Evaluation of the sPPM Benchmark on the Cray X1

Sarah Anderson (Cray)

Scott Parker (NCSA)

Dave Strenski (Cray)

Objectives

- Introduction and History to sPPM
- Programming models available on the Cray X1
- Programming models applied to sPPM
- Performance of sPPM on Cray X1 and Itanium2 Linux cluster

sPPM Numerics and History

- Simplified version of Piecewise Parabolic Method (PPM) by Woodward and Colella at Lawrence Livermore National Laboratory.
- Kernel updates a one dimensional strip of zones, resulting in high data locality.
- OpenMP can be used to update independent block or “pencils” of a one dimensional strip.
- MPI is used on a rectangular domain decomposition communicating across the boundaries.

Programming Models

```
 Example 1 (MSP/OMP)
P----< do k=1, num_k
PM-----< do j=1, num_j
PMV------< do i=1, num_i
PMV         ... work ...
PMV------> end do
PM-----> end do
P----> end do
```

```
 Example 2 (MSP/OMP)
P----< do k=1, num_k
P2-----< do j=1, num_j
P2MV-----< do i=1, num_i
P2MV        ... work ...
P2MV-----> end do
P2-----> end do
P----> end do
```

```
 Example 3 (SSP/OMP)
P----< do k=1, num_k
P2-----< do j=1, num_j
P2V------< do i=1, num_i
P2V         ... work ...
P2V------> end do
P2-----> end do
P----> end do
```

```
 Example 4 (MSP)
M----< do k=1, num_k
M2-----< do j=1, num_j
M2V------< do i=1, num_i
M2V         ... work ...
M2V------> end do
M2-----> end do
M----> end do
```

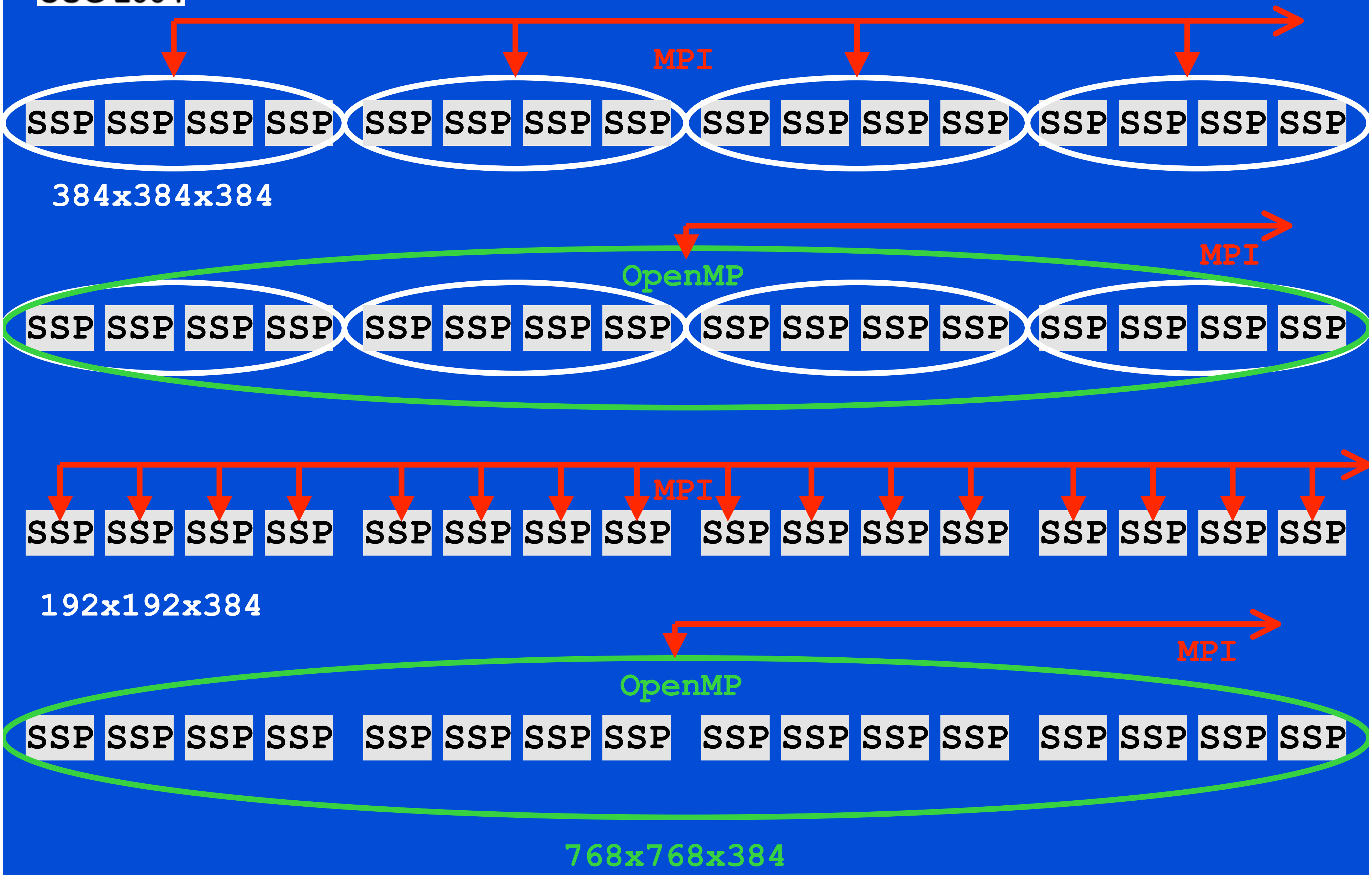
```
do n=1,num_domains             MPI domain Layer
  do mypen=1,nypens * nzpens   OpenMP pencil Layer
    do k= z_start, z_stop    \
      do j = y_start, y_stop  \
        do i = x_start, x_stop \ Extract pencil
        end do                 / of y/z data
      end do                  /
    end do                   /
    do izy=1,(y_stop-y_start)*(z_stop-z_start)
      do i = 2-nbdy, n+nbdy-1      \
      end do                        \
        :                            \
        : Several single nested loops } sppm routine
        :                            /
      do i = 2-nbdy, n+nbdy-1       /
      end do                       /
    end do End strips within a pencil
    do k= z_start, z_stop    \
      do j = y_start, y_stop  \
        do i = x_start, x_stop \
        end do                 / Store pencil data
      end do                  /
    end do                   /
  end do                     End OpenMP pencil layer
end do                       End MPI domain layer
```

Programming Model Applied to sPPM

```
!dir$ ssp_private sppm
!csd$ parallel do
       do 1200 izy= 0, (izstop-izstart+1)*(iystop-iystart+1)-1
         iz= izy/(iystop-iystart+1) + izstart
         iy= mod( izy, (iystop-iystart)+1 ) + iystart
         call sppm( xl, rrho, pp, uux, uuy, uuz,
      &              iy+noffy, iz+noffz,
      &              rhonu(1-5,iy+noffy,iz+noffz),
      &              pnu (1-5,iy+noffy,iz+noffz),
      &              uxnu (1-5,iy+noffy,iz+noffz),
      &              uynu (1-5,iy+noffy,iz+noffz),
      &              uznu (1-5,iy+noffy,iz+noffz),
      &              dtime, courant, gamma,
      &              smlrho, smalle, smallp, smallu, nx, 5 )
  1200 continue
!csd$ end parallel do
```

Problem Size

Nodes	MSP	Model	Thds	MPI			Sub-domain Size			Total Size		
				X	Y	Z	X	Y	Z	X	Y	Z
1	4	MSP_MPI	1	2	2	1	384	384	384	768	768	384
2	8	MSP_MPI	1	2	2	2	384	384	384	768	768	768
4	16	MSP_MPI	1	2	2	4	384	384	384	768	768	1536
8	32	MSP_MPI	1	2	2	8	384	384	384	768	768	3072
16	64	MSP_MPI	1	2	2	16	384	384	384	768	768	6144
1	4	MSP_OMP	4	1	1	1	768	768	384	768	768	384
2	8	MSP_OMP	4	1	1	2	768	768	384	768	768	768
4	16	MSP_OMP	4	1	1	4	768	768	384	768	768	1536
8	32	MSP_OMP	4	1	1	8	768	768	384	768	768	3072
16	64	MSP_OMP	4	1	1	16	768	768	384	768	768	6144
1	4	SSP_MPI	1	4	4	1	192	192	384	768	768	384
2	8	SSP_MPI	1	4	4	2	192	192	384	768	768	768
4	16	SSP_MPI	1	4	4	4	192	192	384	768	768	1536
8	32	SSP_MPI	1	4	4	8	192	192	384	768	768	3072
16	64	SSP_MPI	1	4	4	16	192	192	384	768	768	6144
1	4	SSP_OMP	16	1	1	1	768	768	384	768	768	384
2	8	SSP_OMP	16	1	1	2	768	768	384	768	768	768
4	16	SSP_OMP	16	1	1	4	768	768	384	768	768	1536
8	32	SSP_OMP	16	1	1	8	768	768	384	768	768	3072
16	64	SSP_OMP	16	1	1	16	768	768	384	768	768	6144



MPI

SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP

384x384x384

MPI

OpenMP

SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP

MPI

SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP

192x192x384

MPI

OpenMP

SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP SSP

768x768x384

Results

Cray X1 GFLOPS (%peak)

Nodes	MSPs	MSP_CSD_MPI	MSP_MPI	MSP_OMP	SSP_MPI	SSP_OMP
1	4	14.61(29%)	9.52(19%)	11.06(22%)	10.45(20%)	13.52(26%)
2	8	29.57(29%)	21.49(21%)	22.20(22%)	20.88(20%)	26.16(26%)
4	16	57.24(29%)	42.93(21%)	44.28(22%)	41.72(20%)	53.45(26%)
8	32	118.11(29%)	85.93(21%)	88.62(22%)	83.38(20%)	104.39(25%)
16	64	237.08(29%)	172.39(21%)	177.56(22%)	225.79(27%)	208.99(26%)
31	124	459.61(29%)	334.77(21%)	344.00(22%)	322.97(20%)	402.19(25%)

NCSA Itanium2 Linux Cluster

Nodes	CPUs	Thd	MPI			Sub-domain			Total Size			GFlops(%Peak)
			X	Y	Z	X	Y	Z	X	Y	Z	
1	1	1	1	1	1	384	384	384	384	384	384	0.927(18%)
2	2	1	2	1	1	384	384	384	768	384	384	1.843(18%)
4	4	1	2	2	1	384	384	384	768	768	384	3.603(17%)
8	8	1	2	2	2	384	384	384	768	768	768	7.326(18%)
16	16	1	4	2	2	384	384	384	1536	768	768	14.541(17%)
32	32	1	4	4	2	384	384	384	1536	1536	768	29.113(17%)

Single MSP 64-bit: 3.7 Gflops(29%)
Single MSP 32-bit: 5.7 Gflops(22%)
120 MSPs 32-bit: 663.0 Gflops(22%) 512 x 500 x 250 problem size
252 MSPs 32-bit: 1396.6 Gflops(22%) 1536 x 1536 x 1792 problem size

Summary

- Try to convert OpenMP directives to CSD.
- Make sure the vectors have enough work.
- Try using OpenMP across 16 SSPs.
- Cray X1 outperforms the Itanium2 Cluster, because it has a higher peak per processor and a higher percentage of peak.