DISTRIBUTED RESEARCH ON EMERGING APPLICATIONS & MACHINES
dream-lab.in | **Indian Institute of Science**, Bangalore

DREAM:Lab

SE252:Lecture 20, Mar 24

ILO4: Application Execution Models on Cloud *(Dynamic Scheduling)*

Yogesh Simmhan

DREAM:Lab

Lecture 20: ILO 4

Constrained Scheduling

- Best effort on co$t
- Constraint on a deadline
- Constraint on a budget
- Adaptive scheduling

Multi-criteria Optimization

- Time & Cost Budget
  - Reduce time (best effort) while ensuring below budget (fixed)
- Normalize time & cost into single optimization function
  - Reuse existing algorithm
- Optimize one, then tweak to optimize other
  - `LOSS-GAIN` Approach

Scheduling with Budget Constraints§

- LOSS Approach
  - Schedule using HEFT. If within budget, done.

$$LossWeight(i,m) = \frac{T_{new} - T_{old}}{C_{old} - C_{new}}$$

    - » *T_old* is time taken by task *i* on old machine assigned by HEFT
    - » *T_new* is time taken by task *i* on new candidate machine *m*
    - » *C* is cost on respective machines
  - Try re-assignments using smallest *LossWeight*

Scheduling Workflows with Budget Constraints, Rizos Sakellariou and Henan Zhao, *Integrated Research in GRID Computing*, 2007

Scheduling with Budget Constraints

- GAIN Approach
  - Initially map each task to the cheapest available machine (rather than HEFT)
  - Then make reassignments based on the largest *GainWeight*

$$GainWeight(i,m) = \frac{T_{old} - T_{new}}{C_{new} - C_{old}}$$

  - Continue till budget is exhausted

Dynamism

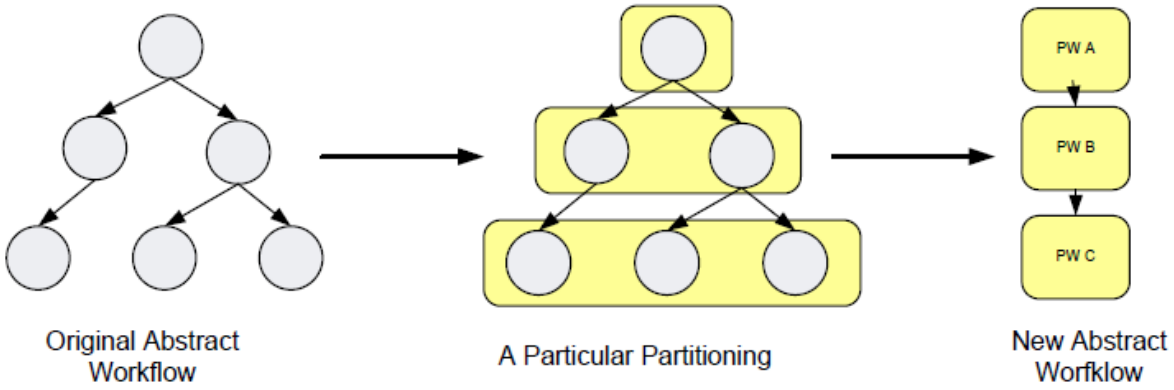
- Variability in application requirements
  - DAG branches non-deterministic
  - Continuous applications
- Variability in Infrastructure
  - Performance of VMs vary in time
  - With/without task migration

DAG Scheduling Strategies under Dynamism

- Full-ahead planning
  - Plan task to machine mapping *a priori* (HEFT)
  - Costly, Does not consider current conditions
- Incremental
  - Add task to queue when ready
  - Simple, can lead to sub-optimal (e.g. wait time for large jobs)

DAG Scheduling Strategies under Dynamism

- Just-in-time planning
  - Pegasus: Stage at a time, when ready



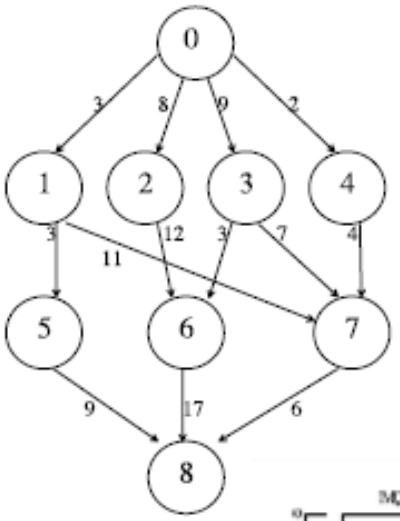
§ *Pegasus: Mapping Scientific Workflows onto the Grid*

Dynamic DAG Scheduling Strategies

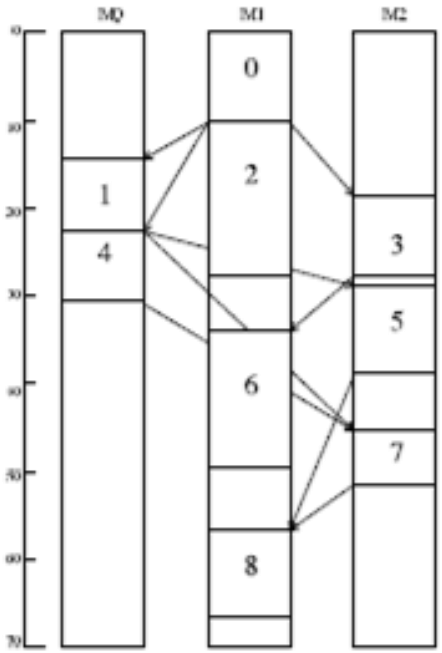
- Just-in-time rescheduling
  - Start with HEFT. At runtime, use actual task time to decide rescheduling.
  - *Slack & spare time* gives us leeway
- **Spare time** between a pairs of dependent nodes
  - DAG/machine dependence
  - Maximal time the source node can execute for *without* affecting the start of the sink node
- **Slack time** is the minimum spare time on any path from a node to the exit node
  - Maximum delay that can be tolerated, *without* affecting the overall makespan.
  - Slack = 0 → Node on Critical Path

§ A low-cost rescheduling policy for efficient mapping of workflows on grid systems, Rizos Sakellariou and Henan Zhao, *Scientific Programming 12 (2004) 253–262*

Dynamic DAG Scheduling



- *ST(j) is the expected start time of node j*
- *DAT(i, j)* is the time at which data required by node *j* from node *i* will arrive on its machine
- *FT(i)* is the finish time of node *i* in the given schedule



Spare time between node i & successor j (4&7)

$$\mathrm{Spare}_{\mathrm{DAG}}(i,j) = ST(j) - DAT(i,j),$$

For adjacent tasks i and j in a machine (3&5)

$$\mathrm{Spare}_{\mathrm{SameMach}}(i,j) = ST(j) - FT(i),$$

Maximal value that can be added to the execution time of this task without affecting the overall makespan (5&7)

$$\mathrm{Slack}(i) = \min_{\forall j \in D_i} (\mathrm{Slack}(j) + \mathrm{Spare}(i,j)).$$

Reading

- A low-cost rescheduling policy for efficient mapping of workflows on grid systems, Rizos Sakellariou and Henan Zhao, *Scientific Programming 12 (2004) 253–262*
- Scheduling Workflows with Budget Constraints, Rizos Sakellariou and Henan Zhao, *Integrated Research in GRID Computing*, 2007

Summary: Scheduling on the Cloud

- Different types of applications
  - Stateful vs Stateless
  - Continuous Dataflows
- Client+Cloud
- Scale up vs Out
- Exclusivity, Variability
- Resource Sizing, Costs
- Elasticity

Scheduling

- List Scheduling
- HEFT Scheduling of DAGs
- Multi-criteria optimization
  - Scheduling on a Budget
- Dynamism

ILO 4

- Application Execution Models on Clouds:
  - *Design* and *implement* Cloud applications that can scale up on a VM and out across multiple VMs. ➲Project
  - *Illustrate* the use of load balancing techniques for stateful and stateless applications.
  - *Characterize* resource allocation strategies to leverage elasticity and heterogeneity of Cloud services ✓