



SGI Developer Tools Update



Jim Galarowicz

***Manager of Performance Tools and
Debugger Tools Group***

SGI Developer Tools Update

- Topics covered in this presentation
 - Introduction to WorkShop and SpeedShop
 - What are the current releases?
 - Feature highlights of the current releases.
 - Features scheduled in the next releases.
 - Appendix with screen shots and examples

SGI Developer Tools Update

- Introduction to WorkShop and cvd
 - WorkShop is collection of tools
 - cvd - GUI based source level debugger
 - dbx - command line source level debugger
 - cvcov - coverage tool - what parts of my program are being executed
 - cvperf - performance analysis viewer
 - cvd <executable> or dbx <executable>
 - cvd mpirun -args -np 64 <mpi_executable>

SGI Developer Tools Update

Introduction to WorkShop and cvd

– cvd features and capability

- OpenMP, MPI for single system image, pthreads, shmem, MP mixed codes
- Fortran (f90,f77), C++, C, Ada, mixed lang. codes
- o32, n32, and 64 bit ABI's
- specialized views for source, data, instruction, register information
- parallel view to control all or specific processes and/or threads

Multiprocess View

Admin Config Process

Cont Stop Step Next Source

PID:	PPID:	Status:
66619	66621	Stopped
66699	66621	on entry main
66673	66621	on entry main
66735	66621	on entry main

WorkShop Debugger (pid 66619)

Admin Views Query Source Display Perf Traps PC Fix+Continue Help

Command: /usr/bin/mpirun -np 4 cvd_test

ALL Cont Stop Step Next Return Source Print Kill Run Debug

Status: Process 66619: Stopped
main(<stripped>) ["cvd_test.c":186, 0x000010001de8]

```

int main(int argc, char **argv)
{
    int cc, len, flag, max_procs, nrank;
    MPI_Comm child_intercomm, parent_intercomm;
    char error_string[MPI_MAX_ERROR_STRING];

    cc = MPI_Init(&argc, &argv);

    MPI_Attr_get(MPI_COMM_WORLD, MPI_UNIVERSE_SIZE, &universe_size, &flag);
    if(flag == 0) {
        fprintf(stderr, "FAILURE: MPI_Attr_get for MPI_UNIVERSE_SIZE attributed FAILED!\n");
        exit(-1);
    }

    MPI_Comm_get_parent(&parent_intercomm);

    if(parent_intercomm == MPI_COMM_NULL) {
        MPI_Comm_size(MPI_COMM_WORLD, &nrank);
        max_procs = *universe_size - nrank;
    }
}

```

File: /B1/gala/pv_bugs/hpp/cvd_test.c (Read Only)

```

cvd>
cvd>
cvd>
Process 66619 stopped at ["cvd_test.c":186, 0x000010001de8]
cvd> where
> 0 main(<stripped>) ["cvd_test.c":186, 0x000010001de8]
! __start(<stripped>) ["crt1text.s":177, 0x000010001958]
cvd>

```

What are the current releases?

- Released Jan 8th, 2002
- WorkShop 2.9.1
 - Download WS 2.9.1 from supportfolio - current patch 4531
- SpeedShop 1.4.3
 - Download 1.4.3 from supportfolio - current patch 4532
- dbx 7.3.3
 - Download 7.3.3 from supportfolio - current patch 4530
- <http://support.sgi.com/colls/patches/tools/relstream/index.html>

Feature highlights in the new WorkShop/dbx release

- Visualizing arrays of pointers/derived types
- F90 debugging improvements
- Pthread debugging improvements
- Auto-dereferencing for pointers of simple types
- Improvements to the speed of execution of cvd
- Other Improvements (merge, other)
- Coverage tool improvements (cvcov)

Feature highlights in the new WS/dbx release

- *F90 debugging improvements*
 - Array browser allows display of derived types
 - All views can now display data declared outside a F90 internal procedure while in the internal procedure.
 - Indexing Fortran90 pointer-arrays within a derived type now works. For example, 'dt%array(i,j)'.
– Other problems have been fixed in the release and in patch 4531 (WS) and 4530 (dbx).

Feature highlights in the new WS/dbx release

- *Pthread debugging improvement*
 - Stepping over `pthread_create` now has consistent behavior. Prior to the fix you could get stuck in “Running state”
 - Fixes to Multiprocess view (MPView).

Feature highlights in the new WS/dbx release

- *Improvements - speed of execution of cvd*
 - Some key server routines were reworked to be faster
 - Raised the compiler optimization level in building cvd
 - Stepping over (next) is faster

Feature highlights in the new WS/dbx release

Other Improvements

- *Due to merging cvd/dbx server*
 - Several cases closed that were fixed in one or the other
 - 64 bit debugging improvements
 - Common source base for fixing server problems.
- *Gui Area*
 - Double clicking from Trap Manager positions source
 - Wide character WorldView fixes - can see comments
 - Clearcase support

Feature highlights in the new WS/dbx release

*WorkShop coverage tool improvements
(cvcov) - reports execution statistics.*

- Improved C++ support (lang:std)
- Updated the default exclusion file
- Reduce cvcov memory leaks - purify,SS
- Improve cvcov speed of execution on large appl.

SGI Developer Tools Update

WorkShop and dbx
Features planned for the next release.

Features planned for the next WS/dbx release.

- *GUI data access improvements*
 - Single mouse action process & data navigation
 - Reengineered Multiprocess Explorer (was MPView)
 - Reengineered Data Explorer (was Structure Browser)
- *OpenMP debugging improvements*
- *cvd/dbx memory usage improvements*

Features planned for the next WS/dbx release.

- *Process & data navigation across cvd*
 - One mouse action access to data
 - Right mouse button down give default action
 - Left or Right mouse button hold gives dynamic menu
 - Data display window for lightweight data display (reusable or throw away)
 - Reusable can access previous data panels

Features planned for the next WS/dbx release.

- *Reengineered Multiprocess Explorer*
 - Show MPI rank in Multiprocess Explorer
 - Optional viewing of information in Multiprocess Explorer
- *Reengineered Data Explorer*
 - Name and minimal type info list for selection
 - Data navigation capabilities
 - Hoping to add multiprocess/thread data navigation

Features planned for the next WS/dbx release.

- *Improved OpenMP support*
 - Show private data w/o frame changing to main program and other improvements
- *No significant memory growth in cvd/dbx on long sessions and re-runs*
 - New garbage collection algorithm to improve memory usage
 - String table re-engineering

SpeedShop

SGI Performance Analysis Tools

SGI Developer Tools Update

Introduction to SpeedShop and cvperf

- SpeedShop is collection of performance analysis tools
 - ssrun - SpeedShop experiment driver
 - cvperf - GUI performance analysis viewer
 - prof - text based performance analysis viewer
- `ssrun -experiment <executable>`
- `mpirun -np 4 ssrun -mpi <mpi_executable>`
- `cvperf <experiment_file>`

Current release Information

SpeedShop 1.4.3

- Released Jan 8th, 2002
- Is delivered as part of ProDev WorkShop 2.9.1
- Download 1.4.3 from supportfolio
- Current patch 4532
- <http://support.sgi.com/colls/patches/tools/relstream/index.html>

SpeedShop 1.4.3

SpeedShop 1.4.3 introduces two new performance experiments:

- MPI Experiment (“ssrun –mpi”)
- NUMA Experiment (“ssrun –numa”)

Includes numerous bug fixes

New MPI experiment

MPI experiment answers four basic questions:

- Which MPI function was called?
- Who made the call?
- Where was the call made? (to the source line)
- How long did the call take?

New NUMA experiment

NUMA experiment answers these questions:

- How often do I access memory on my own NUMA node?
- Where am I accessing memory from a remote NUMA node?
- Are my placement directives working?

SGI Developer Tools Update

SpeedShop

Features planned for the next release.

SGI Developer Tools Update

SpeedShop features for next release

- *R16k Support*
- *Improved by pthread data collection*
- *Pthread data by thread in cvperf*
- *Multiple starts and stops of data collection*
- *MPI and NUMA experiment improvements*
- *Bug fixes and other improvements*

Questions?

SGI Developer Tools Update

Jim Galarowicz

jeg@sgi.com

Developer Tools Appendix

*SGI Developer Tools Update
Additional Slides
showing features discussed in the
Tools Presentation*

SGI Developer Tools Update

WorkShop ***Appendix Information***

Feature highlights in the current WorkShop/dbx release

- WorkShop coverage tool improvements (cvcov) - reports execution statistics.
 - Improved C++ support (lang:std)
 - Updated the default exclusion file
 - Reduce cvcov memory leaks - purify,SS
 - Improve cvcov speed of execution on large appl.

Feature highlights in the current WorkShop/dbx release

- Coverage tool usage (cvcov)
 - `cvcov runinstr a.out`
 - `cvcov -mktest -cmd "a.out -d"`
 - `cvcov runtest test0000`
 - `cvcov lssource funcname test0000` . List of annotated src
 - `cvcov lssum test0000` . Summary of coverage
 - `cvcov lscall test0000` . Lists function call graph

Feature highlights in the current WorkShop/dbx release

Coverage tool usage (cvcov)

cvcov runinstr hashTest

```
runinstr command: /usr/sbin//cvinstr -coverage  
/usr/WorkShop/usr/lib/WorkShop/Tester/default_instr_file -addlibs  
libss.so:libssrt.so -directory  
/data/clink/a01/gala/pv_bugs/swift_probs/ver##0 "hashTest"  
instrumenting /lib32/rld  
instrumenting /usr/lib32/mips3/libssrt.so  
instrumenting /usr/lib32/mips3/libss.so  
instrumenting /usr/lib32/mips3/libc.so.1  
instrumenting hashTest
```

cvcov: Instrument "hashTest" of version "0" succeeded.

cvcov mktest -cmd hashTest

```
cvcov: Made test directory:  
"/data/clink/a01/gala/pv_bugs/swift_probs/test0000"
```

Feature highlights in the current WorkShop/dbx release

Coverage tool usage (cvcov)

cvcov runttest test0000

cvcov: Running test

"/data/clink/a01/gala/pv_bugs/swift_probs/test0000" ...

/data/clink/a01/gala/pv_bugs/swift_probs//ver###0/hashTest_Instr

found: abc with value: 0

found: def with value: 1

found: alsdjk with value: 2

found: Smith with value: 3

found: june with value: 4

found: smith with value: 3

found: June with value: 4

Key: abc, value: 0

Feature highlights in the current WorkShop/dbx release

Coverage tool usage (cvcov)

cvcov lssum test0000

<i>Coverages Weight</i>	<i>Covered</i>	<i>Total</i>	<i>% Coverage</i>	
-----------------------------	----------------	--------------	-------------------	--

<i>Function</i>	<i>8</i>	<i>39</i>	<i>20.51%</i>	<i>0.400</i>
<i>Source Line</i>	<i>71</i>	<i>446</i>	<i>15.92%</i>	<i>0.200</i>
<i>Branch</i>	<i>10</i>	<i>239</i>	<i>4.18%</i>	<i>0.200</i>
<i>Arc</i>	<i>20</i>	<i>176</i>	<i>11.36%</i>	<i>0.200</i>
<i>Block</i>	<i>75</i>	<i>624</i>	<i>12.02%</i>	<i>0.000</i>
<i>Weighted Sum</i>			<i>14.50%</i>	<i>1.000</i>

Feature highlights in the current WorkShop/dbx release

Coverage tool usage (cvcov)

cvcov lsfun test0000

Functions	Files	Counts

is_ival	islval.c	0
is_rval	isRval.c	0
make_hashtable	serv.c	1
free_hashtable	serv.c	1
insert_in_hash	serv.c	7
find_in_hash	serv.c	7
for_all_in_hash	serv.c	1
hashUC	serv.c	14

Feature highlights in the current WorkShop/dbx release

Coverage tool usage (cvcov)

cvcov lssource for_all_in_hash test0000

Counts Source

```
-----  
void for_all_in_hash(Hash* ht, void (*func)(const char *, void  
*))  
1 {  
    int ja;  
    Link* link;  
257    for (ja = 0; ja < ht->num_buckets; ja++)  
257        for (link = ht->buckets[ja].link; link; link=link->next )  
5            (*func)(link->key, link->adt);  
1 }
```

Feature highlights in the current WorkShop/dbx release

- *Visualizing arrays of pointers/derived types*
 - Bring up Array Visualizer and enter array name.
 - Double click on the entry you are interested in.
 - Structure or derived type is displayed in the structure browser.
 - Example on next slide.

Array Browser (pid 309613)

Admin Format Spreadsheet

Array: p1

Indexing Expression: p1[_\$]

Subscript Controls:

Row Col _\$: 1 Min: 1 Max: 1

p1(1) *7ffedc28*

IS	1	2	3
	*7ffedc28*X	*7ffedc30*X	*7ffedc38*X

Structure Browser (pid 309613)

Admin Config Display Mode Help

Expression:

p1(1) [struct XY]

X	5.8
Y	-9.8

p1(2) [struct XY]

X	4.8
Y	-4.8

WorkShop Debugger (pid 309613)

Admin Views Query Source Display Perf Traps PC Fix+Continue Help

Command: /data/c1link/e81/ga1a/pv_bugs/478103/bug

REL [Control] [Pause] [Step] [Next] [Return] [Search] [Print] [Full] [Run] [Debug]

Status: Process 389613: Stopped on breakpoint
MAIN() ["bug.f98":14, "188812d8"K]

```

1  type xy
2      real x
3      real y
4  end type xy
5
6  type (>xy) p1(10)
7      integer n
8
9  do n=1,10
10     p1(n)%x=(10-n)/n
11     p1(n)%y=(n-10)/n
12 end do
13
14 print *,p1(2)%x
15 end
16
17
18
19
20
21

```

File: 81/ga1a/pv_bugs/478103/bug.f98 (Read Only)

```

cvd> p p1
p1 =
(1) = struct XY {
  x = 5.8
  y = -9.8
}
(2) = struct XY {
  x = 4.8
  y = -4.8
}
(3) = struct XY {
  x = 2.8
  y = -2.8
}
(4) = struct XY {
  x = 1.8
  y = -1.8
}
(5) = struct XY {
  x = 1.8
  y = -1.8
}
(6) = struct XY {
  x = 0.888000000e+08

```

Features planned for the next WS/dbx release.

Data navigation across cvd views

- One mouse action access to data
- Right mouse button down give default action
- Left or Right mouse button hold gives dynamic menu
- Data display window for lightweight data display (reusable or throw away)
- Reusable can access previous data panels
 - Example on next slide



Data View (pid 6079926)

Admin Config Display Help

city
[class city_def]

type	0
streets	1000
sewers	50.0
int_ptr	0x101b8768
huge_city_array	(array)
city_workers	35000.0
lights	"never enough"
officials	0x77eb5eac
city_array	(array)

WorkShop Debugger (pid 6079926)

Admin Views Query Source Display Perf Traps Edit+Continue Help

Command: /data/c/link/sbl/ga1a/pv_bugs/stipek_mlx/mlx_cvdtop

Process 6079926: Stopped
matrix(argc = 1, argv = 0x77feddb4) ["csob.c":326, 0x10002c20]

```

dlib_threadset_t dlib;

big_city.type = 1;
big_city.streets = 1;
big_city.sewers = 0;
for(intf(stderr, "big_city.type=%d\n", big_city.type );

i = 0;
i+=3;

city.streets=1000;
city.sewers=50;
city.city_workers=35000;
city.lights="never enough";
worker.how_many = 1;
city.officials = 5worker;
city.officials->how_many = 3;
city.int_ptr = huge_array;

class city_def () city;
Print ...
Assign ...
Structure View ...
Open with data view...
Expression View ...
Array Browser ...

```

Process 6079926 stopped at ["csob.c":326, 0x10002c20]

Process 6079926 stopped at ["csob.c":323, 0x10002c18]

Process 6079926 stopped at ["csob.c":325, 0x10002c20]

Process 6079926 stopped at ["csob.c":326, 0x10002c20]

cvd)



Data View(pid 6186584)

Admin Config Display Help

*city.officials
[class city_worker]

hou_many

WorkShop Debugger (pid 6186584)

Admin Views Query Source Display Perf Traps Edit+Continue Help

Command: /data/c/ink/s81/ga1s/pv_bugs/stipek_mlx/mlx_cvdtop

Process 6186584: Stopped on breakpoint
matrix(argc = 1, argv = 0x7ff4ddb4) ["csub.c":325, 0x18802c28]

```
314 city.streets=1000;  
315 city.sewers=50;  
316 city.city_workers=35000;  
317 city.lights="never enough";  
318 worker.how_many = 1;  
319 city.officials = &worker;  
320 city.officials->how_many = 3;  
321 city.int_ptr = huge_array;  
322  
323 city_road.type = 1;  
324 simple_city.streets = city.streets;  
325  
326 dlib.type = 1;  
327 fprintf(stderr, "dlib.type = %d\n", dlib.type);  
328  
329 general_int_ptr = &1;  
330 printf("Hello world from matrix.\n");  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000
```

File: tulip22/stipek/demo/mlx/csub.c (Read Only)

```
[6] stop entry matrix(int, char**) file /home/tulip22/stipek/demo/mlx/csub.c  
[6] Process 6186584 stopped at ["csub.c":385, 0x18802b84]  
[7] stop at file /home/tulip22/stipek/demo/mlx/csub.c line 325  
[7] Process 6186584 stopped at ["csub.c":325, 0x18802c28]  
cvd>
```


Features planned for the next WS/dbx release.

Reengineered Multiprocess Explorer

- Show MPI rank in Multiprocess Explorer (was MPView)
- More readable Multiprocess Explorer

Reengineered Data Explorer View

- Name and minimal type information list for selection
- Data navigation capabilities
- Hoping to add multiprocess/thread data navigation
 - Example on next slide

Structure Browser (pid 6833886)

Admin Config Display Node Help

Global Variables

Expression:

- common block
 - aa
 - a
 - a1
 - a2**
 - a3
 - a4
 - a5
 - a6
 - a7

Variable details:

- a**: float, 0.00000000e+00
- a1**: float, 1.0
- a2**: float, 2.0

WorkShop Debugger (pid 6833886)

Admin Views Query Source Display Perf Traps Exit+Continue Help

Command: /data/c1/irk/ab1/dps/pv/788462/dpn3/dpn

Debugging controls: Cont, Stop, Step, Next, Return, Search, Print, KIB, Run, Debug

Status: Process 6833886: Stopped
 dps2() ["dps2.f":4, "10000a24*X"]

```

subroutine dps2
  INCLUDE 'dps_inc.f'
  a2=2.00
  print *,a2
  return
end
  
```

File: /ab1/dps/pv/788462/dpn3/dps2.f (Read Only)

Process 6833886 stopped at ["dps1.f":4, "10000d44*X"]
 Process 6833886 stopped at ["dps1.f":5, "10000a48*X"]
 Process 6833886 stopped at ["dps.f":6, "10000d3c*X"]
 Process 6833886 stopped at ["dps2.f":3, "10000e18*X"]
 Process 6833886 stopped at ["dps2.f":4, "10000e24*X"]
 zvd>

Source View (pid 6833886)

File Display Traps Exit+Continue Help

```

common /aa/a,a1,a2,a3,a4,a5,a6,a7
  
```

File: /dps/pv/788462/dpn3/dps_inc.f (Editable)

SGI Developer Tools Update

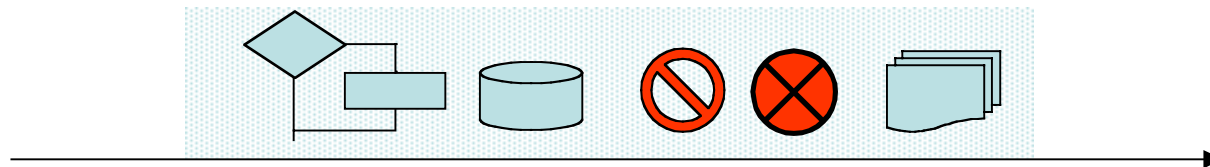
SpeedShop *Appendix Information*

What Is SpeedShop?

- *A collection of tools to determine:*
 - Where is your application's time spent?



- How is your application's time spent?



- What are your application's bottlenecks?

Why Use SpeedShop?

- *It will help you:*
 - Minimize application development time
 - Eliminate bottlenecks and bugs
 - Maximize your application's overall performance
- *How invoke SpeedShop:*
 - `ssrun -<expr-type> <executable>`

What SpeedShop will tell you?

- **SpeedShop reports statistical data**
 - **Function name, source file name, line number, and a statistical data that depends on the experiment you're running**
 - **It includes system functions**
- **Cvperf will let you browse, via a GUI, into your code**
- **Prof will give a text based quick report**

SpeedShop 1.4.3

SpeedShop 1.4.3 introduces two new performance experiments:

- MPI Experiment (“ssrun –mpi”)
- NUMA Experiment (“ssrun –numa”)

Includes numerous bug fixes

New MPI experiment

MPI experiment answers four basic questions:

- Which MPI function was called?
- Who made the call?
- Where was the call made? (to the source line)
- How long did the call take?

MPI Experiment Example

How do I use it?

- NAS CG Parallel Benchmark (MPI)
- Origin 2000 with 16 300Mhz R12000
- `mpirun -np 4 sssrun -mpi cg.A.4`

```
% mpirun -np 4 sssrun -mpi cg.A.4
...
% prof cg.A.4.mpi.f1384250
-----
SpeedShop profile listing generated Wed Jan 23 13:33:52 2002
...
-----
Summary of MPI tracing data (mpi)--
      5044: Total Traced MPI calls
-----
Callee list, in descending order by time taken in MPI call
-----
Seconds   Calls  MPI Function
4.602    1680  MPI_Wait
1.046     1    MPI_Finalize
0.374    1680  MPI_Send
0.204    1680  MPI_Irecv
0.061     1    MPI_Init
0.000     1    MPI_Reduce
0.000     1    MPI_Barrier
```

MPI Example (Continued)

 Call site list, in descending order by time taken in MPI call

Seconds	Calls	MPI Function	Function (dso: file, line)
2.933	400	MPI_Wait	conj_grad (cg.A.4: cg.f, 1177)
1.409	400	MPI_Wait	conj_grad (cg.A.4: cg.f, 1150)
1.046	1	MPI_Finalize	cg (cg.A.4: cg.f, 571)
0.180	400	MPI_Send	conj_grad (cg.A.4: cg.f, 1170)
0.169	400	MPI_Send	conj_grad (cg.A.4: cg.f, 1147)
0.133	400	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1161)
0.099	16	MPI_Wait	conj_grad (cg.A.4: cg.f, 1059)
0.069	16	MPI_Wait	conj_grad (cg.A.4: cg.f, 1361)
0.061	1	MPI_Init	initialize_mpi (cg.A.4: cg.f, 594)
0.048	400	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1139)
0.041	16	MPI_Wait	conj_grad (cg.A.4: cg.f, 1333)
0.037	400	MPI_Wait	conj_grad (cg.A.4: cg.f, 1275)
0.016	400	MPI_Wait	conj_grad (cg.A.4: cg.f, 1221)
0.013	400	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1209)
0.008	16	MPI_Send	conj_grad (cg.A.4: cg.f, 1354)
0.006	16	MPI_Send	conj_grad (cg.A.4: cg.f, 1330)
0.006	400	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1264)
0.005	400	MPI_Send	conj_grad (cg.A.4: cg.f, 1217)
0.005	400	MPI_Send	conj_grad (cg.A.4: cg.f, 1272)
0.002	16	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1345)
0.001	16	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1322)
0.000	16	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1386)
0.000	1	MPI_Reduce	cg (cg.A.4: cg.f, 509)
0.000	16	MPI_Irecv	conj_grad (cg.A.4: cg.f, 1048)
0.000	15	MPI_Wait	cg (cg.A.4: cg.f, 474)
0.000	16	MPI_Send	conj_grad (cg.A.4: cg.f, 1056)
0.000	16	MPI_Send	conj_grad (cg.A.4: cg.f, 1394)
0.000	15	MPI_Irecv	cg (cg.A.4: cg.f, 463)
0.000	15	MPI_Send	cg (cg.A.4: cg.f, 471)
0.000	1	MPI_Barrier	cg (cg.A.4: cg.f, 411)
0.000	16	MPI_Wait	conj_grad (cg.A.4: cg.f, 1397)
0.000	1	MPI_Wait	cg (cg.A.4: cg.f, 376)
0.000	1	MPI_Irecv	cg (cg.A.4: cg.f, 365)
0.000	1	MPI_Send	cg (cg.A.4: cg.f, 373)

New NUMA experiment

NUMA experiment answers these questions:

- How often do I access memory on my own NUMA node?
- Where am I accessing memory from a remote NUMA node?
- Are my placement directives working?

NUMA experiment: How does it work?

NUMA experiment does these items:

- *Statistically samples memory accesses*
- *Looks for a “computable” memory access*
- *Stores “numa info” for each sample*
 - see SpeedShop portion of Appendix for additional information

NUMA Example

- *How do I use it?*
 - NAS CG Parallel Benchmark (OpenMP)
 - Origin 2000 with 16 300Mhz R12000

```
% setenv OMP_NUM_THREADS 4
% srun -numa cg.A
...
% prof -source cg.A.numa.p1350994
-----
SpeedShop profile listing generated Wed Jan 23 13:18:40 2002
...
-----
Summary of NUMA memory profiling data (numa)--
Secondary cache D misses (26): Counter Name (Number)
    100: Counter Average Overflow
   101066: Sampled Memory Accesses
    76615: Remote Memory Accesses
   75.807: Percent Remote Memory Accesses
    1.436: Average ccNUMA Routing Distance
-----
Function list, in descending order by percent remote memory accesses
-----
Sampled  Remote  Pct Rmt  Avg Dist  Function (dso: file, line)
...
 100739   76465   75.904   1.437  conj_grad (cg.A: cg.c, 374)
...
```

NUMA Example

Disassembly listing, annotated with NUMA memory profiling data

```
...
conj_grad (cg.A: cg.c, 374):
...
429: /* rolled version */
430: #pragma omp for private(sum,k)
431:     for (j = 1; j <= lastrow-firstrow+1; j++) {
432:         sum = 0.0;
433:         for (k = rowstr[j]; k < rowstr[j+1]; k++) {
434:             sum = sum + a[k]*p[colidx[k]];
435:         }
436:         w[j] = sum;
437:     }
438:
...
[ 434] 0x100057fc          0x8c440000  lw          a0,0(v0)
      ^----- 220 Sampled, 75.909% Remote, Avg Dist = 1.459 -----^
[ 434] 0x10005800          0x000420c0  sll         a0,a0,3
[ 434] 0x10005804          0xd42d0000  ldc1        $f13,0(at)
      ^----- 85 Sampled, 68.235% Remote, Avg Dist = 1.306 -----^
[ 434] 0x10005808          0x02042021  addu        a0,s0,a0
[ 434] 0x1000580c          0xd48e0000  ldc1        $f14,0(a0)
      ^----- 106 Sampled, 83.019% Remote, Avg Dist = 1.500 -----^
[ 434] 0x10005810          0x24210008  addiu       at,at,8
[ 434] 0x10005814          0x24420004  addiu       v0,v0,4
[ 434] 0x10005818          0x4cee69e1  madd.d      $f7,$f7,$f13,$f14
[ 434] 0x1000581c          0x12600046  beq         s3,zero,0x10005938
[ 434] 0x10005820          0000000000  nop
[ 434] 0x10005824          0x16800698  bne         s4,zero,0x10007288
[ 434] 0x10005828          0000000000  nop
[ 434] 0x1000582c          0x8c440000  lw          a0,0(v0)
      ^----- 351 Sampled, 74.074% Remote, Avg Dist = 1.416 -----^
[ 434] 0x10005830          0xd4220008  ldc1        $f2,8(at)
      ^----- 582 Sampled, 77.491% Remote, Avg Dist = 1.474 -----^
[ 434] 0x10005834          0xd4230000  ldc1        $f3,0(at)
[ 434] 0x10005838          0x000420c0  sll         a0,a0,3
[ 434] 0x1000583c          0x8c430004  lw          v1,4(v0)
      ^----- 57 Sampled, 56.140% Remote, Avg Dist = 1.105 -----^
...

```

NUMA Experiment Notes & Caveats

- Only supported on Origin 2000/3000
- No GUI available (CVPERF not supported), only prof
- Storage requirements can be excessive (32 bytes per sample)
- Do other optimizations first!
 - First Order: algorithm selection
 - Second Order: algorithm implementation details
 - Third Order: NUMA placement