Parallel Programming with MATLAB

Case Study: PI

Christian Terboven <terboven@itc.rwth-aachen.de>
29.01.2015 / Aachen, DE
Stand: 10.12.2013
Version 2.3
Case Study: PI

Calculate PI by numerical integration

\[ \Pi = \int_{0}^{1} \frac{4}{1 + x^2} \, dx \]
V01: Naive Parallelization

```
spmd  % do the following on every worker
    % calculate Pi locally
    fSum = 0;
    fH = 1 / n;
    for i = labindex-1: numlabs: n-1
        fX = fH * (i + 0.5); fX = 4 / (1 + fX^2); fSum = fSum + fX;
    end
    localPi = (fSum / n);

    % Gather all results on worker with labindex 1 in localPi
    if (labindex ~= 1)
        labSend(localPi, 1) % all other workers send their data to labindex 1
    else
    % worker with labindex 1 collects all partial results
        for i=2: numlabs
            localPi = localPi + labReceive(i);
        end
    end
end % back to sequential
```
Speedup: v01 (for n = 15,000,000)
spmd % do the following on every worker
% calculate Pi locally
fH = 1 / n;
vec = [labindex-1: numlabs: n-1];
fX = (vec + 0.5) * fH;
fX = 4 ./ (1 + fX.^2);
sumvec = cumsum(fX);
fSum = sumvec(size(sumvec, 2));
localPi = fSum / n;
% Gather all results on worker with labindex 1 in localPi
if (labindex ~= 1)
    labSend(localPi, 1) % all other workers send their data to labindex 1
else
    % worker with labindex 1 collects all partial results
    for i=2: numlabs
        localPi = localPi + labReceive(i);
    end
end
end % back to sequential
Speedup: v01, v02  (for n = 15,000,000)
Speedup: v02  (for n = 15,000,000)
V03: Saving Memory (reusing vec)

spmd
  % do the following on every worker
  
  % calculate Pi locally
  fH = 1 / n;
  vec = [labindex-1: numlabs: n-1];
  vec = (vec + 0.5) * fH;
  vec = 4 ./ (1 + vec.^2);
  sumvec = cumsum(vec);
  fSum = sumvec(size(sumvec, 2));
  localPi = fSum / n;

  % Gather all results on worker with labindex 1 in localPi
  if (labindex ~= 1)
    labSend(localPi, 1) % all other workers send their data to labindex 1
  else
    % worker with labindex 1 collects all partial results
    for i=2: numlabs
      localPi = localPi + labReceive(i);
    end
  end
end

end % back to sequential
Speedup: v01, v02, v03 (for n = 15.000.000)

Runtime Comparison

- Runtime [s]
- # Worker

- v02
- v03
Thank you for your attention.