Protocol-driven MPI program generation

Nicholas Ng

August 25, 2016

Category 1: Achievements during the project

This line of research work completed in the BETTY project is a framework for developing safe and distributed parallel Message-Passing Interface (MPI) [3] programs. The framework follows a correct-by-construction workflow, starting from a behavioural typed protocol language described below.

We present Parameterised Scribble (Pabble) [5, 6], an extension of the Scribble language [7, 8] to capture scalable protocols, parameterised over the number of roles available at runtime. Pabble language is grounded on theories of parameterised multiparty session types (MPST) [2], and uses a concise and expressive index notation to describe a variable number of participants arranged in multiple dimensions. A large class of parametric, structured message-passing programs which follow Pabble protocols can ensure safety and progress of communications.

We demonstrate an application of Pabble protocols through a top-down code generation framework of MPI programs [4]. The code generation process begins with defining a Pabble protocol for the topology of the MPI application. An MPI parallel program skeleton is automatically generated from the protocol, which can then be merged with sequential code kernels defining the sequential behaviours. The merging process is fully automated through the use of an aspect-oriented compilation tool LARA [1], which can mechanically combine MPI program skeleton with sequential code kernels without additional user input. In addition to custom-defined Pabble protocols, the code generation framework also provides pre-generated MPI skeletons for protocols of common parallel patterns, such as a ring, stencil or map-reduce.

Using the framework, programmers only need to supply the intended Pabble protocol and provide sequential code kernels to obtain parallelised programs. The process not only simplifies the development of MPI programs, the output programs are efficient and scalable MPI applications, that are guaranteed, free from communication mismatch, type errors or deadlocks by construction, improving productivity of programmers.

References

[1] J. a. M. Cardoso, T. Carvalho, J. G. Coutinho, W. Luk, R. Nobre, P. Diniz, and Z. Petrov. LARA: an aspect-oriented programming language for embedded systems. In *AOSD '12*, pages 179–190. ACM, 2012.

[2] P.-M. Denielou, N. Yoshida, A. Bejleri, and R. Hu. Parameterised Multiparty Session Types. *Logical Methods in Computer Science*, 8(4):1–46, 2012.

[3] Message-Passing Interface. `http://www.mcs.anl.gov/research/projects/mpi/`.

[4] N. Ng, J. G. Coutinho, and N. Yoshida. Protocols by default: Safe mpi code generation based on session types. In *CC 2015*, volume 9031 of *LNCS*, pages 212–232. Springer, 2015.

[5] N. Ng and N. Yoshida. Pabble: Parameterised Scribble for Parallel Programming. In *PDP*, pages 707–714. IEEE, 2014.

[6] N. Ng and N. Yoshida. Pabble: Parameterised Scribble. *SOCA*, 9(3-4):269–284, 2015.

[7] Scribble homepage. `http://scribble.org/`.

[8] N. Yoshida, R. Hu, R. Neykova, and N. Ng. The scribble protocol language. In *TGC 2013*, volume 8358 of *LNCS*, pages 22–41. Springer, 2013.