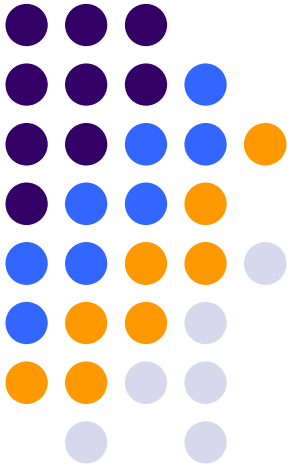
Early Adopter – Parallel Computing Education at the University of Pannonia

Zoltan Juhasz

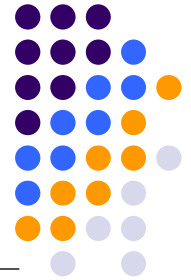
juhasz@virt.uni-pannon.hu

Dept of Electrical Engineering and Information Systems

University of Pannonia, Veszprem, Hungary

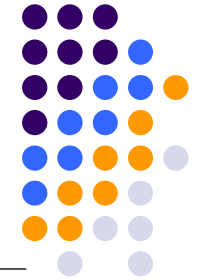


Background



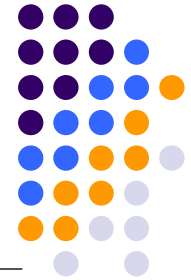
- Research intensive university in a historic town
- Student number is over 10 000
- Faculty of Information Technology
  - around 1200 students
  - Information Technology (Comp Sci with algorithms, programming or business majors) (BEng, MEng and PhD programmes)
  - Electrical Engineering (BEng only)

Current opportunities for covering parallel computing



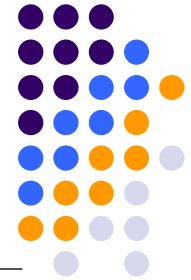
- **Java Programming** – core course, 2nd language, 14 weeks, 4 hours/week
- **Parallel Programming** – core as well as elective, BEng, MEng, 14 weeks, 4 hours/week
- **Degree projects** – bring in research topics as case studies for parallelisation, 2 semesters
- Planned extensions
  - Introduction to computing, embedded systems, telecom systems, web programming

Java as an introductory path



- Core course – 70 to 120 students
- Introduce threads with GUI
- Open up to general problems
- Introduce multicore systems
- Cover fundamentals of concurrency
  - non-determinism, race conditions
  - critical section, synchronisation needs
- Finish with producer/consumer problem
- Many hands-on exercises – make mistakes

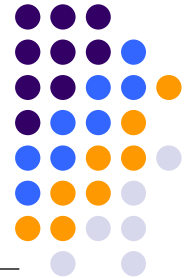
Parallel Programming



- Core and elective course – app. 30 students
- Covers architecture, programming and algorithms
- Main structure
  - Paradigms 2 weeks
  - Architecture 2 weeks
  - Models
  - Languages
  - Algorithms

Models, languages and algorithms evolve together from Java through OpenMP and MPI to Cuda

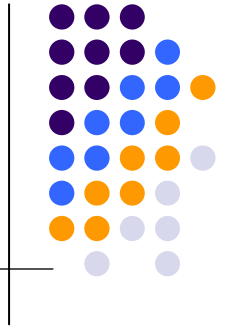
Topics in detail



- Architecture

Taxonomy	Flynn's	K
Data vs control parallelism	SIMD, GPU, multi-core, MIMD, heterogeneous, representative Top500 systems	C
Shared vs distributed memory	Covers all curriculum topics except NUMA	C
Memory hierarchy	Caching, cache coherence	K, also covered in Systems
Floating point	Not covered	Systems
Performance metrics	Run Scimark and NAS PBS benchmarks	A

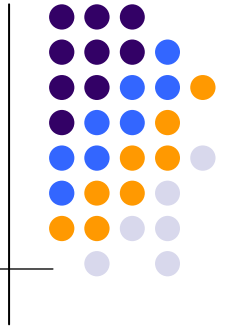
Topics in detail



- Programming

Threading	Java threads	A
Producer Consumer problem	Java	A
SMPD	MPI	A
Data parallel	OpenMP	A
Client server	Java TCP, Java RMI	A
Performance metrics	Run Scimark and NAS PBS benchmarks	A
Hybrid	Cuda	A
Notation	Java, MPI, OpenMP, Cuda, (occam – K only)	A
Correctness	critical section, monitor, semaphore, deadlock, livelock, resource contention	C

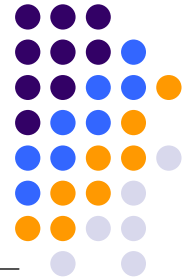
Topics in detail



- Algorithms

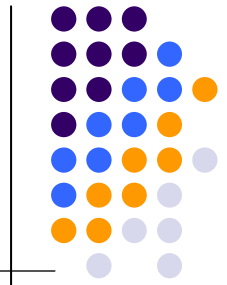
Collective communication operations	Broadcast, gather, gather with vector-matrix examples (MPI)	C
Data distribution	Block and striped data distribution, load balancing (OpenMP, MPI)	A
Matrix algorithms	Transposition, Matrix-vector, matrix-matrix multiplication, (Java, OpenMP, MPI and Cuda implementaitons	A
Searching, sorting	Compare-exchange sort (MPI)	C
PDEs	Not implemented this year	K
Graph algorithms, image processing	Covered in previous years (shortest path, etc)	N
Libraries: CuBlas, Magma	Discussed and used in implementing test programs	C

Programming Examples



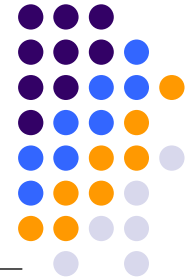
- Hands on exercises are a must. My students enjoyed this the most.
- Use simple examples, toy problems are very useful
- Identify core issues and concentrate on fundamentals
- Benchmarks – Scimark2
- OpenMP, MPI, Cuda sample programs (matrix algorithms, communication samples)
- Mandelbrot set ☺

Degree projects



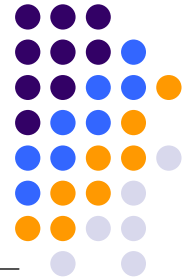
- In the past
  - Mainly Grid and distributed Java programming projects
- Current projects use Cuda
  - Colour segmentation of historic maps
  - EEG/based brain activity imaging and visualisation
  - MRI brain volume segmentation
  - 60-300x speedup achieved

Evaluation



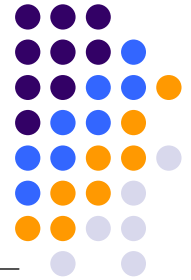
- Theory is evaluated at the exam
- Past experience shows that students grasp the curriculum and reach the ‘C/A’ level on each topic
- Each student wrote several parallel programs individually and in teams using Java, C/OpenMP, C/MPI, C/Cuda. The programs are used as benchmarks and their behaviour is discussed to ensure they reach level A in fundamental parallel programming skills.

Discussion



- Have operating hardware and software at hand, working properly – can waste lot of time with configuration and setup
- Java followed by Par Prog works well
- Reinforce relevant architecture, op sys topics
- Message: Think in parallel
- Problem and application driven course delivery, not “l’art pour l’art”

‘DVD Extras’



- Live runs at lectures are essential
- Performance optimisation demos
- Parallel system access is a must
- Cuda card is invaluable! Teraflops live!
- Show real problems
  - Image processing
  - Log file search
  - Computational science (N-body, fluid simulation)