Distributed Map/Reduce

Guião de Avaliação 2

João Santos - 73761
Diego Trovisco - 80228

Coordinator (1/2)

O Coordinator quando é iniciado começa por "partir" o texto em blobs e introduz as mesmas numa queue com o nome datastore_q.

De seguida pergunta quantos workers pretendemos utilizar para realizar o trabalho de map/reduce. Depois de introduzido o valor este fica à espera dos workers para dar então seguimento à distribuição de tarefas.

Os Workers ligam-se ao Coordinator através de mensagens de Register. Sendo que Coordinator fica bloqueado enquanto não tiver a totalidade dos Workers a si ligada. O Coordinator cria uma thread com cada Worker onde é feita a troca de mensagens entre ambos. As tarefas são assim distribuídas de forma indistinta pelos diferentes Workers.

O Coodinator envia para o Worker duas task's distintas:

- "map_request" - para o Worker fazer o Map do blob enviado;
- "reduce_request" - para o Worker fazer o Reduce do value enviado.

O Coodinator recebe do worker duas task's distintas:

- "map_reply" - o Worker envia quando o Map está concluído;
- "reduce_reply" - o Worker envía quando o Reduce está concluido.

Coordinator (2/2)

Quando é recebida uma mensagem de map_reply de um Worker existem duas hipóteses:

1. datastore_q está vazio, ou seja, não existem mais blobs que necessitam de Map:
   - O Coordinator começa a enviar o "reduce_request" dos Maps.
2. datastore_q não está vazio, ou seja, existem blobs por precessar :
   - O "coordinator" continua a enviar o "map_request" até que não existam mais blob's para executar o Map.

Se a mensagem recebida do Worker é de "reduce_reply" então :

1. o self.map_responses está vazio:
   Se está vazio, este verifica se o self.reduce_responses › 1, ou seja, se tem mais que uma lista para fazer reduce, se sim continua a fazer o "reduce_request". Se o self.reduce_responses == 1, significa que já acabou todo o trabalho e faz a escrita para o ficheiro com as palavras ordenadas por ordem alfabética.
2. o self.map_responses ainda não está vazio:
   Se não está vazio, este vai verificar o tamanho do self.map_responses. Se o tamanho for igual a 1, o "coodinator" vai enviar um "reduce_request" do último mapa e do primeito reduce recebido. Se o tamanho for › 1 envia "reduce_request" ainda dos mapas que se encontram no self.map_responses.

Worker

O Worker liga-se ao Coordinator através de uma mensagem de Register. De seguida a única tarefa dele é receber as mensagens via thread que foi criada pelo Coordenador, processar o seu conteúdo e enviar de volta para o Coordinator.

Nesta entidade o trabalho de Map e Reduce é feito da seguinte maneira:

- MAP - `def handle_map_request(self, blob):`

  Trabalha sobre um blob recebido do Coordinator, este blob é separado por os espaços através da função .split(), depois são eliminadas as pontuações e dígitos que se encontrem no texto. Cada palavra é guardada numa lista e formatada em forma de tuplo de forma à lista final ficar com o tuplo com a palavra e o número 1.
- REDUCE - `def handle_reduce_request(self, value):`

  A lista value é uma lista de listas, onde cada lista contém tuplos. Esse tuplo contém uma palavra e um número. A palavra é colocada em minúscula e é feita a verificação se a palavra já existe na lista words. Caso exista é incrementada ao valor de vezes que esta apareceu anteriormente, caso não exista é apenas adicionada.

No final de processadas as mensagens recebidas pelo Coordinator, são enviados os objetos JSON de volta em forma de dicionário. `json.dumps(dict(task="reduce_reply", value=reduced_list))`

Características do Sistema

Troca de mensagens com tamanho dinâmico:

```
size = len(map_reply)
self.sock.sendall((str(size).zfill(8) + map_reply).encode("utf-8"))
```

É guardado o tamanho da mensagem a enviar, de seguida atribuímos o padding para quando é feito o recv pela outra entidade ela poder saber quantos bytes deve ler.

```
bytes_size = self.sock.recv(8).decode()
xyz = int(bytes_size)
json_msg = self.sock.recv(xyz).decode("utf-8")
```

Utilização de Queues

A utilização de queues para guardar os diferentes blobs, Maps e Reduces revelou-se bastante facilitadora de todo o processo de troca de mensagens entre o Coordinator e o Worker.

Dificuldades

Com apenas 1 Worker conseguimos obter todos os valores no ficheiro csv como seria suposto, quando adicionamos mais workers por vezes algum vai a baixo por causa de algum problema de formatação e assim não conseguimos obter os valores correctos.