The pC++ System Programming Environment

Part III:

Performance Measurements

Allen Malony Bernd Mohr

Dept. of Computer and Information Science
University of Oregon
Eugene, Oregon 97403

{malony,mohr}@cs.uoregon.edu

pC++ Benchmarks

- **Grid**: Block Grid Conjugate Gradient

  solves the Poisson equation on a 2-D grid using finite difference operators and a simple conjugate gradient method without preconditioning (PDE algorithm)

- **Poisson**: Fast Poisson Solver

  uses FFT's and cyclic reduction along the rows and columns of a 2-D array to solve PDE problems (typical of CFD algorithms)

- **Embar**: NAS Embarrassingly Parallel Benchmark

  generates 2^{24} complex pairs of uniform (0,1) random numbers and gathers statistics

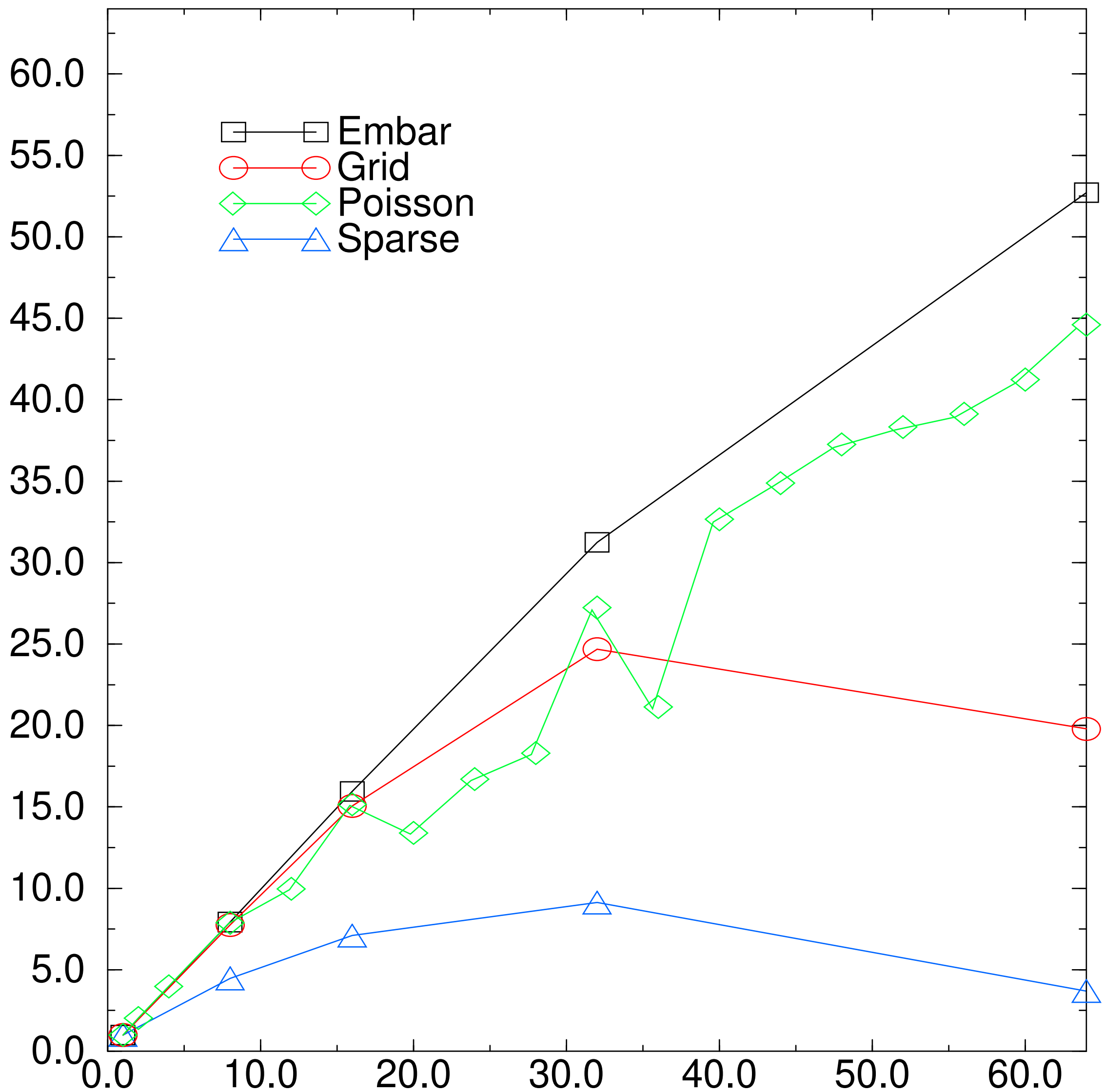
- **Sparse**: NAS Sparse Conjugate Gradient Benchmark

  computes repeated solution to $A \times X = F$ where A is a random sparse matrix

KSR-1 Results

Grid	1	8	16	32	64
mflops	0.88	6.82	13.24	21.70	17.40
seconds	45.91	5.92	3.05	1.86	2.32
speedup	1.00	7.76	15.05	24.68	19.79
Embar	1	8	16	32	64
ymp	0.02	0.12	0.25	0.49	0.83
seconds	739.56	93.26	46.37	23.67	14.03
speedup	1.00	7.93	15.95	31.24	52.71
Poisson	1	8	16	32	64
seconds	39.23	4.98	2.59	1.44	0.88
speedup	1.00	7.88	15.15	27.23	44.60
Sparse	1	8	16	32	64
ymp	0.01	0.03	0.05	0.07	0.03
mflops	0.45	2.03	3.23	4.15	1.68
seconds	146.02	32.66	20.56	16.00	39.48
speedup	1.00	4.47	7.10	9.13	3.70

KSR-1 Speedup



- ❑ remote access intensive applications pay penalty for crossing cluster (= 32 nodes) boundaries

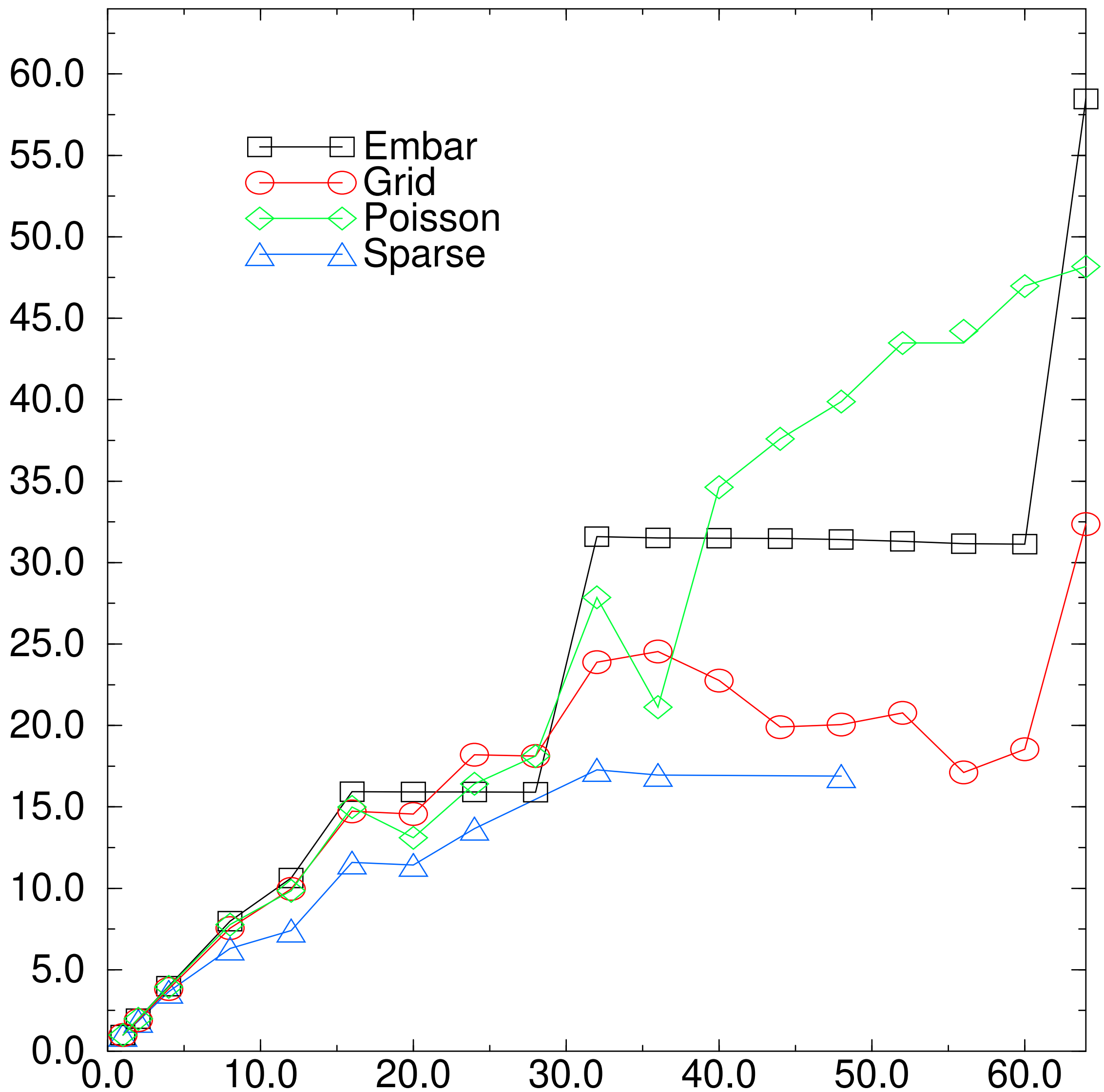
  ➠ Grid, Sparse!

- ❑ current KSR programming model makes it hard to exploit the memory hierarchy

BBN TC2000 Results

Grid	1	8	16	32	64
mflops	0.46	3.47	6.76	10.97	14.87
seconds	87.89	11.64	5.97	3.68	2.72
speedup	1.00	7.55	14.73	23.89	32.36
Embar	1	8	16	32	64
ymp	0.01	0.05	0.10	0.19	0.36
seconds	1898.43	237.91	119.16	60.10	32.47
speedup	1.00	7.98	15.93	31.59	58.47
Poisson	1	8	16	32	64
fft	48.69	6.17	3.13	1.62	0.86
cyclic	12.44	1.69	0.94	0.57	0.41
seconds	61.74	7.94	4.12	2.22	1.28
speedup	1.00	7.77	14.99	27.86	48.17
Sparse	1	8	16	32	64
ymp	< 0.01	0.03	0.06	0.08	---
mflops	0.30	1.89	3.47	5.08	---
seconds	221.46	35.15	19.11	13.06	---
speedup	1.00	6.30	11.59	16.96	---

BBN TC2000 Speedup

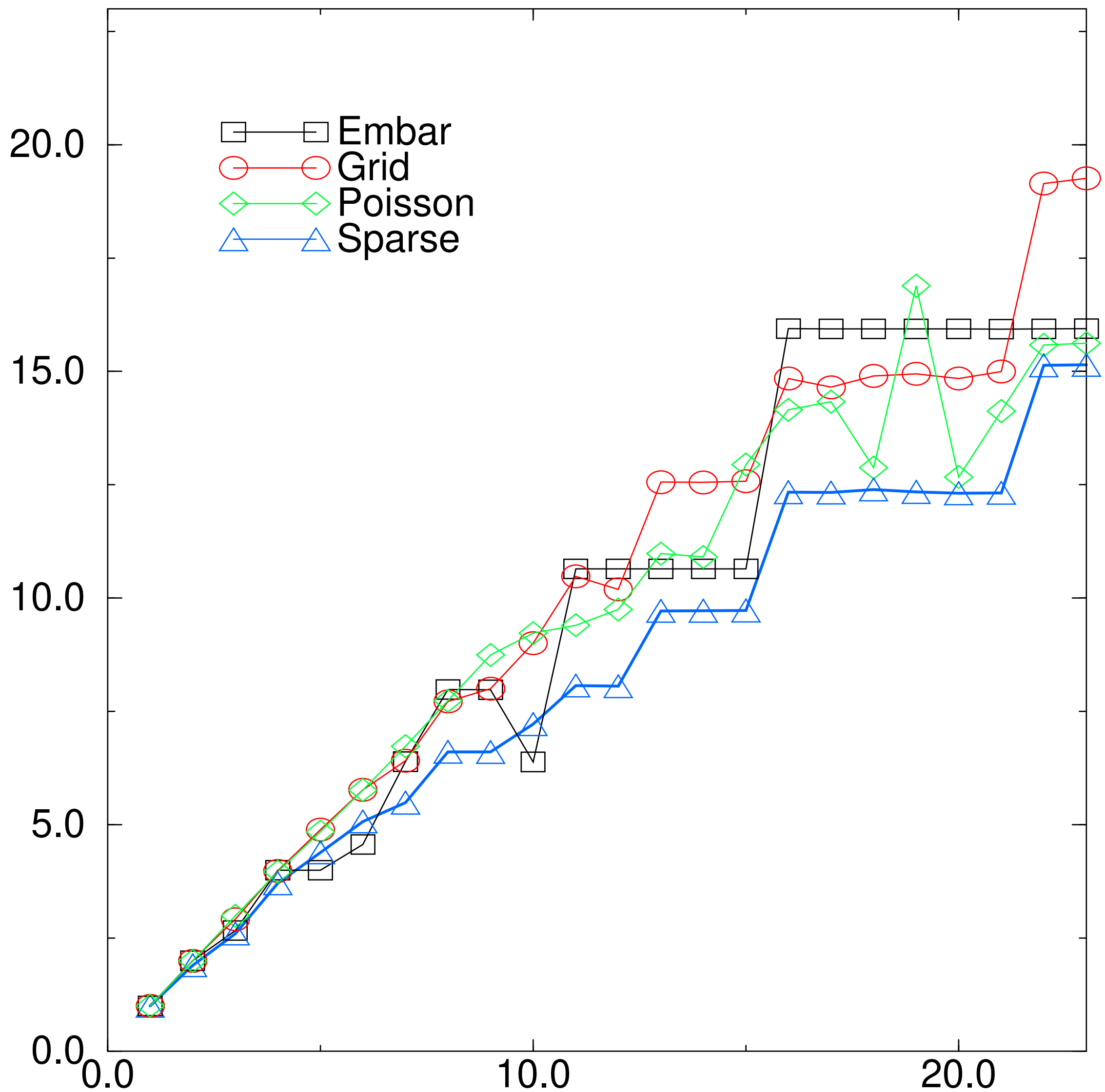


- ❑ remote access intensive applications perform a little better than on the KSR-1, but still show problems over 32 nodes
- ❑ “staircase” speedup curve for Embar and Grid due to mismatch of problem size and processor number
- ❑ general behavior the same for all shared memory machines (e.g., negative peak for Poisson at 36 nodes)

Sequent Symmetry Results

Grid	1	2	4	8	12	16
mflops	0.10	0.19	0.38	0.75	0.99	1.44
seconds	417.46	209.40	104.89	54.10	40.96	28.13
speedup	1.00	1.99	3.98	7.72	10.19	14.84
Embar	1	2	4	8	12	16
ymp	< 0.01	< 0.01	< 0.01	0.01	0.01	0.01
seconds	14507.50	7260.73	3633.16	1817.95	1363.09	909.81
speedup	1.00	2.00	3.99	7.98	10.64	15.95
Poisson	1	2	4	8	12	16
fft	198.82	99.73	50.01	25.22	19.80	12.83
cyclic	51.32	25.90	13.17	7.11	5.85	4.83
seconds	252.32	126.71	63.73	32.63	25.89	17.83
speedup	1.00	1.99	3.96	7.73	9.75	14.15
Sparse	1	2	4	8	12	16
ymp	< 0.01	< 0.01	< 0.01	0.01	0.01	0.01
mflops	0.06	0.11	0.21	0.37	0.46	0.70
seconds	1168.20	616.73	315.05	176.93	145.04	94.72
speedup	1.00	1.89	3.71	6.60	8.05	12.33

Sequent Symmetry Speedup

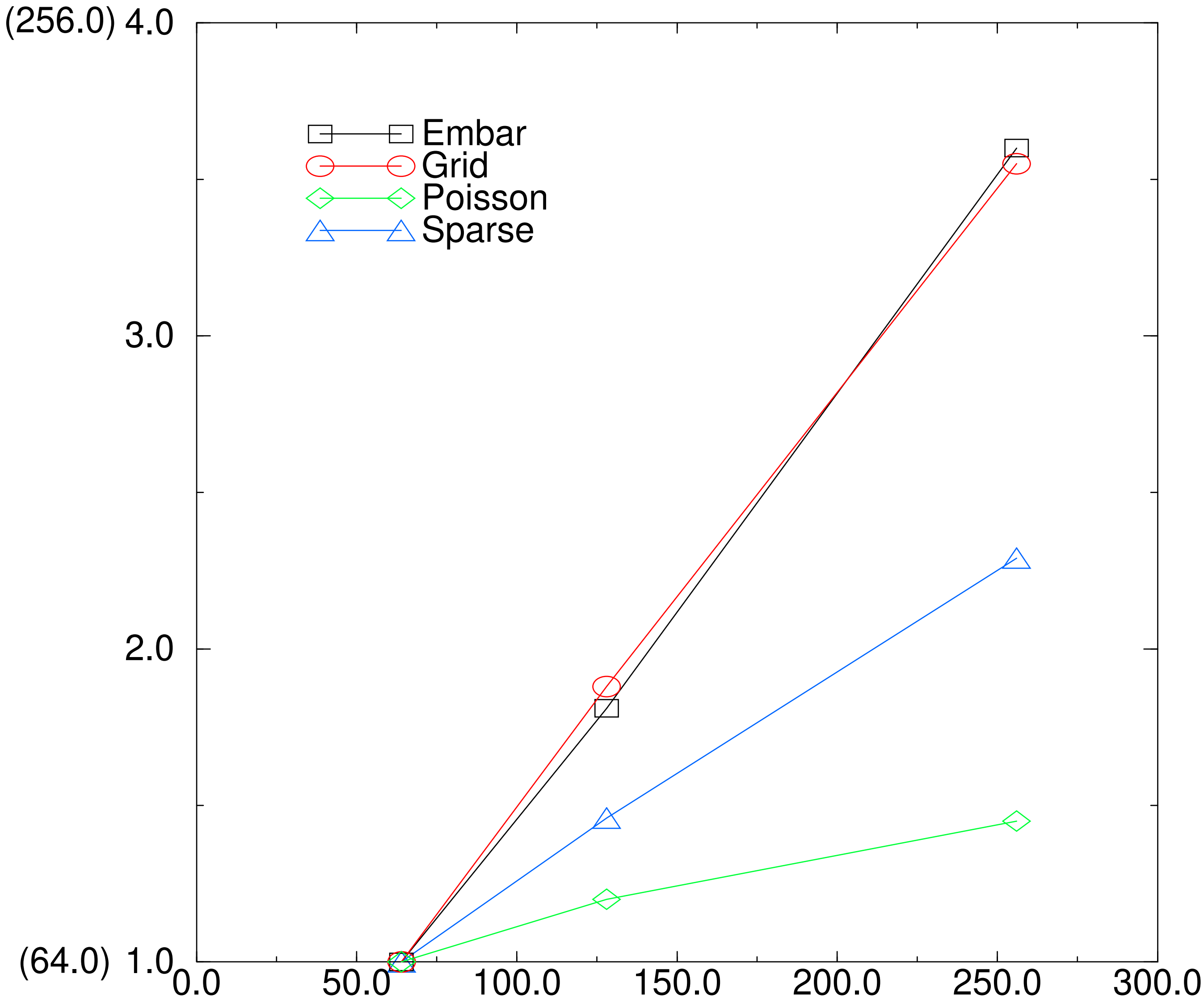


- ❑ uniform memory system hides poor locality effects
- ❑ machine size to small to make serious comparison to other shared memory machines

TMC CM-5 Results

Grid (8,8) bs=64	64	128	256
mflops	67.22	---	---
seconds	0.60	---	---
speedup	1.00	---	---
Grid (16,16) bs=128	64	128	256
mflops	69.82	131.04	248.4
seconds	2.31	1.23	0.65
speedup	1.00	1.88	3.55
Embar	64	128	256
seconds	11.12	6.16	3.09
ymp	1.04	1.87	3.74
speedup	1.00	1.81	3.60
Poisson (2K, 2K)	64	128	256
seconds	13.8	11.5	9.52
fft	6.51	3.60	1.67
cyclic	7.16	7.84	7.68
speedup	1.00	1.20	1,45
Sparse, n = 14000	64	128	256
seconds	50.9	34.9	22.19
ymp	0.427	0.623	0.98
mflops	29.4	42.9	67.56
speedup	1.00	1.46	2.29

TMC CM5 Speedup

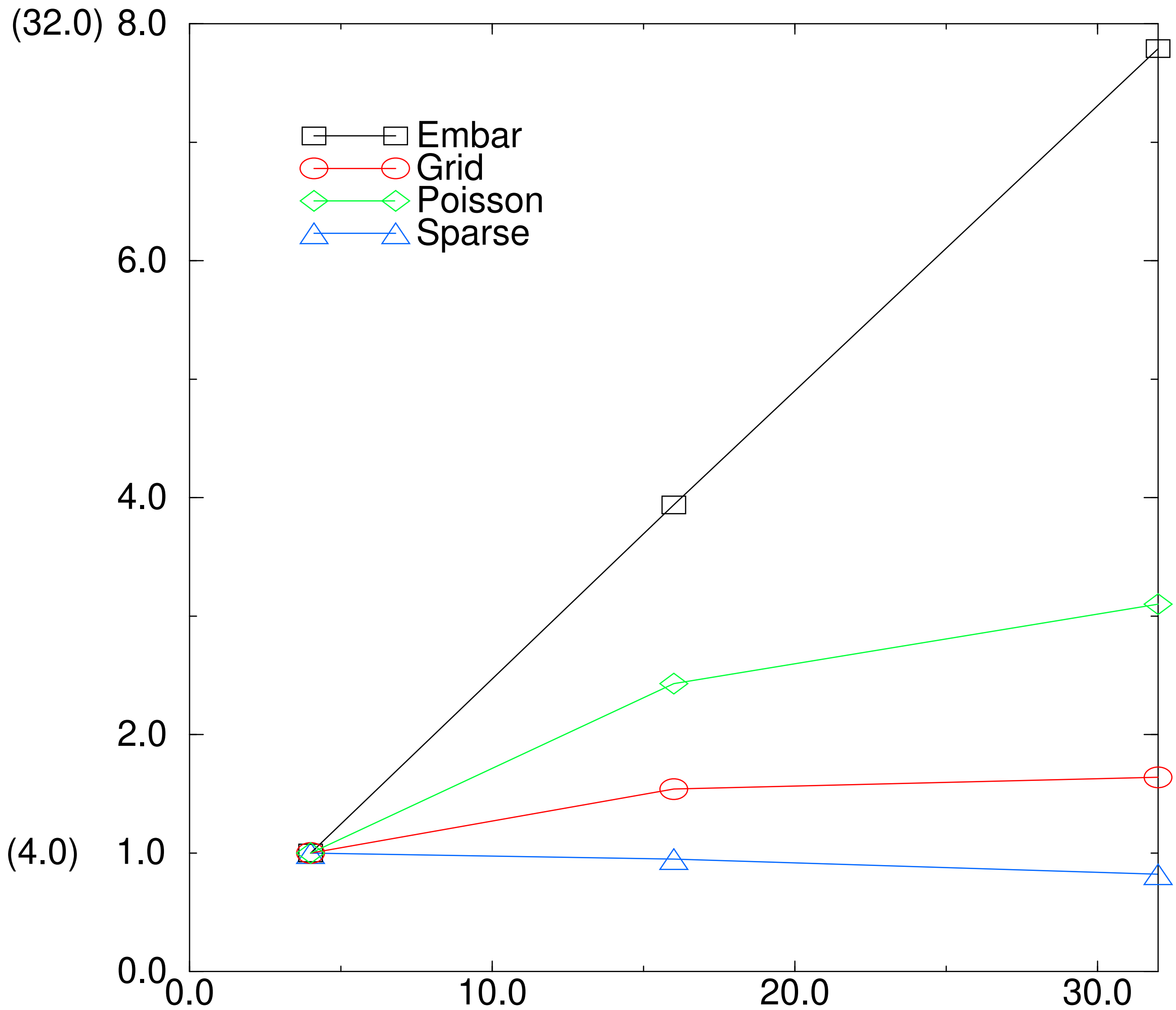


- figure shows speedup compared to 64 node versions!
- low communication latency (active messages) + fast HW barrier
- problem size had to be increased compared to shared memory machines due to size of the machine
  - ➠ Grid and Sparse now show reasonable speedup
- execution time of Embar within 10% of published hand optimized Fortran version!

Intel Paragon Results

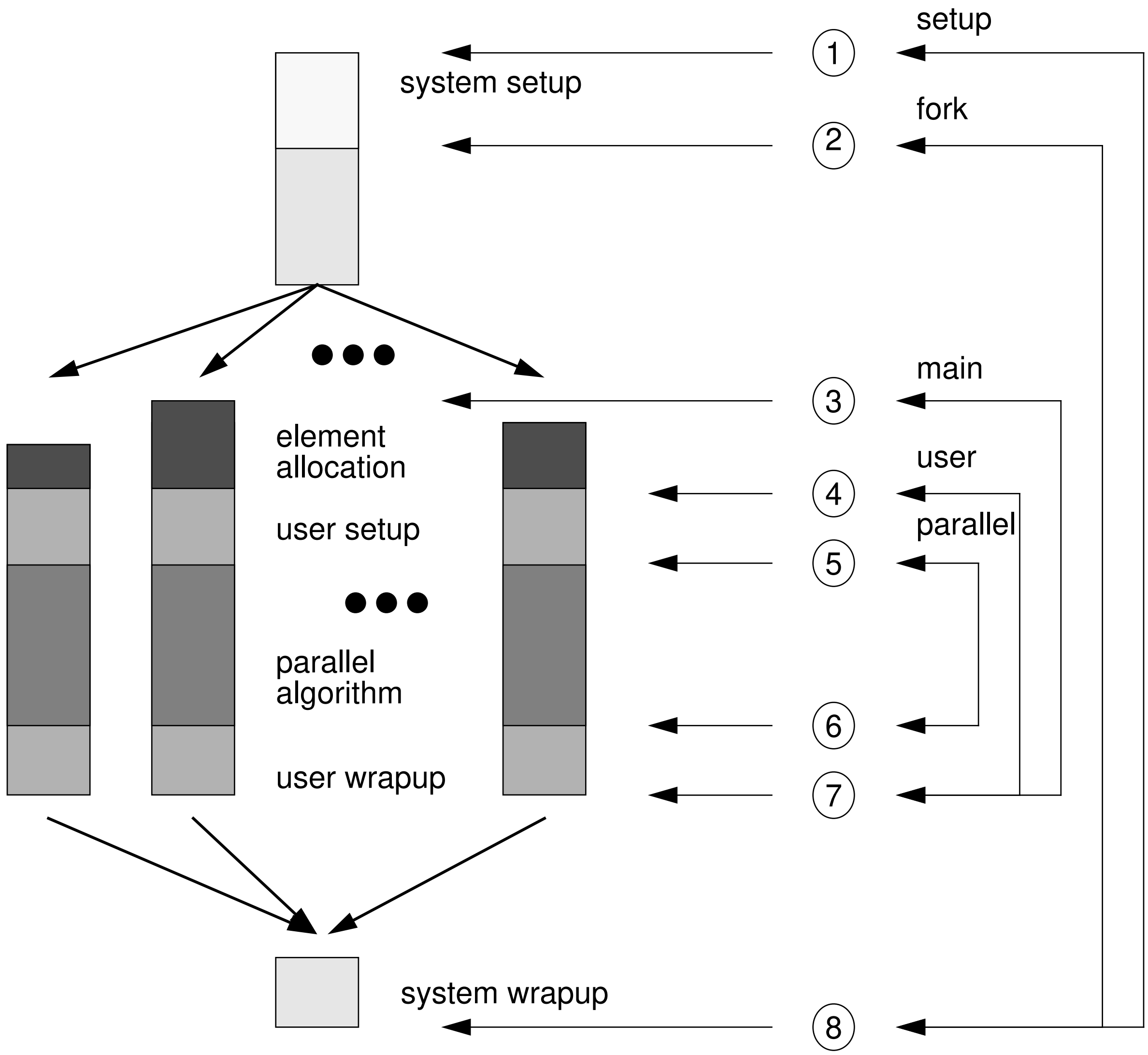
Grid (8,8) bs = 64	4	16	32
mflops	5.55	8.53	9.51
seconds	7.28	4.73	4.43
speedup	1.00	1.54	1.64
Grid (8,8) bs = 128	4	16	32
flops	---	5.55	16.19
seconds	---	9.98	7.41
Embar	4	16	32
seconds	125.42	31.84	16.1
ymp	0.092	0.36	0.71
speedup	1.00	3.94	7.79
speedup w/r F77	2.51	9.9	19.6
Poisson (512, 512)	4	16	32
seconds	4.18	1.72	1.35
fft	3.70	0.76	0.44
cyclic	1.45	0.97	0.89
speedup	1.00	2.43	3.10
Sparse, n = 1400	4	16	32
seconds	52.29	54.82	63.58
ymp	0.02	0.02	0.02
speedup	1.00	0.95	0.82

Intel Paragon Speedup



- speedup based on 4 processor run!
- communication intensive applications (Grid, Sparse) show bad performance due to high message latency, necessity of using a software polling barrier, but only small benchmarks used
- execution time of Embar is 71% of published hand optimized Fortran version!
- SUNMOS version of Sparse up to 7 times better!

pC++ Benchmarks
General Execution Phases



➠ $T_{Total} = T_{Setup} + T_{Fork}(P)$

$+$ foreach C in Coll { $T_{Alloc}(size_C / P)$ }

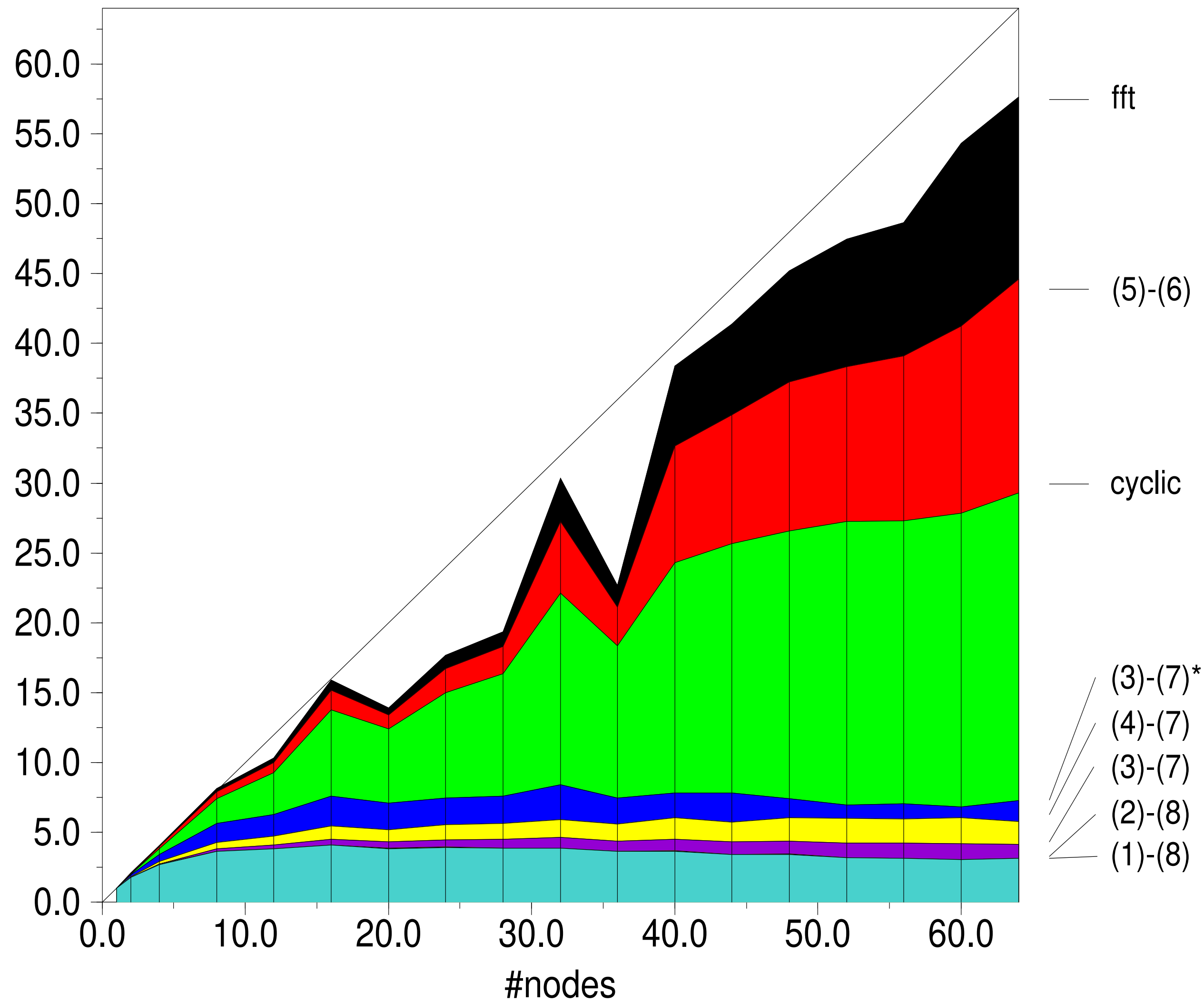
$+ \; T_{UserSetup}(size, P)$

$+ \; T_{ParallelAlg}(size, P)$

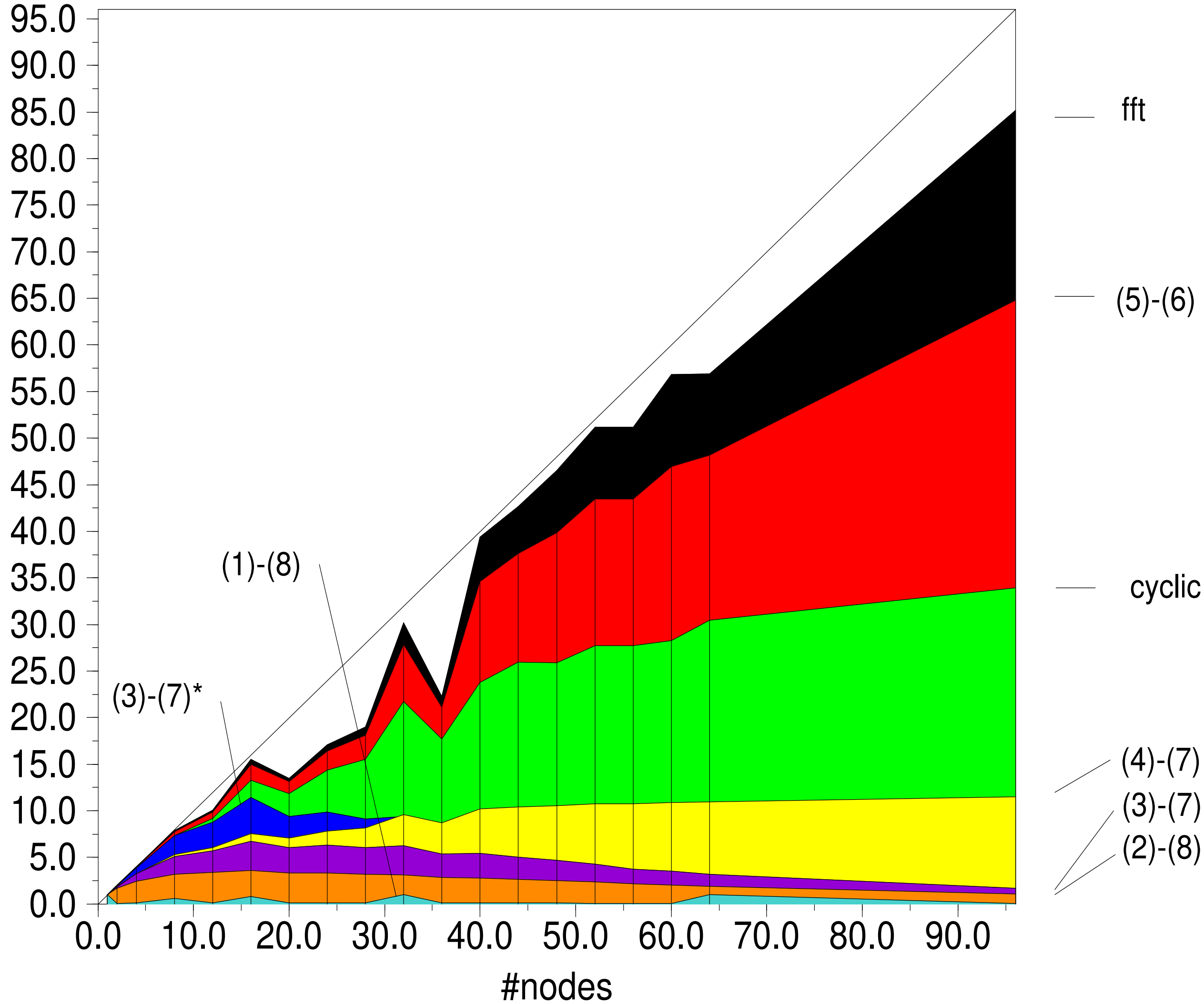
$+ \; T_{UserWrapup}(size, P)$

$+ \; T_{SysWrapup}(P)$

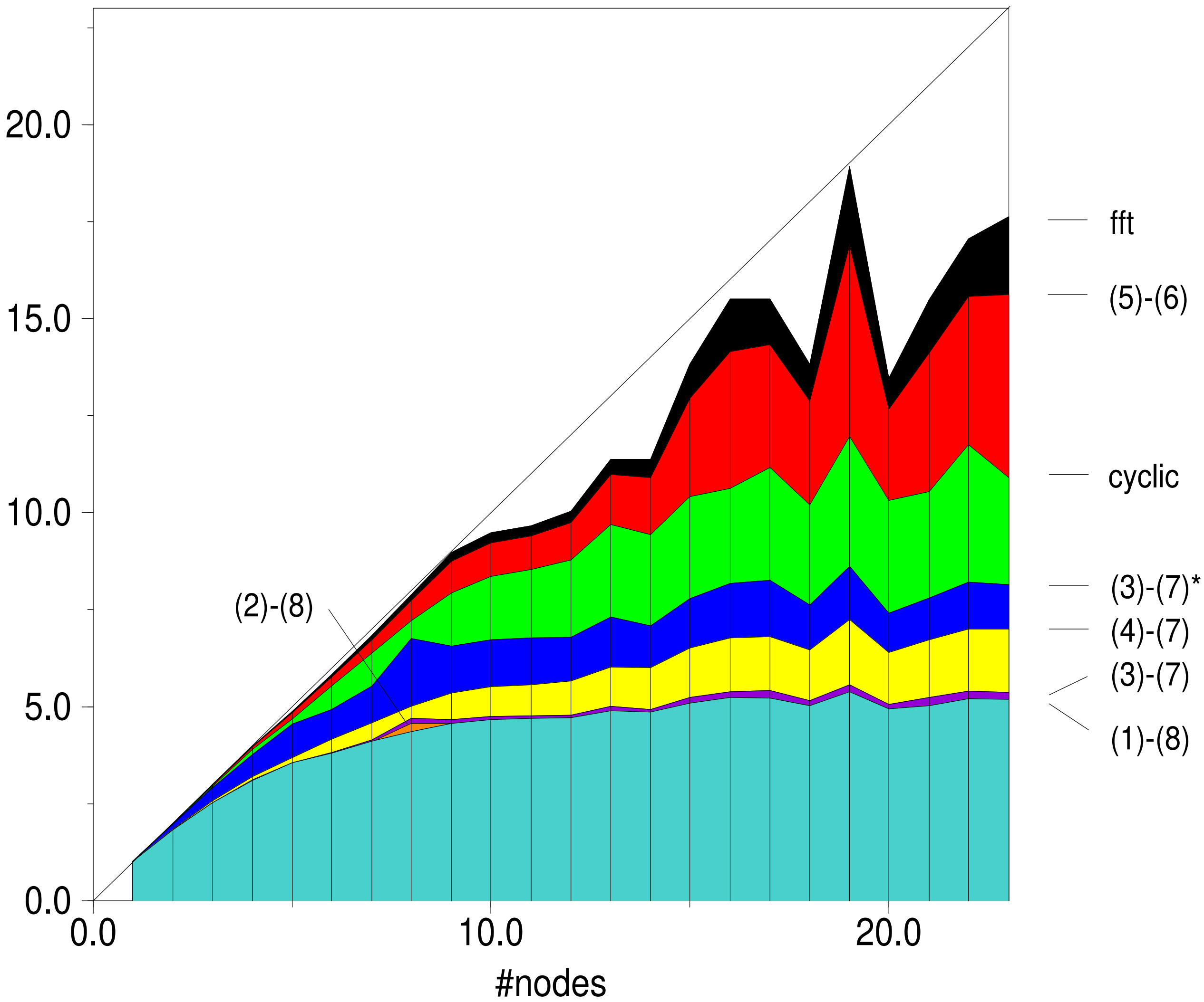
KSR-1
Poisson Speedup Profile



BBN TC2000
Poisson Speedup Profile



Sequent Symmetry
Poisson Speedup Profile



Future Performance Analysis Plans

- redo measurements for new pC++ version and
  get results for new ports (SP-1, ...)

- get detailed speedup curves for the distributed memory machines

- make scalability experiments with changing problem sizes
  - simulated performance extrapolation
  - empirical performance analysis

- compare profiling data for
  - same machine / changing node numbers
  - between machines

- design “synthetic” pC++ runtime system benchmark
  and compare different machines

- measure differences between pC++ “native” runtime system and
  - Nexus (CalTech, ANL)
  - Chant (NASA ICASE)

  based runtime system