What we have done

- We identified courses in which PDC concepts can be relatively easily blended with existing concepts. For example here are some of the courses PDC concepts are introduced:
  - CS160 - Introduction to Computer Science
  - CS271 - Computer Organizations
  - CS372 - Operating Systems
  - CS452 - Internet Communications
  - CS474 - Concurrent Systems
- We revived our Concurrent Systems course
- We started to emphasize on multi-core computers
- Selected students for Western Oregon Academic Excellence Showcase
- Started a survey to learn students’ perspective

What we will do

- Continue our pushing of cover PDC concepts in our curriculum
- Survey students to learn more about the effectiveness of our teaching and try to identify the reasons why some students do not want to take the PDC concept heavy courses
- Continue to select students for Western Oregon Academic Excellence Showcase
- We invite the faculty in the Computer Science Curriculum Committee and across campus to audit classes that involve PDC topics. They will be polled on how appropriate the integration is, and asked for suggestions on how to improve the PDC topic integration efforts.
- Learn from other schools for ways to improve our teaching
- Introduce a graduate level course so we can conduct some research

What we want to do

- The physical limitations, such as the speed of light, lead us to focus on concurrency as a method to increase computational efficiency. So, we want to introduce parallel and distributed computing (PDC) concepts to our students.
- We want our students to become familiar with PDC concepts early, and through many courses.
- We want our students to be able to program multi-core computers and be able to utilize the hardware resources.
- We want our students to be excited about parallel programming as we are.
- We want to attract more and more students to become interested in learning more about parallel programming.
- We want to know how are we doing.

We have started

About CS474 - Concurrent Systems

- It is offered every Winter term.
- It is designed to study parallel architectures, parallel algorithms, and parallel programming.
- In this course, students have written parallel programs on UMA, clusters, and multi-core parallel computers.
- Students write code on Intel's multi-core, IBM's Cell processors, and NVIDIA Tesla C2075 video card.
- We cover concepts such as data dependencies, load balancing, locking, synchronizations, and granularity.
- The class also covers many PDC proposed topics:
  - parallel computer organizations (mesh, hyper-tree, butterfly, hypercube, Shuffle-Exchange networks, etc.), parallel algorithms on PRAM (finding Max in a constant time, fan-in, parallel prefix sum, list ranking, parallel merging, etc.),
  - parallel programming concepts (shared memory vs. distributed memory, speedup, cost of a parallel algorithm, barrier and semaphore synchronizations, data parallelism vs. control parallelism, etc.),
  - computation theory related topics such as NC and P-Complete classes,
  - important parallel processing laws such as Amdahl’s law and Gustafson’s Law
  - parallel sorting algorithms such as Bitonic sort, parallel quick sort, etc.
  - other algorithms such as list ranking, merging of sorted arrays, Monte Carlo,
- Students select, design, implement, and present a term projects.

Students’ Projects