

OpenTuner: An Extensible Framework For Building Autotuners

Jason Ansel, Una-May O'Reilly, Saman Amarasinghe

MIT, CSAIL

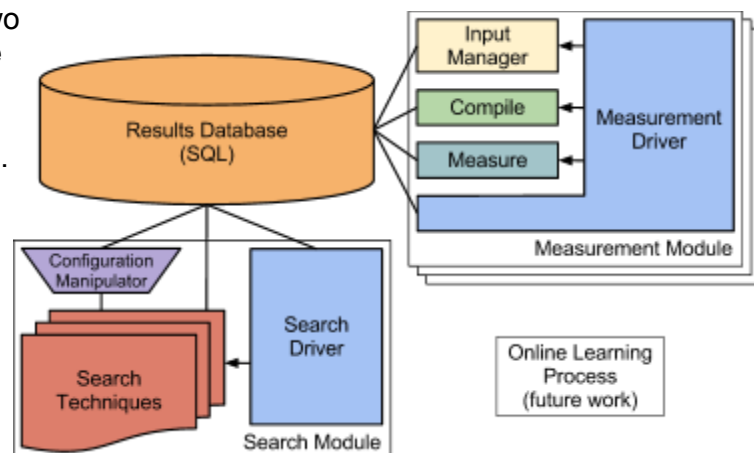
About OpenTuner

Program autotuning has been demonstrated in many domains to achieve better or more portable performance. However, autotuners themselves are often not very portable between projects because using a domain informed search space representation is critical to achieving good results and because no single search technique performs best for all problems. Each search technique has strengths and weaknesses. Incorporating an autotuner today requires expertise in machine learning as well as deep domain knowledge of the problem. Thus, autotuners are still not widely adopted. Most autotuners in use are not fully utilizing the capabilities of machine learning.

OpenTuner is a new framework for building domain-specific multi-objective program autotuners, it is not an autotuner by itself, but rather a toolbox to allow for the rapid construction of custom autotuners to fit a specific problem. OpenTuner supports fully customizable configuration representations, to allow for the representation of complex structures such as decision trees or data layouts; an extensible technique representation to allow for domain-specific techniques, such as existing heuristics or hand coded solutions; and an easy to use interface for communicating with the tuned program. A key capability inside OpenTuner is the use of ensembles of disparate techniques simultaneously, techniques which perform well will receive larger testing budgets and techniques which perform poorly will be disabled. This provides a more robust solution than a single technique, and techniques can share information synergistically to work together to arrive the optimal solution faster. With OpenTuner, we believe that a domain expert can build an effective autotuner that will use appropriate machine learning techniques with minimum effort.

Framework

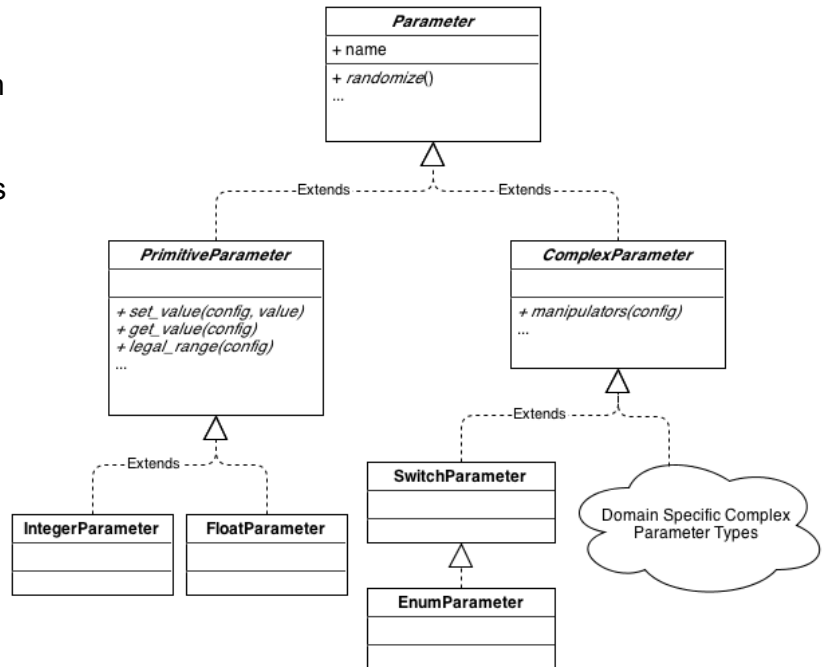
OpenTuner is divided into two main modules. The **search module** manages a set of techniques which read results and produce new suggested configurations to explore. The techniques mutate configurations only through the configuration manipulator to allow custom representations. The **measurement module** can be run many times in parallel and is responsible for determining the



fitness of proposed configurations. Fitness can be a user defined function of time, accuracy, and confidence and is tied both to an *input class* and to a *machine class*, which can be defined in the interface to the measurement module. The **results database** stores all information about tested configuration performance and allows that information to be queried in a variety of ways. This includes long term learning, where results from different inputs, machines, or program versions can be used to guide a new search.

Configuration Representation

OpenTuner techniques mutate configurations through a **configuration manipulator** which is a set of extendable Parameter objects. The class hierarchy of parameter objects (part of which is shown to the right) is divided into two main types. **Primitive parameters** are bounded integer and float types and are the most nature for many types of techniques which may try to do things such as fit curves to the expected values of parameters. **Complex parameters** can not fit on a linear scale and instead defined by a set of custom manipulator functions which make stochastic changes to the underlying configuration structure. All types of parameters can be extended by the user in order to add both domain specific structures and to support arbitrary underlying configuration types.



Using OpenTuner

To use OpenTuner, the user (at a minimum) must:

- Instantiate a configuration manipulator using predefined and/or custom parameter types.
- Implement a measurement routine to run a configuration and return its performance.

OpenTuner is implemented in python and supports most popular database backends, it uses SQLite by default in order to require no database setup.

Contact

We are looking for early adopters to test and help motivate the design of OpenTuner. If interested please email: jansel@csail.mit.edu