Roadmap of Upcoming Research, Features and Releases

Jeff Hollingsworth
hollings@cs.umd.edu

Barton P. Miller
bart@cs.wisc.edu
Supported Systems

- Solaris 8 (SPARC)
- Windows NT 4.0 & 2000 (x86)
- Linux 2.4 (x86)
- AIX 4.3 (Power2/SP2)
- IRIX 6.5 (MIPS) – *DyninstAPI only*
- Tru64 Unix 4.0 (Alpha) – *DyninstAPI only*

Current version: Paradyn v3.3 & Dyninst v3.0 released Jan 17, 2002
Baseline Paradyn UI/Front-end

- Synchronized Call Graph and Where Axis selections
- Numerous Stability Improvements
- Numerous Compiler/OS Compatibility Improvements
Baseline Paradynd/runtime-library

- Shared object handling: all platforms
- Shared-memory metric data sampling
- 64-bit data path
- Support for GCC and native C compilers
- Support for Fortran common blocks
- Support for DyninstAPI CFG generation
- Optimized runtime library
- Improved C++ DyninstAPI tests
- Function name aliasing (simplifies MDL)
- Shared instrumentation code optimizations
- Scalability feature: large numbers of communicators and tags.
- Scalability feature: efficient aggregation from many daemons.
Linux (x86)

+ Hardware wall timers
+ High-resolution timers, includes kernel patch
+ Function relocation to avoid trap-based instrumentation
+ Detach-on-the-fly to minimize trap overhead
+ MPI (MPICH 1.2)
+ Linux 2.4 support
? Multiple glibc versions
Linux (x86) Hardware Timers

We developed libhrtime, a kernel patch for hardware counter access, providing:

- Third party timer access
- User/System time selection
- Memory-mapped access

Mikael Pettersson at Upsalla Univ. developed the perfctr time patch, providing:

- Wider variety of hardware counters
- Memory-mapped access
- Fixed time selection.

We are currently developing a unified version of this patch to promulgate to the Linux community.
AIX (Power)

+ POE MPI
+ Retroactive “catch-up” instrumentation
+ Instrumentation for threading: major development platform.
+ Hardware CPU timers
+ More efficient wall time sampling
Windows NT/2000 (x86)

+ Hardware wall timers
+ Function relocation to avoid traps
+ Visual C++ and Digital Fortran f90 support
+ Dyninst API now available as a DLL
+ Dyninst line number information
  – GCC C support
  – Support for external debugging information
  – Support for frame pointer optimization (FPO)
  – “Detach-and-leave-running” (NT limitation)
Solaris (SPARC)

+ Unified function relocation scheme
+ Dyninst line number information
+ Solaris 8 support
+ Instrumentation for threading: major development platform.
  – SunWorkshop f90, CC, MPI support
IRIX (MIPS)

+ DyninstAPI available from Maryland
Next Release  
(tentative features)

- Memory Instrumentation  
  - Power, SPARC, x86 platforms
- Saving Instrumented Binaries  
  - x86, SPARC, Power
- DPCL on top of Dyninst  
  - x86/Linux, Power
- TruUNIX Port running on release 5.x
- IA-64/Linux port
- Threading  
  - SPARC and Power.
Major Research Initiatives

Scalability
- Multiprocessor front-end
- Mcast/reduction network
- Distributed Performance Consultant
- Scalable visualization of qualitative performance results.

BinInst: Portable, General Purpose Binary Rewriter
- Key tool for many applications
- Uses foundation of saving application and function relocation.

Tool Daemon Protocol
- Launch runtime tools where no tools have gone before.
- Portability of tools between runtime environments: Condor, MPICH, Globus, etc.

Instrumentation for Threaded Programs and OpenMP
Major Research Initiatives

UMD/UT/UW DOE/MICS Project
- Standardize access to processor hardware counters
- Support at both Dyninst and Paradyn levels
- Establish IA-64 port of dynamic instrumentation

UMD in SciDAC ISIC PERC Project
- Extend Dyninst to support memory instrumentation
- Basis for study of cache performance issues.

UW ONR CIP/SW Security Project
- Dyninst as vehicle for White Hat security attacks.
- Potential early customer for binary rewriting tool.

EU APART (Automated Performance Analysis)
A Balancing Act

Paradyn and Dyninst:

• A platform for far-reaching and exciting research ideas.
• Produce usable tools for real users.
  • Direct distribution to users.
  • Indirect distribution through industrial partners (e.g., IBM)

These are compatible and synergistic goals (though they tend to keep us up late at night!).
THE TECHNOLOGY DEMO

THE SOFTWARE ISN'T 100% COMPLETE.

IF IT HAD A USER INTERFACE YOU WOULD SEE SOMETHING HERE... HERE...AND SOMETIMES HERE.

AND THEN YOU'D BE SAYING, "I GOTTA GET ME SOME OF THAT."

ANY QUESTIONS?

Copyright © 2000 United Feature Syndicate, Inc. Redistribution in whole or in part prohibited.