Resource Scheduling Algorithm in Distributed Problem-Oriented Environments

ANASTASIA SHAMAKINA, LEONID SOKOLINSKY

SOUTH URAL STATE UNIVERSITY
THE 24-TH OF JUNE, 2014
Outline

1. Introduction
   - Purpose of the work
   - Clustering algorithms
   - Requirements for a job model

2. POS algorithm
   - Mathematical job model
   - Problem-oriented scheduling (POS) algorithm
   - Example

3. Integration with UNICORE
Purpose

Development of a resource scheduling algorithm in problem-oriented distributed computing environments
Clustering algorithms

1. Kim and Browne’s linear clustering algorithm (KB/L)
2. Sarkar’s algorithm
3. Dominant sequence clustering algorithm (DSC)
Requirements for a job model

• Representing a workflow in the form of a marked-up weighted directed acyclic graph (DAG)
• Clustering vertices
• Scaling individual tasks in a workflow
• Specifying the number of processor cores for computational nodes
• Describing known and new algorithms for clustering
Directed graph

A directed graph is called a quadruple

\[ G = \langle V, E, \text{init}, \text{fin} \rangle, \]

where

\( V \) is a vertices set;
\( E \) is an edges set;
\( \text{init}: E \to V \) is a function which determines an initial vertex of an edge;
\( \text{fin}: E \to V \) is a function which determines a final vertex of an edge.
Weighting function and layout function of a graph

Let us take directed graph \( G = \langle V, E, \text{init}, \text{fin} \rangle \).

- Weighting function \( \delta(e) \) of edge \( e \) determines the amount of transmitted data from a task associated with vertex \( \text{init}(e) \) to a task associated with vertex \( \text{fin}(e) \).

- A layout of graph \( G \) is function \( \gamma : V \rightarrow \mathbb{N}^2 \).
Job graph

A job graph is called a marked-up weighted directed acyclic graph,

\[ G = \langle V, E, \text{init}, \text{fin}, \delta, \gamma \rangle, \]

where

V is a set of vertices which correspond to tasks,

E is a set of edges which correspond to data flows.
Example: a job graph

The layout function,

$$\gamma(v) = (m_v, t_v),$$

where

$m_v$ is the maximum number of processor cores on which task $v$ has a nearly-linear speedup;

$t_v$ is the execution time of task $v$ on a single core.
Computer system

- *Compute node* $P$ is an ordered set of processor cores $\{c_0, \ldots, c_{q-1}\}$.
- *A computer system* is an ordered set of compute nodes,

$$\mathcal{P} = \{P_0, \ldots, P_{k-1}\}.$$
Clustering function

- *Clustering* is called single-valued transformation of vertices set $V$ of job graph $G$ on a set of computational nodes $\mathcal{P}$.

$$\omega: V \rightarrow \mathcal{P}$$

- *Cluster* $W_i$ is a set of all vertices that are displayed on computational node $P_i \in \mathcal{P}$.

- Example:
  $W_0 = \{v_1, v_2, v_8\}$ and $W_1 = \{v_3, v_4, v_5, v_6, v_7\}$.

<table>
<thead>
<tr>
<th>Vertex $v$</th>
<th>$\omega(v)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>$P_0$</td>
</tr>
<tr>
<td>$v_2$</td>
<td>$P_0$</td>
</tr>
<tr>
<td>$v_3$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$v_4$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$v_5$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$v_6$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$v_7$</td>
<td>$P_1$</td>
</tr>
<tr>
<td>$v_8$</td>
<td>$P_0$</td>
</tr>
</tbody>
</table>
Example: a clustered graph

• Clusters:

\[ W_0 = \{v_1, v_2, v_8\} \] and \[ W_1 = \{v_3, v_4, v_5, v_6, v_7\}. \]

• A communication cost is the time of data transmission along edge \( e \in E \).

• A communication cost function,

\[ \sigma(e) = \begin{cases} 0, & \text{if } \omega(\text{init}(e)) = \omega(\text{fin}(e)) \; \text{and} \; e \in W_0, \\ \delta(e), & \text{if } \omega(\text{init}(e)) \neq \omega(\text{fin}(e)). \end{cases} \]
Schedule

A schedule is called single-valued transformation,

\[ \xi : V \to \mathbb{Z}_{\geq 0} \times \mathbb{N}, \]

which maps casual vertex \( v \in V \) on a pair of numbers,

\[ \xi(v) = (\tau_v, j_v), \]

where \( \tau_v \) determines the launch time of task \( v \); \( j_v \) is a number of processor cores allocated to task \( v \).

<table>
<thead>
<tr>
<th>Vertex ( v )</th>
<th>( \xi(v) = (\tau_v, j_v) )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Launch time</td>
</tr>
<tr>
<td>( v_1 )</td>
<td>0</td>
</tr>
<tr>
<td>( v_2 )</td>
<td>1</td>
</tr>
<tr>
<td>( v_3 )</td>
<td>3</td>
</tr>
<tr>
<td>( v_4 )</td>
<td>7</td>
</tr>
<tr>
<td>( v_5 )</td>
<td>7</td>
</tr>
<tr>
<td>( v_6 )</td>
<td>7</td>
</tr>
<tr>
<td>( v_7 )</td>
<td>10</td>
</tr>
<tr>
<td>( v_8 )</td>
<td>15</td>
</tr>
</tbody>
</table>
Communication cost

Communication cost $\chi(v, j_v)$ of task $v$ on $j_v$-th processor cores is determined by the following formula:

$$\chi(v, j_v) = \begin{cases} 
  t_v/j_v, & \text{if } 1 \leq j \leq m_v; \\
  t_v/m_v, & \text{if } m_v < j_v.
\end{cases}$$

<table>
<thead>
<tr>
<th>Vertex $v$</th>
<th>$\chi(v, j_v)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>1</td>
</tr>
<tr>
<td>$v_2$</td>
<td>3</td>
</tr>
<tr>
<td>$v_3$</td>
<td>4</td>
</tr>
<tr>
<td>$v_4$</td>
<td>3</td>
</tr>
<tr>
<td>$v_5$</td>
<td>3</td>
</tr>
<tr>
<td>$v_6$</td>
<td>2</td>
</tr>
<tr>
<td>$v_7$</td>
<td>4</td>
</tr>
<tr>
<td>$v_8$</td>
<td>2</td>
</tr>
</tbody>
</table>
Scheduled graph

• Clustered graph $G$ with specified schedule $\xi$ is called a scheduled graph.
**Critical path**

- Let us take a simple path, \( y = (e_1, e_2, ..., e_n) \), in scheduled graph \( G \). Path cost \( u(y) \) has the following value:

\[
u(y) = \chi\left(\text{fin}(e_n), j_{\text{fin}(e_n)}\right) + \sum_{i=1}^{n} \left( \chi\left(\text{init}(e_i), j_{\text{init}(e_i)}\right) + \max\left(\sigma(e_i), \tau_{\text{fin}(e_i)} - s_{\text{init}(e_i)}\right) \right)\]

- \( Y \) is a set of all simple paths in scheduled graph \( G \). A simple path, \( \bar{y} \in Y \), is called a critical path, if it has a maximum value.
Gantt chart

The critical path equals 17.
Problem-oriented scheduling (POS) algorithm

1. Constructing an initial configuration of a graph,
   \[ G_0 = \langle V, E, init, fin, \delta, \gamma, \omega_0, \xi_0 \rangle. \]
2. \( i := 0. \)
3. Changing over from configuration
   \[ G_i = \langle V, E, init, fin, \delta, \gamma, \omega_i, \xi_i \rangle \]
   to configuration
   \[ G_{i+1} = \langle V, E, init, fin, \delta, \gamma, \omega_{i+1}, \xi_{i+1} \rangle, \]
   such as
   (simultaneously marking up the considered edges).
4. If there remain some unconsidered edges, then
   4.1. \( i := i + 1; \)
   4.2. transfer to step 3.
5. Stop.
Example: a POS algorithm

A job graph,
\[ G = \langle V, E, \text{init}, \text{fin}, \delta, \gamma \rangle \]

A computer system,
\[ \Psi = \{ P_0, P_1, ..., P_7 \}, \]
where
\[ P_0 = \{ c_{00}, c_{01}, c_{02}, c_{03} \}, \]
\[ P_1 = \{ c_{10}, c_{11}, c_{12}, c_{13} \}, \]
...

Canonical tiered-and-parallel form of a graph
Parallel time was increased!
Integration with UNICORE

STATISTICAL INFORMATION

BROKER

UNICORE
Conclusion

The following features have been developed:

- a mathematical job model for the description of known and new algorithms for clustering
- a resource scheduling algorithm for problem-oriented distributed computing environments