IBM Research

HPC Challenge 2005 Awards Competition: UPC on BlueGene/L

C. Caşcaval, C. Barton, G. Almási,
Y. Zheng, M. Farreras, P. Luk, R. Mak
IBM Research and IBM SWG Toronto

Environment

- **Blue Gene characteristics & installations**
  - BG nodes (2 procs. each) have 4M L3 cache, 512 MB local memory; connected by a 3D torus, 175 MB/s/link
  - Blue Gene/X – 1 rack, 2048 procs., 512 GB mem.
  - Blue Gene/W – 20 racks, 40K procs., 10 TB mem.
  - Blue Gene/L – 64 racks, 128K procs., 32 TB mem.
- **Software**
  - An experimental version of the IBM XL UPC compiler
  - An experimental version of the BG/L communication library
- **Benchmarks:**
  - Random Access and EP STREAM Triad

Random Access

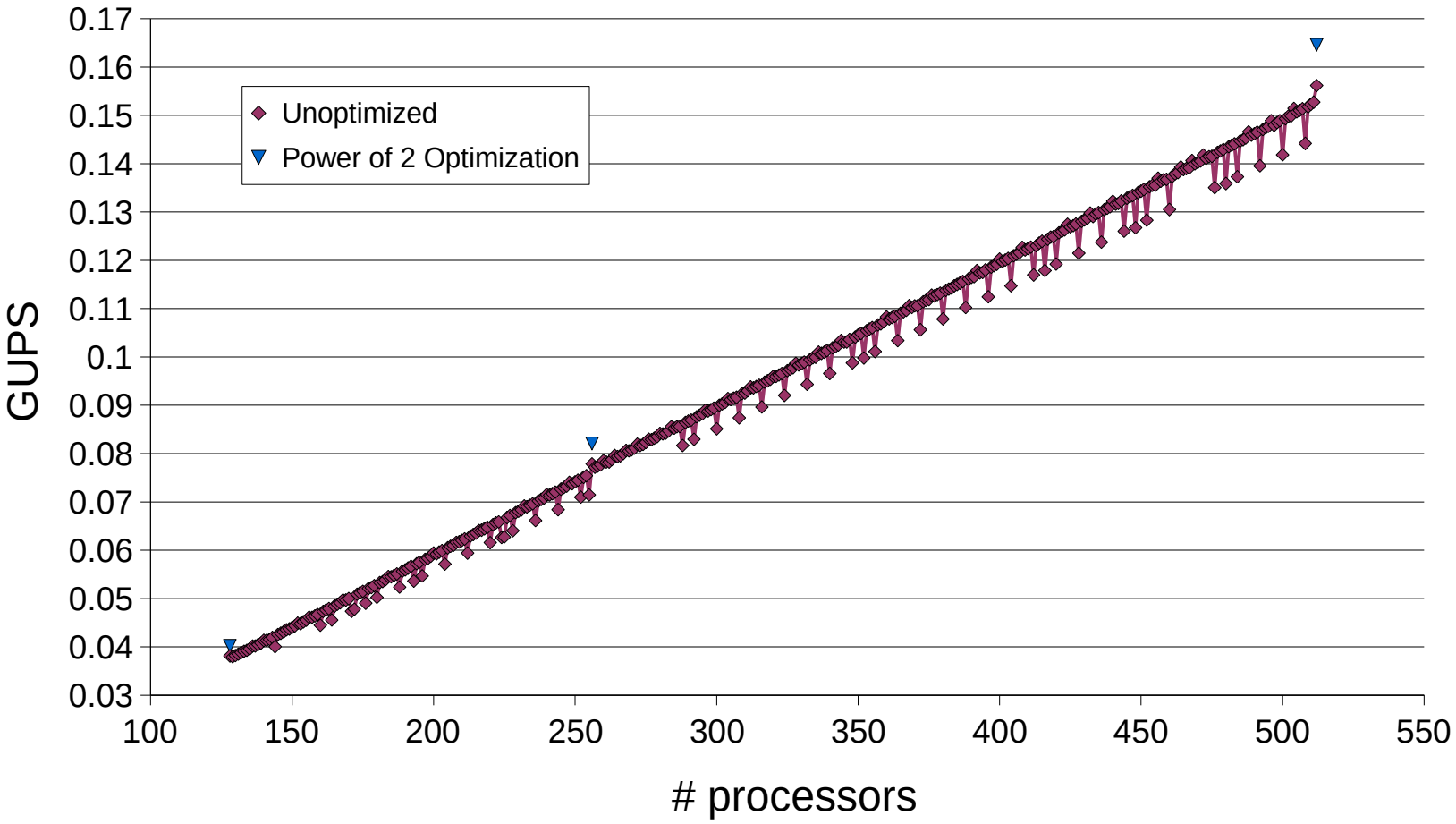
```
u64Int ran = starts(NUPDATE/THREADS * MYTHREAD);
upc_forall (i = 0; i < NUPDATE; i++; i) {
    ran = (ran << 1) ^ (((s64Int) ran < 0) ? POLY : 0);
    Table[ran & (TableSize-1)] ^= ran;
}
```

- **Each update is a packet –** performance is limited by network latency
- **Important compiler optimization:**
  - Identify update operations
  - Translate them to one sided update in comm. library
- **Verification:** run the algorithm twice
- **Lines of code: 111**

Random Access: Performance Results

Processors	Problem Size 2^N	GUPS	Efficiency
1	22	0.00054	
2	22	0.00078	73%
64	27	0.02000	61%
2048	35	0.56000	51%
65536	40	11.54000	33%
131072	41	16.72500	23%

Random Access – Power of 2 Optimization



EP Stream Triad

```
upc_forall (i = 0; i < VectorSize; i++; i) {
      a[i] = b[i] + alpha * c[i];
}
```

- **Embarrassingly parallel:** performance is gated by the individual node memory bandwidth
- **Important compiler optimization:**
  - Identify shared array accesses that have affinity to the accessing thread; transform them into local accesses
- **Verification:** random sampling
- **Lines of code: 105**

EP STREAM Triad – Performance Results

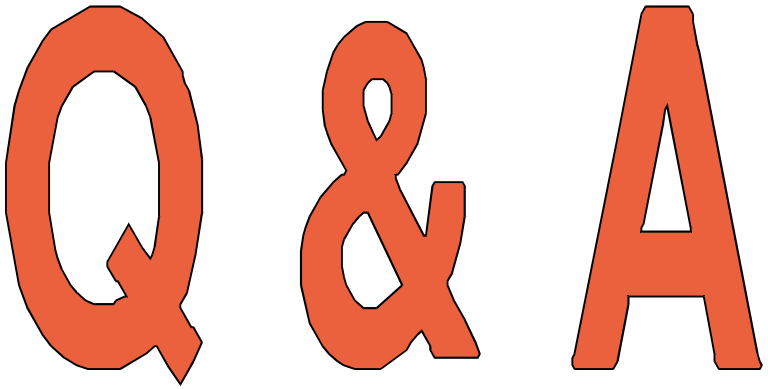
Processors	Problem Size	Memory Used	GB/s
1	2,000,001	45 MB	0.73
2	2,000,001	45 MB	1.46
64	357,913,941	8 GB	46.72
2048	11,453,246,122	256 GB	1472.00
65536	366,503,875,925	8 TB	47830.00
131072	733,007,751,850	16 TB	95660.00

Discussion

- We focused on the simplicity of code and on compiler and runtime optimizations, **not** on algorithmic changes
- **Most challenging issues:**
  - Overcome limitations in compiler indexing decisions and scaling the UPC runtime system to the max. machine size
    - How to index a 16 TByte array on a 32 bit machine?
  - Obtaining single node performance comparable to C
    - Eliminate the shared memory translation overhead
  - Reduce one-sided communication latency
    - Naïve UPC code tends to generate short messages

Acknowledgments

- **A long list of people, in particular:**
  - Roch Archambault
  - Jose Castanos
  - Siddhartha Chatterjee
  - John Gunnels
  - Manish Gupta
  - Roland Koo
  - Fred Mintzer
  - Tom Spelce (LLNL)
- **Work supported in part by the DARPA HPCS Program, contract NBCH30390004**



Backup: Theoretical GUPS limit on large configuration

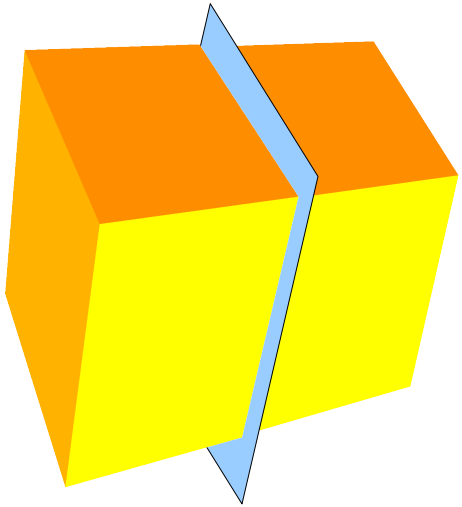
Pre-condition: one packet per update (naïve algorithm)

Update packets:

16 Byte header
4 Bytes target SVD
4 Bytes offset
4 Bytes op type,kind
8 Bytes update value

Packet size: 64 Bytes

75 Bytes on wire



$$\left\{\begin{array}{l}\left\{\begin{array}{l}\text{Packet size: } 75 \text{ Bytes} \\ \text{Wire speed: } 4\,\dfrac{cycles}{Byte}\end{array}\right\} \rightarrow 300\,\dfrac{cycles}{packet} \\ \text{CPU speed: } 700 \text{ MHz} = 1.4\,\dfrac{ns}{cycle}\end{array}\right\} \rightarrow \mathbf{2.38 \cdot 10^6}\ \dfrac{packets}{second \cdot link}$$

Cross-section bandwidth:
64 x 32 x 32 torus:
2 wires/link x **32** x **32** x **2** (torus) = **4096** links
$= 9.74 \cdot 10^9$ packets/s

Half of all packets travel across the cross-section;
GUPS limit = **19.5**