Application of Lagrangian Receding Horizon Techniques to Resource Management in Ad Hoc Grid Environments

R.H. Castain

Los Alamos National Laboratory

W.W. Saylor

SAIC, Inc.

H.J. Siegel

Colorado State University

Problem Description

- Ad hoc computing grids
  - Heterogeneous collection of computing and communication resources without fixed infrastructure
- Challenges
  - Assets can appear/disappear without warning
  - Communication links prone to failure, noise
- Required
  - Resource manager capable of rapid response to changing conditions

Approach

- Lagrangian objective function
  - Combine constraints into objective function using time-dependent parameters (Lagrangian multipliers)
- Receding horizon
  - Optimal control method
  - Predict evolution of system for limited time into future
  - Control based on prediction until next measurement of system state
- Initial experiment
  - Determine performance under different conditions
  - Evaluate sensitivity of critical parameters

Simulation Environment

- Two types of machines – fast, slow
  - Differentiated by cpu speed, energy consumption rate, communication bandwidth
- Single application
  - 1024 inter-communicating subtasks connected in directed acyclic graph (DAG)
  - Two versions of each subtask: 100% and 10%
  - Estimated time to compute provided for each subtask/machine/version triplet
  - 100 ETC/DAG combinations
- Three Cases
  - A: 2 fast, 2 slow
  - B: 2 fast, 1 slow
  - C: 1 fast, 2 slow

Objective

- Maximize number of 100% subtasks completed (T_{100})
  - Within specified time, energy constraints
  - Must complete all subtasks
- Objective function

$$ObjFn(\alpha,\beta,\gamma) = \alpha\frac{T_{100}}{T} - \beta\frac{TEC}{TSE} - \gamma\frac{AET}{\tau}$$

TEC = Total Energy Consumed
TSE = Total System Energy
AET = Application Execution Time
α,β,γ = Lagrangian multipliers [0,1], $\alpha+\beta+\gamma=1$
τ = time constraint

Heuristics: Max-Max (static)

- Provide performance baseline
  - Static heuristic – not suited to dynamic environment
- Two step process
  - For each machine, pick subtask/version pair that maximizes ObjFn
  - From that set, select machine/subtask/version triplet that maximizes ObjFn
- No receding horizon
  - Considered all subtasks, entire mapping simultaneously
  - Selected triplet could be scheduled for any time provided adequate "hole" in existing schedule can be found

Heuristics: SLRH* (dynamic)

- At each time step
  - For each machine, if available…
    - Collect set of all subtasks U whose
      - Precedence constraints are met
      - Adequate energy to execute at least 10% version
      - Meet worst-case communications
    - Evaluate ObjFn for each subtask in U, both versions
    - Order U based on ObjFn
    - Find first subtask/version pair that can be scheduled to start within time horizon H – map it
  - Increment time by time step ΔT

*Simplified Lagrangian Receding Horizon

Two Additional Variants

- SLRH-2
  - Assign all subtask/version pairs until
    - All pairs assigned
    - No additional pairs can be started within time horizon
  - Unable to successfully map all subtasks – dropped
- SLRH-3
  - Re-create, re-evaluate U after each assignment
    - Catch new subtasks that meet precedence constraint
  - Continue assigning pairs until no additional pairs can be stated within time horizon

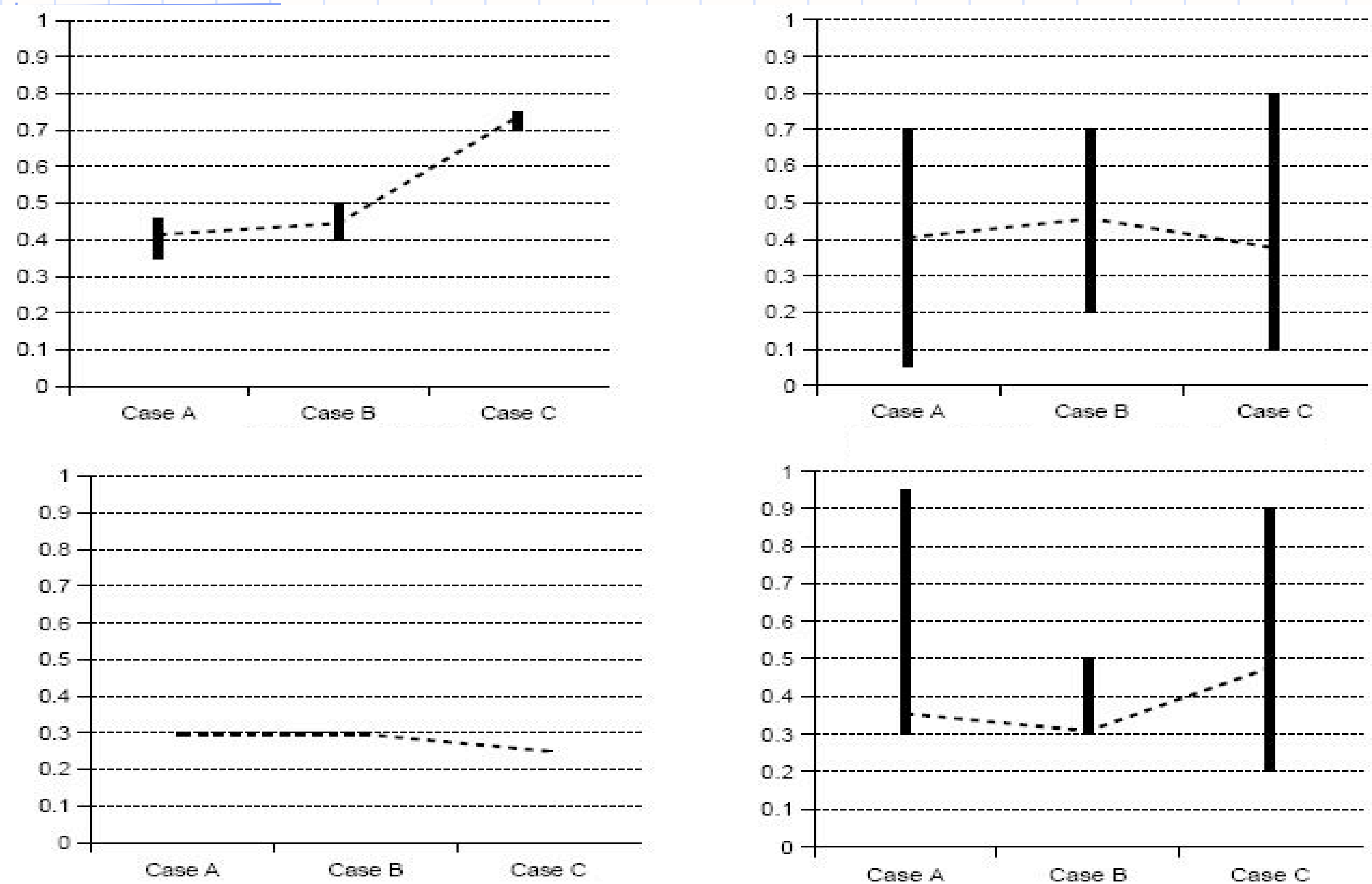
SLRH: Closer Look

- Simplified Lagrangian
  - No dynamic adjustment of α, β, γ
  - Acceptable for this experiment
- No guaranteed non-violation of constraints
  - Explicitly checked execution time constraint, energy constraint
- Setting ΔT & H
  - Experimentally determined
  - ΔT = 10 clock cycles
  - H = 100 clock cycles

ObjFn Parameters: α, β

α

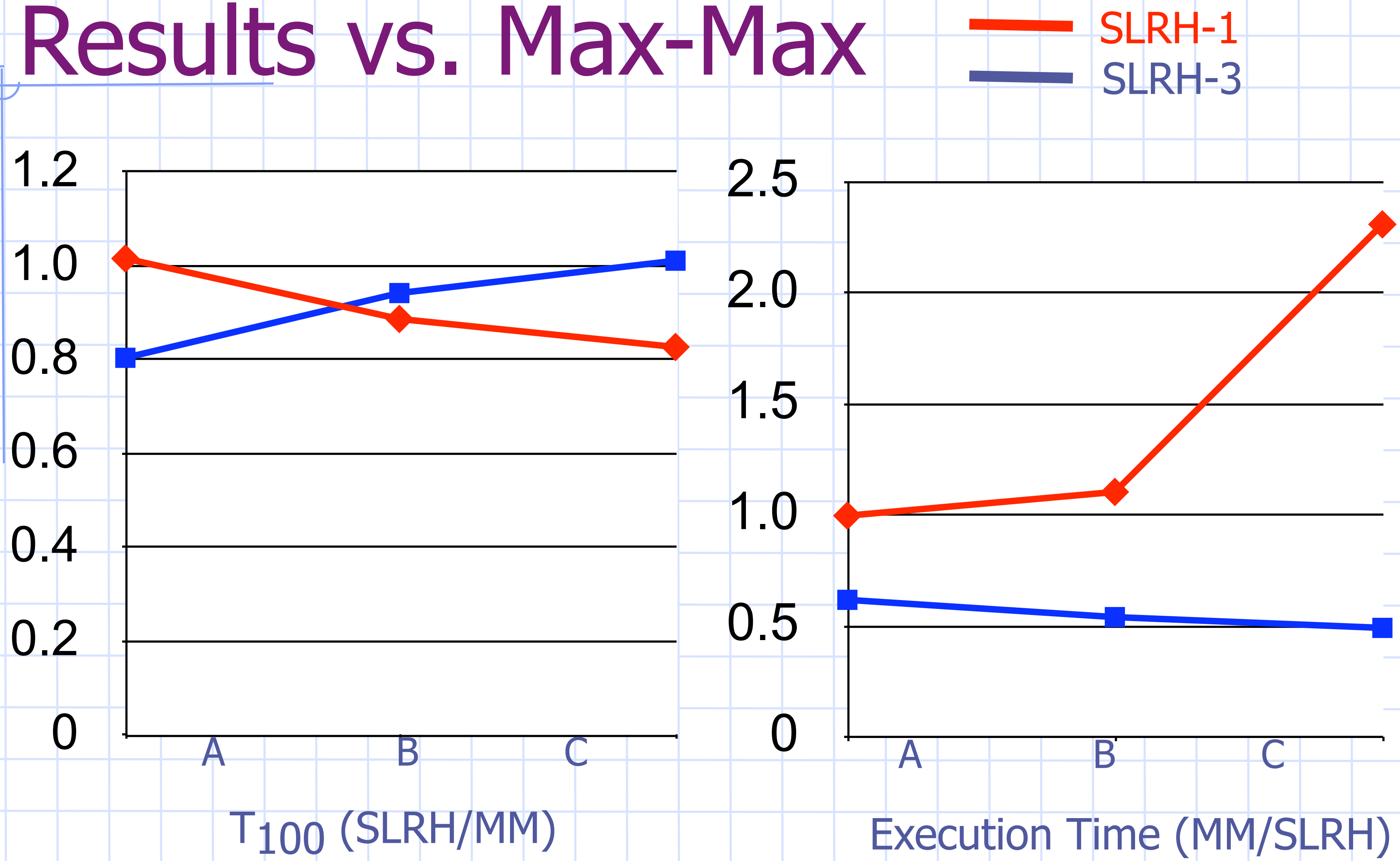
β



SLRH

Max-Max

Results vs. Max-Max



*Averaged over 100 ETC/DAG combinations

Summary

- SLRH performance
  - Comparable to static baseline
  - Appears relatively insensitive to characteristics of application
  - May require dynamic adjustment of the T_{100} Lagrangian multiplier to reflect changes in machine availability
- Speed needs improvement
  - Non-optimized scripting language used
  - Convert and optimize
- Questions