IBM Research

2009 IBM HPC Challenge Class II Submission

George Almási
Ganesh Bikshandi
Călin Caşcaval
David Cunningham
Gábor Dózsa
Montse Farreras
David Grove
Sreedhar Kodali
Nathaniel Nystrom
Igor Peshansky
Vijay Saraswat
Sayantan Sur
Olivier Tardieu
Ettore Tiotto

© 2009 IBM Corporation

Our submission at a glance

- **Two programming languages**
  - X10
  - UPC
- **Three platforms**
  - Power 5+ cluster (Poughkeepsie Benchmark Center)
  - Blue Gene/P (instead of Blue Gene/L)
  - BSC MareNostrum
- **One common distributed runtime**

HPC programming models research at IBM

http://www.alphaworks.ibm.com/tech/upccompiler

http://x10-lang.org

- **xlUPC**
  - UPC moving towards standardization
  - PERCS, BW deliverable
  - Power architectures
- **xlCAF**
  - CAF in Fortran2008 standard
  - Prioritized subsets in future Fortran releases
- **X10**
  - Open source
    - Eclipse Public License
  - X10 2.0 released November 6, 2009
    - Java or C++ back-end
    - Runs on almost any architecture

Common runtime support for all three efforts

HPCC Benchmarks

- **X10:**
  - Benchmarks rewritten for X10 2.0
    - LU, FT: new scalable version
      - Use APGAS collectives
        - > Broadcast, Reduction, Alltoall
    - RA, Stream:
      - Reduced overheads

HPCC submission completed even though it overlapped with PERCS milestone

- **UPC:**
  - Benchmarks *almost* unchanged from 2008
    - FFT
      - Local scatter + alltoall instead of “memput”
    - HPL
      - Reduced loop overhead (compiler opt);
      - Better optimized collectives

1 PW to compile, run, organize

Performance results: Power5+ cluster

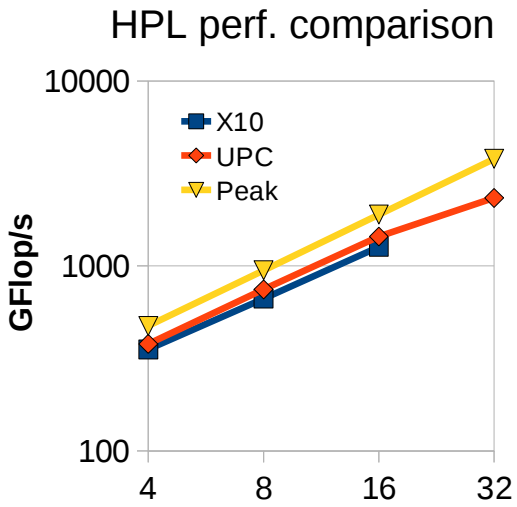
X10	LU	RA	Stream	FFT
nodes	GFlop/s	MUP/s	GBytes/s	GFlops/s
4	354	6.34	325.7	23.67
8	666	12.31	650.5	40.62
16	1268	23.02	1287.8	65.92
32		43.1	2601.5	

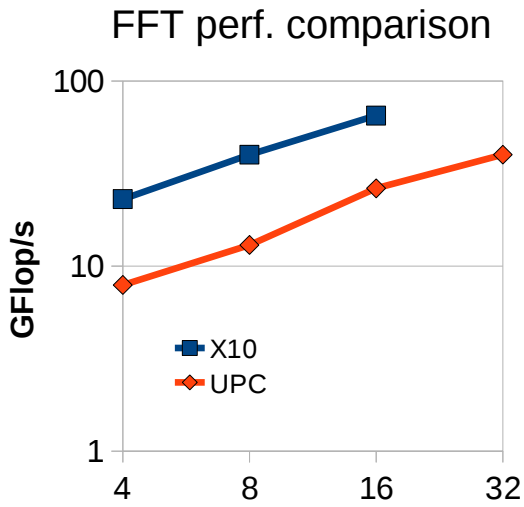
UPC	LU	RA	Stream	FFT
nodes	GFlop/s	MUP/s	GBytes/s	GFlops/s
4	379	5.5	140	7.9
8	747	10.8	256	13
16	1442	21.5	523	26.3
32	2333	43.3	1224	39.8

IBM Poughkeepsie Benchmark Center

32 Power5+ nodes
16 SMT 2x processors/node
64 GB/node; 1.9 GHz

HPS switch, 2 GBytes/s/link





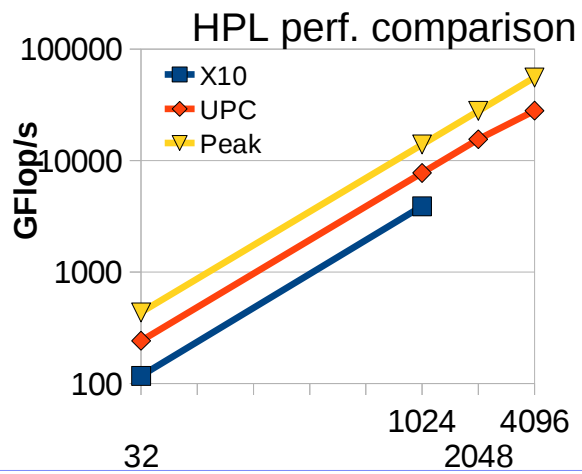
Performance results – Blue Gene/P

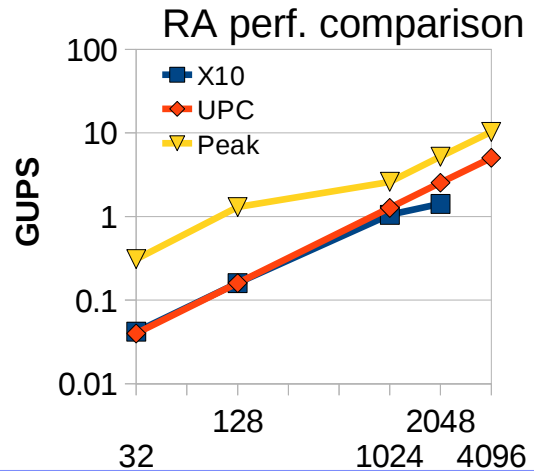
X10	LU	RA	Stream	FFT
nodes	GFlop/s	GUP/s	GBytes/s	GFlops/s
32	186	0.042	141	9.6
128	713	0.16	564	13.9
1024	5874	1.05	4516	
2048		1.42	9032	

UPC	LU	RA	Stream	FFT
nodes	GFlop/s	GUP/s	GBytes/s	GFlops/s
32	242	0.04	168	8.28
128	967	0.16	672	28
1024	7744	1.27	5376	246
2048	15538	2.54		492
4096	28062	5.04		519

IBM TJ Watson Res. Ctr.
WatsonShaheen

4 racks Blue Gene/P
1024 nodes/rack
4 CPUs/node; 850 MHz
4 Gbytes/node RAM
16 x 16 x 16 torus

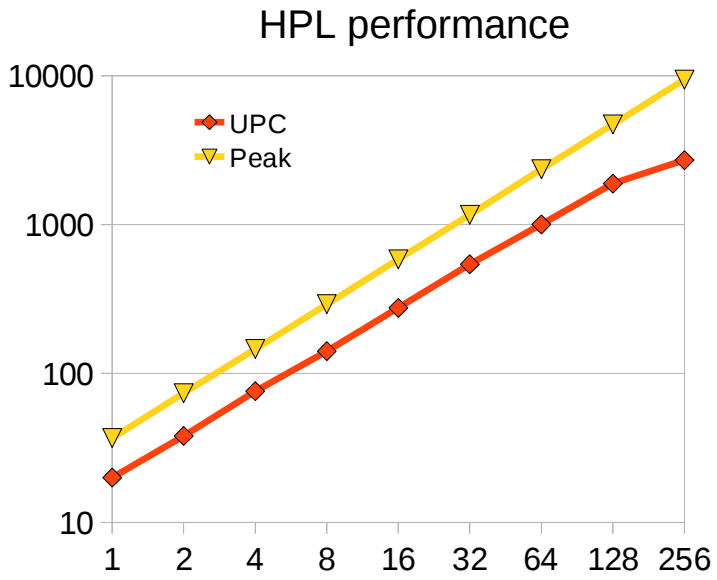




Performance results – MareNostrum

UPC	LU	Stream	FFT
nodes	GFlop/s	GBytes/s	GFlops/s
2x2	20	12.3	0.37
4x2	38		0.7
8x2	76		1.4
16x2	141		2.73
32x2	276		5
64x2	541	391.05	11
128x2	1003		20
256x2	1885	1558	40
256x4	2709		53

GUPS runs prevented by network failures



BSC MareNostrum

CPUs: 2x2x2560 PPC 970MP
2.3 GHz, 8 GB/node
Network: Myrinet 2Gb/s crossbars
10 cabinets 256x256
2 “spines” 1280x1280

Discussion

Platforms

- Power5+:
  - Forgiving machine
  - Performance, scalability is easy
  - Firmware limits RA performance
- BG/P:
  - Hard memory limitations
  - X10 scaling achievable
- MareNostrum:
  - Network issues

Benchmarks

- LU:
  - Global view (UPC)
  - Explicit blocking; SPMD
  - APGAS collectives
- FT:
  - Scatter/transpose algorithm
  - Alltoall collective
- RA:
  - Network performance
  - Low runtime overhead
- Stream:
  - Rely on back-end compiler

Productivity in HPC:
Programs are easy to write;
High performance programs are easy to write.

Our thanks to:

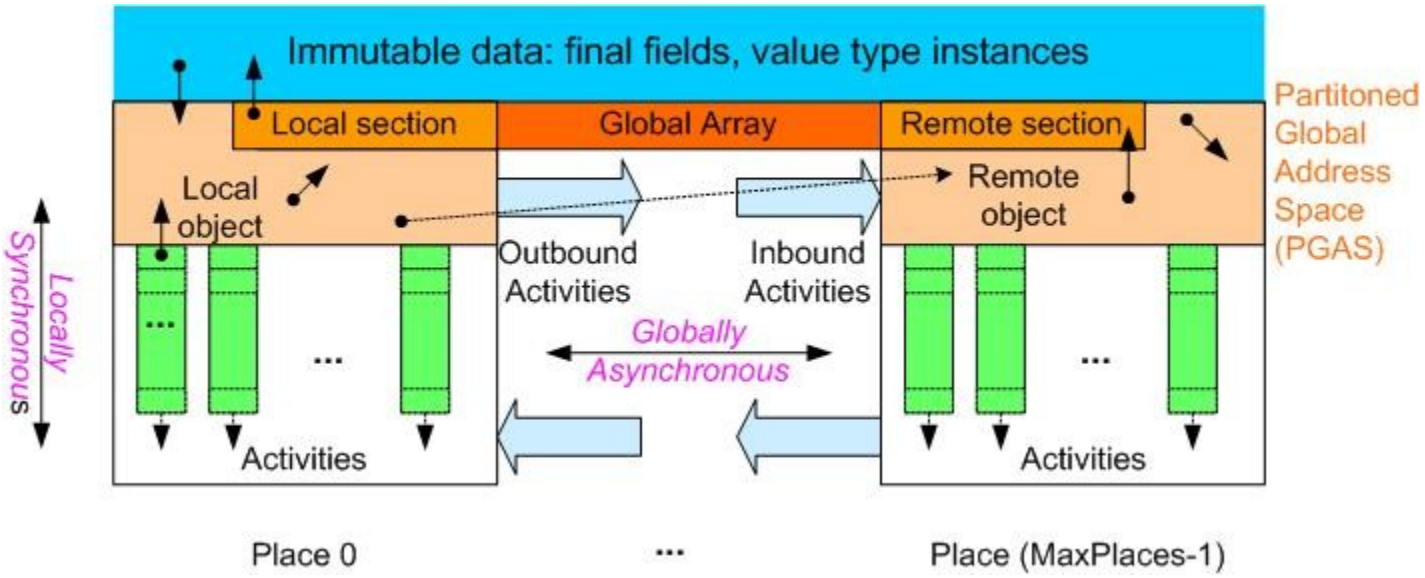
- **IBM Poughkeepsie Benchmark Center: S. Selzo**
- **NCSA BluePrint cluster: M. Showerman, W. Gropp**
- **IBM Research/WatsonShaheen: F. Mintzer, D. Singer, A. Raishubsky, B. Fitch**
- **BSC: David Vicente**
- **Christian Bell (Myricom)**

Backup

Performance factors

- **Runtime:**
  - Low overhead
  - Collective communication
  - Good language support (finish, async, shared arrays)
- **Compiler optimization is crucial**
  - UPC:
    - Locality inference
    - Comm. Aggregation
  - X10:
    - Allocation optimization
    - async/finish optimization

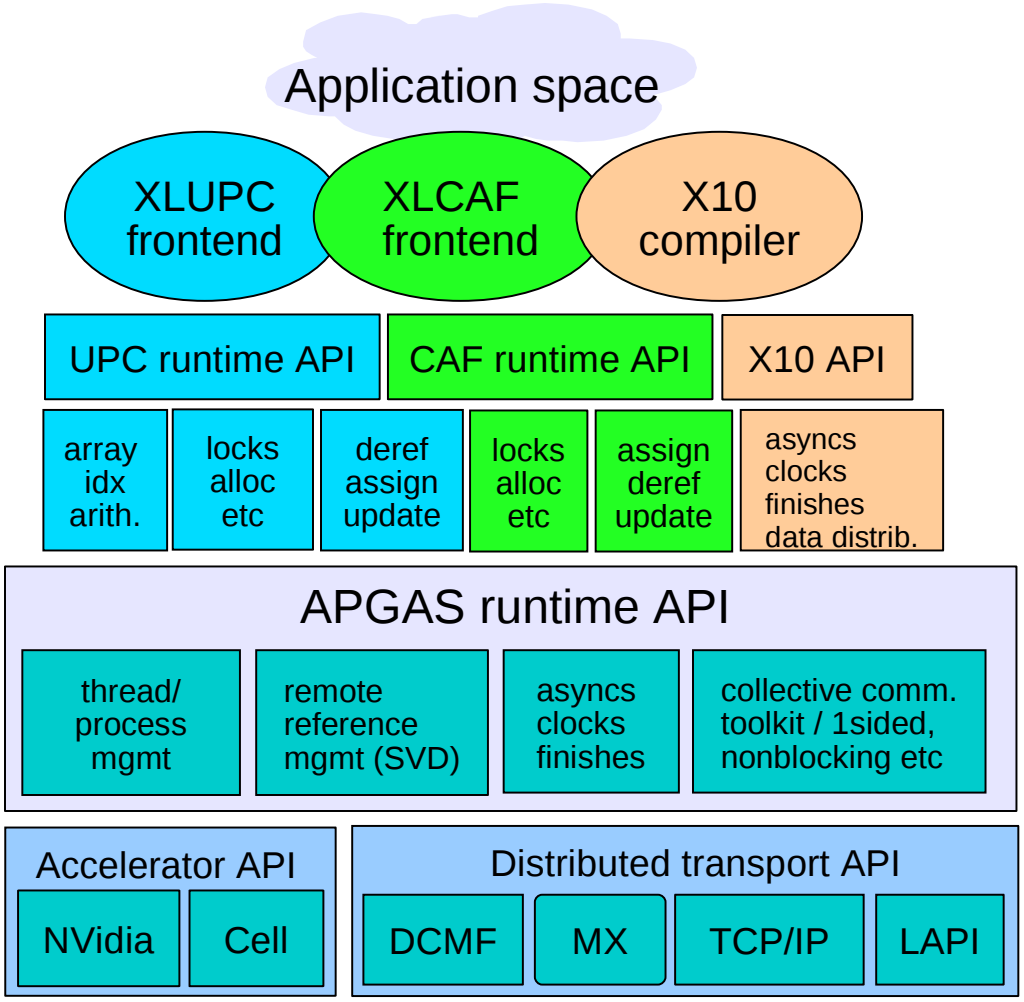
X10: Innovation, Productivity, Scalability



Fine grained concurrency • **async S**	**Atomicity** • **atomic S** • **when (c) S**	**Global data-structures** • **points, regions, distributions, arrays**
Place-shifting operations • **at (P) S**	**Ordering** • **finish S** • **clock**	

Two basic ideas: Places and Asynchrony

APGAS: one library to run them all



- **Support for UPC and CAF**
  - shared arrays; pointers-to-shared; locks; optimized collectives
- **Support for X10**
  - Asyncs & activities; remote references
- **Multiplatform**
  - Power, BG, Intel, Sun etc.
  - LAPI (IB, HPS), DCMF (BG), MX (Myrinet), TCP/IP sockets
- **Interoperable**
  - MPI

Bisection bandwidth calculation (Blue Gene/P)

# Nodes	Torus	Bisection (links)	BW (GB/s)	GUPS limit	FFT limit
32	4x4x2	32	13.6	0.32	39
128	4x8x4	128	55	1.30	176
1024	8x8x16	256	109	2.6	870
2048	8x16x16	512	217	5.2	1741
4096	16x16x16	1024	434	10.3	1562

Torus bisection = smallest diameter x 2 (torus) x 2 (half traffic)
Bisection Bandwidth = Bisection x 0.42 GB/s/link
GUPS limit = Bisection bandwidth / 42 bytes/packet
FFT Gflops = flops * BW / Bytes
FFT Gflops = 5* log(N) * N * N * Bandwidth / 3 * N * N * sizeof(cplx)