Steady State Scheduling

Optimal Scheduling Strategy

Heuristics Vs Deterministic

Introduction

- Allocating a large number of independent, equally-sized tasks to a heterogeneous computing platform is the primary interest of this research.
- We illustrate our approach by scheduling a general Master-Slave distributed computation.
- The specific aim is to construct an optimal steady state scheduler. In order to maintain optimality the scheduler must adapt itself to variations of the system.

Outline

Introduction
Model
Solution
Conclusion
References
An optimal steady state achieves the following condition: For each processor it is determined the number of compute tasks and the number of send/receive tasks, along each of its communication links. Thus, the (averaged) overall number of tasks processed at each time-step is maximum.

We model the computational platform with a node-weighted edge-weighted nonoriented graph. The resources can have different computation speeds and different communication rates.

Let $G = (V, E, w, c)$ be the platform graph model.

- $w_j$ are positive rational numbers ($0 < w_j \leq +\infty$)
- $c_{ij}$ are positive rational numbers ($0 < c_{ij} \leq +\infty$)

$c_{ij} = c_{ji}$, this means that link between $P_i$ and $P_j$ is bidirectional and symmetric. 

$w_j$ is the weight of the node $P_j$, this means that processor $P_j$ requires $w_j$ units of time to process one task.

$c_{ij}$ is the weight of the edge $e_{ij}$, it represents the time needed to communicate one task between the nodes $P_i$ and $P_j$ in either direction.
Scenario

- There are several scenarios for the operation of the processors. We concentrate on the full overlap, single-port model.
- Each node can simultaneously receive data from one neighbor, perform some computation, and send data to one neighbor.

Communication model

- If \( P_i \) sends a task to \( P_j \) at time-step \( w \), then
- \( P_j \) cannot start executing or sending this task before time-step \( w + \delta w \).
- \( P_j \) cannot initiate another receive operation before time-step \( w + \delta w \).
- \( P_j \) cannot initiate another send operation before time-step \( w + \delta w \).

Solution

The equations above constitute a LPP, whose objective function is the number of tasks consumed within one unit of time (throughput).

Maximize \( \sum_{i=1}^{n} \frac{\alpha_i}{w_i} \) Subject to

\[
\begin{align*}
\forall i, \quad & 0 \leq \alpha_i \leq 1 \\
\forall i, \forall j \in n(i), \quad & 0 \leq s_{ij} \leq 1 \\
\forall i, \forall j \in n(i), \quad & 0 \leq r_{ij} \leq 1 \\
\forall e_{ij} \in E, \quad & s_{ij} + r_{ij} \leq 1 \\
\forall i \neq m, \quad & \sum_{j \in n(i)} s_{ij} = \frac{\alpha_i}{w_i} \\
\forall i \in n(m), \quad & r_{m} = 0 \\
\end{align*}
\]

Master-Slave Scheduling Problem MSSG(\( G \))

Steady State Operation

Let \( n(i) \) denote the index set of the neighbors of processor \( P_i \).

During one time unit:

- \( \alpha_i, s_{ij}, r_{ij} \) are fraction of time spent:
- \( \alpha_i \) by \( P_i \) computing.
- \( s_{ij} \) by \( P_j \) sending tasks to \( P_i \) for \( j \in n(i) \).
- \( r_{ij} \) by \( P_i \) receiving tasks from \( P_j \) for \( j \in n(i) \).

\( 0 \leq \alpha_i, s_{ij}, r_{ij} \leq 1 \)

One-port model

\( \forall i \sum_{j \in n(i)} s_{ij} \leq 1 \quad \sum_{j \in n(i)} r_{ij} \leq 1 \)

Limited bandwidth.

\( \forall e_{ij} \in E, \quad s_{ij} + r_{ij} \leq 1 \)

Conservation laws.

\( \forall i \neq m, \quad \sum_{j \in n(i)} s_{ij} = \frac{\alpha_i}{w_i} + \sum_{j \in n(i)} r_{ij} \)

Master \( P_m \).

\( \forall i \in n(m), \quad r_{m} = 0 \)

Solution

When we have the optimal solution, we take the least common multiple of the denominators and, thus, we derive an integer period \( T \) for the steady state operation.

\( \alpha_i T = \) the time spend for \( P_i \) computing
\( \frac{\alpha_i T}{w_i} = \) integer number of task computing \( P_i \)

\( s_{ij} T = \) integer number of task \( P_j \) sending to \( P_i \)
\( \frac{s_{ij} T}{c_{ij}} = \) integer number of task \( P_j \) sending to \( P_i \)

\( r_{ij} T = \) integer number of task \( P_i \) receiving from \( P_j \)
\( \frac{r_{ij} T}{c_{ij}} = \) integer number of task \( P_i \) receiving from \( P_j \)
Solution

• The previous values characterize the solution, but do not describe the actual order of task execution at every time step, during a period of the scheduler.

• To generate a schedule we first create a bipartite graph out of the original platform graph.

• Then, we decompose the bipartite graph into a set of subgraphs (matching) where a node (sender or receiver) is occupied by at most one communication task.

• This decomposition of the graph is reduced to edge coloring (The weighted edge coloring algorithm)

Example 1

If we feed the values $w_i$ and $c_i$ into the linear program, and compute the solution using a tool such as LINDO we obtain the optimal throughput $n_{task}(G) = 7/4$. This means that the whole platform is equivalent to a single processor with processing $w = 4/7$.

In this example we get:

$\alpha_1 = 1, \alpha_2 = 1, \alpha_3 = 1, \alpha_4 = 1, s_{12} = 1/2, s_{13} = 1/2, s_{23} = 3/4, s_{34} = 1/4, t_{13} = 1/2, t_{14} = 1/2, t_{24} = 3/4, t_{34} = 1/4$

This makes a total of $12 + 3 + 4 + 2 = 21$ tasks every 12 time steps, and then we have $n_{task}(G) = 21/12 = 7/4$, according to the solution.
To generate the schedule we must use an algorithm of weight edge coloring for bipartite graphs.

Construction of the schedule

A bipartite graph of the problem

Algorithm of weight edge coloring

• Find and Remove successively matching until there are no remaining values on the edge.

\[
M' = \text{Count} = \frac{T_{\text{task}}}{w_i} \\
S_p = S_{\text{partition}} / \text{Count} \\
c_{ij} = \frac{S_{\text{partition}}}{\text{Band}_{ij}}
\]

Steady state

Define \( w_i \) and \( c_{ij} \) (systematic)

To define \( w_i \) and \( c_{ij} \):
- Let \( t_i \) be the amount of time (measured in microseconds) in a time slice of processor \( P_i \).
- Let \( G_i \) be the speed rate (Mips).
- \( T_n = \text{LCM}(t_{\text{cpu}}) \) where \( P \) the set of node in the Platform Graph.
- \( G = F(G_{\text{cpu}}) \) where \( F \) is Min, Max or Average.
- Let \( T_{\text{task}} \) be task computation time (in some processor).
- Let \( S_{\text{task}} \) be the size of the data measured in MB.
- \( \text{Band}_{ij} \) be the communication bandwidth between nodes \( i \) and \( j \).

\[
\begin{align*}
\frac{w_i}{G_i} &= \frac{T_n}{G_i} \\
\text{Count} &= \frac{T_{\text{task}}}{w_i} \\
S_{\text{partition}} &= \frac{S_{\text{task}}}{\text{Count}} \\
c_{ij} &= \frac{S_{\text{partition}}}{\text{Band}_{ij}}
\end{align*}
\]
Example 2

<table>
<thead>
<tr>
<th>$G_i$</th>
<th>$t_i$</th>
<th>$T_{mk}$</th>
<th>Band$_{ij}$</th>
<th>$S_{mk}$=12.5MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6000 Mbps</td>
<td>20 ms</td>
<td>2s</td>
<td>Band$_{ij}$=50 Mbps</td>
</tr>
<tr>
<td>2</td>
<td>2000 Mbps</td>
<td>40 ms</td>
<td>2s</td>
<td>Band$_{ij}$=12.5 Mbps</td>
</tr>
<tr>
<td>3</td>
<td>1500 Mbps</td>
<td>40 ms</td>
<td>2s</td>
<td>Band$_{ij}$=50 Mbps</td>
</tr>
<tr>
<td>4</td>
<td>1500 Mbps</td>
<td>50 ms</td>
<td>2s</td>
<td>Band$_{ij}$=12.5 Mbps</td>
</tr>
</tbody>
</table>

Example 2 (Solution)

Max $11/6$

• We have shown how to determine the best steady state scheduling strategy for a general interconnection graph, using a linear programming approach.

• This method supposes that we know a priori some information about the system; this can be a disadvantage, but this information in many cases is known.

• We construct a reliable measure that satisfy the requirements of the LP Model already designed. This is a robust measure to $w_j$ and $c_{ij}$.
In order to cope with system variations, monitor the system and update the initial parameters. If necessary, recalculate \((c_{ij}, w_{ij})\) and generate a new scheduler for upcoming periods.

Solve the MSSG to obtain the characterization of the solution. Generate the schedule by using a bipartite graph and edge coloring algorithm.

Estimate the initial parameters of the system through experimentation or technical specifications. This information must be translated into reliable measurements for providing the initial values of the LP model already designed \((c_{ij}, w_{ij})\).

- **Future Work**
  - **Adaptive Schedule**
    - **Phase 1**
    - **Phase 2**
    - **Phase 3**

- **References**