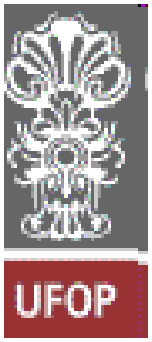


PCC104
Projeto e Análise de Algoritmos

Joubert de Castro Lima
joubertlima@gmail.com
Professor Adjunto – DECOM

UFOP 2010/1

Saindo um pouco da teoria….

Java Threads

Material obtido do site:
http://java.sun.com/docs/books/tutorial/essential/concurrency/index.html

Threads

- Uma porção de código (uma task, por exemplo!!!) que pode ser executada independentemente de outras Threads

- Um processo possui pelo menos uma Thread

Most implementations of the Java virtual machine run as a single process.

Threads

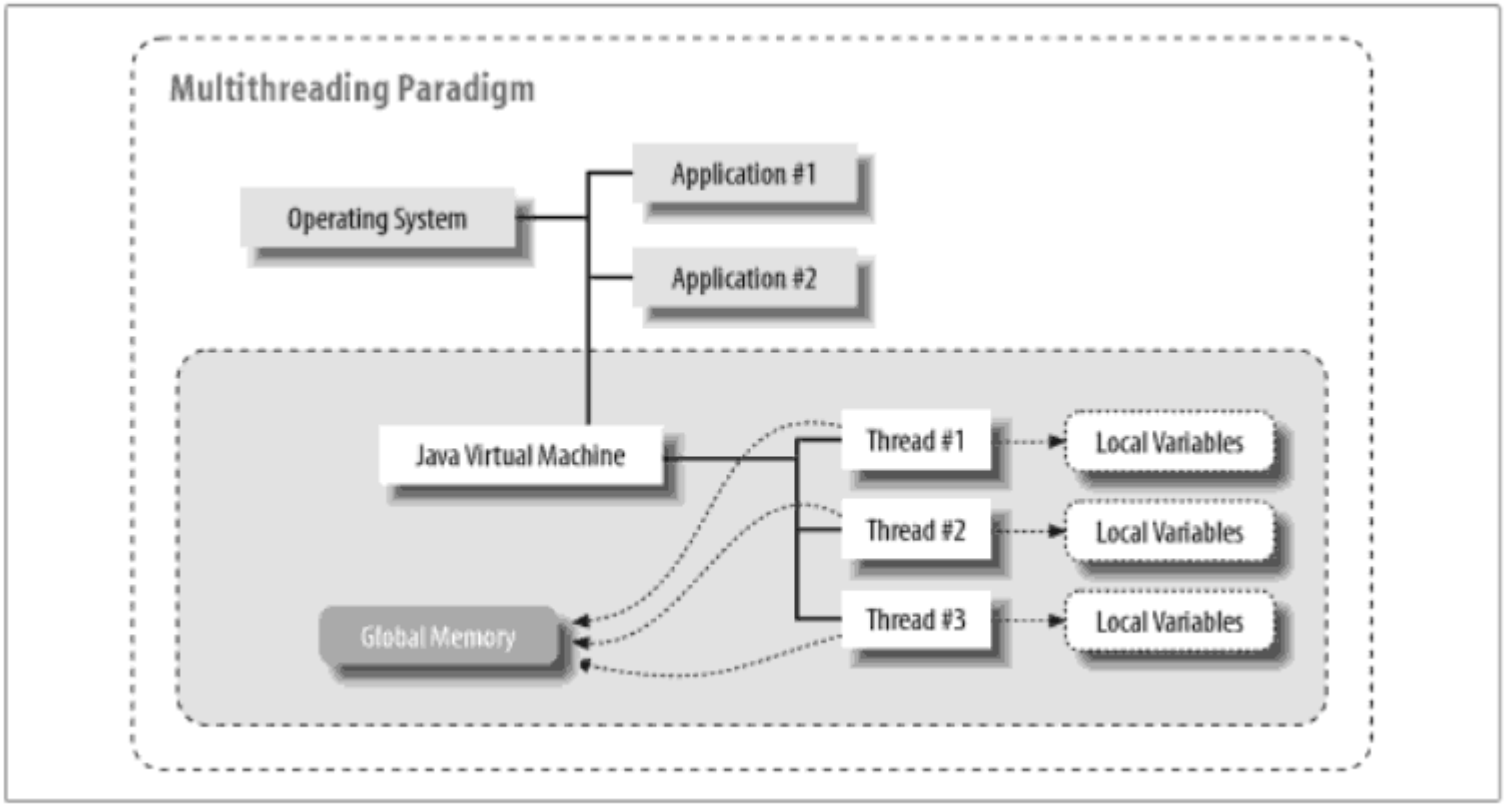


Figure 2-2. Threads in a multithreaded environment

Retirada do livro Java Threads
Scott Oaks

Iniciando Threads em Java

- Usando a interface Runnable

```
public class HelloRunnable implements Runnable {

    public void run() {
        System.out.println("Hello from a thread!");
    }

    public static void main(String args[]) {
        (new Thread(new HelloRunnable())).start();
    }

}
```

Iniciando Threads em Java

- Usando a classe Thread

```
public class HelloThread extends Thread {

    public void run() {
        System.out.println("Hello from a thread!");
    }

    public static void main(String args[]) {
        (new HelloThread()).start();
    }

}
```

Thread ou Runnable

- A classe Thread é uma implementação de Runnable, portanto Thread é uma sub-classe

- Em Java não temos herança múltipla, portanto a classe Runnable é necessária

Class minhaClasse extends minhaSuperClasse implements Runnable

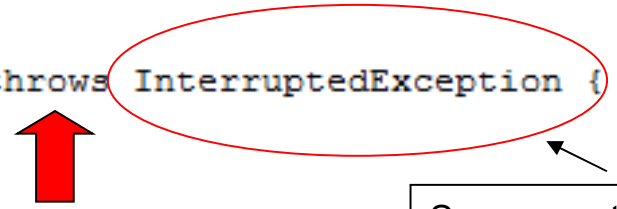
minhaClasse é uma thread
minhaClasse é uma minhaSuperClasse

Parando a execução

Thread.sleep causes the current thread to suspend execution for a specified period.

This is an efficient means of making processor time available to the other threads of an application or other applications that might be running on a computer system.

```
public class SleepMessages {
    public static void main(String args[]) throws InterruptedException {
        String importantInfo[] = {
            "Mares eat oats",
            "Does eat oats",
            "Little lambs eat ivy",
            "A kid will eat ivy too"
        };

        for (int i = 0; i < importantInfo.length; i++) {
            //Pause for 4 seconds
            Thread.sleep(4000);
            //Print a message
            System.out.println(importantInfo[i]);
        }
    }
}
```



Se uma outra thread interromper esta thread a exceção é lançada

Se SleepMessages extends Thread, basta colocar sleep(4000);

Joins

The join method allows one thread to wait for the completion of another.

If t is a Thread object whose thread is currently executing, t.join(); causes the

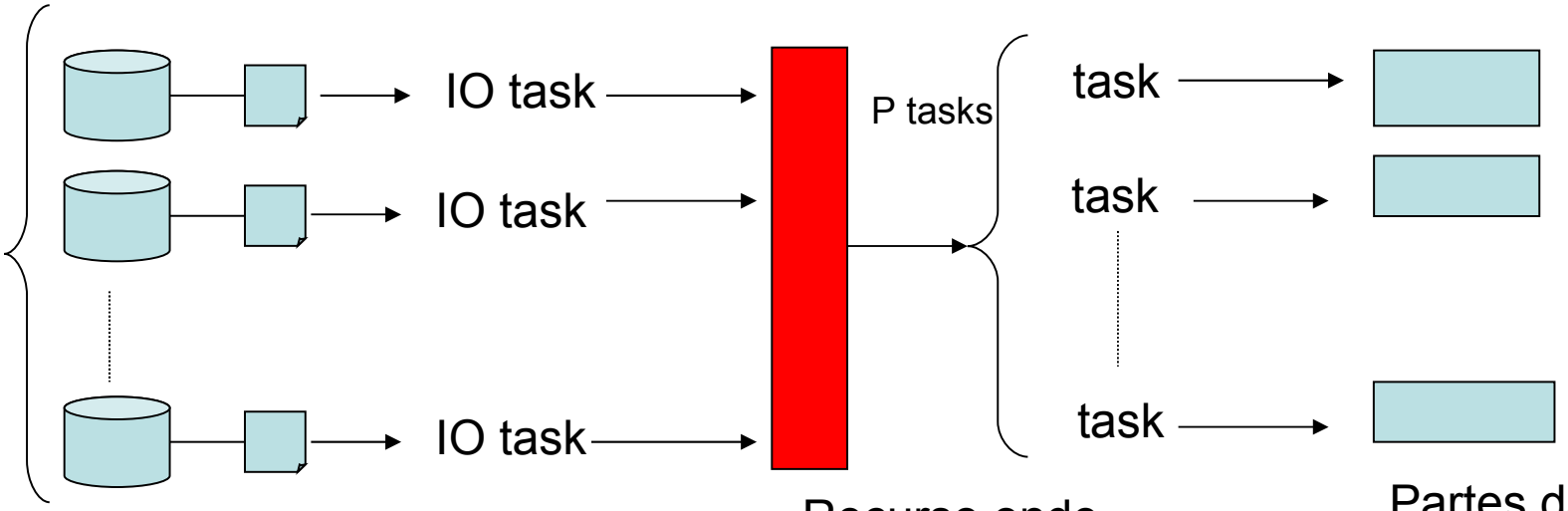
current thread to pause execution until t's thread terminates.

```
public class FiboII extends Thread{
    private int x;
    public int answer;

    public FiboII(int x) {
        this.x = x;
    }

    public void run() {
        if( x < 2 )
            answer = 1;
        else {
            try {
                FiboII f1 = new FiboII(x-1);
                FiboII f2 = new FiboII(x-2);
                f1.start();
                f2.start();
                f1.join();
                f2.join();
                answer = f1.answer + f2.answer;
            }
            catch(InterruptedException ex) { }
        }
    }

}
```

Sincronização, Wait e notify

EXEMPLO

PRODUTOR - CONSUMIDOR



D discos
D arquivos

Cada arquivo com n/D linhas

Recurso onde as linhas são armazenadas

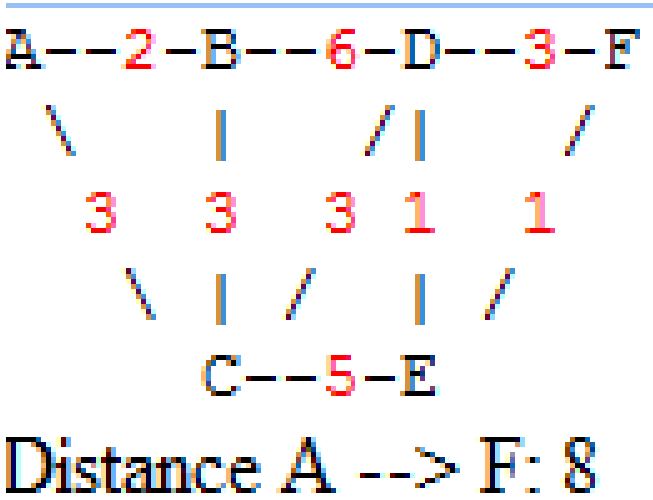
Partes da matriz transposta

USAREMOS CÓDIGO para explicar estes conceitos!!!

TP

- USAR MÁQUINAS DO NOVO LAB DO DECOM (3 CORES) OU AGENDAR HORÁRIO NA MÁQUINA NO TERRALAB

- Tarefa: fazer um algoritmo paralelo para o problema all-pairs shortest path



$A \rightarrow B, A \rightarrow C, A \rightarrow D, A \rightarrow E, A \rightarrow F$

$B \rightarrow C, B \rightarrow D, B \rightarrow E, B \rightarrow F, B \rightarrow A$

$\vdots$

calcular:

$$E \qquad \frac{S}{p}.$$

$$C = p \times T_P$$

speedup

escalabiliadade

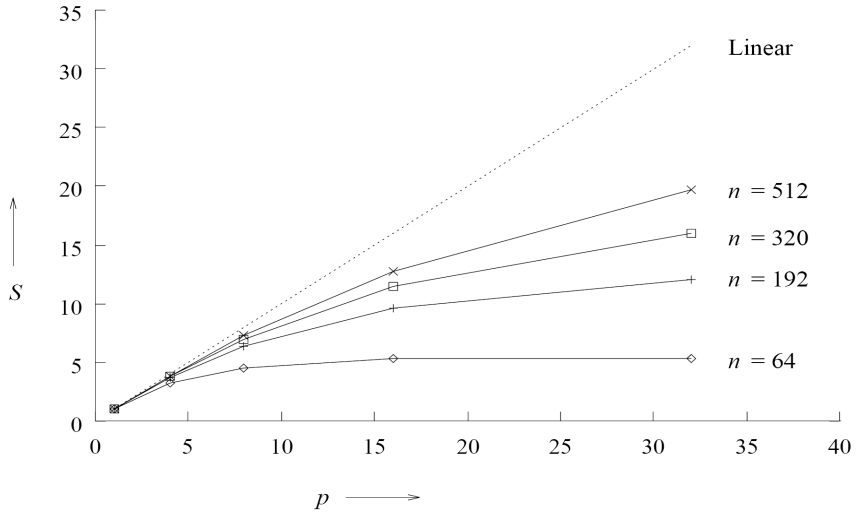


Figure 5.8 Speedup versus the number of processing elements for adding a list of numbers.

TESTE SEU CÓDIGO FIXANDO O INPUT E O NÚMERO DE THREADS
Conforme ilustrado abaixo!!!!

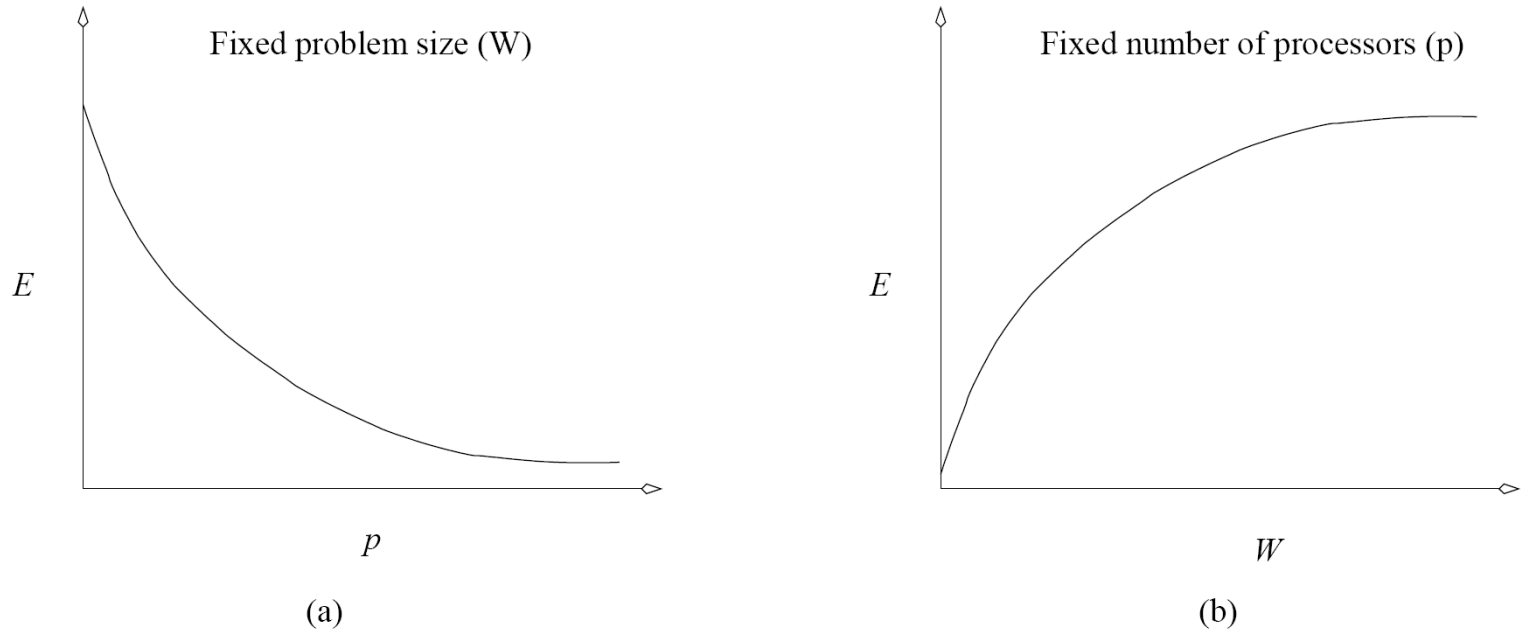


Figure 5.9 Variation of efficiency: (a) as the number of processing elements is increased for a given problem size; and (b) as the problem size is increased for a given number of processing elements. The phenomenon illustrated in graph (b) is not common to all parallel systems.