Curso de Fortran

básico ao intermediário

Átila Saraiva Quintela Soares

Historia do Fortran

Desenvolvido pela IBM em 1950 para aplicações para a ciência e engenharia.

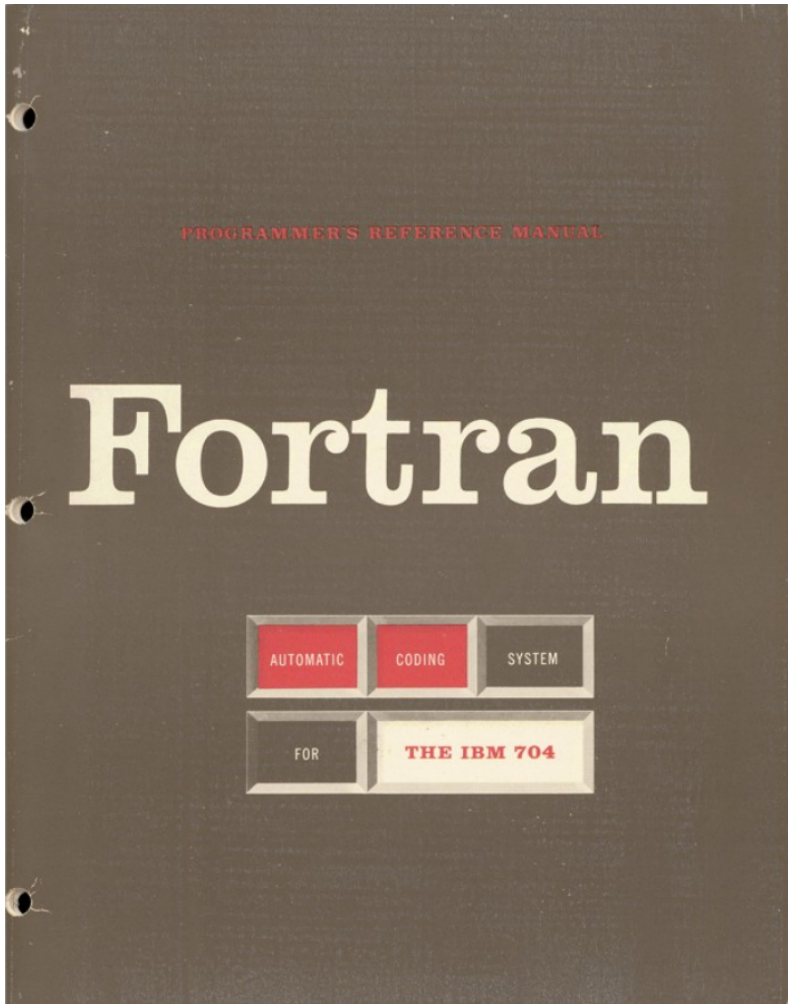

Figure 1: Primeiro livro de referência de FORTRAN

Historia do Fortran

A galera naquela época escrevia o código de máquina na mão. O FORTRAN revolucionou propondo uma linguagem de alto nível.

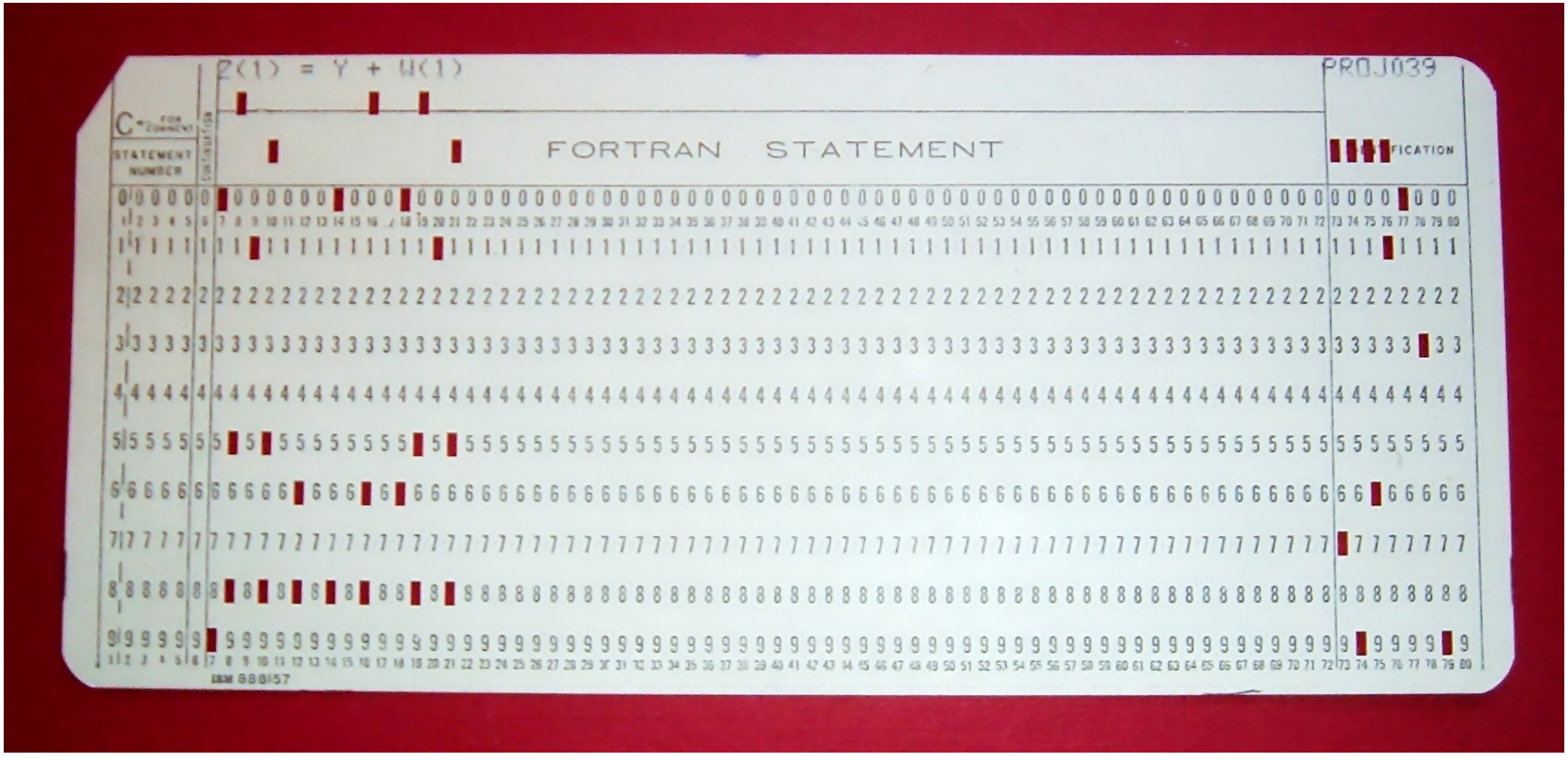

Figure 2: Cartão de furar que contém um pedaço de código FORTRAN

Historia do Fortran

Figure 3: Mainframe IBM 704

Historia do Fortran

O Fortran teve algumas revisões:

Revisões não padronizadas

- FORTRAN
- FORTRAN II (1958)
- FORTRAN III (1958, não liberada)
- IBM 1401 FORTRAN (1959)
- FORTRAN IV (1962)

Historia do Fortran

O Fortran teve algumas revisões:

Revisões padronizadas (ANSI)

- FORTRAN 66
- FORTRAN 77
- Fortran 90
- Fortran 95
- Fortran 2003
- Fortran 2008
- Fortran 2018

Historia do Fortran

Hoje Fortran é utilizado sorrateiramente em diversas aplicações ainda hoje:

- Predição numérica de clima, oceano, e surfe
- Predição e ciência do clima
- Software de dinâmica de fluido, usado em engenharia mecânica e civil
- Solucionadores de aerodinâmica para projetar carros, aviões, e espaçonaves
- Bibliotecas de algebra linear rápidas usadas por bibliotecas de aprendizado de máquina
- Fazer benchmark dos supercomputadores mais rápidos do mundo

Milan Curcic; Modern Fortran - Building Efficient Parallel Applications

Características do Fortran

- Compilada
- Tipagem estática
- Multiparadigma
- Paralel
- Madura
- Fácil de aprender

Porque aprender Fortran?

Orientada para arrays

```
do j = 1, jm
    do i = 1, im
        c(i,j) = a(i,j) * b(i,j)
    end do
end do
```

pode ser expresso como:

```
c = a * b
```

Porque aprender Fortran?

- A única linguagem paralela desenvolvida por um comitê normativo (ISO)
- Bibliotecas maduras para ciência, engenharia e matemática
- Ecosistema para programação “general-purpuse” em crescimento
- Performance imbatível

Vantagens e desvantagens

Muitas das características do Fortran são tanto uma vantagem quanto uma desvantagem, por exemplo:

- É uma linguagem específica de domínio (DSL)
- Linguagem nichada
- Linguagem fortemente e estaticamente tipada

Comparação com Python

Language	Fortran	Python
First appeared	1957	1991
Latest release	Fortran 2018	3.8.5 (2020)
International standard	ISO/IEC	No
Implementation language	C, Fortran, Assembly (compiler-dependent)	C
Compiled vs. interpreted	Compiled	Interpreted
Typing discipline	Static, strong	Dynamic, strong
Parallel	Shared and distributed memory	Shared memory only
Multidimensional arrays	Yes, up to 15 dimensions	Third-party library only (`numpy`)
Built-in types	`character`, `complex`, `integer`, `logical`, `real`	`bool`, `bytearray`, `bytes`, `complex`, `dict`, `ellipsis`, `float`, `frozenset`, `int`, `list`, `set`, `str`, `tuple`
Constants	Yes	No
Classes	Yes	Yes

Comparação com Python

Language	Fortran	Python
Generic programming	Limited	Yes
Pure functions	Yes	No
Higher order functions	Limited	Yes
Anonymous functions	No	Yes
Interoperability with other languages	C (limited)	C
OS interface	Limited	Yes
Exception handling	Limited	Yes

Comparação com Python

Python x Octave x Fortran

Fortran em paralelo, exemplo

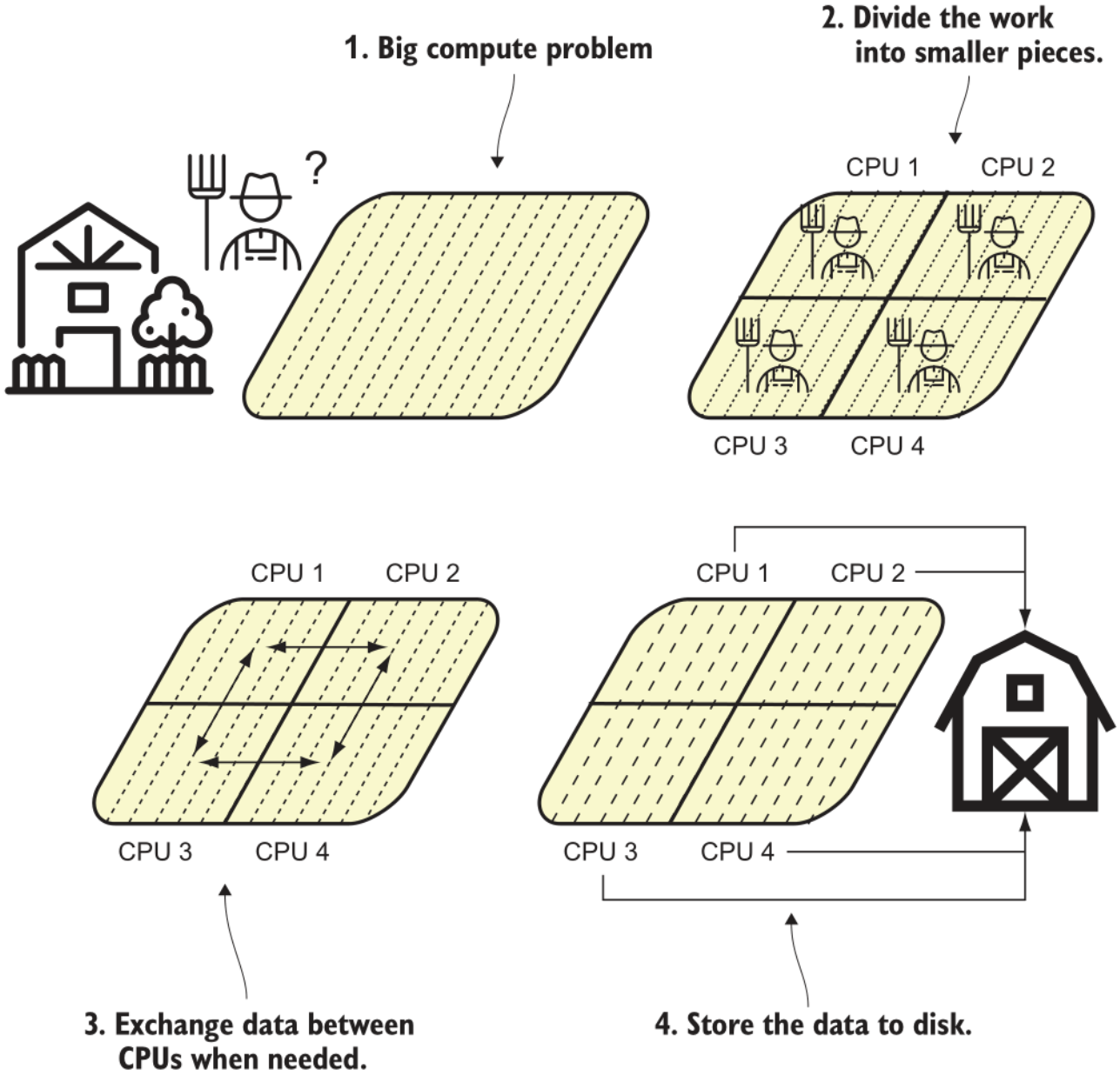

Figure 4: Padrões de programação em paralelo: dividir o problema, trocar

Preparando ambiente de desenvolvimento

Para garantir que vamos trabalhar com a mesma versão dos programas, preparei um ambiente para a gente, siga os comandos abaixo

```
git clone https://github.com/AtilaSaraiva/Curso-fortran-202
cd Curso-fortran-2022/codigos
sh prep.sh
```

Hello world

Agora vamos escrever um código de hello world

Abrindo arquivo

```
cd 1-helloworld
notepadqq oi.f90 &
```

Código

```
program hellou
    implicit none

    print*, "E ai galera"
end program hellou
```

Compilar e executar

```
gfortran oi.f90 -o oi
./oi
```

Estrutura básica de um programa

```
program main
    implicit none
    integer :: a

    a = increment(34)
    write (*,*) a

contains

    function increment(input) result (output)
        integer :: output
        integer :: input

        output = input + 1
    end function increment

end program main
```

Comentário

```
a = b ! Isso é um comentario

c = d ! Isso!! também é um comentário
```

Variáveis

```
real                :: numeroDecimal = 3.141592
integer             :: numeroInteiro = 3
character           :: caractere     = "a"
character(len=5)    :: nome          = "atila"

print*, "Número real: ", numeroDecimal
print*, "Número inteiro: ", numeroInteiro
print*, "Caractere único: ", caractere
print*, "String: ", nome
```

Números complexos

```
program numerosComplexos
    implicit none
    complex, parameter :: i = (0, 1)   ! sqrt(-1)
    complex :: x, y, z
    x = (7, 8)
    y = (5, -7)
    write(*,*) i * x * y
    z = x + y
    print *, "z = x + y = ", z
    z = x - y
    print *, "z = x - y = ", z
    z = x * y
    print *, "z = x * y = ", z
    z = x / y
    print *, "z = x / y = ", z
end program numerosComplexos
```

Funções instrísecas

```
x = sin(3.14159)
y = exp(0)
z = log(1)
w = acos(-1)
```

I/O Básico

Para ler variável do terminal

```
read(*,*) variavel
```

Para imprimir o valor de uma variável na tela

```
print*, variavel
```

ou

```
write(*,*) variavel
```

Exercício

Escrever código que lê dois números reais do terminal e imprime o resultado da soma deles

Loops

Dois tipos básicos

```
print*, "do simples"
do i=1,3
    do j=1,3
        print*, i,j
    end do
end do

print*, "do while"
i=1
do while(i<=3)
    j = 1
    do while(j<=3)
        print*, i,j
        j = j + 1
    end do
    i = i + 1
end do
```

Loops

Do concurrent

```
print*, "do concurrent"
do concurrent(i=1:3, j=1:3)
    print*, i,j
end do
```

Exercício

Escreva o código dos últimos dois slides e veja o resultado