Relatório Projeto de Computação Distribuída

Leonardo Almeida 102536
Pedro Rodrigues 102778

Neste relatório, será apresentada de forma muito breve uma pequena análise do projeto desenvolvido, mais concretamente as diferentes decisões tomadas, bem como a sua justificação. Serão também apresentados outros aspetos que consideramos relevantes, como por exemplo a topologia da rede P2P, o protocolo definido, a eleição da melhor imagem, etc.

Topologia da rede P2P

A rede P2P criada consiste numa rede ***full mesh***, quer isto dizer que todos os nós da rede desenvolvida encontram-se ligados diretamente a todos os outros existentes nessa rede. Dada a escala deste projeto, ou seja, o número total de *daemons* é relativamente baixo (comparando com uma situação real), a utilização deste tipo de topologia não influenciará o desempenho da rede, no entanto, é importante realçar que num caso real justifica-se a utilização de redes mais eficientes, como por exemplo o *CHORD* com *finger table*.

Protocolo de transporte

O protocolo de transporte utilizado foi TCP/IP, ao invés de UDP. Esta escolha deveu-se ao facto das inúmeras vantagens que este protocolo de transporte apresenta e que se justificam no projeto desenvolvido, nomeadamente:

1. mais fiável
2. orientado à conexão e por isso possui mecanismos para iniciar, manter e encerrar a conexão
3. existe fragmentação de pacotes
4. entrega ordenada dos pacotes
5. possui mecanismos de verificação e detecção de falhas
6. possui mecanismos de confirmação da receção de mensagens

Escolha de imagens

Sempre que imagens têm o mesmo hash, a melhor é escolhida tendo em conta o número de cores diferentes que contém e o número de pixels.

Consideramos 2 tipos de imagens em relação ao número de cores: imagens coloridas, que considerámos como tendo mais de 70000 cores diferentes e imagens não coloridas. O critério de escolha da melhor imagem é dado por:

- Caso ambas sejam do mesmo tipo em relação ao número de cores, é escolhida a que tiver um maior número de pixels
- Caso a imagem A seja colorida e B não, é escolhida a A se esta tiver mais que 70% dos pixeis de B e vice-versa.

Protocolo utilizado

O protocolo desenvolvido surge com o intuito de permitir a comunicação entre os diferentes nós da rede através da troca de mensagens com diferentes fins. As mensagens trocadas pelos diversos nós são **codificadas em *pickle*** e, dependendo da mensagem, encapsulam estruturas de dados importantes, tais como:

general_map:

{ hashkey: [(address1, port1), (address2, port2), num_colors, num_pixeis, num_bytes] }

Guarda informação sobre as imagens, tal como o *daemon* principal e backup que a contém.

update:

[(address1, port1), (address2, port2), num_colors, num_pixeis, num_bytes, hashkey]

Sempre que há uma alteração no *general_map* esta é enviada para outros *daemons* através desta mensagem.

image:

As imagens são enviadas como objeto **Image** obtido através da biblioteca PIL.

Em relação ao protocolo temos as seguintes mensagens:

first_connect:

{"**type**": "first_connect", "**node_type**": node_type, "**recv_host**": recv_host, "**recv_port**": recv_port}

Utilizada quando algo entra na rede.

node_type → client ou *daemon*
(recv_host, recv_port) → endpoint do nó que entrou na rede

connect:

{"**type**": "connect", "**node_type**": node_type, "**recv_host**": recv_host, "**recv_port**": recv_port}

Utilizada quando um *daemon* se conecta a outro sem ser quando entra na rede.

connect_ack:

{"**type**": "connect_ack", "**nodes**": nodes}

Envia uma lista com todos os *daemons* ao novo *daemon* que entrou na rede.

req_general_map

{"**type**": "req_general_map"}

O novo *daemon* pede o general_map para saber o que existe na rede.

general_map:

{"**type**": "general_map", "**general_map**": general_map}

O *general_map* é enviado para o novo *daemon*.

request_list:

{"**type**": "request_list"}

Utilizada quando o cliente pede a lista de hash das imagens disponíveis na rede.

image_list:

{"**type**": "image_list", "**image_list**": image_list}

Utilizada para enviar ao cliente a lista de hash de imagens disponíveis.

request:

{"**type**": "request", "**hashkey**": hashkey}

Utilizada quando um cliente pede uma imagem.

request_ack:

{"**type**": "request_ack", "**image**": image}

Devolve a imagem pedida.

image_backup:

{"**type**": "image_backup", "**update**": update, "**image**": image}

Enviada para o *daemon* que vai conter o backup de uma imagem.

update:

{"**type**": "update", "**update**": update}

Sempre que um *daemon* alterar o *general_map* irá enviar esta mensagem para os restantes *daemons*.

Informação relevante

Cada imagem é identificada pela sua *hash*, quer seja entre *daemons* e clientes, quer seja na forma como é armazenada localmente.

A escolha do *daemon* de backup da imagem é **determinada pelo armazenamento ocupado**, ou seja, é escolhido o que tiver menos espaço ocupado. Por isso, cada imagem vai estar armazenada em 2 nós e todos os *daemons* irão ter aproximadamente o mesmo espaço ocupado.

Quando um *daemon* se desliga os restantes irão refazer o *backup* das imagens de modo a esta continuar em 2 *daemons*.