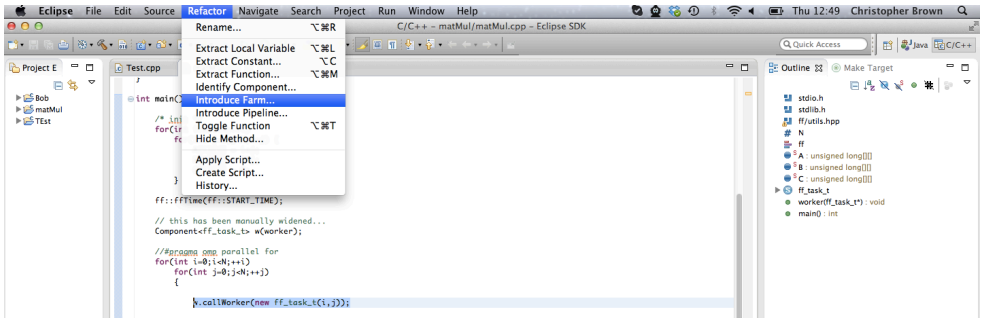
RePhrase

ParaFormance

The ParaFormance Tool Prototype

University of St Andrews

- Integrated into Eclipse (CDT)
- Supports full C++(11) standard
- Layout and comment preserving
- Undoable
- Preview feature



RePhrase

ParaFormance

Our Refactorings

University of St Andrews

- Refactorings to Introduce:
  - Farm/Map, Parallel-For and Pipeline patterns
  - FastFlow
    - Components
    - Farm
    - Pipeline
  - TBB
    - Lambda
    - Function Class
    - Parallel For/Pipeline
  - OpenMP
    - Parallel-For

RePhrase ParaFormance

The ParaFormance Toolkit

University of St Andrews

1. Prediction/Estimation
   - Accurate and advance performance modelling
2. Discovery
   - Automatic discovery of (instances of) parallel patterns
3. Insertion
   - Automatically insert the parallel “Business Logic”
4. Elimination
   - Remove exisiting/legacy parallelism
5. Validity
   - Fundamental condition checks
6. Profile
7. Shaping

RePhrase ParaFormance

Image Convolution

University of St Andrews

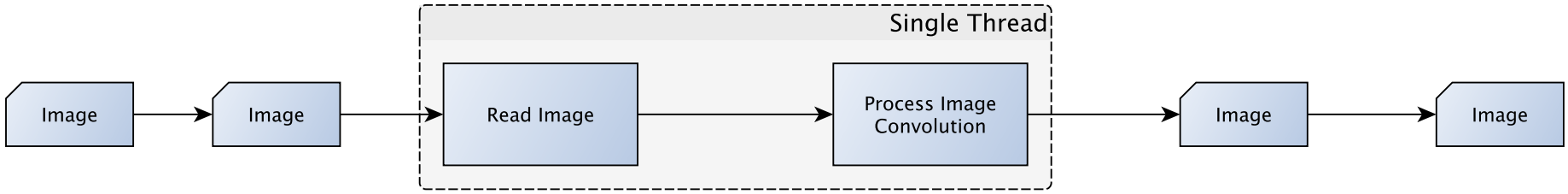
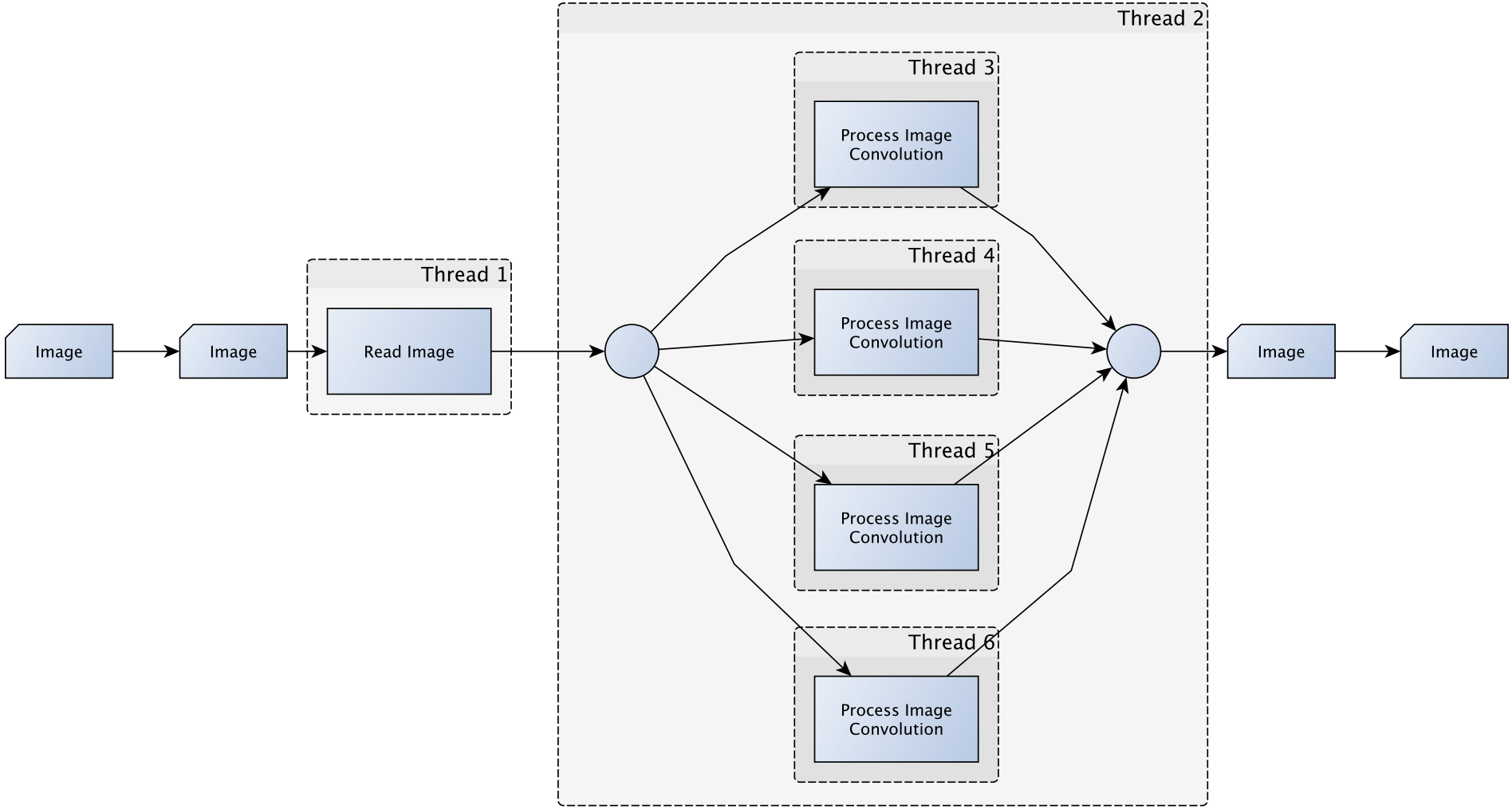


Image Convolution, Refactored

University of St Andrews



RePhrase

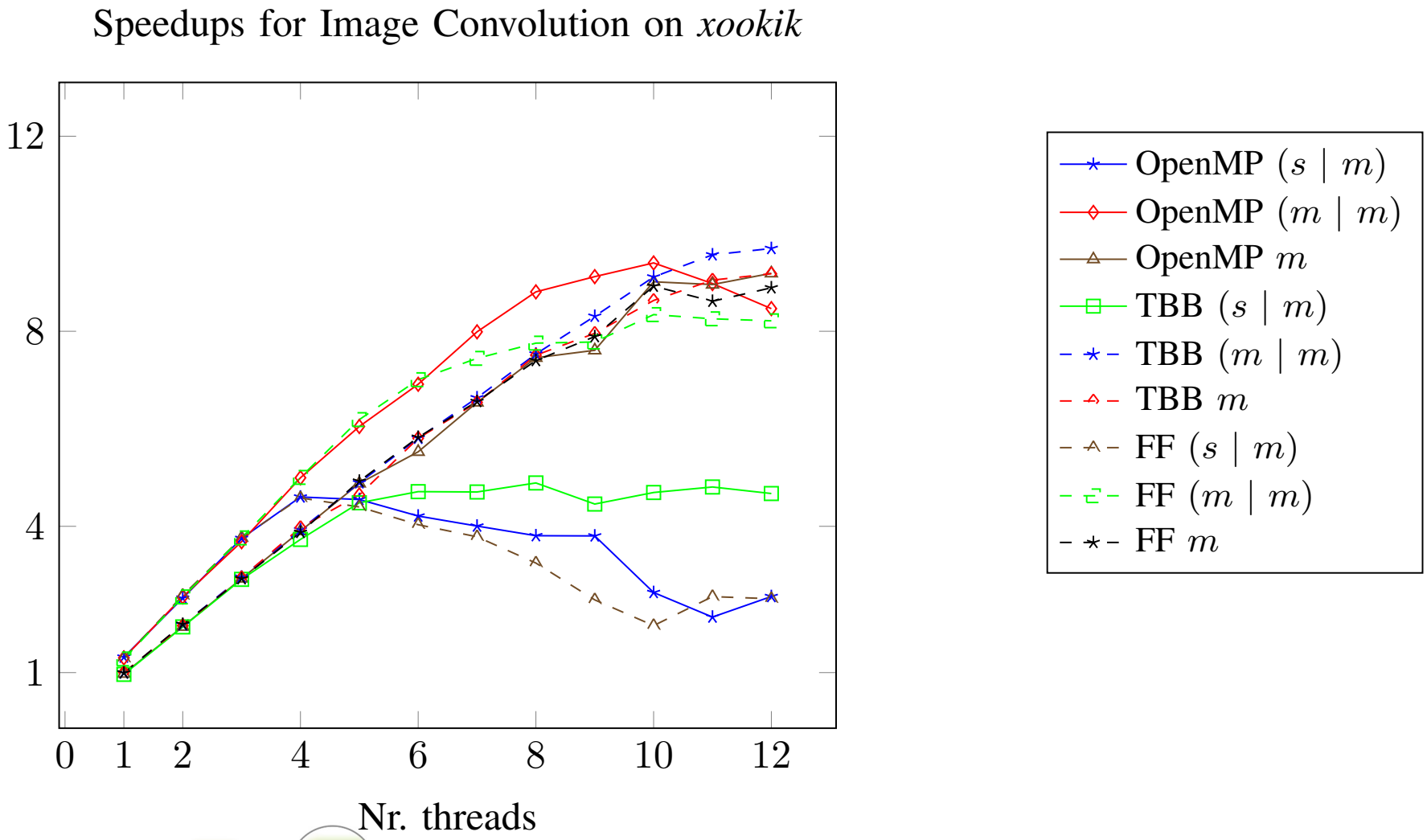
ParaFormance

C++ Refactoring Demo

ParaFormance

Image Convolution

University of St Andrews

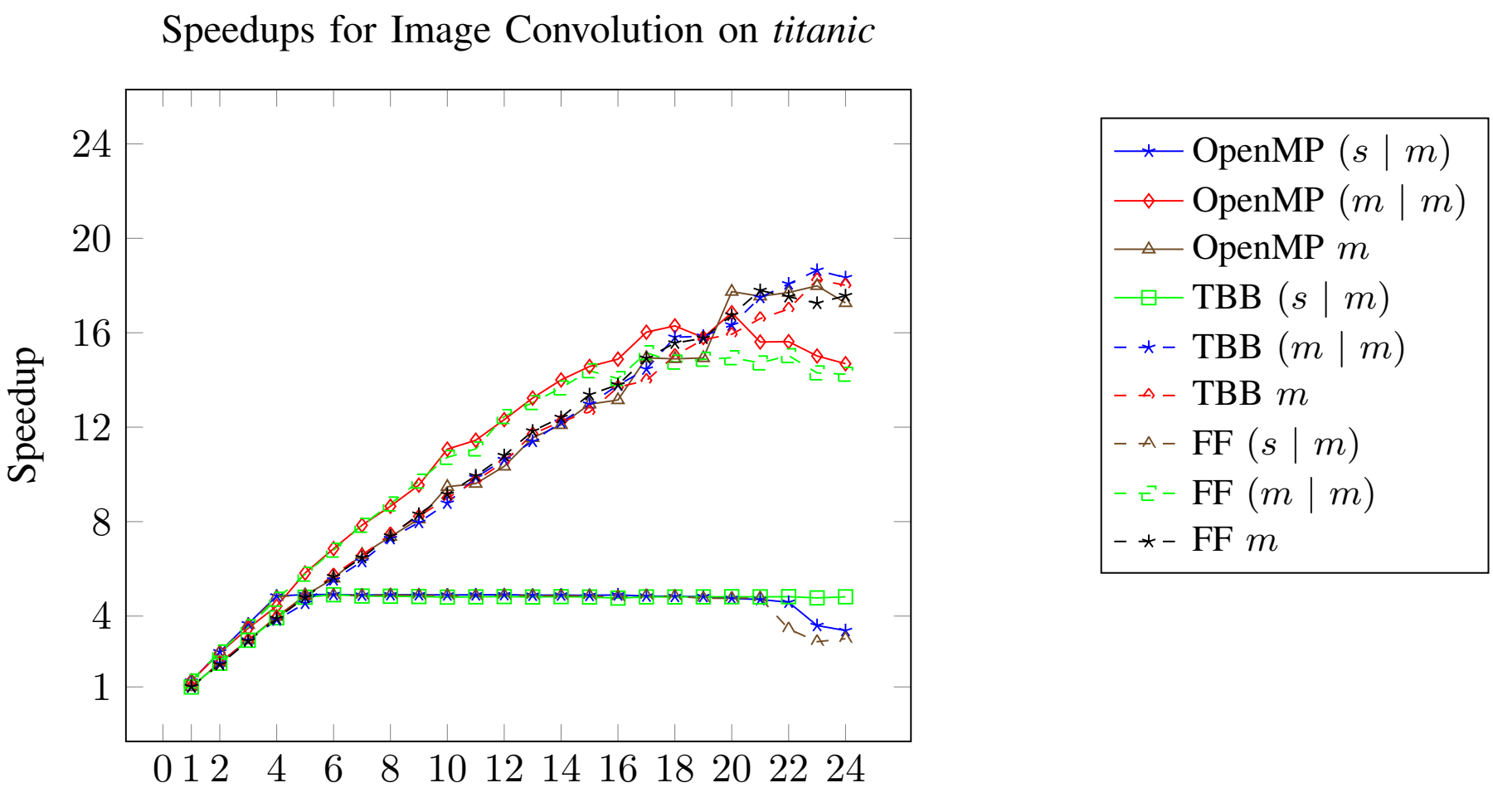


RePhrase

ParaFormance

Image Convolution

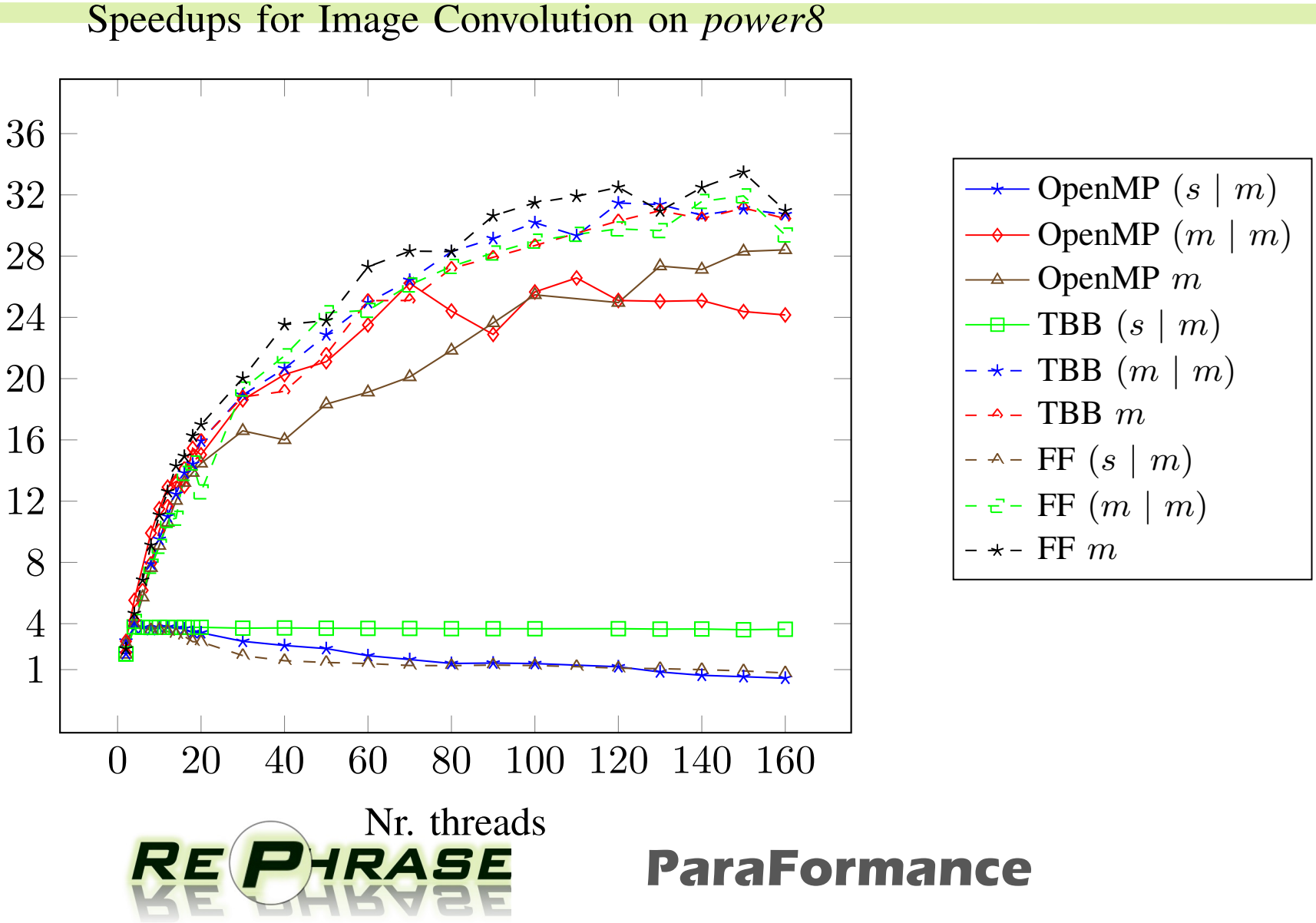
University of St Andrews



RePhrase

ParaFormance

Image Convolution – 20 Cores!



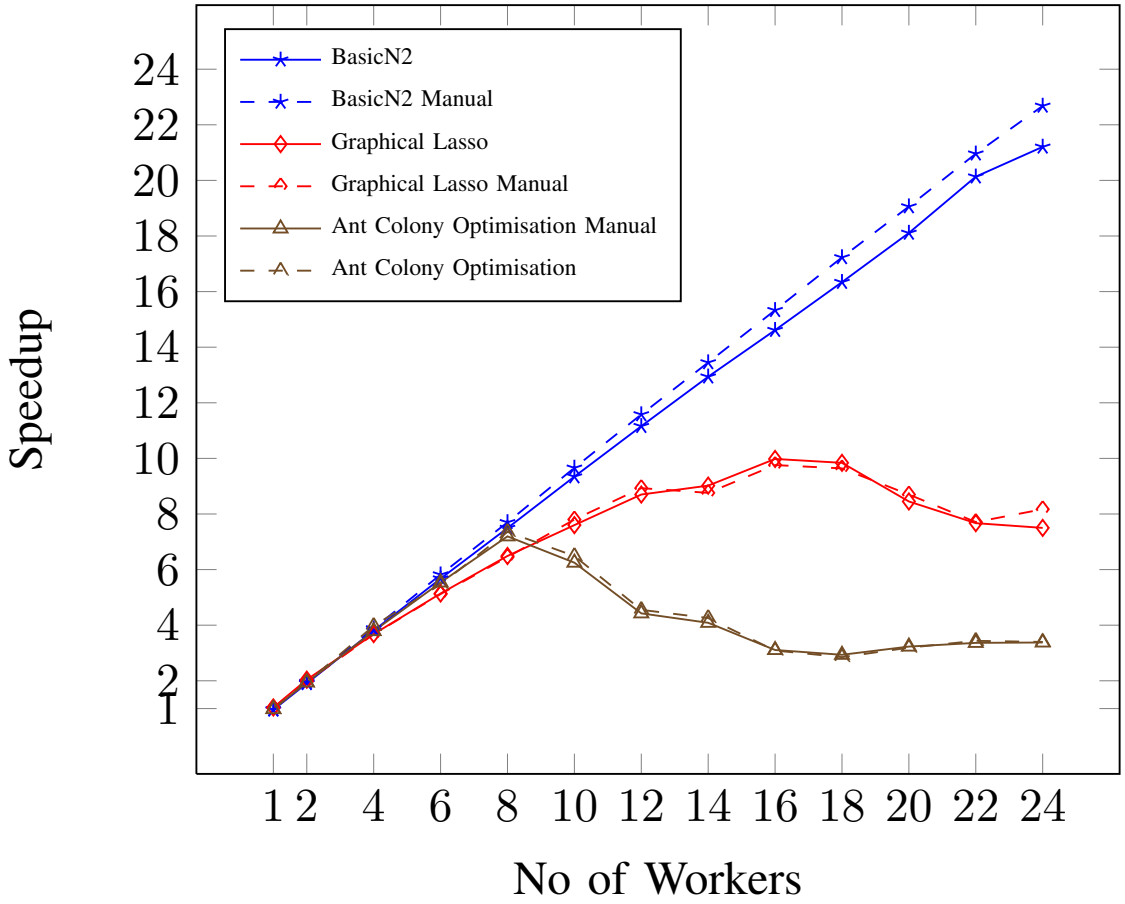
RePhrase

ParaFormance

Comparable Performance

University of St Andrews

Speedups for Ant Colony, BasicN2 and Graphical Lasso



RePhrase

ParaFormance

Conclusions

University of St Andrews

- Refactoring tool support:
  - Guides a programmer through steps to achieve parallelism
  - Warns the user if they are going wrong
  - Avoids common pitfalls
  - Helps with understanding and intuition
  - Reduces amount of boilerplate code
    - Allows programmer to concentrate on algorithm, rather than parallelism.

RePhrase ParaFormance

ParaFormance/Rephrase Needs You!

University of St Andrews

- Please join our mailing list and help grow our user community
  - news items
  - access to free development software
  - chat to the developers
  - free developer workshops
  - bug tracking and fixing
  - Tools for both Erlang and C++
- Subscribe at

https://mailman.cs.st-andrews.ac.uk/mailman/listinfo/rephrase-news

- We're also looking for open source developers...

THANK YOU!

http://rephrase-ict.eu

@rephrase_eu