Summer Of Code 2007 Projects Proposal

here is the (draft) list of harmony tasks for G SOC 2007. Candidates will have to come to the mailing list and discuss details (regularly).

* Java bytecode translator refactoring

Problem: interaction between `ByteCodeParser`, `JavaLabelPrepass`, `JavaByteCodeTranslator` classes is very complicated and error-prone.

Task: Re-factor Java bytecode translator in the Jitrino.OPT to make the code cleaner and simplify the data structures used.

* One High-Level Optimization implementation and tuning

Problem: some of important proved as effective optimiztions in Jitrino.OPT JIT compiler are not impremented

Task: Implement one of the following classic compiler optimizations: `loop_peeling`, `partial_redundancy_elimination`, `operator_strength_reduction`. Compose a number of microbenchmarks uncovering optimization flaws in existing implementation. Show that your code performs better on these microbenchmarks, passes the regular tests, does not decrease performance on our high-priority benchmarks: `DaCapo` and `SciMark`. Increasing the popular benchmark scores is also highly desirable.

* Parallelize our ant-based build system

Problem: current build system cannot utilize more than one core in a multicore/multiprocessor SMP box

Task: easily deducible from the problem

* Implement a JDK command-line tool

Problem: not all jdktools are implemented

Task: pick a tool (or a set of tools) to implement, estimate the number of effort with help of Harmony community. And.. do the clean-room implementation.

* Enlarge the heap on x86_64

Problem: currently only < 2 GB large heaps are supported on the x86_64 platform (implemented with compressed pointers) while the opportunity for larger heaps is promising

Task: design the system so that it can support compressed and uncompressed pointers in the same VM instance in an effective way

* Implement Exception to branch translation in Jitrino.OPT optimizing JIT compiler

Problem: Throwing an exception is relatively expensive optimization in VM. Locally handled exception can be translated to the usual branch. This optimization should improve performance of the application extensively using exceptions, such as javac.

Task: Implement exceptions to branch optimization in Jitrino.OPT. Show performance improvement on the micro-benchmarks and javac.

* Implement bytecode-based edge profile and value profile instrumentation in the Jitrino.JET JIT compiler

Problem: Currently edge profile and value profile are supported in the Jitrino.OPT compiler only. Jitrino.JET which facilitates fast application startup lacks this support. We can combine advantages of the client mode (fast startup) and the server mode (good performance) in one mode if we implement full edge and value profile support in Jitrino.JET.

Task: Implement instrumentation of the code in Jitrino.JET JIT compiler to facilitate gathering edge and value profile. Since bytecode based mapping is the only mapping available in Jitrino.JET, Jitrino.OPT profile support should be adjusted to use bytecode-based mapping as well. Implement optimization mode with 2 compilation stages:

1. Compilation from Jitrino.JET with edge and value profile support

2. Recompilation by the optimizing Jitrino.OPT compiler with use of the profiles.