Figure 1: Parallel Partition

if $g < 2$ **then**
serialPartition A
else
for $i \in \{0, 1, \ldots, s-1\}$ **do**
$X[i] \leftarrow$ a random integer from $[0, g-1]$
end for
– We implement this parallel for-loop with a sequence of recursive spawns, and which facilitates computing v_{min}, v_{max} without storing v_ys
for all $y \in \{0, 1, \ldots, g-1\}$ in parallel **do**
– Now we perform a serial partition on U_y
– Initialize ALowIdx to be the index of the first element in U_y
ALowIdx $\leftarrow ((X[0] + y) \bmod g) \cdot b$
– Initialize AHighIdx to be the index of the last element in U_y
AHighIdx $\leftarrow n - g \cdot b + ((X[s-1] + y) \bmod g) \cdot b + b - 1$
while ALowIdx $<$ AHighIdx **do**
while A[ALowIdx] $\leq$ pivotValue **do**
ALowIdx $\leftarrow$ ALowIdx+1
if ALowIdx on block boundary **then**
– We perform a block increment
$i \leftarrow$ # of block increments so far (including this one)
– Increase ALowIdx to start of block i of G_y
ALowIdx $\leftarrow ((X[i] + y) \bmod g) \cdot b + i \cdot b \cdot g$
end if
end while
while A[high] $>$ pivotValue **do**
AHighIdx $\leftarrow$ AHighIdx-1
if AHighIdx on block boundary **then**
– We perform a block decrement
$i \leftarrow$ # of block decrements so far (including this one)
– Decrease AHighIdx to end of block $s-1-i$ of G_y
AHighIdx $\leftarrow ((X[s-1-i]+y) \bmod g) \cdot b + i \cdot b \cdot g + b - 1$
end if
end while
Swap A[ALowIdx] and A[AHighIdx]
end while
end for
Recurse on $A[v_{min}], \ldots, A[v_{max} - 1]$
end if