**Computação Paralela**

Laboratório corre independente da teoria. Mario + Calebe.

Eles só abordam o ambiente Unix (Linux).

-------------------------------------------------------

**Objetivo**: reduzir tempo de processamento de determinada tarefa.

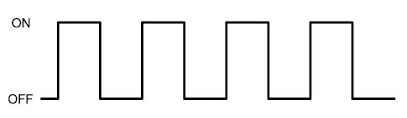
**Como?** Dividindo a tarefa em tarefas menores e submetendo-as simultaneamente a processadores distintos.

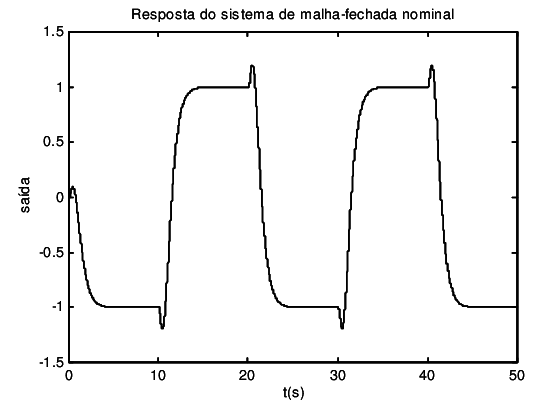


As tarefas tem que ser independentes (o início de uma não depende de resultados da outra).

**Analogia:** construção do prédio - um andar só pode ser construído após o de baixo estar construído -> não dá para paralelizar; pintura dos apartamentos com todo o prédio já construído -> dá para paralelizar.

**Histórico:** as técnicas de paralelismo não são tão recentes, mas ganharam interesse em virtude das limitações de hardware - atrasos e aquecimento.

Clock teórico:

Clock real: 

O programa precisa ser escrito com instruções específicas de paralelismo, caso contrário rodará em sequência, independentemente da quantidade de processadores disponíveis.

**Técnicas:**

* Processamento multi-core: Threads, Pthreads, Fork, OpenMP.
* Processamento distribuído: MPI (message passing interface).

As Pthreads vêm do padrão POSIX (Portable Operating System Interface) do IEEE. O objetivo do POSIX é padronizar as chamadas (programáticas) aos SOs para possibilitar a portabilidade dos códigos-fonte. O X foi incorporado ao nome por ter sido baseado no modelo Unix e também para dar maior sonoridade e deixar o nome mais fácil de lembrar.

**Situações em que utilizar o paralelismo:**

* Diversas tarefas independentes
  + Para paralelizar, as tarefas têm que ser independentes.
* Uma tarefa sobre um conjunto de dados.
  + Dados que não precisam ser processados sequencialmente.

**Linguagem C:**

Foi criada entre 1969-1973 nos laboratórios da AT&T Bell por Dennis Ritchie especificamente para desenvolvimento de um novo sistema operacional que já tinha uma primeira versão em Assembly, o Unix.

1978 o livro "The C Programming Language" divulgou a linguagem. A partir daí começou-se a utilizar mais a C em detrimento da Basic para programação de microcomputadores.

1983: O ANSI cria um comitê para especificar a linguagem.

1989: A especificação ANSI X3.159-1989 é concluída. O nome informal dela é C89 e também é conhecida por ANSI C.

1990: O ISO (International Organization for Standardization) assume a especificação de linguagem e cria um grupo de trabalho (working group) chamado WG14 [2] para cuidar da especificação do C, que é renomeada para ISO/IEC 9899:1990, informalmente conhecida como C90. Não houve mudanças na linguagem. Esta especificação se mantém por vários anos e define a linguagem C que “todos conhecem”.

1999: O WG14 publica a especificação ISO/IEC 9899:1999, informalmente chamada de C99. Esta já traz várias novidades que serão tratadas mais à frente neste artigo.

2011: O WG14 publica a ISO/IEC 9899:2011, conhecida como C11.

2018: C18 - corrigiu defeitos na C11 sem introduzir novos recursos.

**Linguagem C++:**

1983: Bjarne Stroustrup desenvolveu o C++ (originalmente com o nome C with Classes, que significa C com classes em português) no Bell Labs como um adicional à linguagem C.

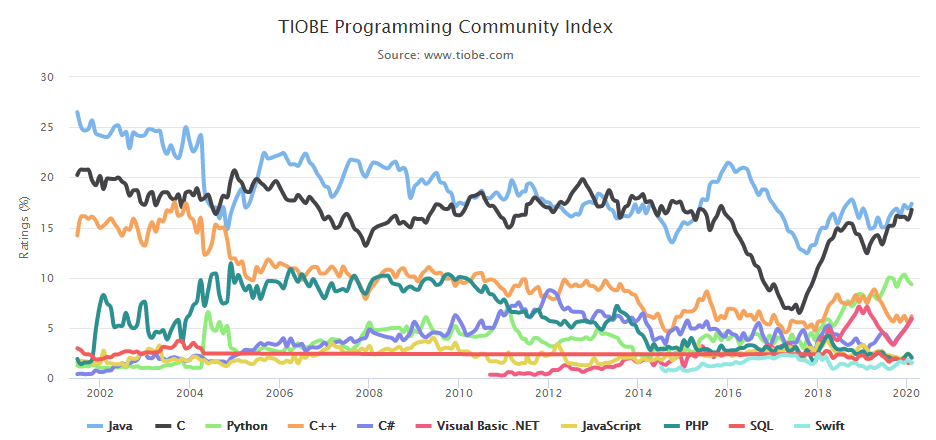
1986: Bjarne Stroustrup publica o livro "The C++ Programming Language", o que divulgou a linguagem.

1997: ANSI publica o Draft Standard da linguagem C++.

2005: relatório técnico (tr1) contendo extensões para a linguagem C++ e biblioteca padrão.

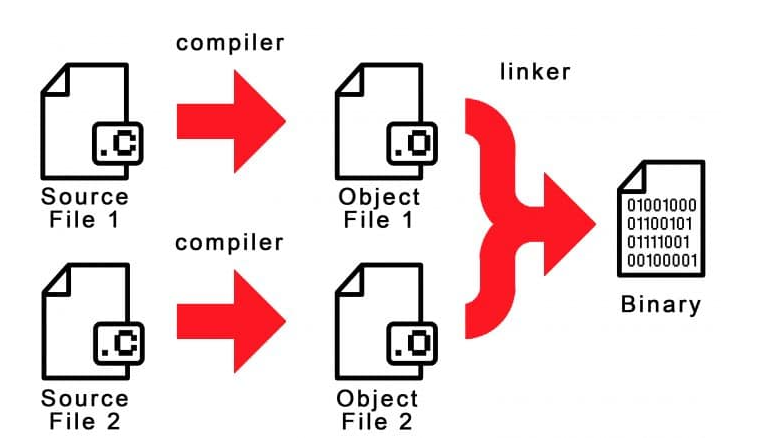
* C++98
* C++11
* C++14
* C++17
* C++20 (em andamento)

2012: foi fundada a The Standard C++ Foundation (isocpp.org), organização sem fins lucrativos cuja finalidade é apoiar a comunidade de desenvolvedores de software C++ e promover a compreensão e o uso do Standard C++ moderno em todos os compiladores e plataformas.



**Compiladores:**

A dita compilação de um programa para geração do arquivo executável envolve duas etapas - a compilação propriamente dita que transforma os arquivos fonte em arquivos com um formato binário. Esses arquivos não são executáveis, pois estão separados, e cada um não possui ainda o cabeçalho necessário para que o SO o reconheça como executável. Esses binários individuais precisam ser montados pelo *link editor* ou *linker* em um arquivo único e com os necessários atributos para ser executado.

****

**GCC** - Originalmente criado como *GNU C Compiler* e posteriormente designado *GNU Compiler Collection*.

**G++** - Para compilar os programas C++.

**Exemplo**: programa com 3 arquivos: main.cpp, classe\_a.h, classe\_a.cpp.

Compilar e editar:

**g++ -o prog main.cpp classe\_a.cpp**

* **-o** = indica o nome do programa executável
* **prog** é o nome do programa a ser gerado (a extensão .exe é acrescentada automaticamente).
* em seguida vem a relação de arquivos cpp.

Se desejarmos apenas compilar:

**g++ -c main.cpp classe\_a.cpp**

* **-c** indica para se executada apenas a parte de compilação
* por padrão os arquivos compilados têm a extensão .o

Editar os arquivos compilados

**g++ -o prog main.o classe\_a.o**

**Make:**