INTRODUCTION
- Data Structures is a fundamental component in any Computer Science curriculum.
- A freshman course in IIIT-H Programming knowledge is a prerequisite.
- A good chance to sensitize students about parallelism in computing.

RATIONALE
- Data Structures lays foundation for other CS courses.
- Early exposure to the need for parallelism is required.
- Treat Data Structures as the first course in a sequence of courses including Algorithms, Advanced Algorithms, Advanced Computer Architecture, Compilers, Advanced Compilers, and the like.

OBJECTIVES
- Initiate discussion on parallelism in computing early on.
- Bring out algorithmic and data structure differences
- Focus on seeking independent operations.
- Prepare ground for later courses
- May not introduce parallel architectures, parallel algorithmic models, and the like.

SYLLABUS-I
- Introduction to data structures
- Introduction to asymptotic analysis – mainly O(.) notation
- Array as a data structure, sorting, parallel sorting, parallel prefix
- Stacks and queues
- Linked lists, ideas from list representation and ranking and its difficulty in the parallel setting.

SYLLABUS-II
- Hashing
- Trees, applications to evaluation, searching, balanced search trees, scope for parallel operations
- Graph traversal techniques, shortest paths, spanning trees, solutions in the parallel setting
- Advanced data structures such as Union-Find, B-trees, Suffix tree.

TEACHING MODEL AND LOGISTICS
- Close to 200 students taught in two separate sections.
- Common material and exams
- Fourteen teaching weeks
- Three hours of lectures per week
- Guided laboratory of three hours per week.
- Weekly tutorial – one for every group of 25 students.
- Help from 8 teaching assistants.

EVALUATION MODEL – I
- Level of coverage of topics related to parallelism in computing
- Impact on the traditional contents was kept to a minimum
- Roughly 15% of lecture time spent on parallelism.

EVALUATION MODEL – II
- Questions on parallelism in computing included in the semester examinations.
- Online (Google Forms) survey conducted to get feedback.
- In class discussions

RESOURCES
- Presentation slides available for download.
- URL : http://cstar.iiit.ac.in/~kkishore/DS_Spring11/
- Lecture notes available
- Course material includes homeworks, exams, and quiz.
- Some material based on the books by Mark Weiss, JaJa.

FUTURE DIRECTIONS – I
- Work on similar changes to other introductory courses such as Computer Systems Organization, Algorithms, and Operating Systems.
- Other advanced courses such as Advanced Algorithms, Multicore architectures can benefit from the background

FUTURE DIRECTIONS – II
- IIIT-H offers an honors stream for UG students in parallel computing.
- Every year plan to offer the following
  ▶ Advanced Algorithms
  ▶ Multicore Architectures
  ▶ Parallel Programming
  ▶ Concurrent Data Structures
  ▶ Advanced Compilers

SAMPLE FEEDBACK COMMENTS - I
- Parallel computing improves efficiency, reduces run time.
- Parallel prefix can be done in parallel efficiently, requires more space than the sequential algorithm, the standard sequential algorithm has too many dependencies.
- Merge sort can be done in parallel using binary search.

SAMPLE FEEDBACK COMMENTS – II
- Storing a linked list can be done using either the successor or the predecessor for every element.
- Expression evaluation differs significantly from the sequential algorithm.
- Cannot use the standard sequential algorithm for list ranking due to too many dependencies

CONCLUSIONS
- Can introduce parallelism in computing independent of architectures and languages.
- Some difficulty experienced in discussing solutions for a large processor base.
- Present coverage weak in analysis of parallel algorithms. Can try other approaches such as XMT.
- Feedback suggests that it is not early to introduce parallelism in Data Structures.

Kishore Kothapalli
kkishore@iiit.ac.in

Acknowledgements: NSF, IBM Hybrid Multicore Center
NSF EduPar Workshop @ IPDPS ’11
May 16, 2011