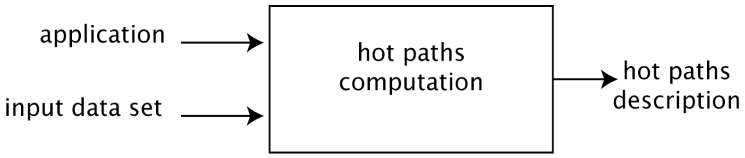
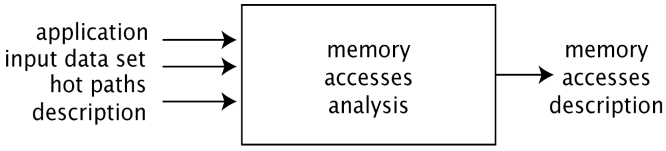
ASTEX

Code partitioning tool for hardware accelerators

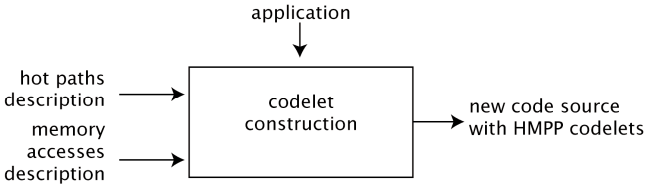
ASTEX[1] is a prototype tool to partition C codes between general purpose cores and hardware accelerators (GPU, FPGA, coprocessor, ...). The input of the tool is a runable application. Using runtime data, ASTEX extracts part of the code and provides speculative information on data structure accesses. HMPP™ directives[2] are used to express the partitioning in the application code. The code to be executed on the hardware accelerator is set up as HMPP™ codelets. A codelet is a compute intensive function with well-defined inputs and outputs. ASTEX works in three steps: the first collects hot paths in the application, the second analyzes at run-time the memory accesses, the last step builds the codelets.



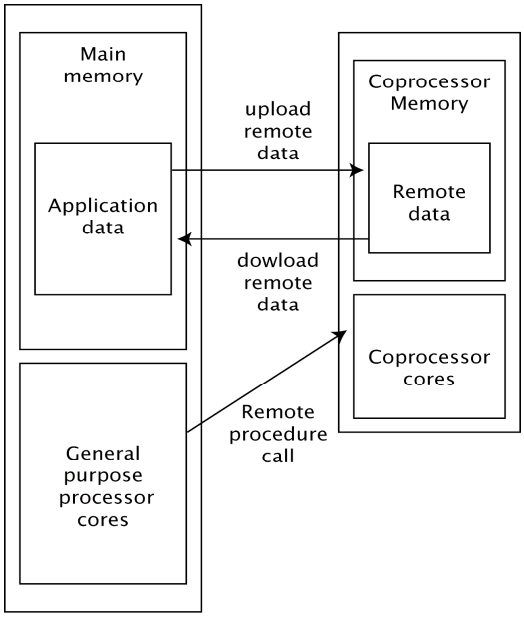
Memory access runtime analysis
This dynamic phase collects data size, aliases and data structure accesses for each potential codelets.



The memory analysis provides speculative information that can then be used at the codelet level to determine data transfers between the main computing units and the accelerators. These information can be used to guard the use of the codelet. For instance, the hardware accelerator can be activated only if the data size is large enough to get speedup.



Hot paths collection
The hotpaths found[3] are used to determine part of the code that can potentially be transformed as codelets. This step tries to find a set of potential codelets that covers most of the execution time of the application. Current implementation uses basic block frequencies stored in a compress trace.



HMPP™ codelet construction
This static step transforms the application to explicitly insert the codelets into the application code. Hot paths are converted into functions and HMPP™ directives are added. The HMPP directives are used to express in and out parameters as well as other conditions to be verified at run-time.

References
[1] E. Petit and F. Bodin. "Extracting Threads Using Traces for System on a Chip". In the proceedings of the Compilers for Parallel Computers (CPC 2006), A Coruna, Spain, January 2006.
[2] R. Dolbeau, S. Bihan and F. Bodin. "HMPP: A Hybrid Multi-core Parallel Programming Environment". In the proceedings of the Workshop on General Purpose Processing on Graphics Processing Units (GPGPU 2007), Boston, Massachussets, USA, October 4th, 2007.
[3] G. Pokam and F. Bodin. "An Offline Approach for Whole-Program Paths Analysis using Suffix Arrays". In the proceedings of the 17th International Workshop on Languages and Compilers for Parallel Computing (LCPC 2004), West Lafayette, Indiana, USA, September 2004.

This work is partially funded by ANR PARA, FAME2-Systematics and Sceptre-Minalogic projects.