Vision

- In future, all programming will require parallel thinking and practice
  - Ensure core concepts are on required curriculum path
  - All programming assignments become parallel programming assignments
- Paths forward:
  - Work on integrating parallel concepts across curriculum
  - Start with parallelism in the first year and continue
  - Outreach to teach parallelism at highschools

Where we are now

- There are classes where parallelism is already pervasive, can augment emphasis to highlight it:
  - architecture
  - operating systems
- There are low-resistance ways to teach parallelism
  - Senior elective
  - Parallelism for non-CS majors
    - Ex: UIUC teaching CUDA

Challenges

- Lack of consensus among ourselves on what parallel techniques should be taught
- Resistance from colleagues who themselves may not understand how to teach parallel thinking
- Slow pace of curriculum change
- Lack of tools (debugging still very primitive)

Approaches to teaching

- Bottom up thinking
  - Need to understand fundamentals
    - Threads/Locks, communication
  - Need to appreciate true costs (time)
- Emphasize parallelism in data structures & complexity
  - (e.g. mergesort versus qsort)
  - PRAM

Recommendations

- Provide a potpourri of approaches
  - Incremental low-resistance change
    - Senior electives
    - Convince theory faculty to teach parallel algorithms
    - Simple augmentations to receptive courses
    - Teach a non-majors parallel programming course
  - Work at CRA level to impart urgency of the need to teach parallel practices throughout the curriculum
    - Value on day 1 (freshmen courses)
- Survey university approaches to teach parallelism
  - Look for what works
  - Aggregate to understand what is done now