Ciências ULisboa | Informática

Sistemas Operativos

2022/23

Trabalho de projeto – 2ª Fase

Objetivo

Continuação da realização do trabalho de programação em Python envolvendo a sincronização de processos, a manipulação de ficheiros e o tratamento de sinais e alarmes.

Descrição do trabalho

Este trabalho pretende estender o comando `pgrepwc` (**apenas a versão desenvolvida com processos**, na 1ª Fase trabalho de projeto de avaliação) com algumas funcionalidades adicionais.

Pretende-se que os alunos concretizem o comando `pgrepwc` descrito de seguida:

NOME

`pgrepwc` – pesquisa uma determinada palavra num ou mais ficheiros, escrevendo na saída as linhas de texto que contêm pelo menos uma ocorrência isolada da palavra. O comando também pode escrever na saída o número total de ocorrências isoladas da palavra, bem como o número total de linhas onde esta foi encontrada. O nível e a forma de paralelização da procura e contagem de ocorrências são determinados pelo utilizador.

SINOPSE

```
./pgrepwc [-c] [-l] [-p n] [-e k] palavra {ficheiros}
```

DESCRIÇÃO

Todas as opções e parâmetros funcionam como especificado na 1ª fase do projeto. Com exceção de duas:

`-t`: esta opção que existia na 1ª fase deixa de ser válida.

`-e k`: opção que permite ativar o modo de paralelização especial, que agora funciona com múltiplos ficheiros. O parâmetro `k` define o **número máximo de bytes** que compõem um bloco de trabalho.

Às funcionalidades já descritas na 1ª Fase do trabalho de projeto, os alunos devem melhorar as já implementadas e adicionar as seguintes.

Divisão mais equitativa de ficheiros pelos processos: quando executado na paralelização convencional (i.e., sem a opção `-e`), o processo pai deve distribuir os ficheiros pelos processos tendo em conta o tamanho dos ficheiros de tal forma que os "filhos" realizem uma quantidade de trabalho tão similar quanto possível, sem dividir um ficheiro por vários processos.

Por exemplo, considere a seguinte i

nvocação do programa com os ficheiros com os tamanhos definidos na tabela ao lado:

```
./pgrepwc -c -p 3 teste f1.txt f2.txt f3.txt f4.txt f5.txt f6.txt
```

Na 1ª fase do projeto, uma invocação como essa deveria distribuir dois ficheiros por cada processo. Se essa distribuição fosse feita em *round robin*, o processo 1 teria de buscar a palavra em `f1.txt` e `f4.txt` (totalizando 120kB), enquanto o processo 2 fica com os ficheiros `f2.txt` e `f5.txt` (totalizando 50kB) e o processo 3 trabalharia nos ficheiros `f3.txt` e `f6.txt` (totalizando 5kB).

Ficheiro	Tamanho
`f1.txt`	80kB
`f2.txt`	20kB
`f3.txt`	1kB
`f4.txt`	40kB
`f5.txt`	30kB
`f6.txt`	4kB

Nessa 2ª fase, a distribuição deve feita de forma mais equitativa, por exemplo: o Processo 1 teria de buscar a palavra em `f1.txt` (totalizando 80kB), o Processo 2 ficaria com os ficheiros `f3.txt`, `f4.txt` e `f6.txt` (totalizando 45kB) e o Processo 3 trabalharia nos ficheiros `f2.txt` e `f5.txt` (totalizando 50kB).

Note que este tipo de distribuição pode ser feito através da seguinte heurística:

1. Defina uma lista de ficheiros atribuídos para cada processo, inicialmente vazia.
2. Ordene os ficheiros por tamanho, do maior para o menor e coloque-os numa lista *L*.
3. Remova o maior ficheiro de *L* e atribua-o ao processo cujo tamanho total dos ficheiros atribuídos seja menor.
4. Repita o passo 3 até que *L* esteja vazia.

Paralelização especial: quando a opção `-e` é usada, o programa executa uma paralelização por blocos de linhas e não por ficheiros. Neste modelo, o processo pai deve ler todos os ficheiros e distribuir blocos de trabalho com um número variável de linhas, mas que nunca deve ultrapassar os `k` bytes definidos no argumento do programa. Mais precisamente, a ideia será concretizar um mecanismo produtor-consumidores[1] onde um produtor (o processo pai) produz blocos de trabalho (sequência de linhas lidas dos ficheiros, totalizando no máximo `k` bytes) que serão processados por `n` consumidores (processos filhos), de forma similar ao feito na paralelização convencional.

A fila de trabalho a ser distribuído deve ser configurada de tal forma a nunca conter mais de 1MB de dados.

Note que neste modelo, os ficheiros serão processados por múltiplos processos, desta forma, as mensagens a serem impressas pelo processo pai têm de ser separadas por ficheiros.

Tratamento do sinal *Ctrl+C* (SIGINT): caso o processo pai receba o sinal SIGINT (i.e., CTRL+C), o processamento dos ficheiros deve terminar corretamente, isto é, os processos devem concluir o processamento nos ficheiros correntes e terminar de seguida. Por seu turno, o processo pai escreve para *stdout* o número de ocorrências encontradas de cada a palavra a pesquisar ou o número de linhas onde cada palavra foi encontrada até ao momento, considerando apenas os ficheiros que foram processados pelos processos.

Impressão de resultados parciais: A cada 3 segundos o processo pai deve escrever para *stdout* o estado da contagem até ao momento, com a seguinte informação: (1) número de ocorrências da palavra e/ou número de linhas resultantes da pesquisa; (2) número de ficheiros completamente processados; (3) número de ficheiros em processamento; (4) tempo decorrido desde o início da execução do programa (em microssegundos).

Note que a função de tratamento de alarmes temporais tem de adquirir os *locks* apropriados para ter acesso às variáveis necessárias para imprimir as informações acima mencionadas.

Desafios

- Como dividir o trabalho entre os filhos de forma mais equitativa possível sem ter de ler o ficheiro mais de uma vez durante a execução do programa?
- Como concretizar a paralelização especial garantindo um uso de memória limitado?
- Como fazer a impressão não-intercalada das linhas encontradas nos ficheiros?
- Como fazer a impressão de resultados parciais a cada 3 segundos de forma consistente?

Ficheiros iniciais e de teste

Juntamente com o enunciado do projeto, serão disponibilizados um ficheiro ZIP (`grupoXX.zip`) com a estrutura inicial do projeto e quatro ficheiros de texto que servirão para os alunos testarem as duas soluções do comando `pgrepwc`. Os alunos terão de descarregar estes ficheiros para a sua máquina. Não os deverão abrir no Moodle, principalmente o ficheiro `file1.txt` por este ser grande (~500 MB).

[1] Sugere-se o uso de uma `multiprocessing.Queue` para concretizar esse mecanismo.

Entrega

A entrega do trabalho é realizada da seguinte forma:

- Os grupos inscrevem-se atempadamente, de acordo com as regras afixadas para o efeito, no Moodle.
- Colocar os ficheiros `pgrepwc` e `pgrepwc_processos.py` do projeto numa diretoria cujo nome deve seguir exatamente o padrão **grupoXX** (por exemplo grupo01 ou grupo23). Juntamente com os dois ficheiros, incluir um ficheiro de texto README.txt (não é .pdf nem .rtf nem .doc nem .docx) que deve conter:
    - A identificação dos elementos do grupo;
    - Exemplos de chamadas do comando `pgrepwc`;
    - As limitações da implementação;
    - A abordagem usada para a divisão dos ficheiros pelos processos;
    - Outras informações que acharem pertinente sobre a implementação do projeto.
- A diretoria será incluída num ficheiro ZIP cujo nome deve seguir exatamente o padrão **grupoXX.zip**. Esse ficheiro deverá ser submetido no Moodle (um por grupo).

De notar que **a entrega deve conter apenas a diretoria com o ficheiro `pgrepwc`, o ficheiro .py e o ficheiro README.txt, pois qualquer outro ficheiro será ignorado**.

Se não se verificar algum destes requisitos o trabalho é considerado não entregue.

Não serão aceites trabalhos entregues por mail nem por qualquer outro meio não definido nesta secção.

Prazo de entrega

O trabalho deve ser entregue até dia **11 de dezembro de 2022 (domingo) às 23:59h**.

Avaliação dos trabalhos

A avaliação do trabalho será realizada:

(1) pelos alunos, pelo preenchimento do formulário de contribuição de cada aluno no desenvolvimento do projeto. O formulário será disponibilizado no Moodle e preenchido após a entrega do projeto.

(2) pelo corpo docente, em uma discussão presencial com cada grupo a ser realizada entre 12 e 16 de dezembro de 2022. Todos os elementos do grupo terão de comparecer à avaliação que será feita **individualmente**. Deste modo, cada elemento do grupo deve estar preparado para responder a qualquer questão relacionada com os trabalhos e com a matéria das aulas teórico-práticas.

Os seguintes parâmetros serão avaliados: funcionalidade, estrutura, desempenho, algoritmia, comentários, clareza do código, validação dos parâmetros de entrada e tratamento de erros.

Divulgação dos resultados

A data prevista da divulgação dos resultados da avaliação dos trabalhos é 31 de dezembro de 2022.