CS 704: Suggested Projects

- Research topic/open problem
- Content, length, difficulty, complexity
- Advice: links on the web page
- “An Evaluation od the Ninth SOSP Submissions”
- “The Task of a Referee”
- Project Presentations

  - what are your references?
- Practical Importance (argue about it...)
- Lessons, applicability
- Choices (alternatives and justification)
- Context (assumptions, insights)
- Focus (irrelevant? follow?)
- Presentation (organization, terms, definitions, abstract, information, related work, simplicity)
- Writing Style
- Convincing...

What is suitable for publication?

- Paper types
  - real system, implementation
  - ideas or techniques that technical community should know (validation necessary though)
  - theoretical
- Original idea/open question
  - is the idea new?
  - how do you know?
  - do you state it concisely?
  - what is it exactly?
  - how big an innovation?
  - how *different* from existing work?
  - what do others do?

Reviewers point of view...

- Quality control...
- Journals, conferences
- Refereeing process
- Reputation (both ways)
- Seek for: correctness, significance, innovation, interest, timeliness, excitement, readability, and more...
- Think professionalism!

CS704 Projects and Deadlines

- Goal: operating systems research
- What about implementation?
- Use one of your favorite papers as a model
- Pose a question, answer the question
- Conduct the research, write up experiences and results
- Deliverables...

Project Suggestions

- Reasonable to be completed in 2-3 months
- Tangible results
- Learn
- My bias
  - Scheduling Support for Parallel Processing in NOWs
  - To migrate or not?
  - Multi-Process Prefetching and Caching
  - Long-term and short-term file access patterns

Deliverables

- Project proposal (due next week)
- Literature Review
  - Related Work (careful here!)
  - Experimental Setup
  - Schedule, preliminary plan
  - Design of experiments
- Progress Report I
- Progress Report II
- Final Report
  - Final product

Scheduling Support for Parallel Processing in NOWs

- NOWs as a favorable platform for parallel computation
- Efficiency?
- Scheduling Problem
  - message-passing parallel systems
  - static, adaptive, dynamic: space-sharing
  - time-sharing, hybrid methods
  - restrictions posed by SPMD problems
- in NOWS: tougher problem
  - volatility of the environment
  - non-homogeneous

- Departmental network of workstations
- Scenarios...
- Implementation of dynamic space sharing
- Else: check-pointing
- System-level versus user level
- Advantages/Disadvantages
- My choice: user level
- Why?
- MPI (Message Passing Interface)
- PSPARSELIB, PETSc, possibly others

To migrate or not?

- NOWs again
- Cost-effectiveness of process migration
- Prior work: availability of systems in CS departments
- Measurements on our network
- Load-balancing
- Different behavior at different times of the day
- Could you predict heavy load situations (bursts)?
- Could you predict load changes?

- Requirements: parallel programming (MPI)
- Libraries: iterative methods
- Rewriting for redistribution of resources
- Use of real (production) programs
- Develop a scheduler
- Programs and scheduler "talking" to each other
- More issues to consider: migration cost, memory, disk space, heterogeneity
- Advantage: most people talk about these things but never really implement
- Limited serious implementation attempts
- Platforms: local network of workstations, then Beowulf cluster

To migrate or not?

- Analyze measurements
- Development of a "model"
  - Work of Harchol-Balter and Downey (Sigmetrics'96)
  - Stochastic learning automaton
  - Hidden-Markovian Models (as predictors) (ISCA'97, Joseph and Grunwald)
  - Fuzzy logic issues
- Model should drive migration policies:
  - how long should a process stick around?
  - can your policy be "adaptive"
  - implications for parallel programs
- Platform: local network again, different labs

- Measurements (local + other traces)
- Drive simulation model to evaluate suggested policies
- Incorporate Quality-of-Service guarantees

Multi-Process Prefetching and Caching

- Tools, issues
- Disk simulator (from Dartmouth)
- Traces of application request patterns (from Illinois)
- Simulator
  - page interrequest distribution
  - load on the disk (contention)
  - policy implementation
- Modeling
  - simple Markovian model of prefetching depth

Multi-Process Prefetching and Caching

- Mental model
- Single disk, multiple disks
- Single program, multiple programs
- Parallel programs (many processes)
- Single cache at the disk(s) side (server side)
- Share it effectively among many programs
- Share it effectively among many processes
- Research questions:
  - how far ahead should you prefetch?
  - producer-consumer relationship
  - "adaptive" sharing strategies

Long-term and short-term file access patterns

- Measurement project
- Main idea: which files would you move into a "permanent" storage?
- Personal experience
- Disk space (on each workstation)
- Analyze data
- Creating vs. accessing vs. modifying
- Frequency
- Adaptive techniques to decide what to migrate and when
- Integrate time

- Mass storage systems for different environments
- Algorithms, simulation studies