Cachepy

Cachepy is a framework for disk based caching of arbitrary python function calls. It has been conceived primarily with various forms of compilation caching in mind, but it should be useful in other contexts as well. The two obvious benefits are the elimination of duplicate work, and the elimination of dependencies, by generating a distributable cache.

Features:

* Caches are thread/process/NFS\* safe. Multiple processes will coordinate to execute the cached operation no more often than necessary.
* No possibility of hash collisions; a unique representation of the cached arguments is stored entirely, and hashes are only used to accelerate cache lookup
* Arbitrary key objects are allowed, and serialized in a unique and deterministic\* way. This makes caching of functions mapping complex datastructures to complex datastructures painless
* Caches will recompile given different function arguments, but will also optionally monitor the state of a given environment description, which may have relevance to the outcome of the cached process
* Easy to use API; either annotate functions or subclass CachedOperation

\*WIP features

Possible extensions/improvements/generalizations:

* Tools for distribution of caches. This may free the end user of cumbersome dependencies, such as compiler installations.
* More different key-value store backends specialized to serve different use cases
  + An in-memory backend may also be useful, in case we do not care about persistence between sessions, but rather frequent calls within the same session. The deterministic serialization and integral key framework is orthogonal to the backend storage. Being able to set a flag to toggle to in-memory caching, or effective memoization, would be neat
  + Server process type database backend.
* Automatic cache cleanup: we may monitor the cost of executing the cached operation, and in conjuction with usage statistics of cache entries, a disksize target could be maintained.
* Integrate efficient handling of numpy arrays in the key-value store. This can be copied from joblib

Note that cachepy is similar in intent to joblib; but considering it has somewhat different aims, a new project has been started. Both projects are sufficiently similar that ideally, the features of both should be merged into one package in the future. But for now, implementing the specific needs of compilation caching would clash with backwards compatibility in joblib.

Example:

    @cached

    def compile(source, templates):

        print 'compiling'

        sleep(1)

        return source.format(\*\*templates)

    print compile('const {dtype} = {value};', dict(dtype='int',value=3))     #calls compile

    print compile('const {dtype} = {value};', dict(dtype='int',value=3))     #gets value from cache