1 Aims and Objectives of the STSM

Several pieces of work on algorithmic skeletons argue that predictability is one of their strengths. Several cost models, in terms of execution times, have been defined, and are good evidence of this. However, most of these cost models abstract away many aspects that have an important impact on the execution times of parallel applications, such as the cost of communication and synchronisation.

As part of our ongoing work, we are producing a new set of cost models, in terms of execution times, for common algorithmic skeletons on multicore architectures. The two main novelties of these cost models are that,

i they are tied to a new formal queue-based semantics of algorithmic skeletons that captures some low-level implementation details that have a great impact on execution times; and

ii they combine several type-based techniques, such as sized types, that have been proven to be useful for cost modelling in other contexts.

Aim. Define cost models, in terms of execution times, for algorithmic skeletons on multicore architectures, in a formally verifiable way.

We identified 4 research questions related to this aim, which we need to address:

RQ1: Can type-based mechanisms associate costs of algorithmic skeletons on multicore architectures with costs of parallel programs, described as combinations of these algorithmic skeletons, in a formally verifiable way?

RQ2: Can we integrate type-based costs of algorithmic skeletons with other type-based analysis, such as sized types, that have been proven useful for cost modelling of programs in other contexts?

RQ3: Can we define a model, or operational semantics, of algorithmic skeletons that captures the low-level implementation details that have the greatest impact on the execution times of parallel programs?

RQ4: Can we formally associate costs of algorithmic skeletons with their operational semantics?

RQ5: Can we use these mechanisms for reasoning, not only about execution times, but also about other extra-functional properties, such as energy consumption?

2 Goals Achieved

The work carried out during the STSM proved very useful to solve some of the research question, which will help us follow future directions for addressing our aims and objectives. Moreover, we succesfully identified the common ground between both lines of research, which opens the possibility for future collaboration.

2.1 Work Carried Out during the STSM

During the STSM we had several meetings to discuss the lines of work of both the University of St. Andrews, and the IMDEA Software Institute. On an initial meeting, we discussed informally the type-based approach that is carried out in St Andrews [1], and the abstract interpretation approach carried out at the IMDEA [2,3,4]. This led to sharing a number of papers for a more formal, technical discussion about finding a common ground between the work in St. Andrews and at the IMDEA [5,6,7]. This involved doing a talk of the work described in [1], extended with work-in-progress operational semantics and cost models for algorithmic skeletons. The technical discussions motivated a number of improvements for the operational semantics and cost models for extending [1], namely: a more composable description of the operational semantics, and an improvement to the work-in-progress integration with sized types.

2.2 Results

The main result of this STSM was the identification of common ground for collaboration between the two institutions, the identification of possible future collaboration based on this common ground, and the improvement of the work in [1] inspired by the work done in the IMDEA Software Institute. As an example of the common ground, sized-types are obtained via the sized-types abstract domains in [4], while [1] was being extended by adding sized types to the type structure and inference rules. A similar situation happens with cost equations. This opens the possibility to improve [1] by a number of alternatives:

1. Adapting the results in [4,5,7].
2. Directly using the abstract interpretation framework [2] by converting the programs that are analysed in [1] to a suitable form.
3. Embedding the elements in [1] into the abstract interpretation framework [2] by defining the suitable abstract domains (if needed).

In parallel to these conclusions from the technical discussions, we also improved the current status of the operational semantics and cost models for [1] in a number of ways.

1. The operational semantics for [1] was redefined in a more composable way by splitting it into two components: a low level, easy to understand language for describing the behaviour of the workers of a parallel process; and a way to combine workers of parallel processes so that many alternative parallel skeletons can be described in a uniform way.

2. The cost models for [1] were extended so that the sizes considered do not just cover an upper bound on the input sizes, but also lower bounds are described. This improvement was inspired by [4].

There is still much room for improvement, since these results just defined the first steps towards achieving our goals. However, the wide range of possibilities for collaboration, and improvement of the work in [1] are promising.

3 Future Collaboration with the IMDEA Software Institute

The work in [1] addresses correct program transformation techniques for parallelism, while the work in [2,3,4,5,6] focuses mostly on formal resource analysis, but also on other analysis that can be embedded into the abstract interpretation framework. Apart from the inspiration for improving the work in [1] independently, we identified the following possibilities of collaboration:

1. Porting the approach in [1], mostly the program transformation for parallelism, into the abstract interpretation approach.
2. Integrating [1] with the resource analysis of [2,3,4,5,6]. There are a number of alternatives that should be evaluated:
   (a) Obtain a low-level representation of the parallel processes in [1] via a suitable operational semantics, and translate them to the Horn Clause Intermediate Representation (HC IR) in a similar fashion to [5,6,7].
   (b) Investigate the possibility to translate directly high-level structures as in [1] to a suitable representation for using the resource analysis of the abstract interpretation framework.
   (c) Define an alternative operational semantics for [1] that is more suitable for using the resource analysis of the abstract interpretation framework. As an example, we may want to define the operational semantics in terms of some low-level language that their analysis techniques already handle.

A Approval Letter from the IMDEA

To be included.

Bibliography

[1] David Castro, Kevin Hammond, and Susmit Sarkar. Farms, pipes, streams and reforestation: reasoning about structured parallel processes using types and hylomorphisms. Proc. Int. Conf. on Functional Programming, 2016.

[2] P. Lopez-Garcia, M. Klemen, U. Liqat, M. Hermenegildo. A General Framework for Static Profiling of Parametric Resource Usage. Theory and Practice of Logic Programming, 32nd Int'l. Conference on Logic Programming (ICLP'16) Special Issue, To Appear, Cambridge U. Press, 2016.

[3] R. Haemmerl, P. Lopez-Garcia, U. Liqat, M. Klemen, J. P. Gallagher, M. V. Hermenegildo. A Transformational Approach to Parametric Accumulated-cost Static Profiling. 13th International Symposium on Functional and Logic Programming (FLOPS 2016), LNCS, Vol. 9613, pages 163–180, Springer, March 2016.

[4] A. Serrano, P. Lopez-Garcia, M. Hermenegildo. Resource Usage Analysis of Logic Programs via Abstract Interpretation Using Sized Types. Theory and Practice of Logic Programming, 30th Int'l. Conference on Logic Programming (ICLP'14) Special Issue, Vol. 14, Num. 4-5, pages 739–754, Cambridge U. Press, 2014.

[5] U. Liqat, S. Kerrison, A. Serrano, K. Georgiou, P. Lopez-Garcia, N. Grech, M. Hermenegildo, K. Eder. Energy Consumption Analysis of Programs based on XMOS ISA-Level Models. Proc. Int. Sym. Logic-Based Program Synthesis and Transformation (LOPSTR), 2013.

[6] U. Liqat, K. Georgiou, S. Kerrison, P. Lopez-Garcia, M. Hermenegildo, J. P. Gallagher, K. Eder. Inferring Parametric Energy Consumption Functions at Different Software Levels: ISA vs. LLVM IR. Int. Work. Foundational and Practical Aspects of Resource Analysis (FOPARA), 2015.

[7] U. Liqat, Z. Bankovic, P. Lopez-Garcia, M. Hermenegildo. Inferring Energy Bounds Statically by Evolutionary Analysis of Basic Blocks. HIP3ES Workshop 2015