Googla-lhes 2019

Neste projeto pretende-se que seja desenvolvido uma *desktop grid* que permita efetuar a procura de palavras num grande conjunto de notícias.

Computação em Grelha (Grid)

A computação em grid é uma forma distribuída de computação onde existem diversos trabalhadores (Workers) que executam as tarefas definidas por um cliente. No caso de uma desktop grid estes trabalhadores correm nos computadores de uma organização quando estes não estão a ser utilizados.

Para além dos trabalhadores existe um servidor que garante toda a gestão da atribuição dos trabalhos a serem efetuados aos workers e de devolver as respostas aos clientes.

Os clientes ligam-se ao servidor e enviam-lhe as tarefas que pretendem ver executadas pelos workers e esperam que o servidor lhes envie os resultados da execução.

Descrição do Sistema

A aplicação a ser implementada permite usar uma desktop grid para a procura de palavras num grupo de noticias. Assim o utilizador deve correr um cliente que lhe permite inserir as palavras e iniciar a procura. O cliente deve enviar esta palavra ao servidor que irá criar tarefas que são executadas pelos workers. Cada tarefa consiste na pesquisa de uma expressão ou frase no texto de uma notícia. O worker deve devolver ao servidor uma lista que contém todos os índices das ocorrências da palavra no texto da notícia. Após todas as tarefas terem sido executadas, o servidor agrupa os resultados e envia ao cliente. A informação que o servidor deve enviar ao cliente consiste numa lista com os títulos das notícias em que a palavra ocorre bem como uma lista dos índices das ocorrências para cada uma dessas notícias. Após recebidas os títulos das notícias onde a palavra aparece, o cliente mostra ao utilizador estes resultados. Quando o utilizador selecicona uma das noticias da lista, o cliente deve enviar uma mensagem ao servidor a pedir o texto da notícia.

Quando o servidor arrancar deve ler todas as notícias que fazem parte do corpus. Estas notícias encontram-se num conjunto de ficheiros numa pasta que é passada ao servidor. O servidor ao receber um novo pedido de pesquisa vai criar um conjunto de tarefas, uma para cada notícia, que consiste em procurar a expressão numa das notícias.

Servidor

A solução a desenvolver deverá seguir uma arquitetura cliente-servidor. Deverá existir um *servidor* ao qual se vão ligar os vários clientes e os vários workers. O servidor deverá aceitar pedidos de conexão por parte de clientes e dos workers, através de uma ligação TCP/IP num porto com um número bem definido (e do conhecimento dos clientes e dos workers). Ao ser estabelecida uma ligação, os clientes e os workers informam o servidor se são clientes ou workers. Para efeitos de simplificação, não é exigido nenhum processo de autenticação dos clientes e dos workers.

O servidor deverá funcionar em *multi-threading*, permitindo receber novas ligações, receber pedidos dos vários clientes e enviar tarefas para os workers. Para isso deve criar um conjunto de threads que permitam ao servidor executar todas estas tarefas de uma forma concorrente.

O servidor deverá poder ser lançado num processo Java normal, por exemplo executando:

```
java Server pasta_notícias
```

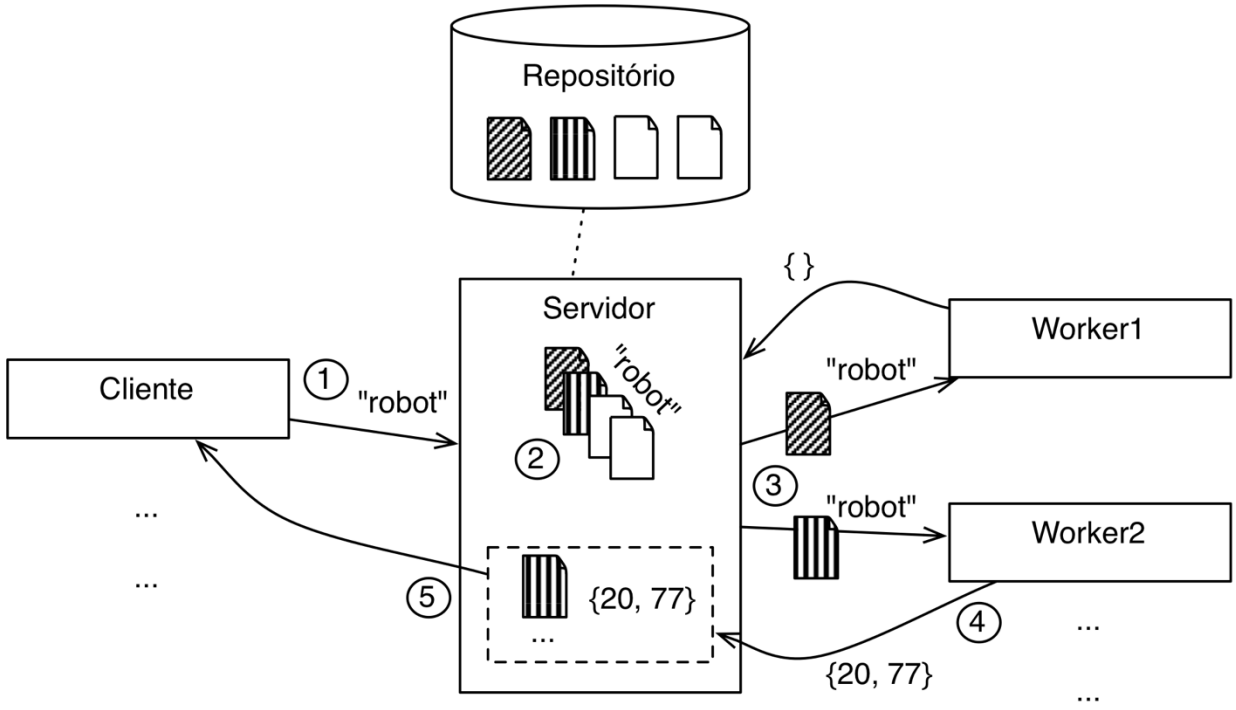
O Servidor é iniciado e carrega em memória todas as notícias constantes na pasta_notícias. Após ler as notícias o servidor fica disponível para receber ligações de clientes e workers.

Se um cliente se ligar, o servidor deve ficar disponível para aceitar novos pedidos deste cliente. Quando recebe um destes pedidos deve criar um conjunto de tarefas a submeter aos workers. Cada uma destas tarefas consiste na procura da expressão em cada uma das noticias lidas da pasta de notícias. As notícias onde foi encontrada a expressão a procurar, devem ser ordenadas pela relevância, número de vezes que a palavra ocorre, e enviadas ao cliente.

Se um worker se ligar o servidor deve ficar disponível para receber pedidos de tarefas. Caso receba um pedido deve enviar a próxima tarefa que ainda não foi executada para o worker e esperar pelo resultado. Quando receber o resultado este deve ser colocado na lista de resultados.

Quando todos os resultados da pergunta de um determinado cliente tiverem chegado ao servidor este deve devolver a resposta ao cliente.

Exemplo: Assumindo que apenas existem 4 notícias previamente carregadas pelo servidor e que existe um cliente (cliente) e dois workers (worker1 e worker2) já ligados ao servidor:



- Os worker 1 e 2 pedem tarefas e ficam bloqueados pois não existem tarefas.
- O cliente envia uma mensagem ao servidor a pedir os documentos com a palavra “robot” e fica à espera da resposta (1).
- Servidor cria 4 tarefas, uma para cada notícia, e coloca-as na lista de tarefas por executar. Cada tarefa consiste em procurar a palavra “robot” no texto da notícia. O cliente fica à espera do resultado da execução das 4 tarefas (2).
- A existência de tarefas permite enviar uma tarefa a cada worker (3).
- Os workers procuram todas os índices da palavra "robot" no texto da notícia.
- Após processadas as tarefas os workers enviam o resultado ao servidor que os adiciona à lista de tarefas prontas do cliente (4).
- Os workers pedem novas tarefas e recebem as restantes tarefas.
- Quando o servidor recebe os últimos resultados da pergunta, ordena todos os resultados pelo número de vezes que a palavra aparece, e envia-os ao cliente (5).
- O cliente mostra ao utilizador o resultado.

Cliente

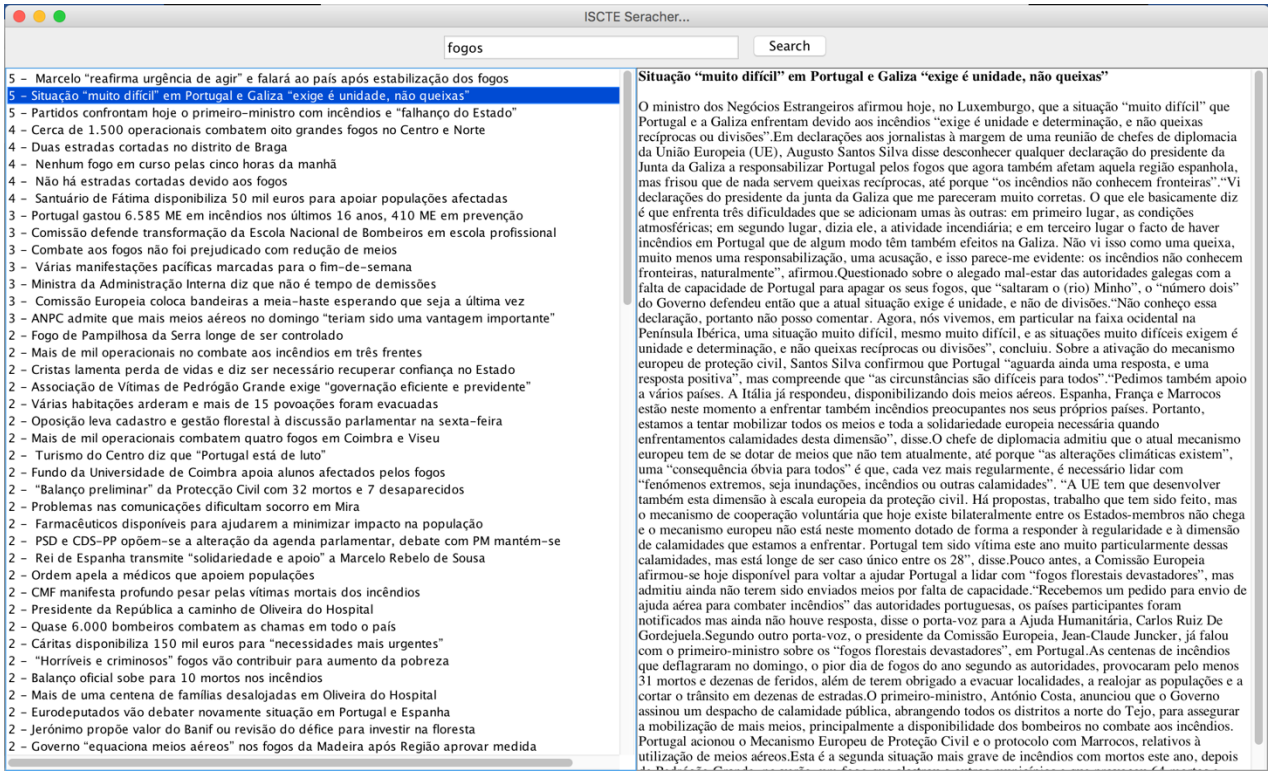
O cliente deverá poder ser lançado num processo Java normal, por exemplo executando:

```
java Cliente endereço_servidor
```

Onde o endereço_servidor é o endereço IP da máquina onde está a correr o servidor. Quando o cliente é lançado, deve ligar-se ao servidor e enviar a indicação de que é um cliente. Após a inicialização deve mostrar a interface gráfica que deve conter os seguintes elementos:

- Caixa de texto onde o utilizador pode escrever a expressão a procurar
- Botão para se iniciar a procura
- Lista de notícias resultado da procura
- Área de texto onde aparece o texto da notícia seleccionada da lsita de resultados

A interface do cliente deve ter o seguinte aspecto:



Na lista de respostas deve ser possível ver o nº de vezes que aparece a palavra na notícia bem como o título da notícia.

Deve implementar as threads necessárias para que a aplicação cliente possa esperar pelos resultados sem bloquear a interface gráfica.

Worker

O worker deverá poder ser lançado num processo Java normal, por exemplo executando:

```
java Worker endereço_servidor
```

Onde o endereço_servidor é o endereço IP da máquina onde está a correr o servidor. Quando o worker é lançado deve ligar-se ao servidor e enviar a indicação de que é um worker. Após ter feito a inicialização deve repetir os seguintes passos:

1. esperar por uma nova tarefa,
2. executar a tarefa,
3. devolver o resultado da sua execução ao servidor.

Caso a ligação ao servidor falhe o worker deve tentar reestabelecer esta ligação ignorando o trabalho que entretanto possa ter efectuado.

Fases, Avaliação, e Entrega

São propostas as seguintes fases de desenvolvimento como metas para a avaliação, e de forma a que seja mais fácil abordar o problema:

Fase 1: Desenvolver uma versão local sem threads e com a interface gráfica do cliente.

Fase 2: Separar versão anterior em Cliente e Servidor.

Fase 3: Fazer a execução de cada tarefa numa thread diferente dentro do servidor.

Fase 4: Desenvolver o worker.

Devem ser usados mecanismos de coordenação que garantam o correto funcionamento de todas as aplicações. Implemente todos estes mecanismos que necessitar, como por exemplo listas bloqueantes, barreiras ou semáforos.

As notas do projeto serão atribuídas de acordo com a realização das fases propostas:

- **A e B**: completar todas as fases,
- **C** : completar as fases (1, 2, 3),
- **D** : não são cumpridos os requisitos mínimos (reprovação à UC).

Grupos: Cada grupo de trabalho é composto por dois alunos, preferencialmente da mesma turma prática.

Entrega Intercalar : Para a entrega intercalar é necessário terminar a fase 1.

Entrega Final: O projeto desenvolvido deve ser entregue sob a forma de um projeto arquivado usando a funcionalidade de *Export/Archive File* do Eclipse. As entregas serão feitas no e- learning até às 24h de dia 24 de Maio. Os grupos deverão comparecer na última semana à aula prática onde estão inscritos para realizarem a discussão do trabalho.