



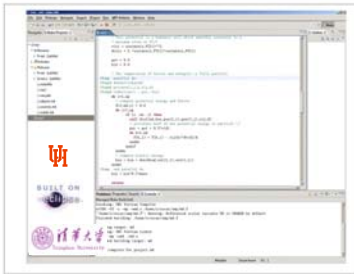
# OpenUH and OpenMP Validation Suite

<http://www.cs.uh.edu/~openuh>



## OpenUH: A Portable and Optimizing OpenMP Compiler

We are pleased to announce the release of the **OpenUH** compiler by the High Performance Computing Tools (HPCTools) group of the University of Houston in collaboration with Tsinghua University. OpenUH is a robust, optimizing, portable OpenMP compiler, which translates *OpenMP 2.0* ([www.openmp.org](http://www.openmp.org)) directives in conjunction with C, C++, and FORTRAN 77/90 (and some FORTRAN 95). OpenUH is available as stand-alone software or with the Eclipse integrated development environment. Our release also includes Dragon, a tool for browsing an application's source code, callgraph, flowgraph, profile information and more.



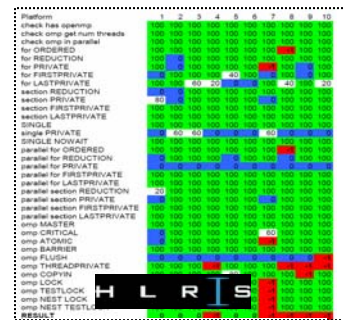
Openuh development environment

OpenUH is based on SGI's open source Pro64 compiler, which targets the IA-64 Linux platform. Our compiler's major optimization components are the *interprocedural analyzer*, the *loop nest optimizer* and *global optimizer*. In order to achieve portability while preserving most optimizations, we have enhanced the suite's IR-to-source translators to produce compilable code immediately before the code generator. To achieve greater stability we merge work from the two major branches of Open64 (ORC and Pathscale) to exploit all upgrades and bug fixes.

We have strong working relations involving our compiler and tools with multiple partners in the research and business sectors. We have installed OpenUH at the National Center for Supercomputing Applications and NASA Ames for a pilot evaluation. We are using this compiler for research into OpenMP language extensions and into novel translation techniques. OpenUH can be [downloaded](http://www.cs.uh.edu/~openuh) at: [www.cs.uh.edu/~openuh](http://www.cs.uh.edu/~openuh).

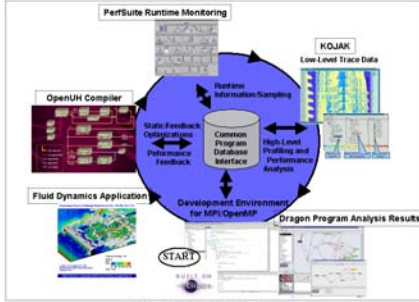
## OpenMP Validation Suite

In addition, we are announcing the release of the *first* public OpenMP validation suite. The validation suite is the result of a collaborative effort between the HPCTools group and the High Performance Computing Center at Stuttgart, Germany. The test suite is in conformance with OpenMP specifications 1.0 and 2.0, is complete for both *Fortran* and *C*, and permits tests to be individually or collectively executed. [The test suite can be downloaded at: www.cs.uh.edu/~openuh](http://www.cs.uh.edu/~openuh).



Sample output from OpenMP validation suite.

## COPPER: Compilation and OPTimization with PERformance feedback



COPPER integrated development life cycle

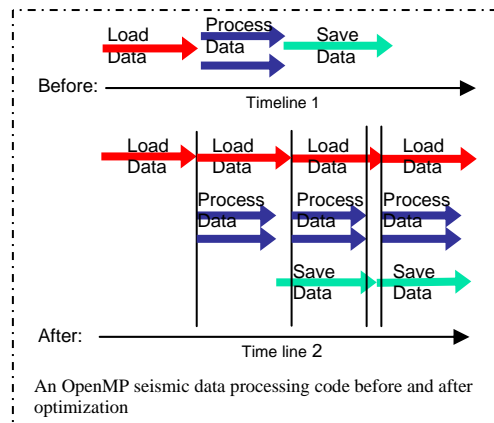
In a collaboration with colleagues at NCSA (UIUC), the University of Tennessee (UTK), and Virginia Tech (VT), we are designing and implementing interfaces and strategies for compiler and performance tools interaction. The goal is to better support the process of developing and tuning high performance codes in MPI and OpenMP. HPCTools contributes OpenUH and Dragon to this effort, NCSA contributes PerfSuite, UTK provides KOJAK and PAPI, and VT implements applications to support the development and evaluation of the new environment. We are also exploring the need for additional support by the

OpenMP standard for tools, particularly for profiling (KOJAK) and hardware counter information (PerfSuite and PAPI). For wider interoperability, we enhanced PDB (program database toolkit) for selective instrumentation and extended the Fortran 90 front end to support PDB.

## OpenMP Language Research

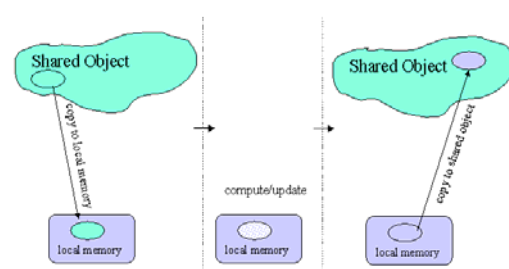
We implemented an OpenMP version of TracePak, an I/O intensive program from Seismic Micro-Technology, Inc. to process 2-D and 3-D seismic data for oil searching. Our OpenMP implementation of TracePak achieves very encouraging performance on SMPs with or without Chip MultiThreading.

We use the experiences gained in this work to explore and propose modest extensions to the OpenMP API that permit a subteam of threads to work independently. This kind of feature may help distribute work to threads appropriately on multithreaded platforms, enable the overlapping of I/O and computation, and may facilitate modular application development.



## Cluster-Enabled OpenMP

Given the importance of cluster computing, we are evaluating approaches to providing OpenMP on clusters. The traditional approach relies on software distributed shared memory, which incurs high overheads and (unless it is integrated with a compiler) is not amenable to important code optimizations. An alternative solution involving translation to MPI is hard to implement. We chose instead to explore a translation using Global Arrays



The mechanism of Global Arrays

Our solution is straightforward and permits a variety of compile-time improvements. An implementation is currently under way.