Anticipated Distributed Task Scheduling for Grid Environments
Thomas Rauber and Gudula Rünger

Multiprocessor Tasks
• Decomposition of an application algorithm into a set of modules realized as M-Tasks
  – Well-defined independent parts
  – No relation between internal computations of different M-Tasks
• Input/output data form dependencies between modules (Task Graph)
• Each M-Task is executed on an arbitrary amount of processors

Example: DIIRK

The Grid Environment
• Abstract/fuzzy concept
  – “the technology that enables resource virtualization, on-demand provisioning, and service (resource) sharing between organizations.” (Plaszczak/Wellner)
  – “a service for sharing computer power and data storage capacity over the Internet” (CERN)
  – “a computer facility operating ’like a power company or water company’” (Corbató)

The Grid Environment (cont’d)
“... a service for sharing computer power ... over the Internet”
• Open standards
• Clients
• Servers
• Communication

Server Node
Scheduling

1. Partition Task Graph into a sequence of layers, \( L_i, i = 1, 2, \ldots, n \)
   - The different layers are executed in sequence.
   - Minimize number of layers.
2. Schedule each layer
   - Sequentially or concurrent
   - Locally or remote

Anticipated Task Placement

- The decision for the placement of layer \( L_{i+1} \) is taken after layer \( L_i \) is placed.
- When \( L_{i+1} \) is placed, the total cost, \( T_i(S) \) for the servers \( S \) is known.
- A task \( M \) of layer \( L_{i+1} \) should only be migrated if the migration cost \( C(M) \) can be hidden.
  - Each server maintains sets of migratable and non-migratable tasks.

Migration of Tasks

- A server \( S \) with neighboring servers \( S_j \) sends tasks as long as there exist a server \( S_j \), such that
  - \( T_i(S) < T_i(S_j) \)
  - \( T_i(S) = T_{\text{reg}}(S_j, L_{i+1}) \)
- A task \( M \) to be migrated is selected as
  - After the migration, the above still holds
  - The execution time of \( M \) is as large as possible

Sub-Optimality Bound

- We consider a ratio \( \alpha \), such that
  \[
  T(M_x, p_1) = \alpha \cdot (T(M_x, p_1) - T(M_x, p_1))
  \]
- Where
  - \( M_x \) is the task with the smallest execution time
  - \( T(M_x, p_1) \) is the execution time of task \( M \) on \( p_1 \) processors

Sub-Optimality Bound (cont’d)

- Assuming \( 0 < \alpha < 1 \) and \( 1 - \alpha \cdot \frac{P_i}{P_n} > 0 \) the accumulated execution time \( T_{i+1} \) of the scheduling algorithm has the following sub-optimality bound

\[
T_{i+1} \leq \frac{1 + \alpha}{1 - \alpha \cdot \frac{P_i}{P_n}} T_{\text{OPT}}
\]
Sub-Optimality Bound (cont’d)

• For a large numbers of tasks $\alpha \to 0$, which in turns means that

$$\frac{1 + \alpha}{1 - \alpha \cdot \frac{p_1}{P_o}} \to 1$$

Anticipated Distributed Task Scheduling for Grid Environments
Thomas Rauber and Gudula Rünger