Transformation of Functional Programs for Identifying Parallel Skeletons

Venkatesh Kannan. Supervised by: Dr. Geoff Hamilton

1 Parallel Hardware and Software

› Limitations in sequential execution of programs.

› Omnipresence of multi-core processors.

› Need for parallel execution of programs.

» **Functional programs are naturally parallelisable.**

› Free from side effects → Parallel evaluation of expressions with no data dependency.

› Mathematical soundness → Easy to analyse, reason about and transform into desirable form.

» **Existing Program Parallelisation Approaches.**

› **Parallel Skeletons.**

› Algorithmic forms of parallel computations.

› Used as building-blocks in parallel programming.

› Eg.: *map, reduce, scan, zip, divide-and-conquer, ...*

» **Program Transformation.**

› Systematic analysis and manipulation of programs.

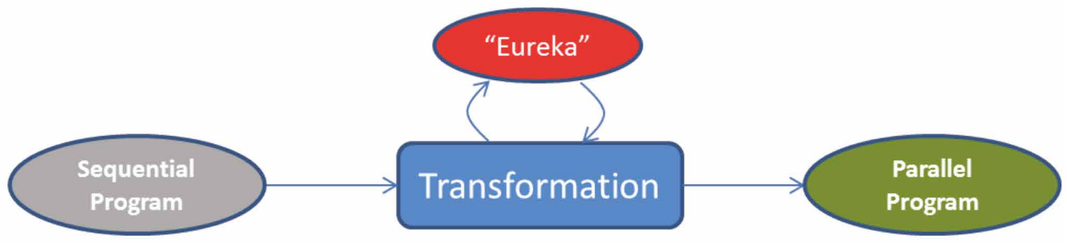
› Removes inefficiencies in a program.

› Extracts potential parallel computations.

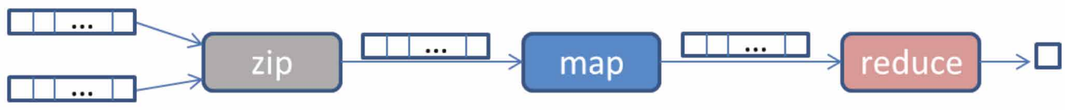
2 Existing Limitations

› Restrictions on number of recursive inputs and/or data types.

› Manual inputs required for some transformation techniques.



› Use of skeletons may introduce intermediate data structures.



» Objective.

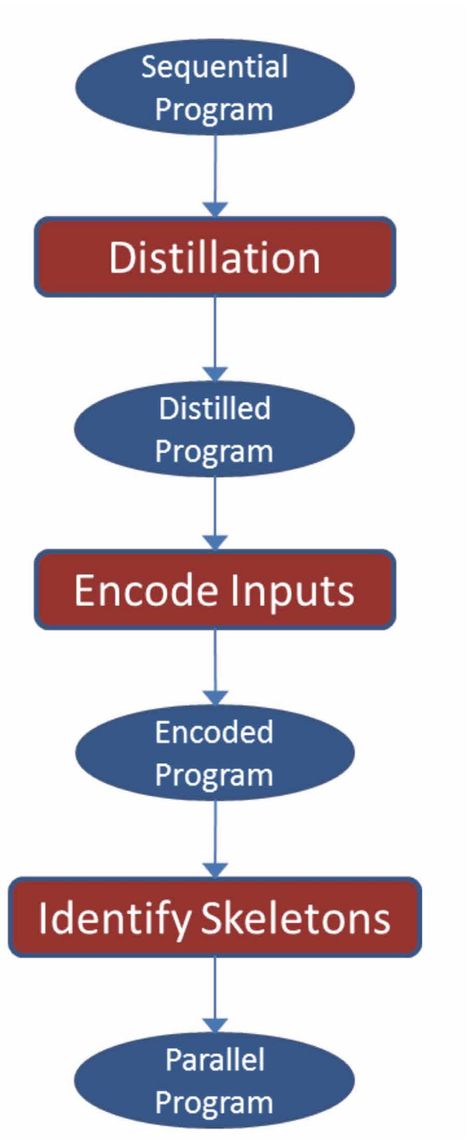
› Eliminate intermediate data structures using distillation [3].

› Transform distilled program by combining all inputs into a single recursive input.

› Transformed program is more likely to contain instances of data parallel skeletons (eg. map-reduce).

› Automatically identify skeletons instances in the encoded program.

3 Methodology



4 Results to Date

› Methods to encode inputs into a new data type or a list.

› Defining polytypic and list-based parallel skeletons.

› Method to identify skeleton instances.

› Tested on programs such as matrix multiplication, dot-product on trees, etc..

» **Next Steps.**

› Perform benchmark tests of the transformation technique on a wider range of programs.

› Evaluate the performance of parallel programs.

› Parallel implementations for polytypic parallel skeletons are not available yet, though they have been studied.

› Encode inputs into a list and use skeleton implementations that operate over lists.

» **References.**

1. Venkatesh Kannan and G. W. Hamilton, *Extracting Data Parallel Computations from Distilled Programs*, META 2014.

2. Venkatesh Kannan and G. W. Hamilton, *Program Transformation to Identify Parallel Skeletons*, Sumbitted.

3. G. W. Hamilton and Neil D. Jones, *Dsitillation with Labelled Transition Systems*, PEPM 2012.

Ireland's European Structural and Investment Funds Programmes 2014-2020
Co-funded by the Irish Government and the European Union

European Union
European Regional Development Fund

SOUTHERN Regional Assembly
Promoting our Region