Static Parallelization of AOBB

April 1st, 2011
Recap

• Dynamic Parallelization with $p$ workers.
  • Generate at most $p$ subproblems
  • More only if previous one finishes.
  • Complexity threshold $L$ (=30 minutes)

• Load Balancing:
  • Split subproblem if complexity above threshold $L$.
  • Learn from completed subproblems.
    – Effective branching factor $b$
    – Avg. leaf node depth $D$ (via avg. increment $inc$)
Issues with static scheme

• Doesn't scale well:
  • As $p$ grows, subproblem traffic increases
    – Becomes almost continuous.
  • Distributed scheduler cannot keep up.
    – Becomes unresponsive / crashes.

• Not suitable in many practical settings:
  • Shared grid resources with common job queue.
    – Number of workers $p$ fluctuates.
  • Challenging for complex grids hierarchies.
Alternative: Static

- **Set entire** parallelization frontier ahead of time.

- **Simplest approach:**
  - Threshold number of subproblems, at least $L$.
  - Iteratively deepen frontier:
    - Split most complex subproblem.
    - Stop once $L$ subproblems generated.

- **Possible secondary objective:**
  - Enforce upper/lower complexity bound.
Deepen Frontier

- At every step, order nodes \( \{n_i\} \) by subproblem complexity \( C_i \)
  - Absolute values not required for comparison.
  - \( C_i = b^{a_0 + a_1 U_i + a_2 L_i + a_3 h_i + a_4 w_i} \)
- Compare only exponent (log space)
- Learn \( a_j \) through linear regression.
Linear Regression

- Run static with “ad-hoc” metric to get subproblem data
  - \( C_i = b ^ {(U_i - L_i) * h_i * w_i} \)
  - Varying thresholds yield multiple runs per problem
- Regression on small random sample
  - Test on same run or different run
Same run (p=320)
Regression pedigree7_110302-134208-i19-w12-t0-p320

Same run (p=320)
Regression pedigree31_110222-191258_i18-w10-t0-p160

\[ p=640 \rightarrow p=160 \]
Regression pedigree13_110302-141031-i20-w12-t0-p160

p=320 → p=160
$p=640 \rightarrow p=160$
Regression pedigree7_110302-134554_i19-w12-t0-p160

p=320 → p=160