BITS Pilani
Pilani Campus

# Performance Analysis

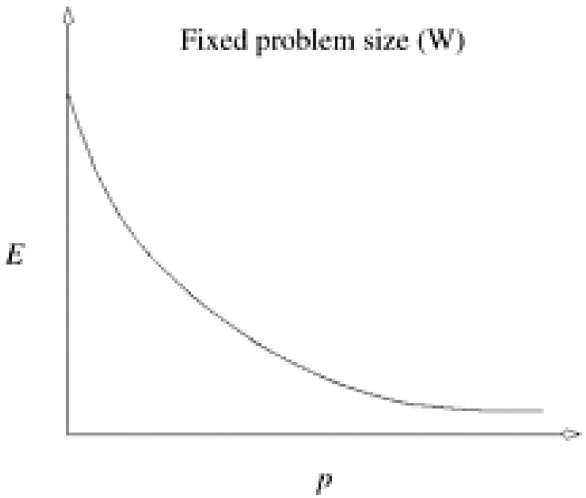
K Hari Babu
Department of Computer Science & Information Systems

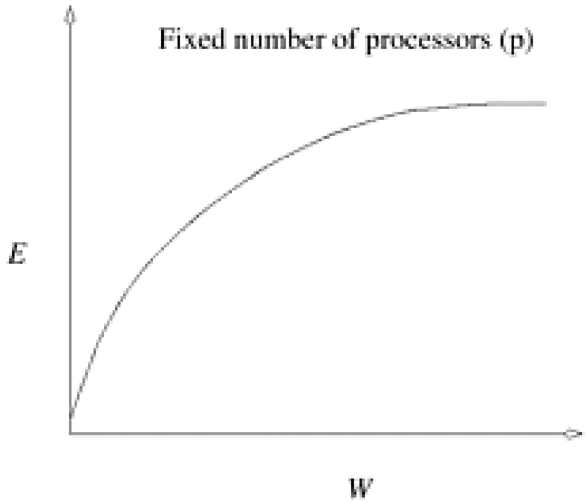
# Scalability of Parallel Systems T1 5.4

# Efficiency vs Problem Size and Processors

BITS Pilani

- For a given problem size, as we increase the number of processing elements, the overall efficiency of the parallel system goes down
  - This phenomenon is common to all parallel systems
- In many cases, the efficiency of a parallel system increases if the problem size is increased while keeping the number of processing elements constant
  - not common to all parallel systems



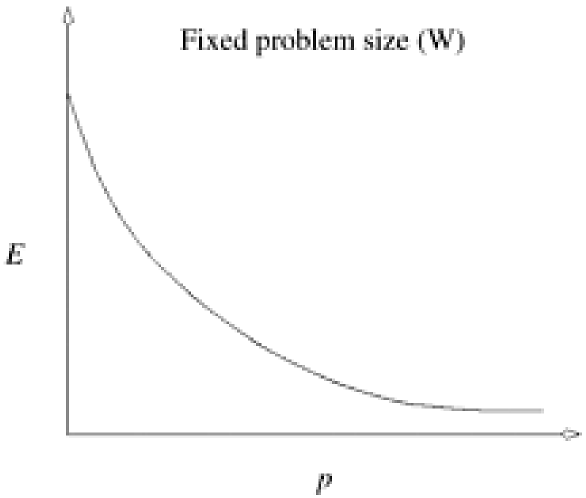



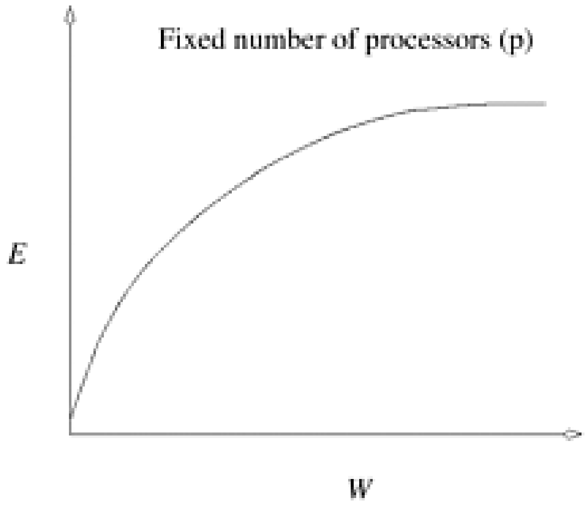
# Efficiency vs Problem Size and Processors

BITS Pilani

- If increasing the number of processors reduces efficiency, and increasing the problem size increases efficiency, we should be able to keep efficiency constant by increasing both simultaneously

- A **scalable parallel system** is one in which the efficiency can be kept constant as the number of processing elements is increased, provided that the problem size is also increased



# The Isoefficiency Function

- A natural question at this point is:
  - At what rate should we increase the problem size with respect to the number of processors to keep the efficiency fixed?

# The Isoefficiency Metric

- $T_1$=time taken by an algorithm to execute on a single processor
  - This is taken as problem size (W).
  - Usually input size is taken as problem size. That varies with each algorithm. To make a uniform measurement, number of computations done by a sequential algorithm is taken as problem size.
- $T_p$=time taken by an algorithm to execute on p processors
- $T_0$=total time spent by all processors doing work that is not done by sequential execution
  - Is a function of W and number of processors = $T_0(W,p)$.
- $p * Tp = T_1 + T_0$
- $T_p = \frac{T_1 + T_0}{p}$
- Speedup
  - $S = \frac{T_1}{T_p} = \frac{p * T_1}{T_1 + T_0}$

# The Isoefficiency Metric

BITS Pilani

- Speedup
  - $S = \frac{T_1}{T_p} = \frac{p * T_1}{T_1 + T_0}$
- Iso-Efficiency
  - $E = \frac{S}{p} = \frac{T_1}{T_1 + T_0} = \frac{1}{1 + \frac{T_0}{T_1}}$
    - Efficiency depends entirely on the ratio between the parallel overhead and the serial execution time

# The Isoefficiency Metric

BITS Pilani

- Iso-Efficiency
  - $E = \frac{S}{p} = \frac{T_1}{T_1 + T_0} = \frac{1}{1 + \frac{T_0}{T_1}}$
  - $E = \frac{1}{1 + \frac{T_0(W,p)}{W}}$
    - If the problem size W is constant while p increases, then the efficiency decreases because the total overhead $T_0$ increases with p
    - If W increases while p is constant, then, for scalable parallel systems, the efficiency increases because $T_0$ grows slower than W
  - We can maintain the efficiency for these parallel systems at a desired value (between 0 and 1) by increasing p, provided W also increases

# The Isoefficiency Metric

BITS Pilani

- Iso-Efficiency
  - $E = \frac{1}{1+\frac{T_0(W,p)}{W}}$
    - For different parallel systems, we must increase *W* at different rates with respect to *p* to maintain a fixed efficiency
  - Suppose W might need to grow as an exponential function of p
    - Such systems are poorly scalable: It is difficult to obtain good speedups for a large number of processors on such systems unless the problem size is enormous
  - Suppose *W* needs to grow only linearly with respect to *p*
    - Such systems are highly scalable: Its speedups increase linearly with respect to the number of processors for problem sizes increasing at reasonable rates

# Isoefficiency Function



- For scalable parallel systems, we can maintain efficiency at a desired value ($0 < E < 1$) if $T_0/W$ is constant
  - $E = \frac{1}{1+\frac{T_0(W,p)}{W}}$
  - $\frac{T_0(W,p)}{W} = \frac{1}{E} - 1$
  - $W = \left(\frac{E}{1-E}\right) * T_0(W,p)$
    - If $K = E/(1 - E)$ is a constant that depends on the efficiency, then we can reduce the last equation to $W = KT_0$
  - This is the system's isoefficiency function
  - Highly scalable systems have small isoefficiency function

# Sources of Parallel Overhead

BITS Pilani

- Inter processor communication:
  - Increase data locality to minimize communication
- Load imbalance:
  - Distribution of work(load) is not uniform. Inherent parallelism of the algorithm is not sufficient
- Extra computation:
  - Modify the best sequential algorithm may result in extra computation. Extra computation may be done to avoid communication

# Example - Parallel Summation

- Adding n numbers on a sequential machine
  - Problem size = W = n
- Parallel algorithm
  - P processors
  - Each allocated n/p numbers
  - Processor locally adds n/p numbers in $\Theta\left(\frac{n}{p}\right)$time.
  - Each processors communicates its sum to its neighbor
    - There will be log(p) communications and additions, assuming each takes 1 unit of time
    - Total parallel execution time is $T_p = \frac{n}{p} + 2\log(p)$
    - Total overhead $T_0 = 2 * p * \log(p)$
    - Speedup: $S = \frac{T_1}{T_p} = \frac{n}{\frac{n}{p}+(2\ logp)}$
    - Efficiency: $E = \frac{S}{p} = \frac{n}{n+(2p\ logp)}$

# Example - Parallel Summation

- Isoefficiency function
  - W= $KT_0$
  - $W = K * 2 * p * log(p)$
  - So iso-efficiency function for parallel reduction is $\Theta(p \log p)$
- If the number of processors is increased from p to p’,
  - the problem size (in this case n) must increase by a factor of $(p' \log p')/(p \log p)$ to maintain the same efficiency
  - If p=10, p’=11, n=1000, then what should be the n’ to keep the efficiency same?
    - Factor=26.37/23=1.14
    - N’=1000*1.14=1140

# References


- Chapter 5 of T1. Ananth Grama, Anshul Gupta, George Karypis & Vipin Kumar Introduction to Parallel Computing, Second Edition, Pearson Education, First Indian Reprint 2004.
- https://www.cse.wustl.edu/~roger/569M.s09/Isoefficiency.pdf


BITS Pilani

# Thank You