Sermon: Project Risk v. Efficiency

<Figure: "Mortar shot graphs" from a research paper>

Parable – "mortar shot graph" in parallel programming

- general shape: rise quickly, level off, fall
- adding more processors never increases efficiency (compare perfect speedup to real speedups; divide x by y for efficiency – efficiency always < 1)
- some programs better than others
    - turns out – directly related to how fine pieces are – how much time spent coordinating/communicating

moral – chopping problems into smaller pieces increases overhead, reduces efficiency
very general theorum: turns out applies to humans as well

obvious inference → chop projects into large, coarse-grained pieces for efficiency
e.g. you do problem 1, I'll do 2, she'll do 3
then we'll get together and pull the pieces together

Seems efficient

- only need 2 meetings: at beginning and at end of projects
- everyone gets to go to lab when most convenient during day/week/projects
- code my section however most convenient to me (as long as I obey the interface)

BUT I believe this is the **wrong approach**
Problem: high-risk strategy (even if efficient in best case)

1) need to get interface exactly right at beginning of project
    a) hard to do at start of project when you know the least
    b) hard to get interface right for big pieces (v. small pieces)
    c) few bother even to spend the time to try to get it right
2) what if a part takes longer than expected (unexpectedly hard, person unexpectedly busy, person is a slacker)?
    - ♦ if deadline comes and part1 doesn't work, nothing works
    - ♦ if deadline comes and part 2 doesn't work, ¼ done
    - ♦ if deadline comes and part 3 doesn't work, ½ done
    - ♦ if deadline comes and part 4 doesn't work ¾ done
      → expected grade35%

    alternate strategy – work as team, end up finishing first 3 parts and some of part 4 → expected grade > 75% (actually better than that – *whole team* realizes what is wrong and can act as a team to fix it early

3) Dependencies among parts (see 2)
4) Exam – the reason you're taking this class is to learn this stuff—get your money's worth by learning the entire project, not just a piece
    (This is true in real life – it is good to know the whole project so that next month/next year when you get moved to a different part of the project)

Better approach – plan to spend a lot of time coordinating – be willing to accept "reduced efficiency" for much increased likelihood of everything going smoothly (would you rather take 20% longer on a project or take a 20% risk of it taking twice as long?)