Jitted Object Stores for Virtual Machines

Master Project Proposal, Summer Term 2016
Software Architecture Group, Prof. Dr. Robert Hirschfeld

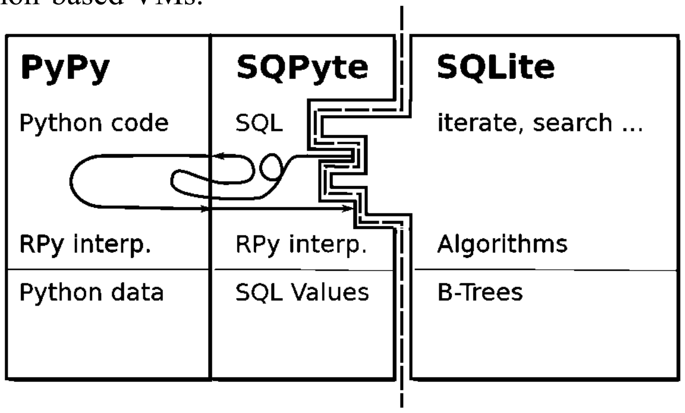
RSqueakVM

The RSqueakVM is a virtual machine for Squeak/Smalltalk written in RPython and developed at our group. It is based on the RPython translation toolchain that also powers the fast PyPy interpreter, and re-uses its virtual machine translation mechanism and garbage collection implementation. The RSqueakVM implements the standard Squeak bytecode set and combines powerful techniques such as storage strategies and shadow objects with a meta-tracing JIT to achieve performance that makes it feasible to run many of the functions that are traditionally implemented as part of the VM from pure Smalltalk.

BitBlt benchmark	in C	in Smalltalk
Interpreter VM	650ms	389,660ms
	1 x C	599 x C
Cog JIT VM	790ms	336,490ms
	1 x C	423 x C
R/SqueakVM	880ms	20,310ms
	1 x C	23 x C

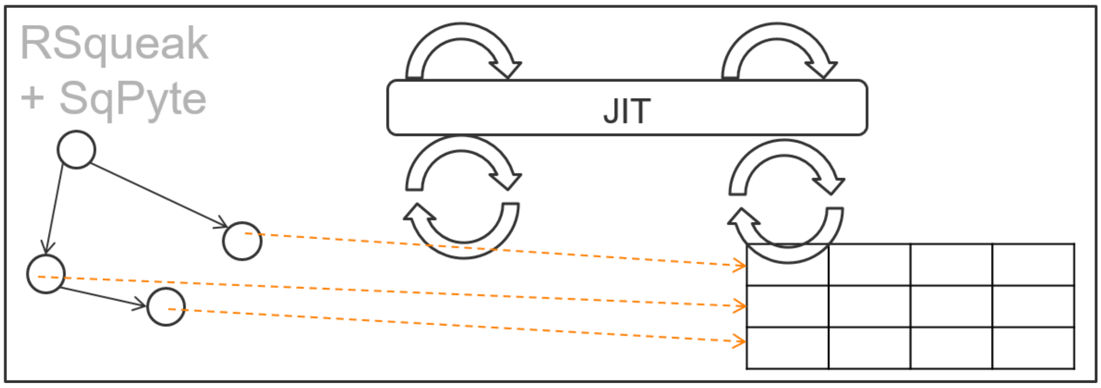
SQPyte

SQPyte is an implementation of the SQLite bytecode on top of RPython and its JIT. SQLite is the most commonly used embedded database, but getting the data across the boundary into a general purpose programming language is slow, due to conversion between program code and SQL as well as the different object representations. SQPyte implements the SQLite bytecodes in RPython to provide a platform for composing it with other RPython-based VMs.



Project Objectives

The core hypothesis of this project is that a composition of RSqueakVM with SQPyte can circumvent the inherent slowdown of crossing the language boundaries to the database by integrating both systems with the same RPython JIT. This way, it will no longer be necessary to write large parts of the application logic in SQL for performance, because high-level language code can be optimized just the same. This is especially important for algorithms such as Dunning or Elo, which cannot be implemented directly inside SQL. SQPyte itself is already faster than plain SQLite for micro-benchmarks, so the goal of this project is to achieve a combination of it with RSqueakVM and to implement some algorithms (such as Dunning) and measure their performance.



Contact

Prof. Dr. Robert Hirschfeld, Tobias Pape, Tim Felgentreff, Jens Lincke, Toni Mattis, Johannes Henning, Patrick Rein
Software Architecture Group, Potsdam
http://www.hpi.uni-potsdam.de/swa, hirschfeld@hpi.uni-potsdam.de

Further Reading

Carl Friedrich Bolz, Darya Kurilova, and Laurence Tratt. *Making an Embedded DBMS JIT-friendly*

Tim Felgentreff, Tobias Pape, Lars Wassermann, Robert Hirschfeld, Carl Friedrich Bolz. *Towards Reducing the Need for Algorithmic Primitives in Dynamic Language VMs Through a Tracing JIT*