PyPS a programmable pass manager

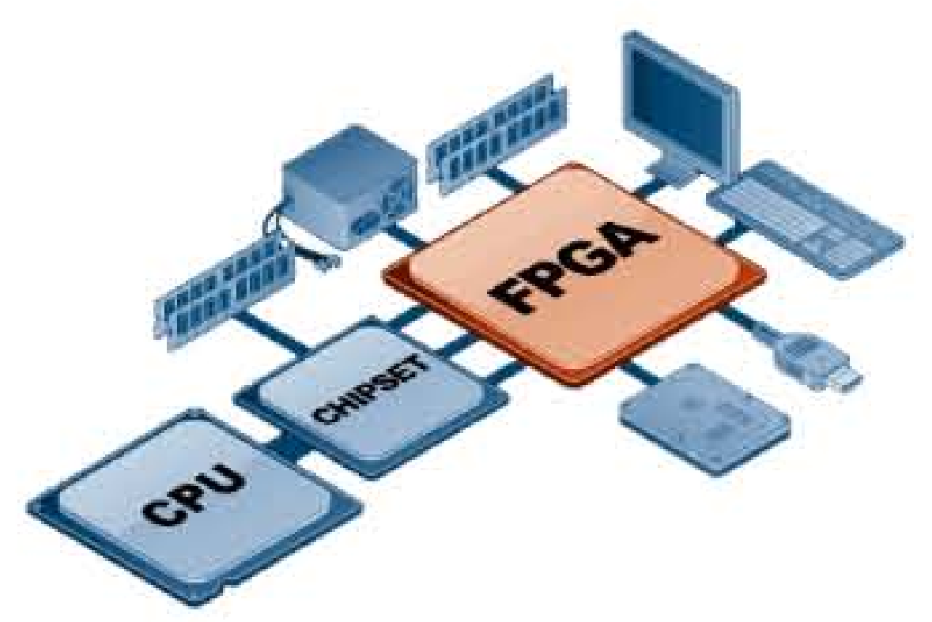
Serge Guelton, Mehdi Amini, Ronan Keryell, Béatrice Creusillet

http://www.pips4u.org/

PyPS
a programmable pass manager

1 - Complex Environment

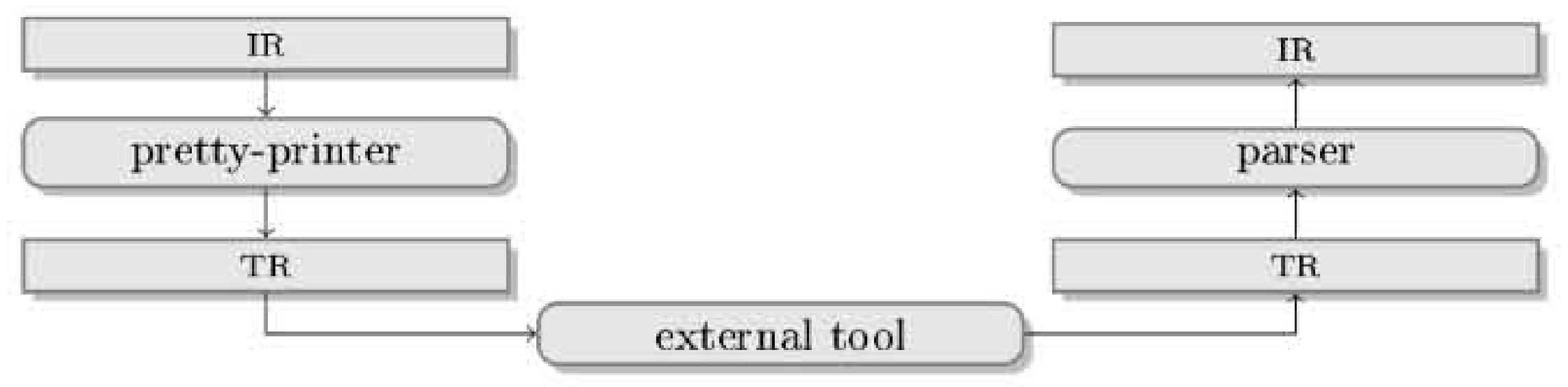
Parallel and heterogenous hardware
Compilers must be multi-target.
collaborate, and be specialized.

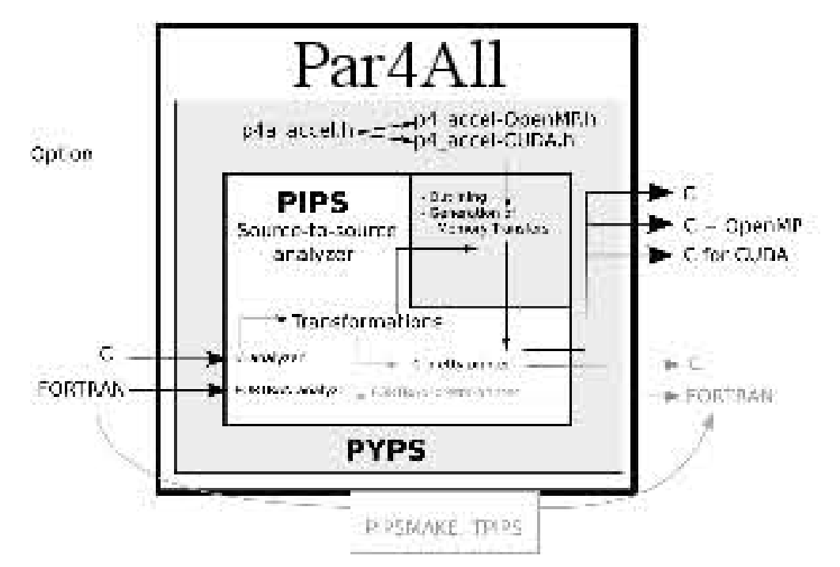


2 - Source-to-Source

Many source-to-source successful compilers.
Flexible transformation systems for heterogeneous computing :

- → parallelism detection algorithm,
- → variable privatization,
- → communication generation,
- → etc.





A code transformation is an application P → P that preserves the semantics of the program, that is to say:

$$\forall p \in \mathcal{P},\ \forall v_{in} \in \mathcal{V}_{in}(p), \quad \mathbf{P}(p, v_{in}) = \mathbf{P}(t(p), v_{in})$$

3 - Model for Code Transformations

Sequencing of code transformations is the compiler core.
Formal point of view for interactions between passes : several transformation composition rules.

A failsafe operator :

$$\forall t \in \mathcal{T},\ \forall p \in \mathcal{P}, \quad \tilde{t}(p) = \begin{cases} t(p) & \text{if} t(p) \neq \text{error} \\ p & \text{otherwise} \end{cases}$$

A failsafe composition:

$$\forall t_0, t_1 \in \mathcal{T} \times \mathcal{T}, \quad t_1 \tilde{\circ} t_0 = \tilde{t_1} \circ \tilde{t_0}$$

A conditional composition:

$$\forall t_0, t_1, t_2 \in \mathcal{T}\times\mathcal{T}\times\mathcal{T},\ \forall p \in \mathcal{P} \quad ((t_1, t_2) \underset{\sim}{\circ} t_0)(p) = \begin{cases} (t_1 \circ t_0)(p) & \text{if } t_0(p) \neq \text{error} \\ t_2(p) & \text{otherwise} \end{cases}$$

An error propagation operator :

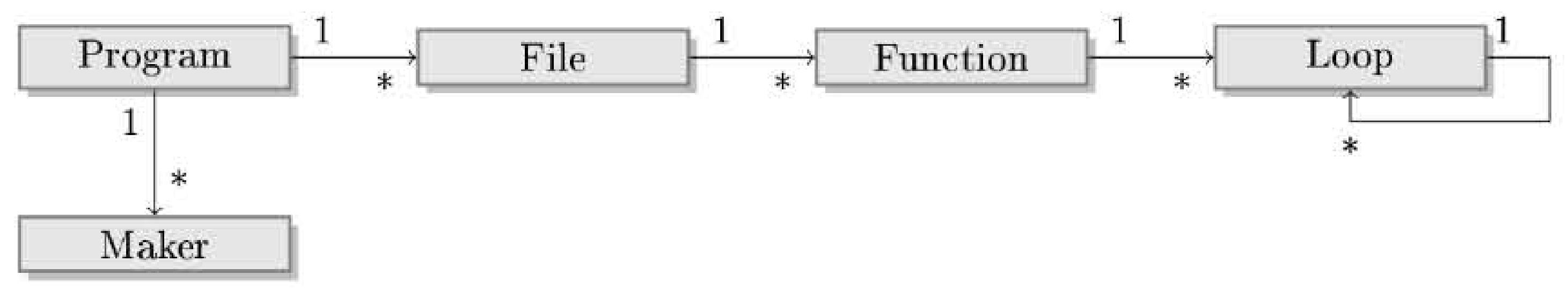
$$\forall t_0, t_1 \in \mathcal{T} \times \mathcal{T}, \quad t_1 \vec{\circ} t_0 = (t_1, \mathrm{id}_{\mathcal{T}}) \underset{\sim}{\circ} t_0$$

4 - Based on a Scripting Language

on the shoulders of giants

No DSL, does not reinvent the wheel, build over a high level language with a rich ecosystem wich widens the set of possibilities.

5 - Abstractions



Classes: high level program representation
high level compilation scheme
Methods: compose transformations into more complex ones
Inheritence: compose scheme for heterogeneous targets

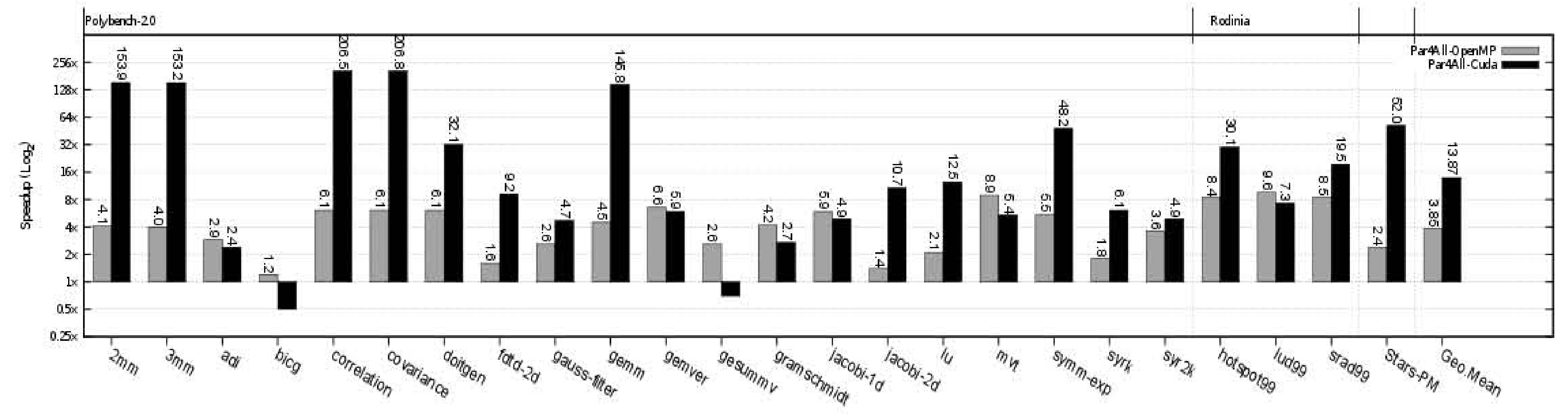
6 - Control Structures

Conditionals: manage switches, choose different compilation schemes.
For loops: iterate over the callgraph, loop nests.
Exceptions: recover from compilation failure, impossible transformations, etc.
While loops: look for fixed point for sequence of passes.

7 - Targets

OpenMP ; classic scheme but still illustrating basic functionalities.
TERAPIX is a fpga based accelerator for image processing from thales.
SAC vectorizer targets AVX, SSE, or NEON.
An Iterative Compiler ; try different compilation schemes or transformation flavors.
Par4All makes use of other compilers to provide automatic parallelization of applications to multiple hybrid architectures.

http://www.par4all.org/



Related Work

Automated Programmable Control and Parameterization of Compiler Optimizations. Yi. CGO 2011.
Finding effective optimization phase sequences. Kulkarni et al. LCTES 2003.
MILEPOST GCC: machine learning based research compiler. Fursin et al. GCC Developers' Summit, 2008.

Serge Guelton, Mehdi Amini, Ronan Keryell, Béatrice Creusillet