Scheduling Algorithms

This document provides pseudo codes representing the implemented algorithms, that we consider in our simulation study. In Table 1, we explain the notations used in the description of these algorithms.

Notation	Description
$l(m_i)$	Remaining load on core m_i
L	Maximum load caused by the new job
d_{lim}	Deadline threshold
$t_{i,j}$	Total idle time caused by allocating multi-core job j to core m_i

Table 1: Definition of notations used in the description of algorithms

Algorithm 1 Greedy Balanced

initialize $l(m_h) = 0$ for all cores
for the next job j **do**
    update $l(m_h)$ for all cores
    $L = l(m_{c_j}) + p_j$
    if $L \leq d_j - r_j$ **then**
        accept j
        allocate j to cores m_1 to m_{c_j}
        update $l(m_h)$ to L for $1 \leq h \leq c_j$
        sort all cores in increasing order of $l(m_h)$
    else
        reject j

Algorithm 2 Greedy BestFit

initialize $l(m_h) = 0$ for all cores
for the next job j **do**
    update $l(m_h)$ for all cores
    if $l(m_{c_j}) + p_j + r_j \leq d_j$ **then**
        accept j
        m_i: last core with $d_j - r_j \geq l(m_i) + p_j$
        $L = l(m_i) + p_j$
        allocate j to cores m_{i-c_j+1} to m_i
        update $l(m_h)$ to L for $i - c_j + 1 \leq h \leq i$
        sort all cores in increasing order of $l(m_h)$
    else
        reject j

Algorithm 3 Threshold

initialize $l(m_h) = 0$ for all cores and $d_{lim} = 0$
for the next job j **do**
    update $l(m_h)$ for all cores
    determine d_{lim}
    if $d_j < d_{lim}$ **then**
        reject j
    else
        accept j
        use the allocation part of Algorithm 2

Algorithm 4 Greedy MinIdle

initialize $l(m_h) = 0$ for all cores
for the next job j **do**
    update $l(m_h)$ for all cores
    $L = l(m_{c_j}) + p_j$
    if $L \leq d_j - r_j$ **then**
        accept j
        for all m_i in $c_j \leq i \leq m$ **do**
            determine $L = l(m_i) + p_j$
            if $L \leq d_j - r_j$ **then**
                initialize $t_{i,j} = 0$
                for all m_a in $i - c_j + 1 \leq a \leq i$ **do**
                $t_{i,j} = t_{i,j} + l(m_i) - l(m_a)$
        determine m_i with $\min_i\{t_{i,j}\}$
        allocate j to cores m_{i-c_j+1} to m_i
        update $l(m_a)$ to L for $i - c_j + 1 \leq a \leq i$
        sort all cores in increasing order of $l(m_h)$
    else
        reject j

Algorithm 5 Greedy Balanced BackFill

initialize $l(m_h) = 0$ for all cores
for the next job j **do**
    if backfilling is possible **then**
        allocate j via backfilling
    else
        apply Algorithm 1

Algorithm 6 Greedy BestFit BackFill

initialize $l(m_h) = 0$ for all cores
for the next job j **do**
    if backfilling is possible **then**
        allocate j via backfilling
    else
        apply Algorithm 2