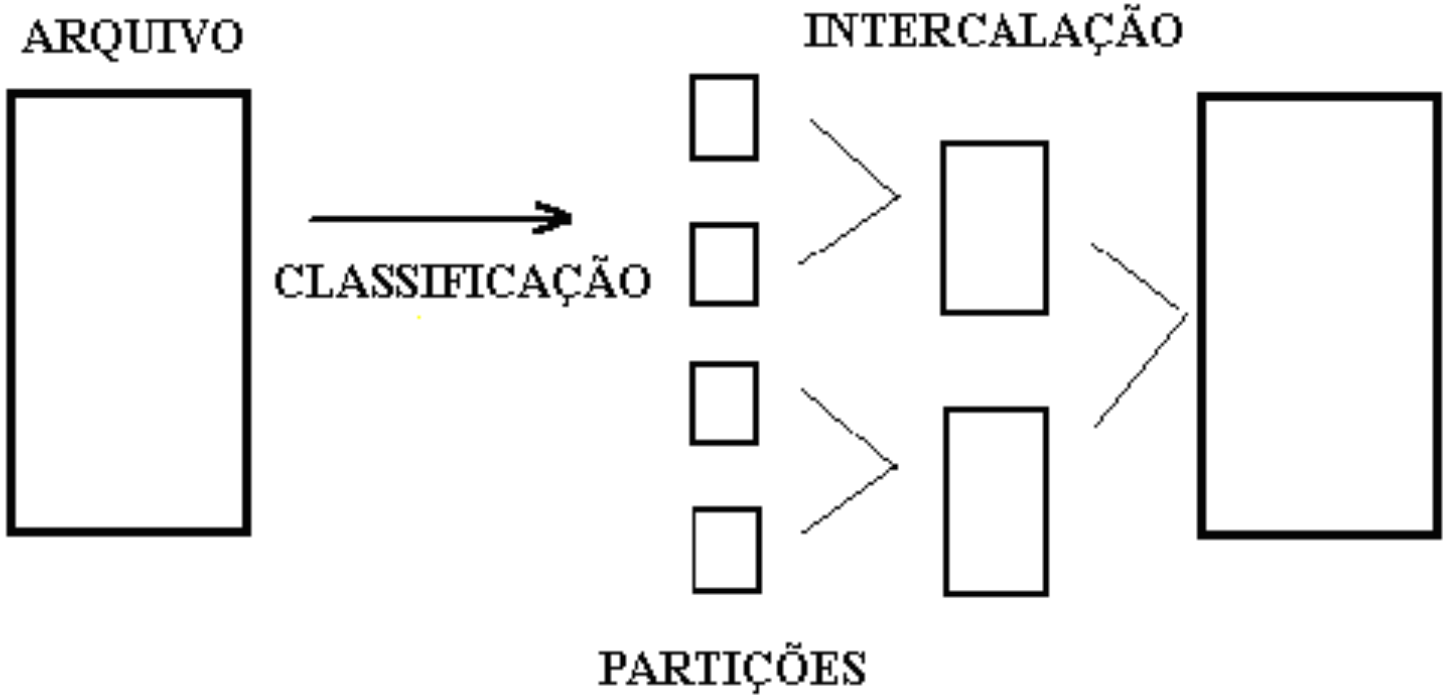
Classificação Externa: Intercalação de Partições Classificadas

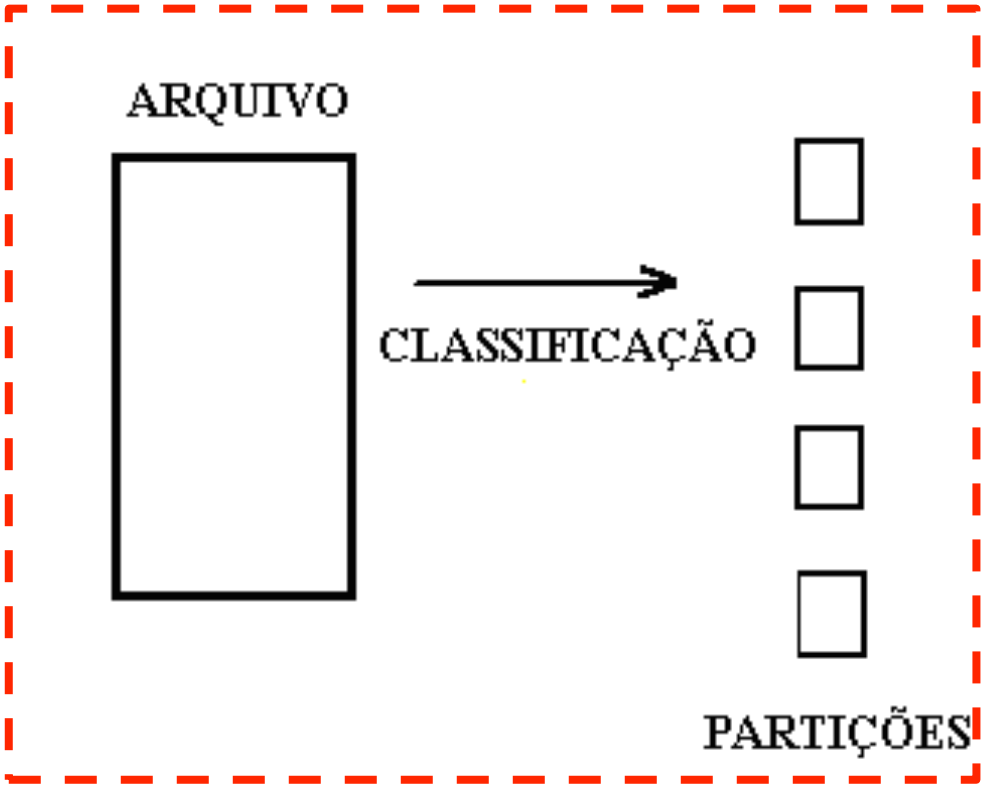
Vanessa Braganholo

Baseado no Material de: Inhaúma Neves Ferraz (IC/UFF)

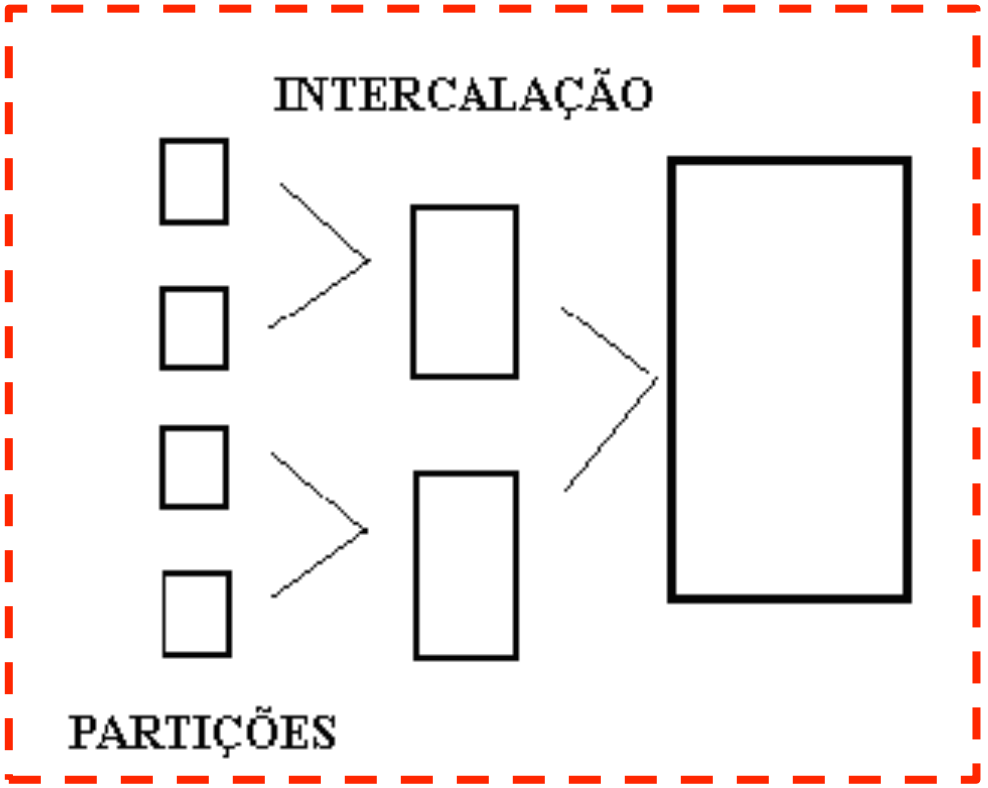
Relembrando:
Modelo da Classificação Externa



Aula Passada: Etapa de Classificação



Aula de Hoje: Etapa de Intercalação



Objetivo da Etapa de Intercalação

- Transformar um conjunto de partições classificadas por determinado critério, em um único arquivo contendo todos os registros de todas as partições originais do conjunto
- O arquivo gerado deve estar classificado pelo mesmo critério de classificação das partições iniciais

Problema

- Considere a existência de R partições geradas pelo processo de geração de partições
- Como gerar o arquivo a partir das R partições?

Problema

- Seria ideal poder intercalar todas as partições de uma só vez e obter o arquivo classificado, utilizando, por exemplo, a árvore de vencedores, mas:

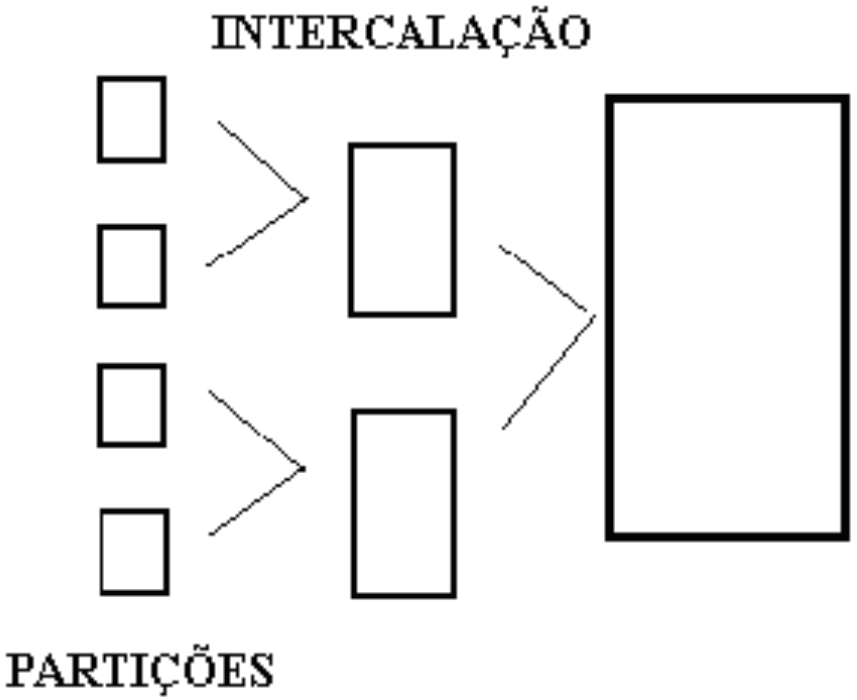
  (i) O número de arquivos a intercalar pode gerar uma árvore de vencedores maior do que a capacidade da memória

  (ii) Sistemas Operacionais estabelecem número máximo de arquivos abertos simultaneamente
  - Esse número pode ser bem menor do que o número de partições existentes
  - Curiosidade: ver o número máximo de arquivos que podem ser abertos no linux:
    - ulimit –Hn

Solução

- A intercalação vai exigir uma série de fases durante as quais registros são lidos de um conjunto de arquivos e gravados em outro (partições)

Etapa de Intercalação



Estágio de Intercalação

Estratégias de distribuição e intercalação:

- Intercalação balanceada de N caminhos
- Intercalação ótima

Medida de Eficiência

- Uma medida de eficiência do estágio de intercalação é dada pelo número de passos sobre os dados:

$$\text{Número de passos} = \frac{\text{No. total de registros lidos}}{\text{No. total de registros no arquivo classificado}}$$

- Número de passos representa o número médio de vezes que um registro é lido (ou gravado) durante o estágio de intercalação

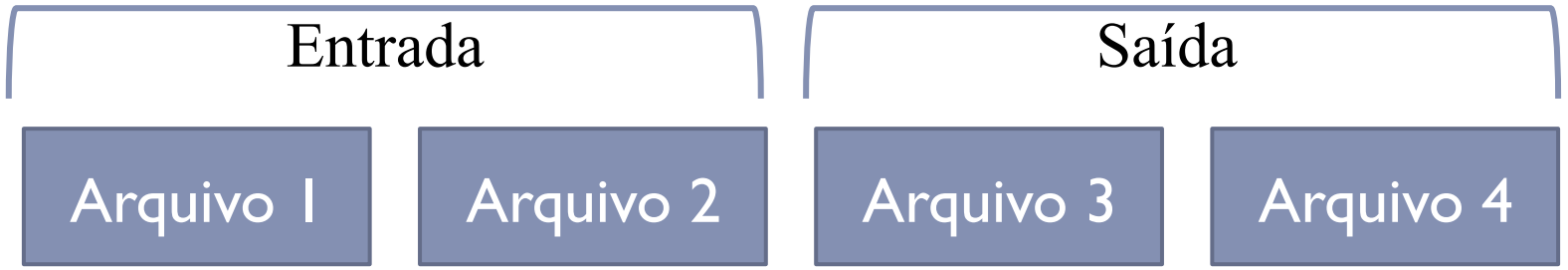
Intercalação balanceada de N caminhos

Intercalação Balanceada de N caminhos

- Primeiro passo: determinar o número de arquivos (*F*) que o algoritmo irá manipular
  - Metade dos arquivos (*F*/2) será usada para leitura (entrada)
  - A outra metade (*F*/2), para escrita (saída)

Exemplo

- Número de arquivos $F = 4$

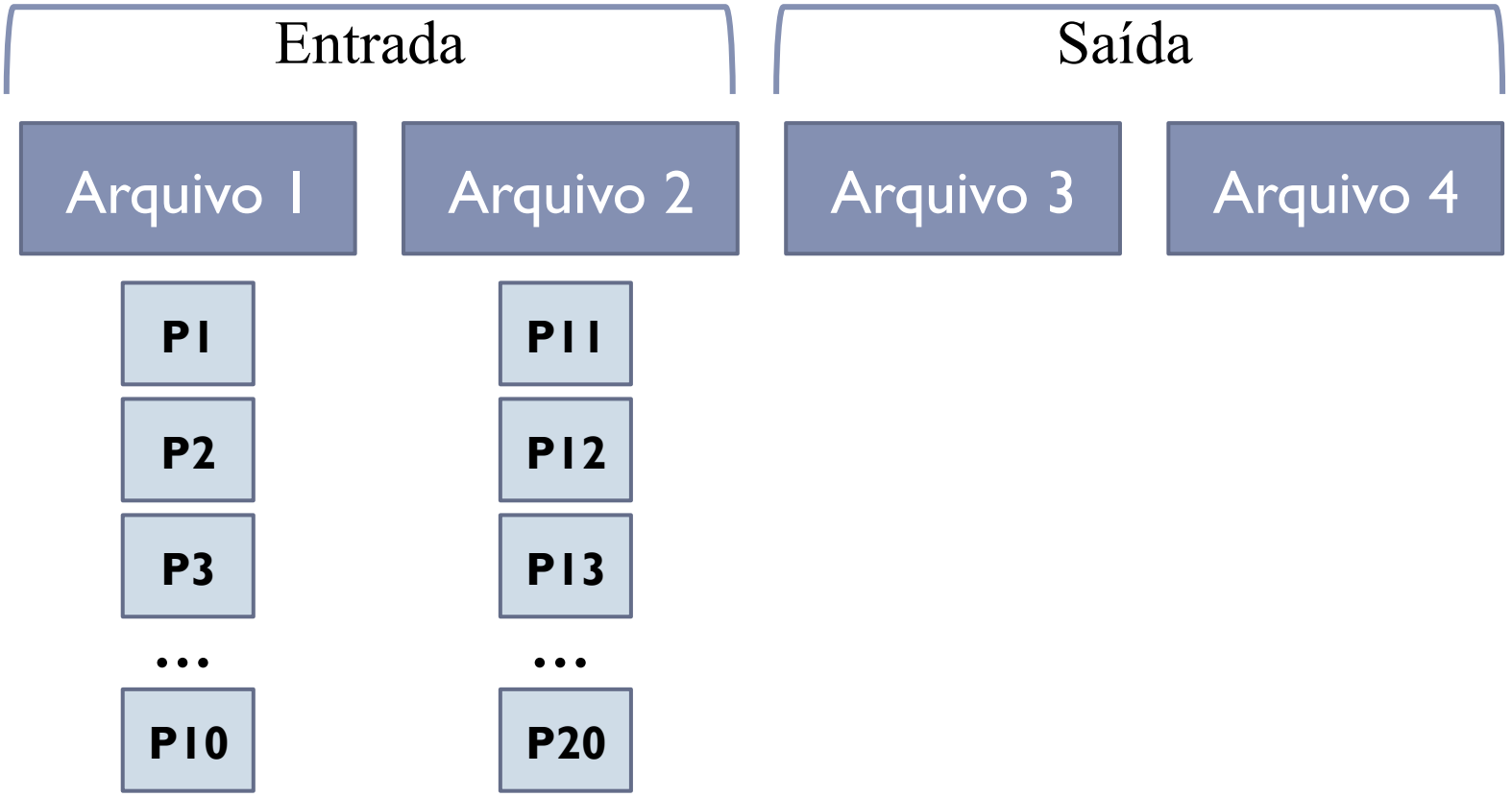


Intercalação Balanceada de N caminhos

- Passo 2: Distribuir todas as partições, tão equilibradamente quanto possível, nos *F/2* arquivos de entrada
  - Atenção: aqui fazemos apenas uma "fila" para ver que "variável de arquivo" vai processar cada partição

Exemplo

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20

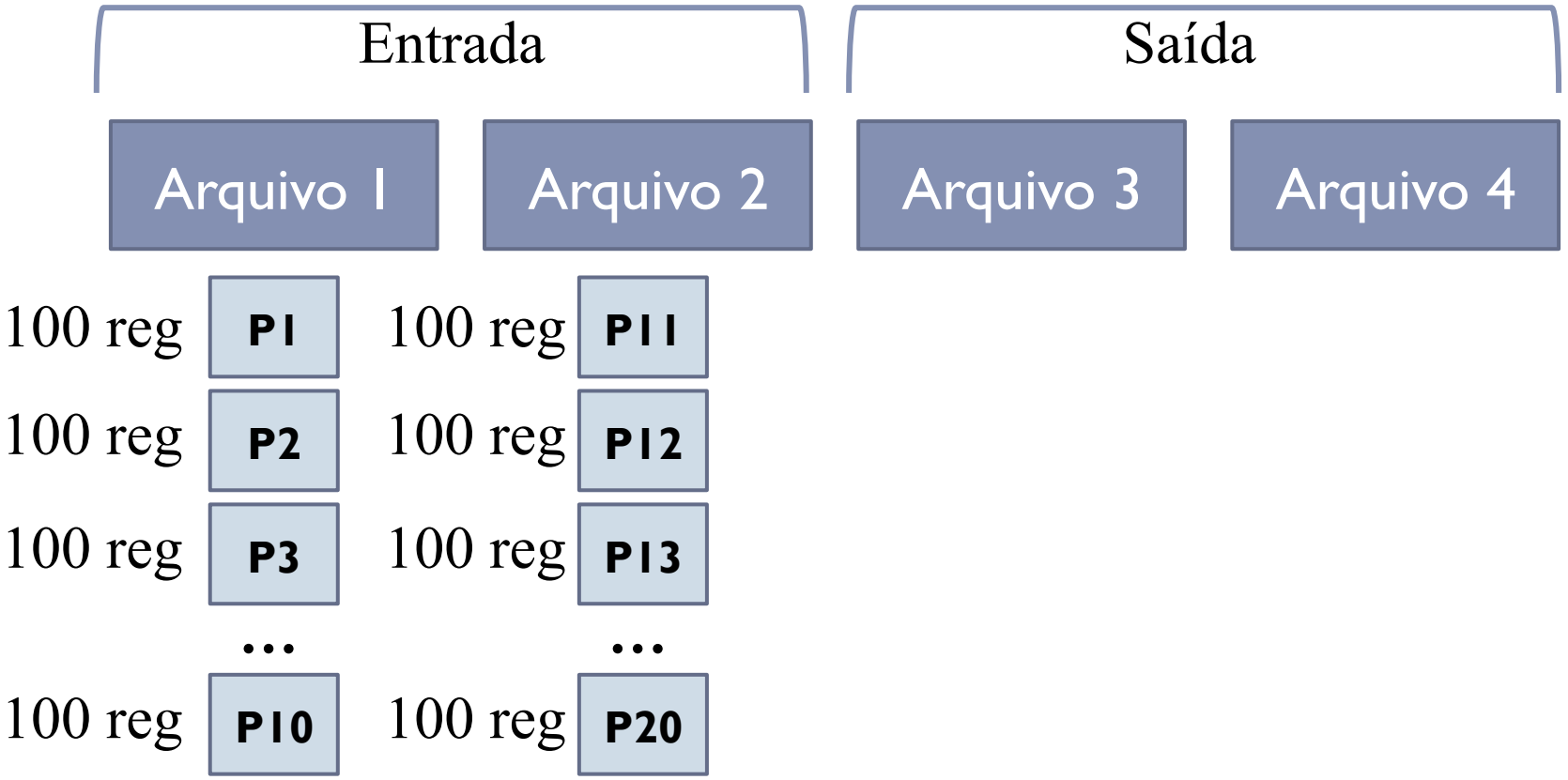


Intercalação Balanceada de N caminhos

- Início da fase de intercalação: intercalar as primeiras F/2 partições, gravando o resultado em um dos F/2 arquivos de saída

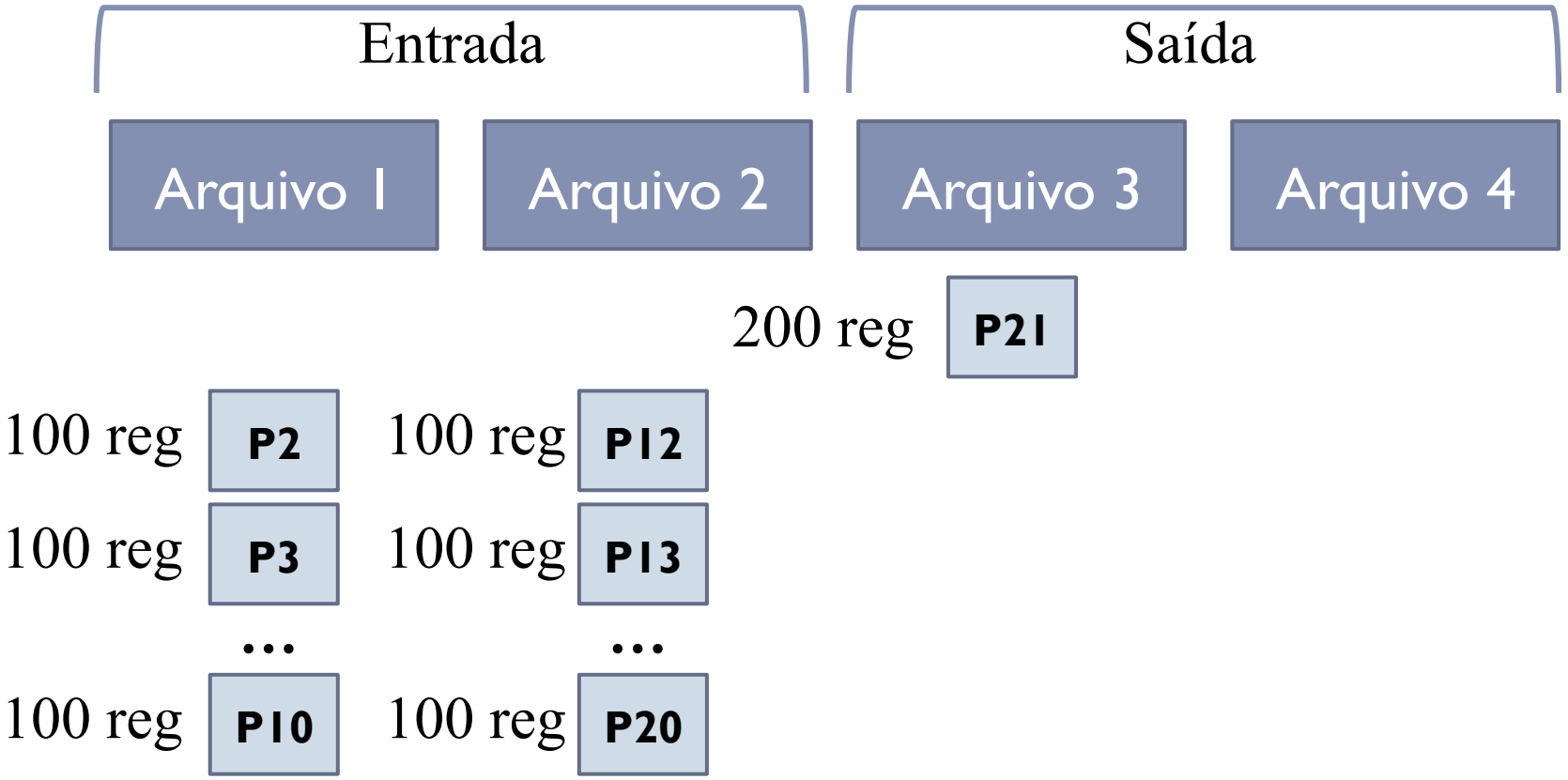
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



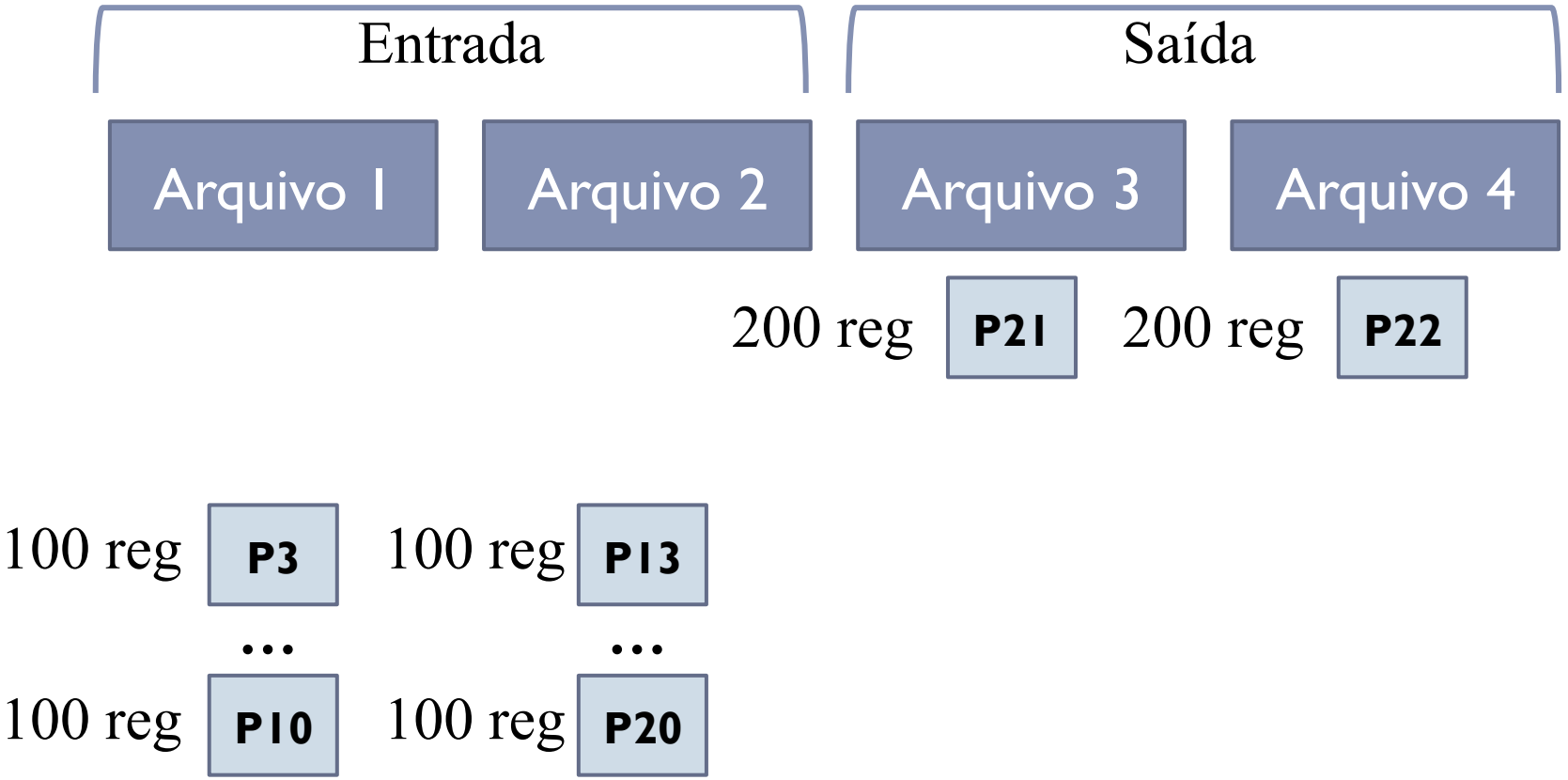
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



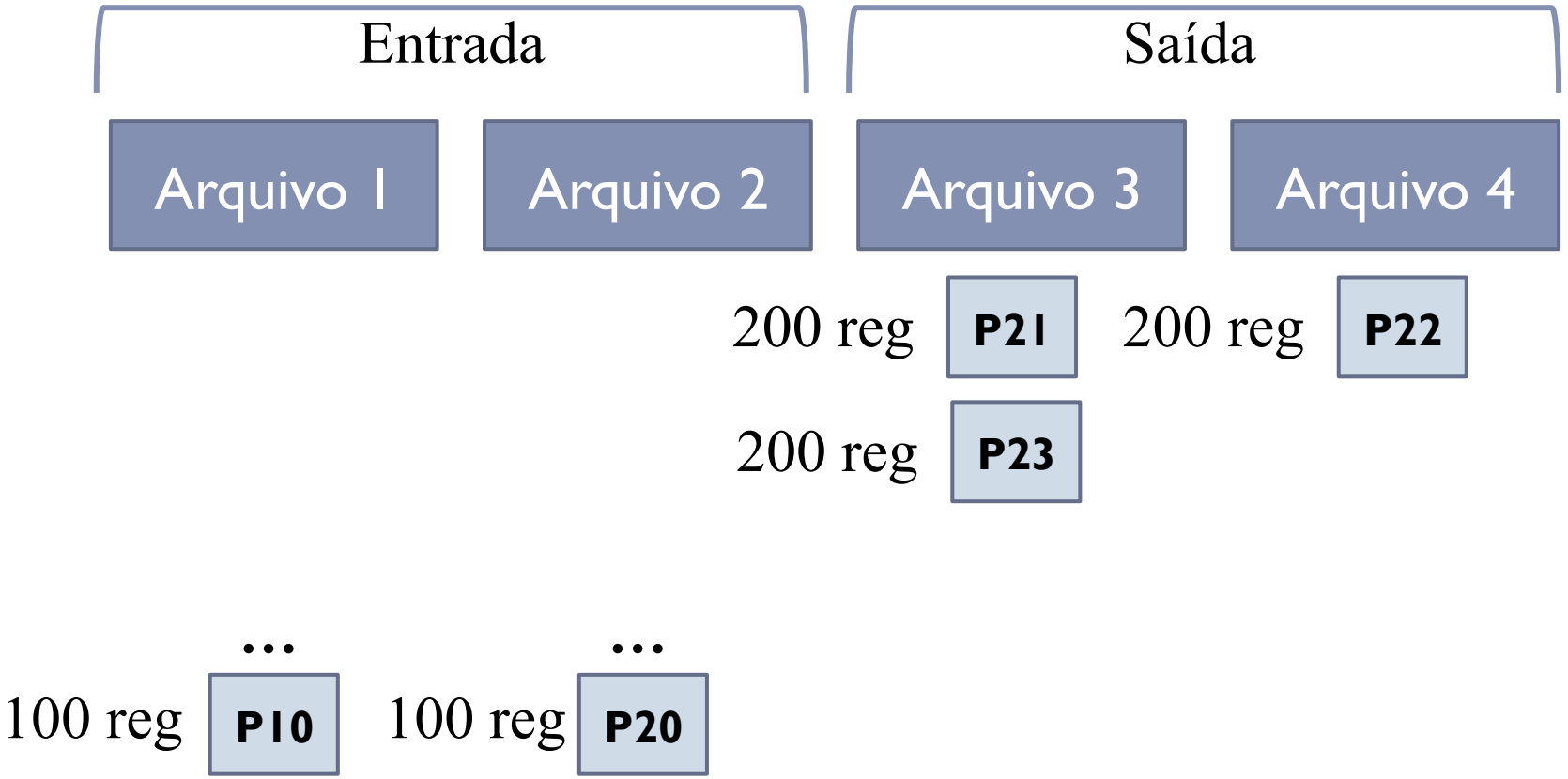
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



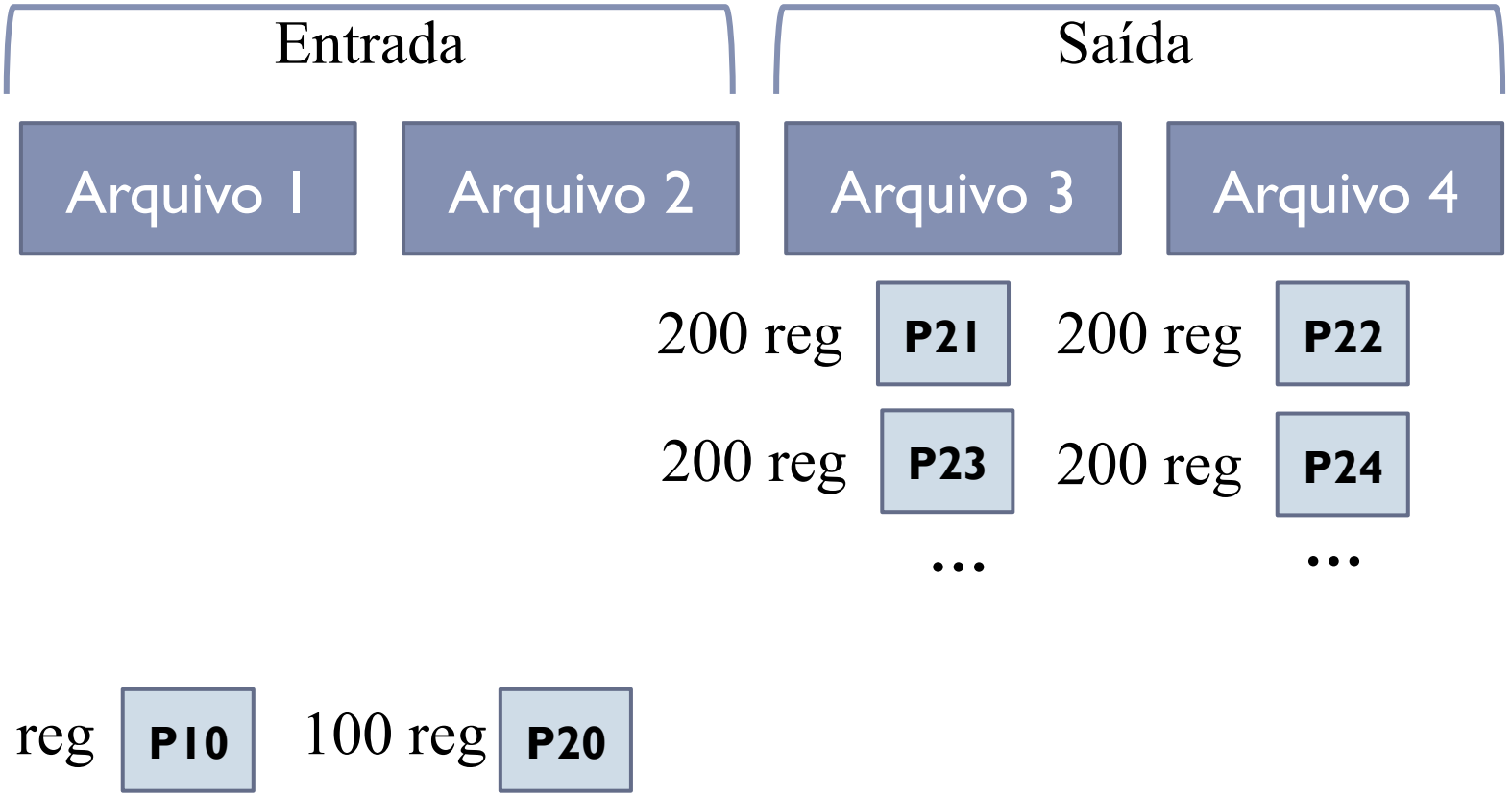
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



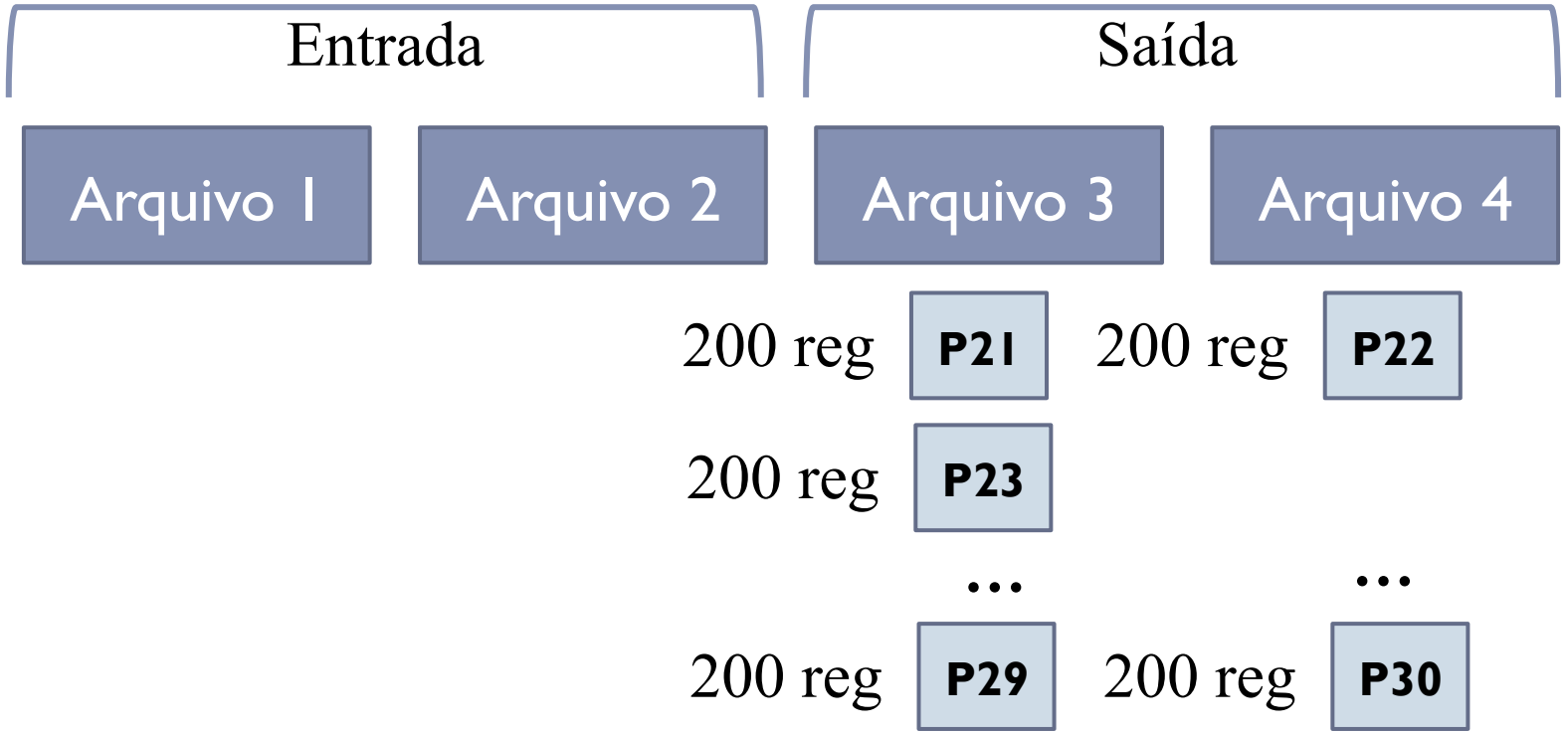
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



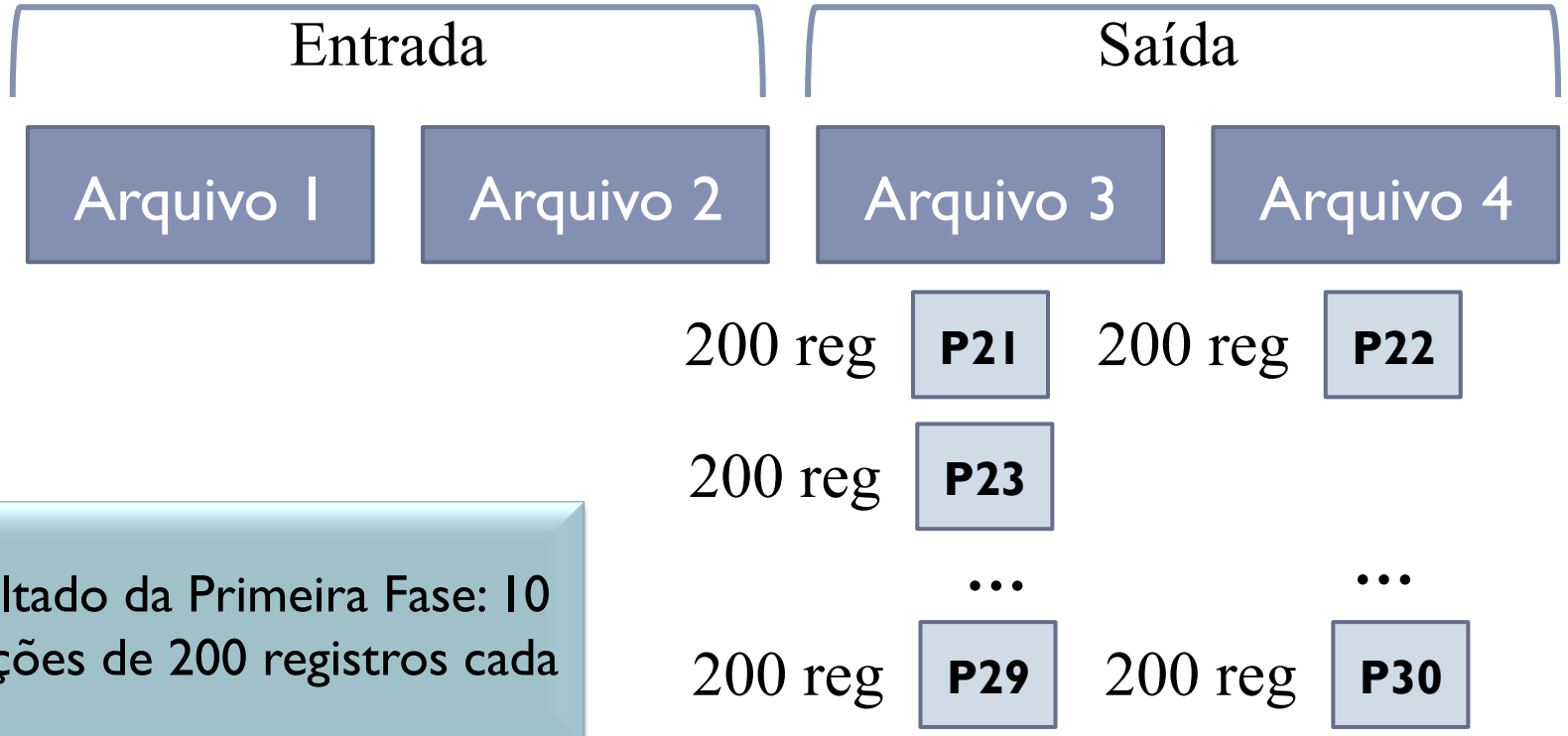
Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



Exemplo: Fase 1

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



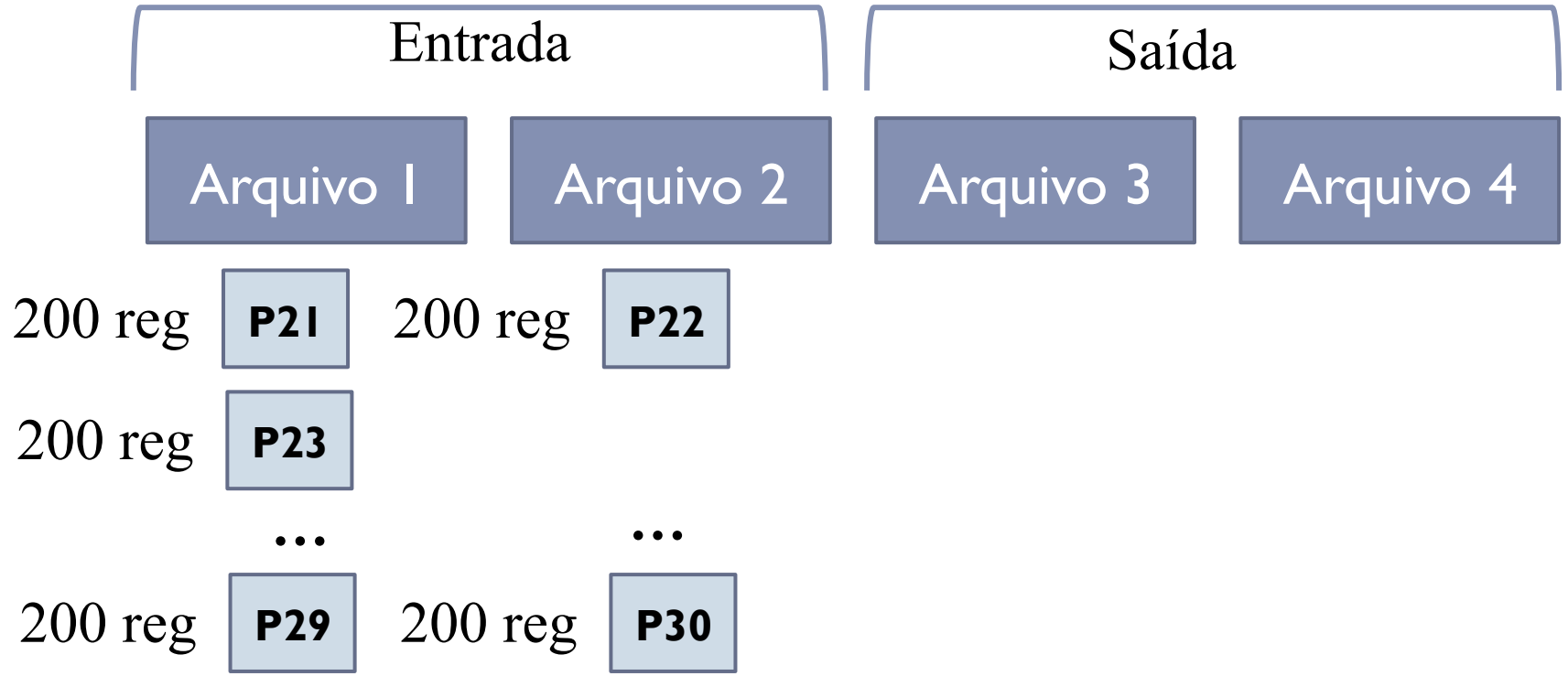
Resultado da Primeira Fase: 10 partições de 200 registros cada

Intercalação Balanceada de N caminhos

- No final de cada fase, o conjunto de partições de saída torna-se o conjunto de entrada

Exemplo: Fase 2

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 10 (200 registros cada)



Intercalação Balanceada de N caminhos

- A intercalação termina quando, em uma fase, grava-se apenas uma partição

Número de Fases: 5

- Fase 1: 20 partições com 100 registros cada
- Fase 2: 10 partições com 200 registros cada
- Fase 3: 5 partições com 400 registros cada
- Fase 4: 2 partições com 800 registros cada + 1 partição com 400 registros cada
- Fase 5: 1 partição com 1600 registros + 1 partição com 400 registros

- Resultado da Fase 5: 1 partição com 2000 registros

Número de Fases: 5

- Fase 1: 20 partições com 100 registros cada
- Fase 2: 10 partições com 200 registros cada
- Fase 3: 5 partições com 400 registros cada
- Fase 4: 2 partições com 800 registros cada + **1 partição com 400 registros cada**
- Fase 5: 1 partição com 1600 registros + **1 partição com 400 registros**

Pode ocorrer que partições sejam copiadas de um arquivo para outro sem qualquer processamento

Número de Passos

$$\text{Número de passos} = \frac{\text{No. total de registros lidos}}{\text{No. total de registros no arquivo classificado}}$$

- Número de registros lidos
  - Fase 1: 20 partições com 100 registros cada = 2000
  - Fase 2: 10 partições com 200 registros cada = 2000
  - Fase 3: 5 partições com 400 registros cada = 2000
  - Fase 4: 2 partições com 800 registros cada + 1 partição com 400 registros cada = 2000
  - Fase 5: 1 partição com 1600 registros + 1 partição com 400 registros = 2000

2000 * 5 = 10000 registros lidos

Número de Passos

$$\text{Número de passos} = \frac{10000}{\text{No. total de registros no arquivo classificado}}$$

- Número total de registros no arquivo classificado: 2000

Número de Passos

$$\text{Número de passos} = \frac{10000}{2000} = 5$$

- Portanto: número de passos = número de fases

Intercalação Balanceada de N caminhos

- O balanceamento do processo baseia-se em colocar nos arquivos de entrada aproximadamente o mesmo número de registros

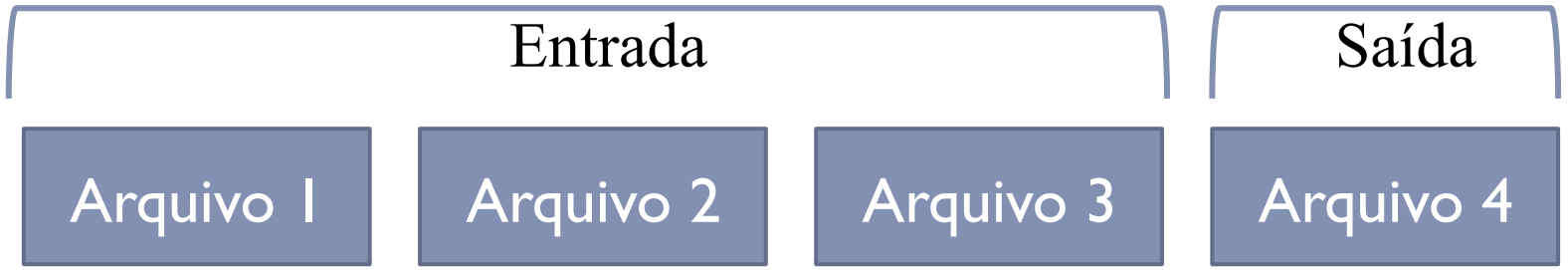
Intercalação Ótima

Intercalação Ótima

- F arquivos
  - F – 1 para entrada
  - 1 para saída

Exemplo

- Número de arquivos *F* = 4

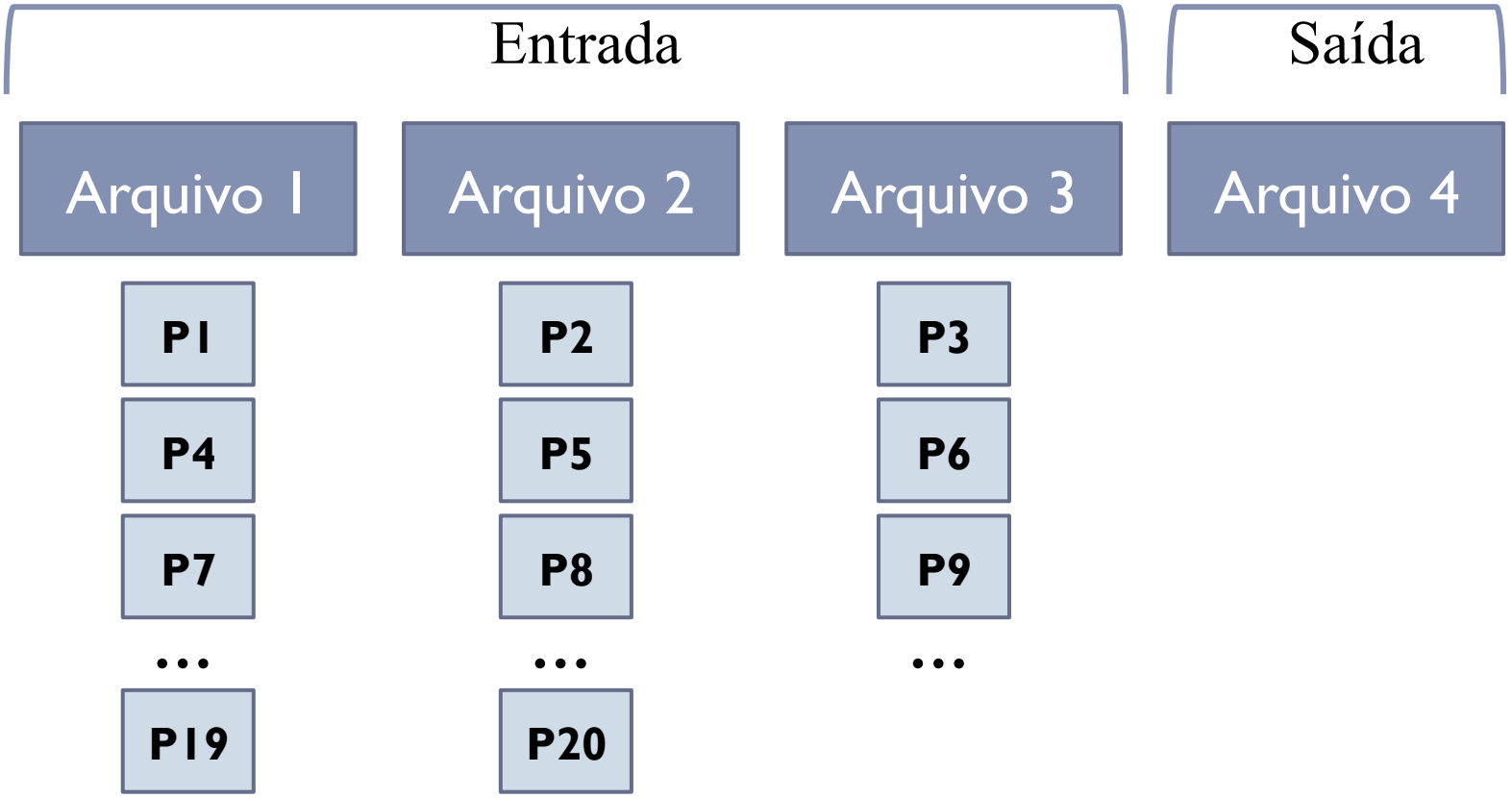


Intercalação Ótima

- Durante cada fase do algoritmo, F-1 partições são intercaladas e gravadas no arquivo de saída

Exemplo

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20

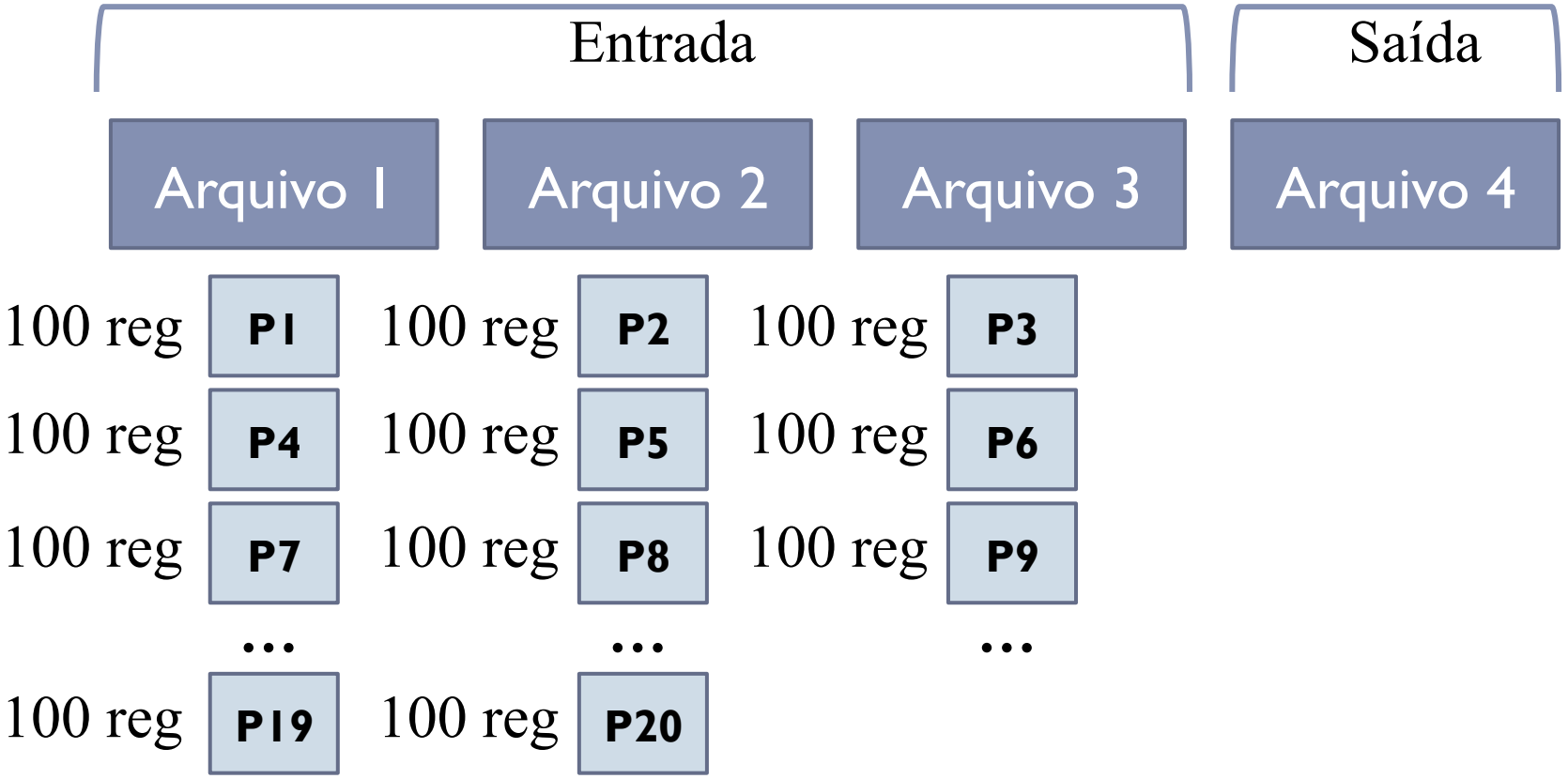


Intercalação Ótima

- Do conjunto inicial de partições removem-se as partições intercaladas e a ele agrega-se a partição gerada na intercalação
- Algoritmo termina quando este conjunto tiver apenas uma partição

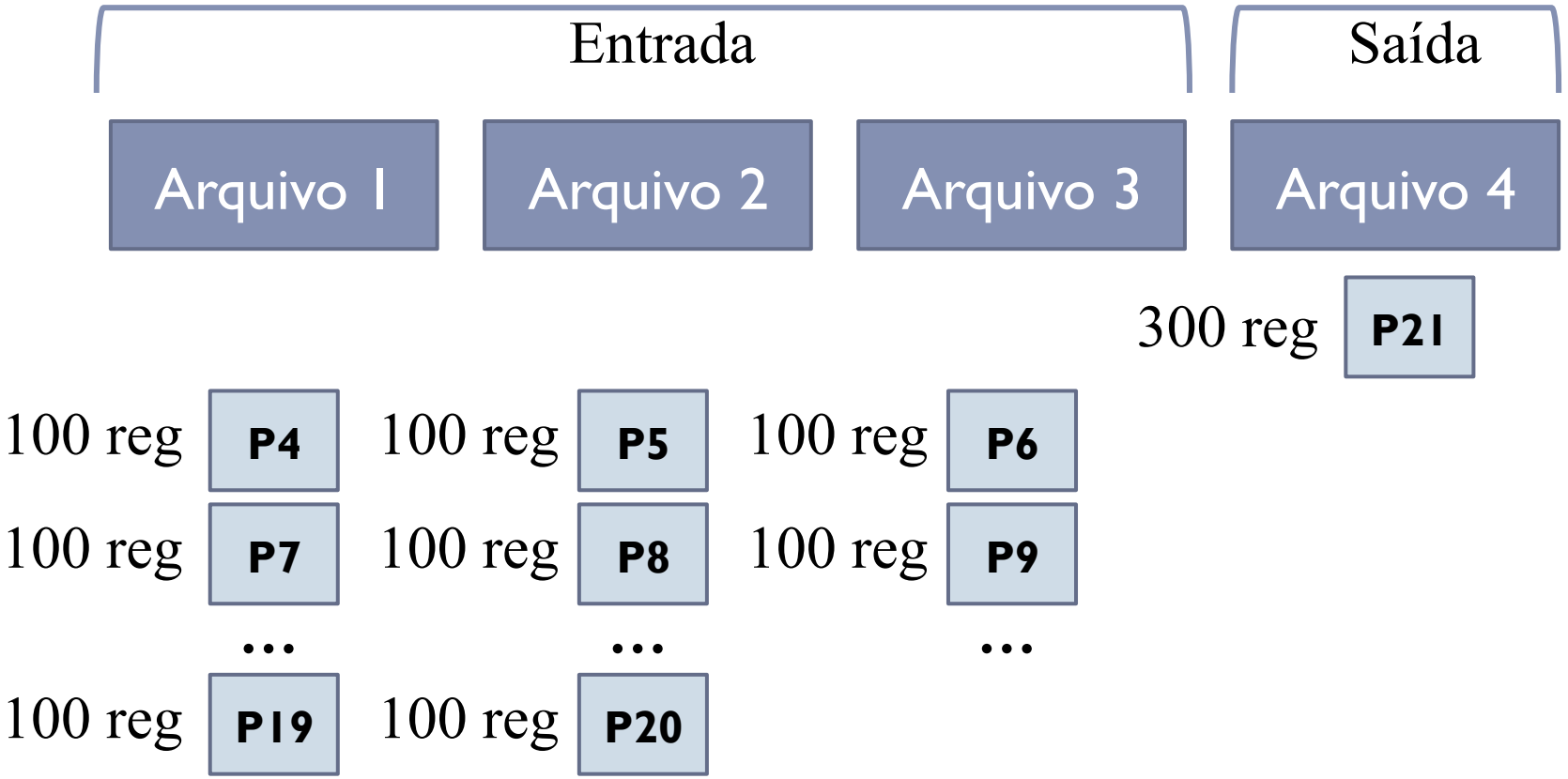
Exemplo

- Número de arquivos $F = 4$
- Partições a serem intercaladas = 20 (100 registros cada)



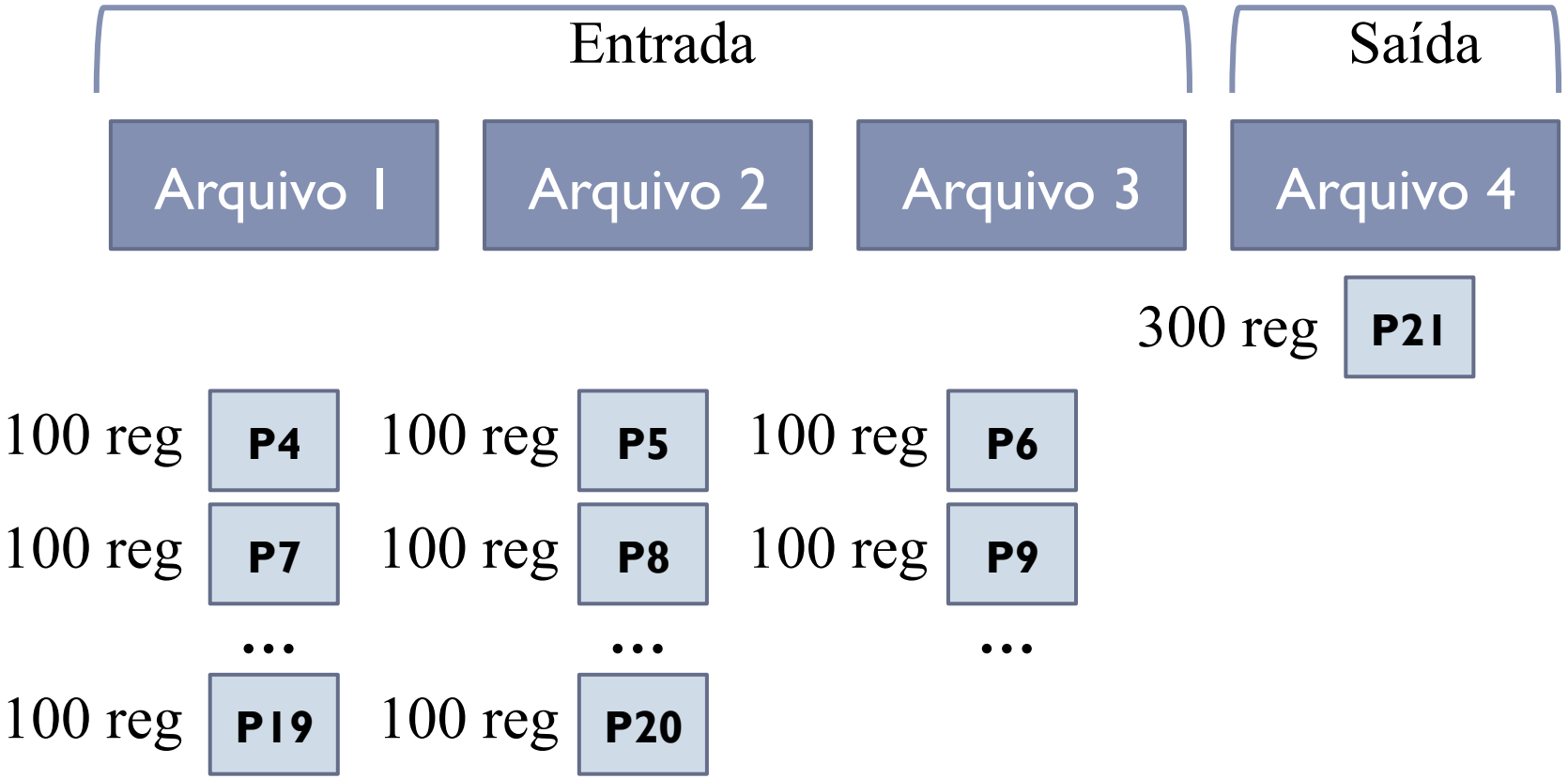
Exemplo

- Número de arquivos $F = 4$
- Partições a serem intercaladas = 20 (100 registros cada)



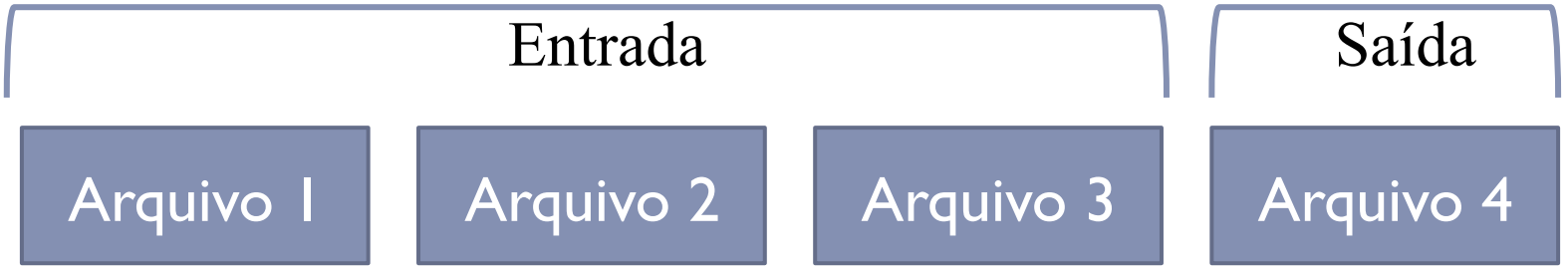
Exemplo

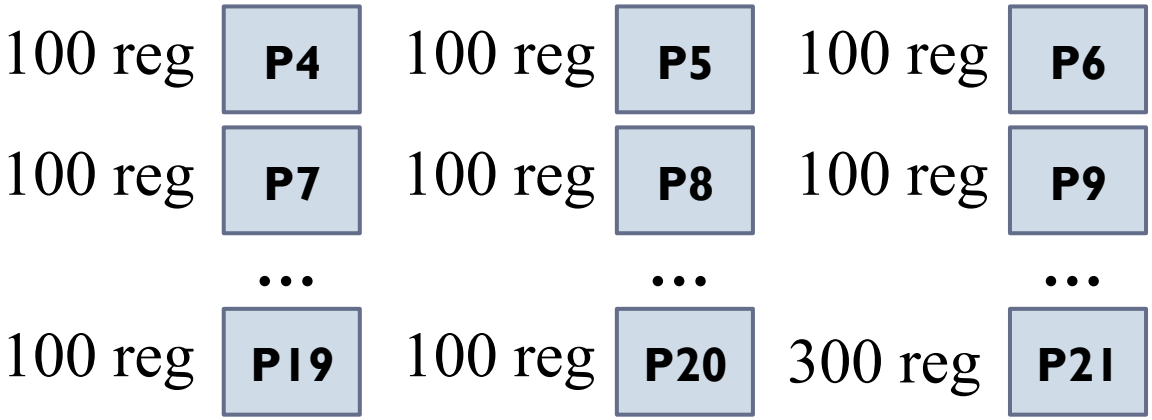
- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



Exemplo

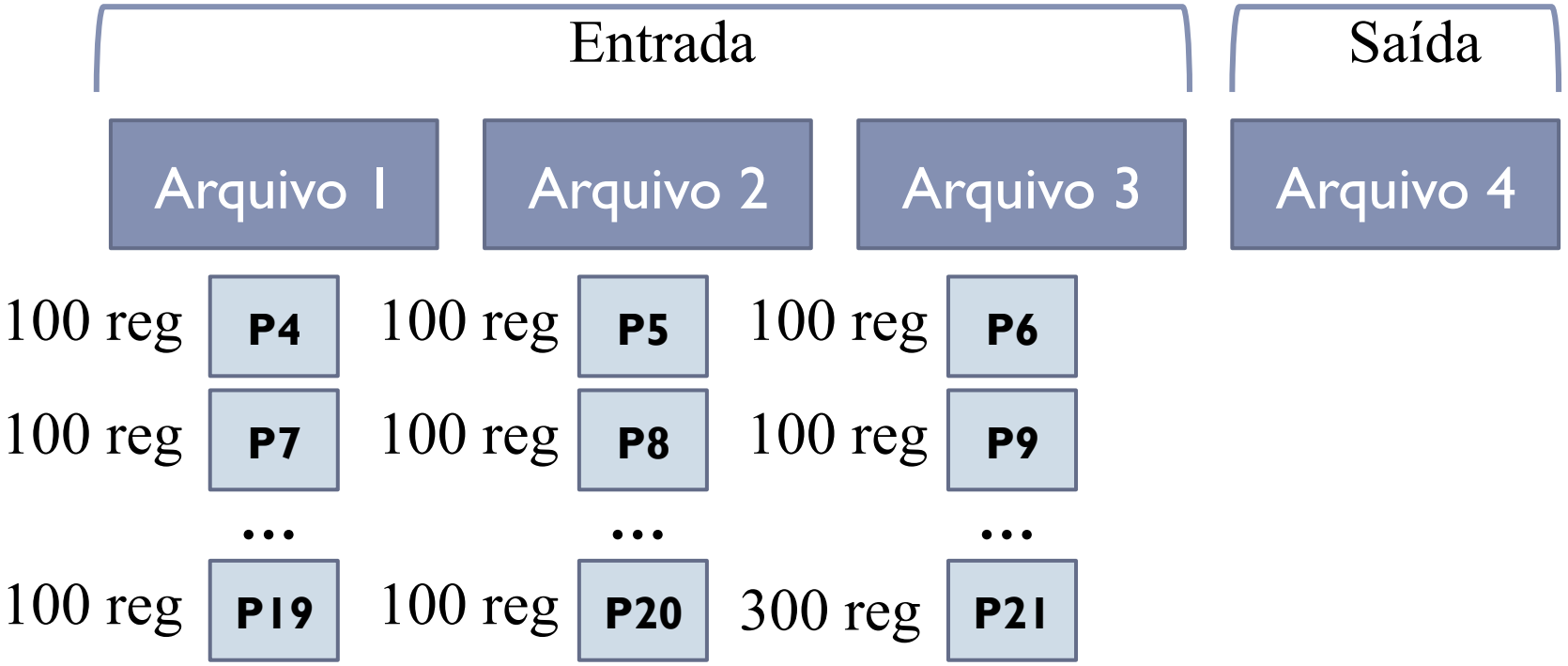
- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)





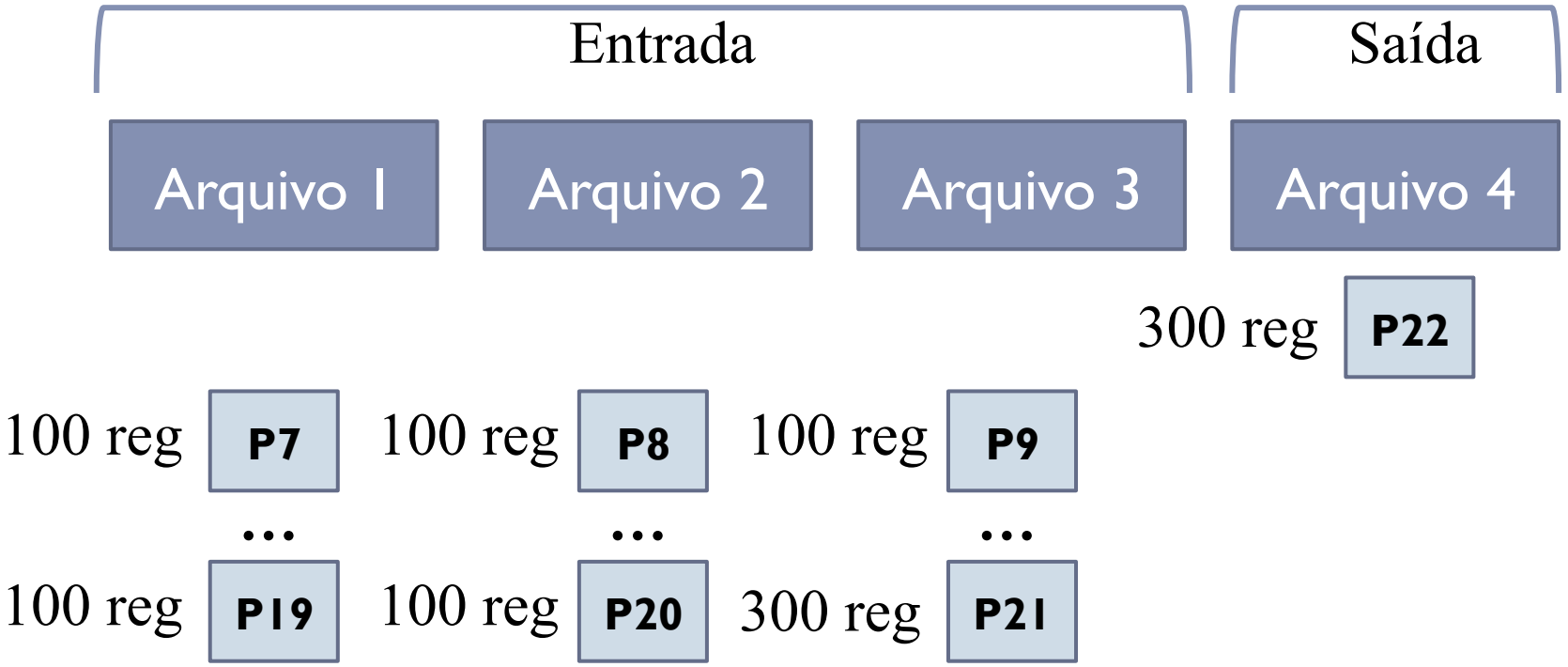
Exemplo

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



Exemplo

- Número de arquivos *F* = 4
- Partições a serem intercaladas = 20 (100 registros cada)



Resumo do Exemplo

Fase	Arquivo1	Arquivo2	Arquivo3	Arquivo4	N°. de leituras
1	1:100	2:100	3:100	21:300	300
2	4:100	5:100	6:100	22:300	300
3	7:100	8:100	9:100	23:300	300
4	10:100	11:100	12:100	24:300	300
5	13:100	14:100	15:100	25:300	300
6	16:100	17:100	18:100	26:300	300
7	19:100	20:100	21:300	27:500	500
8	22:300	23:300	24:300	28:900	900
9	25:300	26:300	27:500	29:1100	1100
10	28:900	29:1100	----------	30:2000	2000
				TOTAL	6300

$$Número\ de\ passos = \frac{6300}{2000} = 3{,}15$$

Resumo do Exemplo

Fase	Arquivo1	Arquivo2	Arquivo3	Arquivo4	N°. de leituras
1	1:100	2:100	3:100	21:300	300
2	4:100	5:100	6:100	22:300	300
10	28:900	29:1100	----------	30:2000	2000
				TOTAL	6300

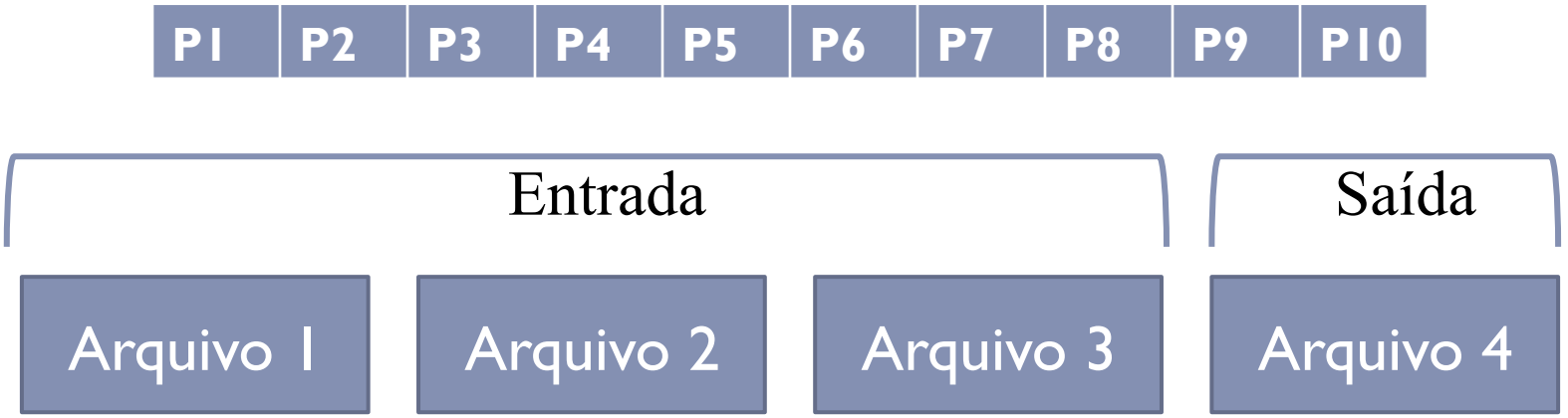
Notar que o conceito de **Fase** desse algoritmo é diferente do usado no algoritmo anterior. Usando esse conceito o algoritmo anterior teria 21 fases.

$$Número\ de\ passos = \frac{6300}{2000} = 3{,}15$$

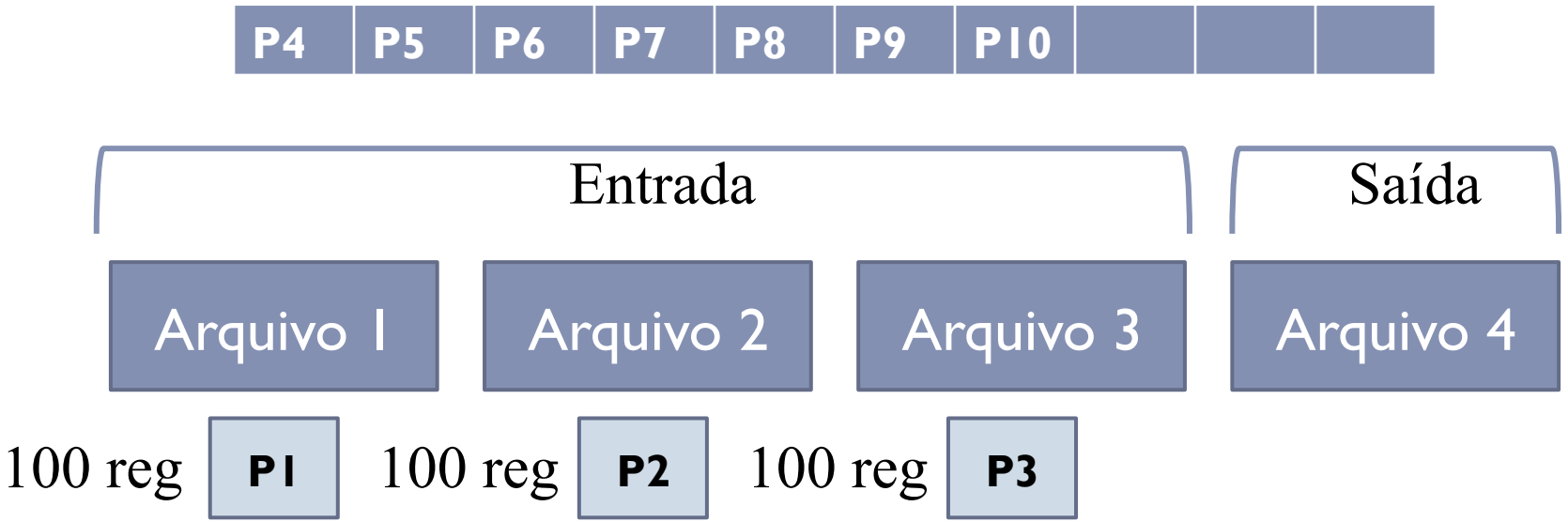
Possível forma de Implementação

- Usar uma lista que contém os nomes dos arquivos a ordenar
- A cada passo do algoritmo, retirar os 3 primeiros itens da lista, intercalá-los, colocar o arquivo resultante no final da lista
- O algoritmo pára quando a lista tiver apenas 1 arquivo (que será o resultante)

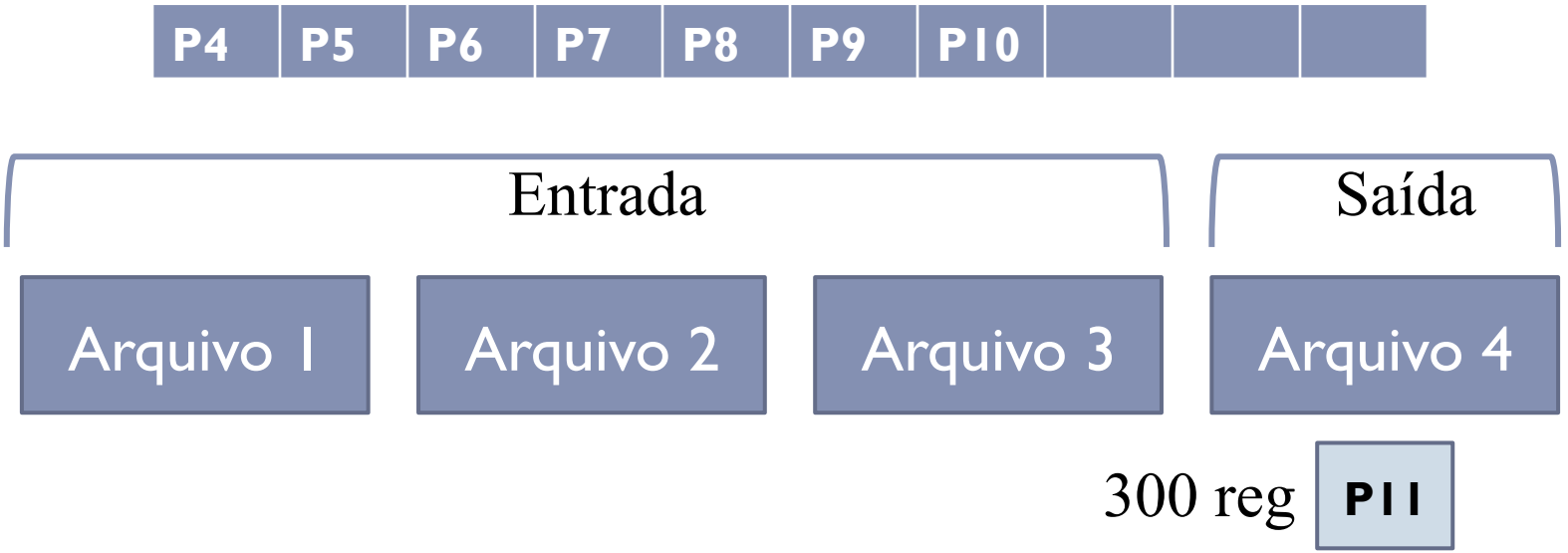
Exemplo (F = 4 e 10 partições):



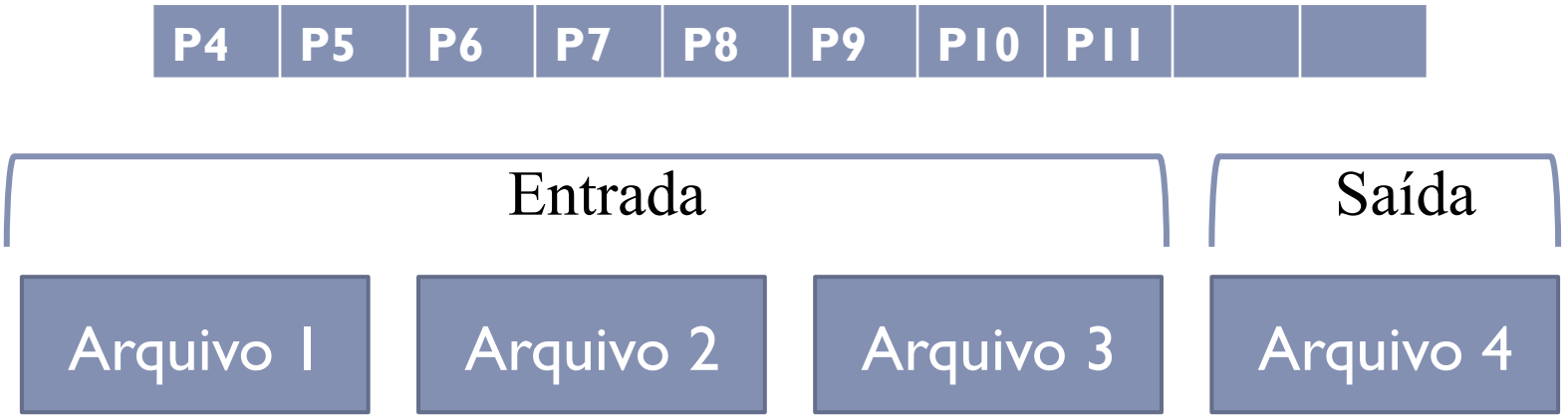
Exemplo (F = 4 e 10 partições):



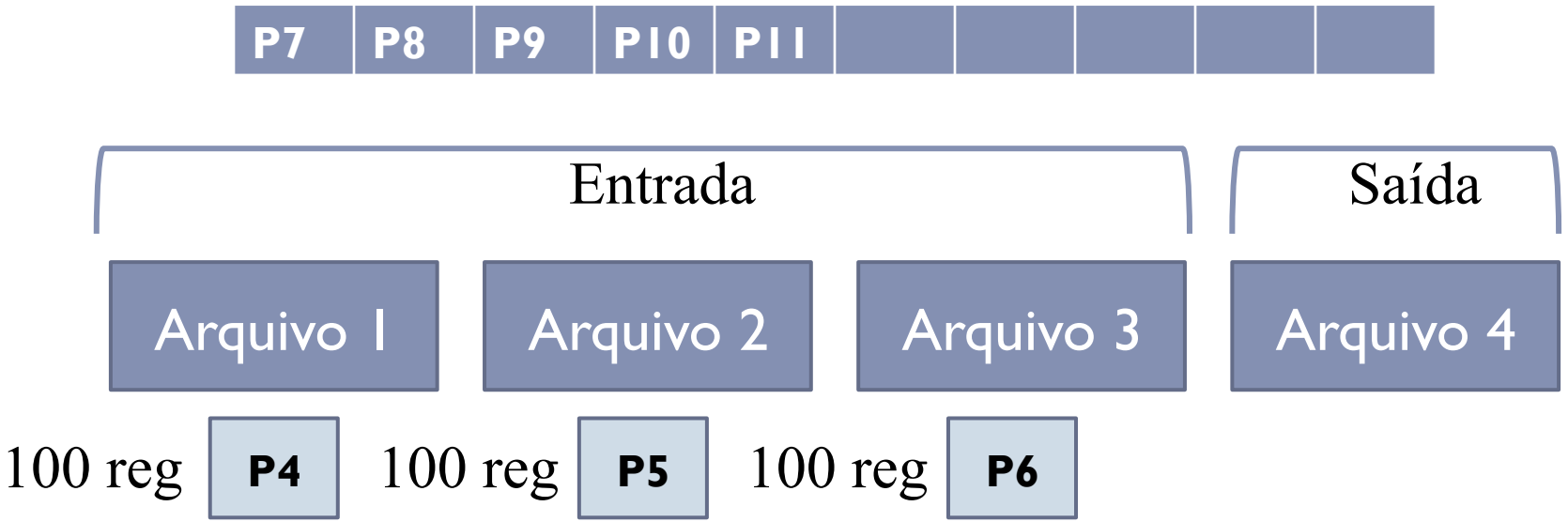
Exemplo (F = 4 e 10 partições):



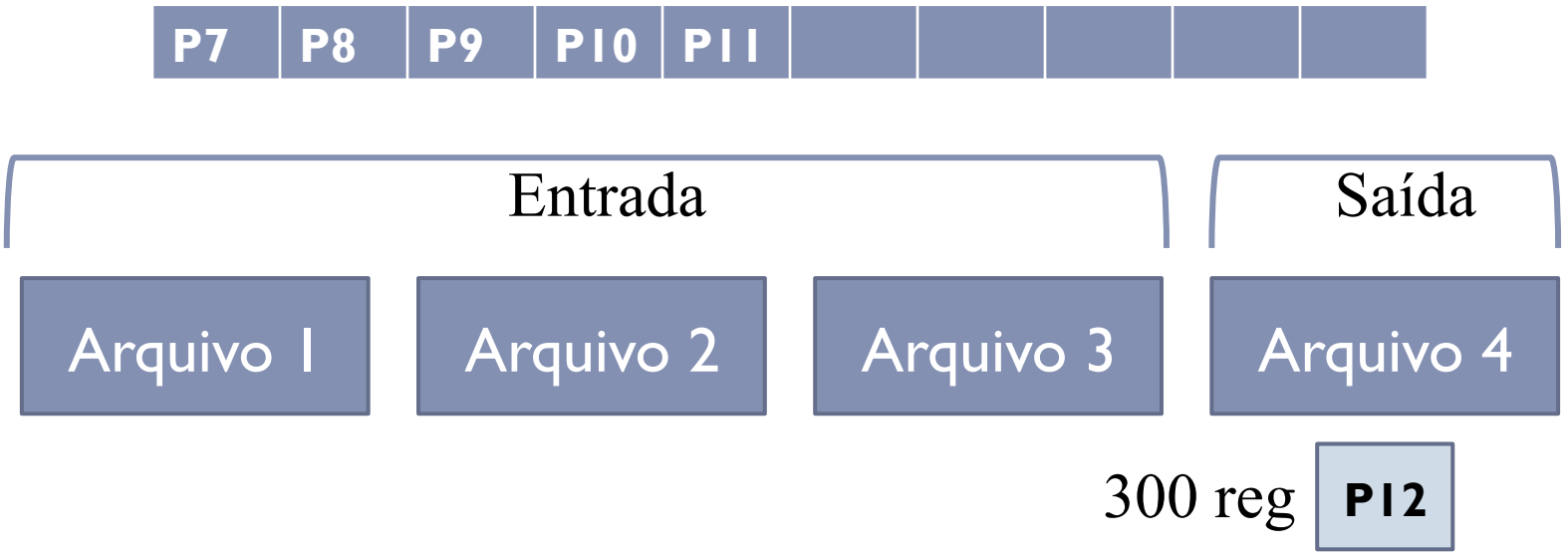
Exemplo (F = 4 e 10 partições):



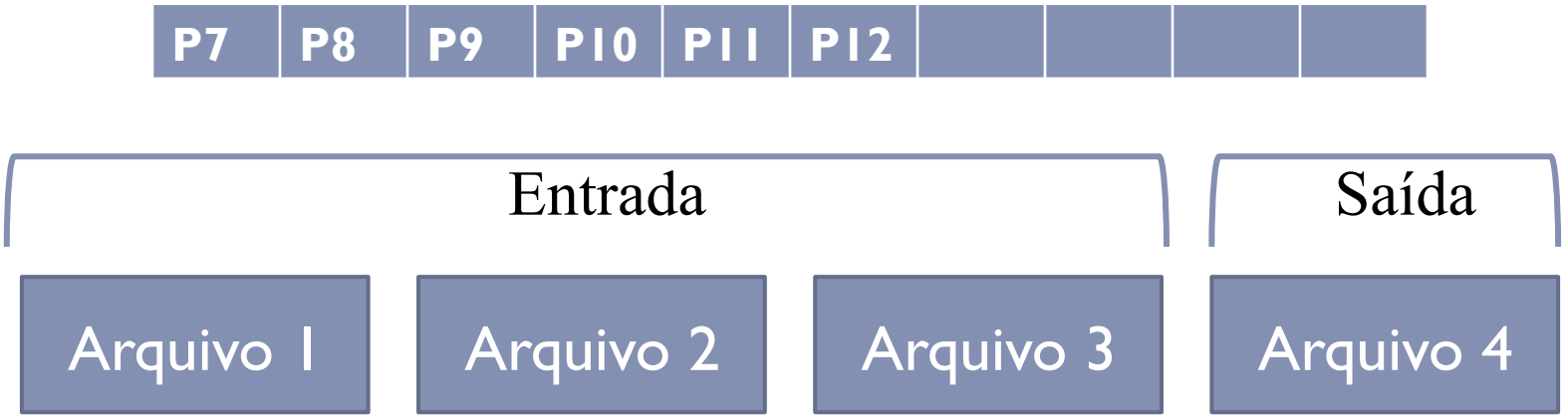
Exemplo (F = 4 e 10 partições):



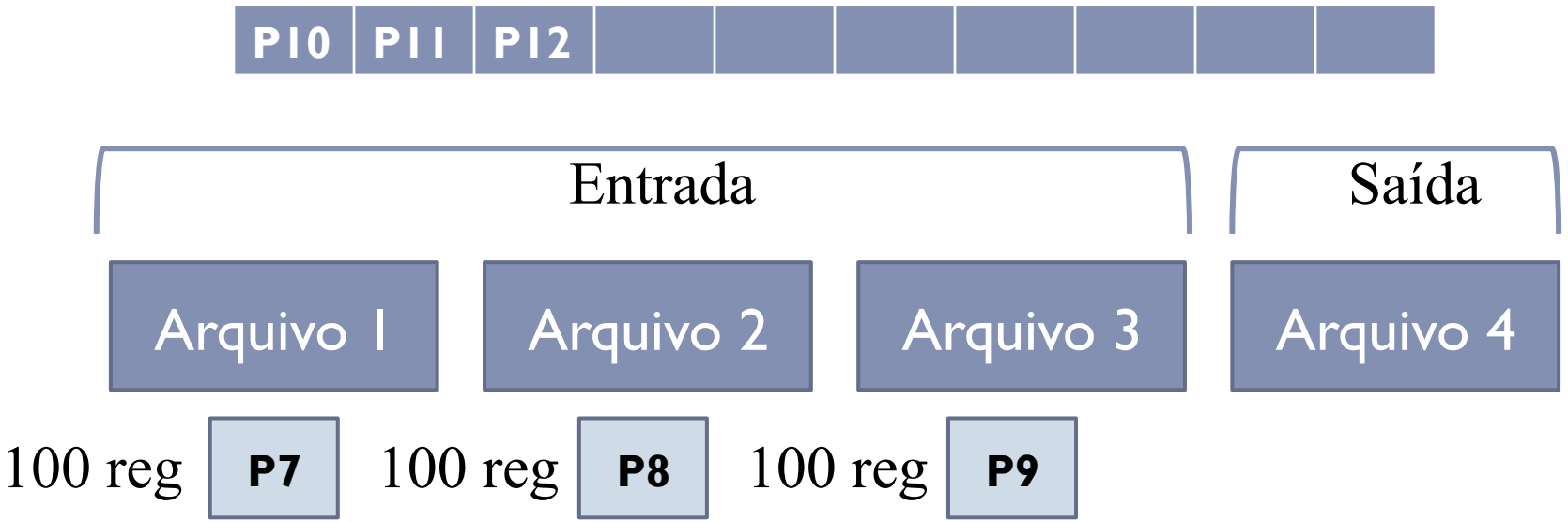
Exemplo (F = 4 e 10 partições):



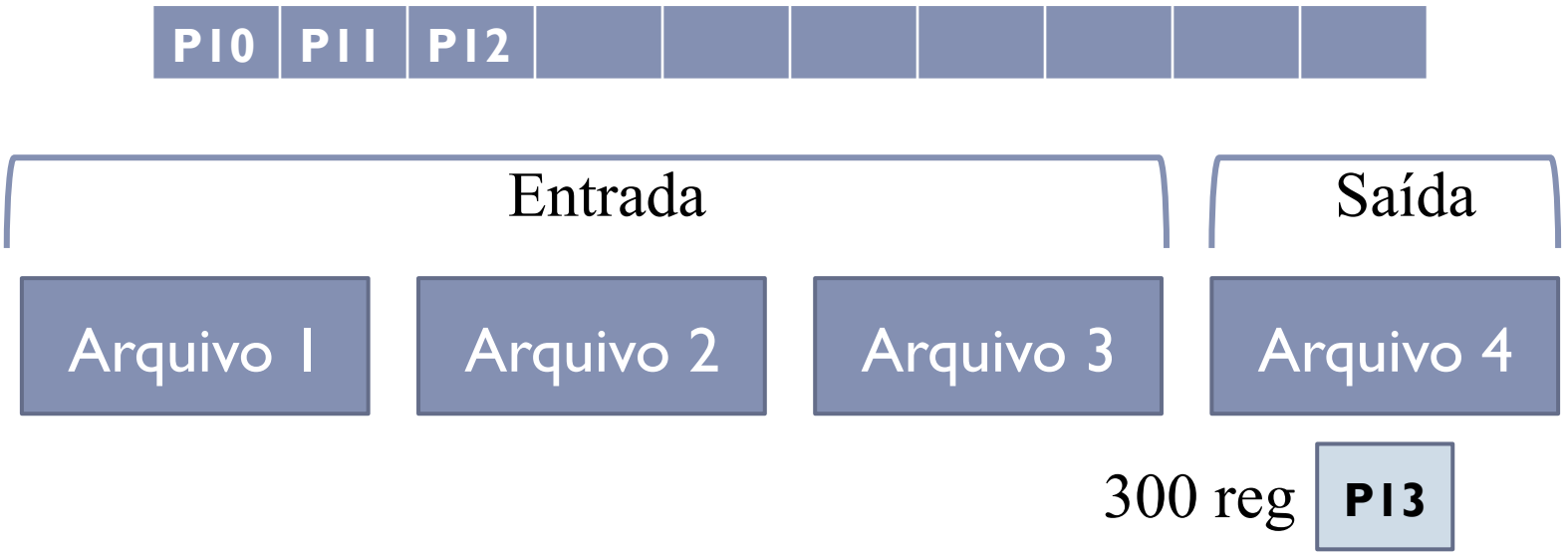
Exemplo (F = 4 e 10 partições):



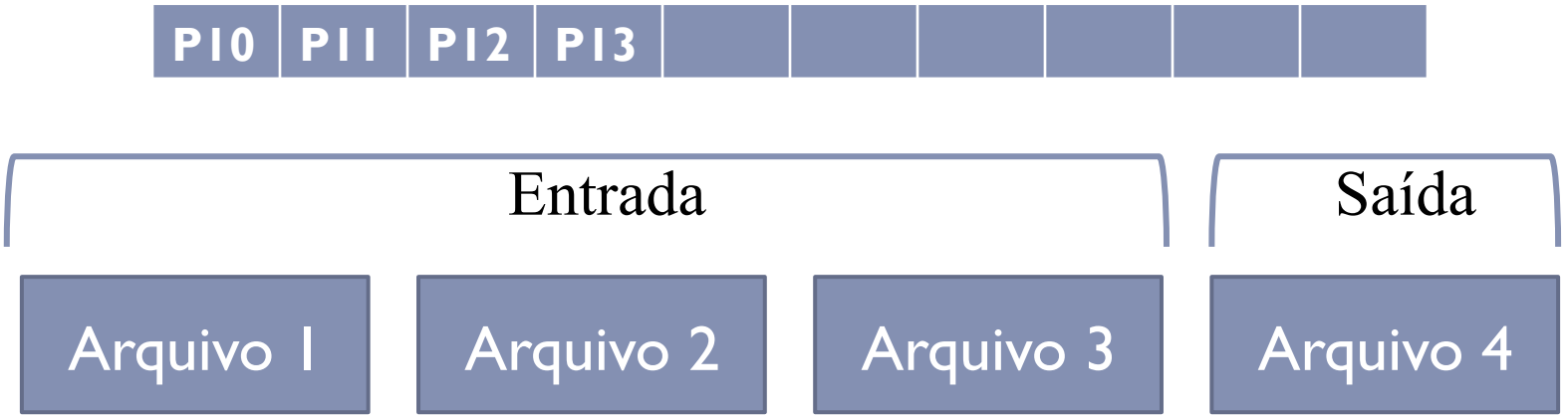
Exemplo (F = 4 e 10 partições):



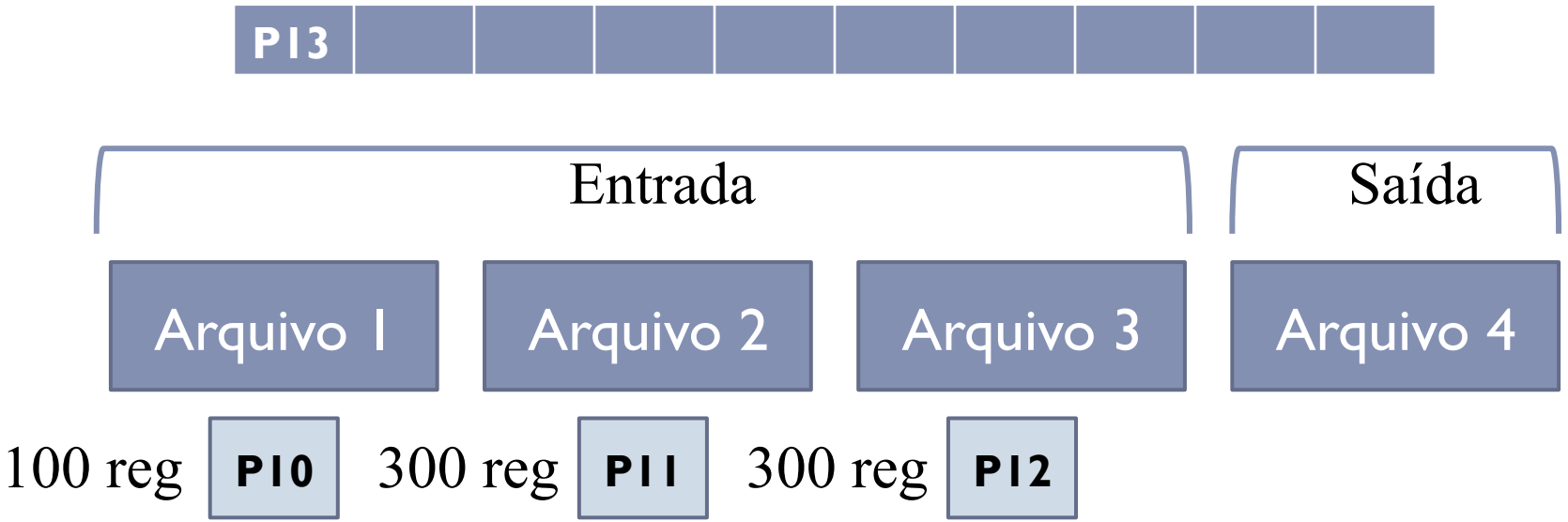
Exemplo (F = 4 e 10 partições):



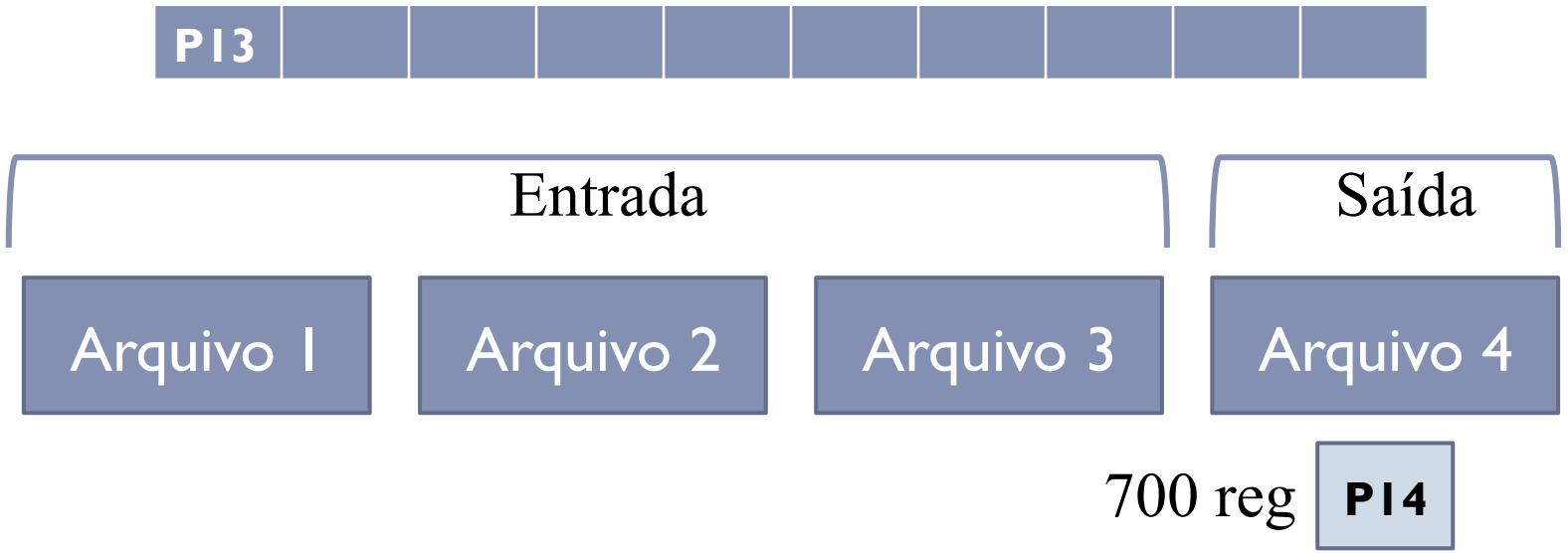
Exemplo (F = 4 e 10 partições):



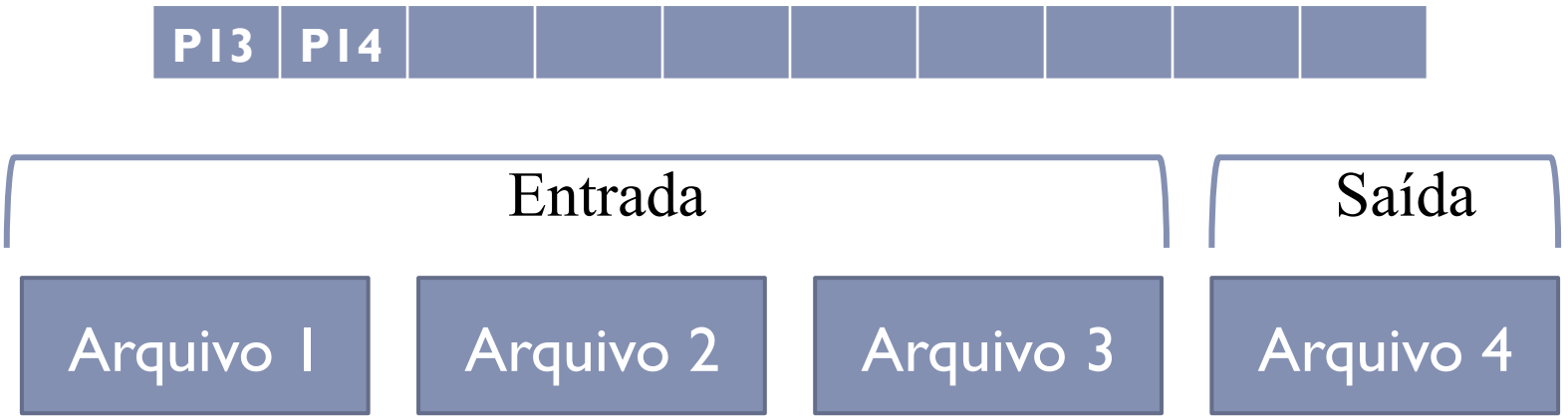
Exemplo (F = 4 e 10 partições):



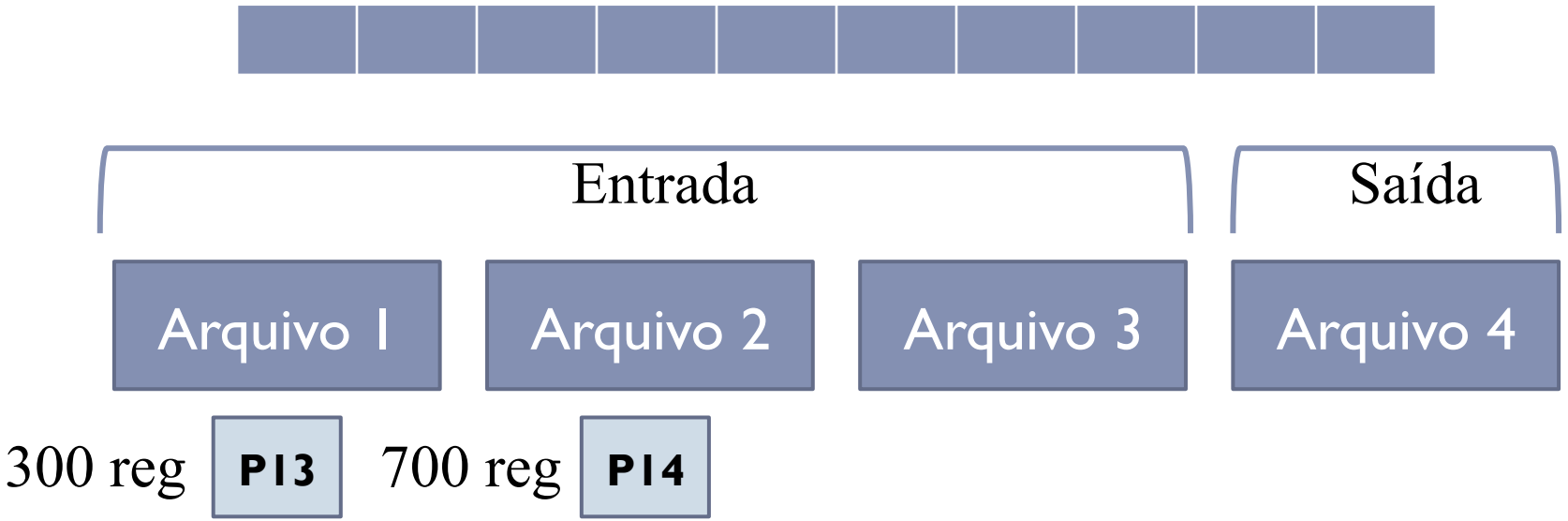
Exemplo (F = 4 e 10 partições):



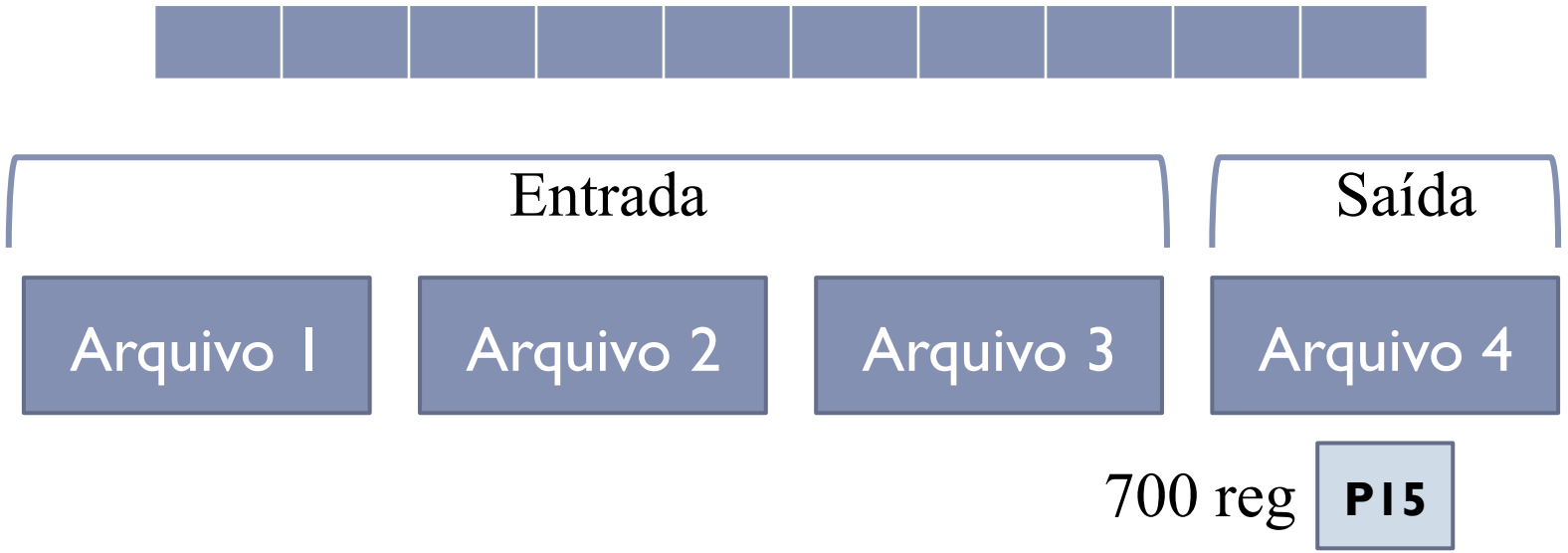
Exemplo (F = 4 e 10 partições):



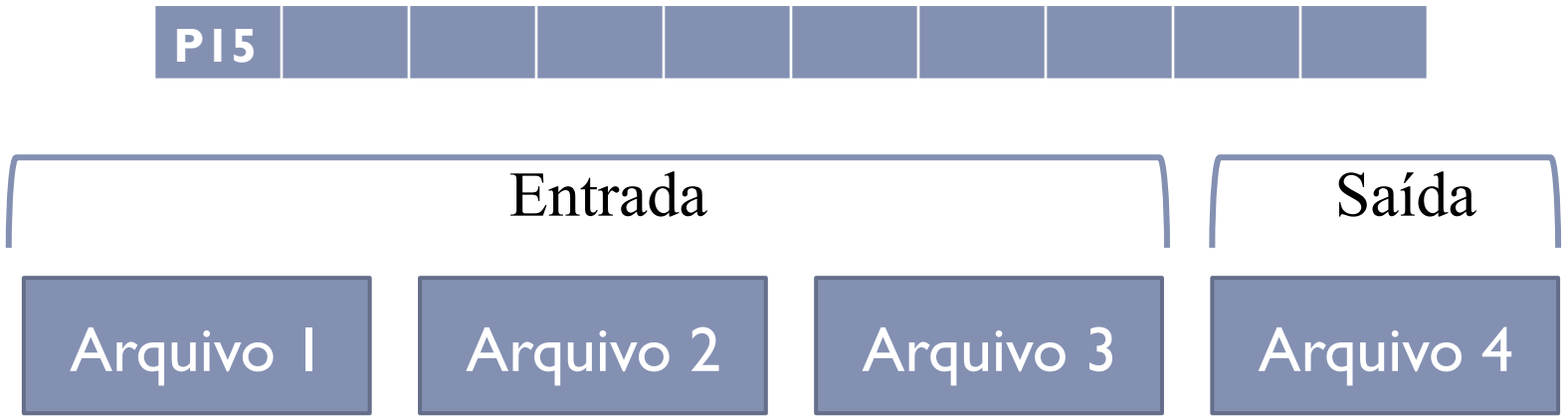
Exemplo (F = 4 e 10 partições):



Exemplo (F = 4 e 10 partições):



Exemplo (F = 4 e 10 partições):



- DICA: agora basta renomear a partição P15 para o nome do arquivo de saída desejado
- Em Java, pode-se usar o método renameTo da classe File

Implementação

- Problema: fazer intercalação de N partições ordenadas para gerar um único arquivo ordenado
- Restrição: Sistema Operacional pode lidar com apenas 4 arquivos ao mesmo tempo
- Entrada:
  - Lista com nomes dos arquivos a intercalar
- Saída:
  - Arquivo resultante ordenado, chamado "saida.dat"