Luiza Wille (GRR20141014)
Adolfo Tognetti (GRR20152278)

Relatório do segundo trabalho de Redes 1

Área de concentração: *Computer Science*.

Curitiba PR
2016

Sumário

Capítulo 1

Introdução

Neste trabalho o objetivo foi implementar um jogo de batalha naval de 4 jogadores usando socket datagram e redes em anel. Ou seja, 4 jogadores escolhem seu tabuleiro e as 4 maquinas diferentes se conectam de forma circular. Um dos jogadores inicia o jogo com o bastão, gerando uma mensagem de ataque. O jogo termina quando apenas 1 jogador permanece com algum navio(ou parte de um navio).

Para a implementação, usamos a linguagem Python adicionando a biblioteca “socket” para fazer a conexão entre as máquinas. O Timeout foi implementado para que mensagens cheguem ao destino mesmo que o destino tenha ficado desligado por algum tempo.

Capítulo 2

Sobre o funcionamento do programa

2.1 Conexão

Primeiramente cada máquina deve executar o programa principal separadamente com o comando “python main.py". A conexão será feita manualmente em cada uma das máquinas que se deseja conectar. Primeiramente, cada jogador deve informar qual o IP da maquina que enviara mensagens para esta e qual o IP da maquina que recebera as mensagens desta. Também deve ser informado o Port para a comunicação, e todas as maquinas devem ser ligadas no mesmo Port.

2.2 Tabuleiro e navios

O programa pergunta ao usuário qual o tamanho do tabuleiro, qual a quantidade de navios, os tamanhos e as posições de cada navio. Um exemplo de tabuleiro com navios é apresentado ao usuário da seguinte forma:

```
_| 0  1  2  3  4
0| -- n0 n0 n0 --
1| -- -- -- -- --
2| -- n1 -- -- --
3| -- n1 -- -- --
4| -- n1 -- -- --
```

O tabuleiro é enumerado para facilitar a visualização. O simbolo “- -” representa água(posições sem navio). “n0” simboliza uma parte do navio de número 0 e “n1” simboliza uma parte do navio de número 1.

2.3 Mensagem

As mensagens a serem enviadas são strings que têm o seguinte formato:

```
[TIPO]_[QUEM ENVIOU]_[PARA QUEM ENVIOU]_[POSICAO X]_[POSICAO Y]_[SITUACAO]
```

O campo “tipo” se refere ao tipo de mensagem enviada, que pode ser:

1 = ataque

2 = navio afundou

3 = jogador perdeu

4 = ataque acertou, mas navio não afundou

5 = ataque errou

6 = mensagem aberta(todas as máquinas devem lê-la) de que navio afundou

7 = mensagem aberta de que jogador perdeu

9 = bastão

“Quem enviou” se refere ao jogador remetente da mensagem, que em um jogo de 4 jogadores pode ser um número de 1 a 4.

“Para quem enviou” se refere ao jogador destinatário da mensagem, que em um jogo de 4 jogadores pode ser um número de 1 a 4.

“Posicao x” e “posicao y” representam a posição de ataque a um navio.

Dependendo da mensagem, ela deve ser lida por todos os jogadores(se for mensagem de tipo 6 ou 7) ou apenas pelo remetente(se for mensagem de algum outro tipo). Por isso, o campo “Situação” pode assumir um dos seguintes valores:

9 = NACK

0 = Nao lida

1 = ACK (ou lido por 1 jogador)

2 = Lido por 2 jogadores

3 = Lido por 3 jogadores

Como resposta às mensagens do tipo 1,2,3,4 e 5 a situação da mensagem deve ser alterada de 0 para 9 ou 1 e deve voltar para o jogador que a colocou na rede. Como resposta å mensagem do tipo 6 e 7, a situação dela deve ser alterada para 3(em um jogo de 4 jogadores), indicando que todos os outros jogadores leram a mensagem, ou 9 caso ocorra algum erro. Caso a mensagem do tipo 1,2,3,4 e 5 volte ao remetente com status 9 ou 0 a mensagem é reenviada e caso mensagem do tipo 6 e 7 volte com status diferente de 3 (se existem 4 jogadores) ela também é reenviada. Caso a mensagem não retorne ao remetente dentro de um período esperado de tempo, ocorre um timeout e a mensagem original é reenviada (ver próximo tópico).

2.4 Timeout

Para implementar o timeout das mensagens, usamos uma lista chamada “listaTimeout” que guarda as mensagens enviadas que ainda não receberam resposta do destinatário junto com o tempo em ticks máximo que se deve aguardar pela resposta daquela mensagem. Assim, no laço principal do programa, através da função “checaTimeouts” todas as mensagens guardadas em “listaTimeout” e seus timeouts são checados para ver se o tempo máximo de espera por uma

resposta já acabou. Caso o tempo tenha acabado, a mensagem é reenviada e o timeout para a resposta a ela é atualizado.

2.5 Ataque

Para um jogador colocar uma mensagem nova no anel, ele precisa ter o bastão. Por isso, apenas o jogador com o bastão realiza um novo ataque, repassando o bastão para o próximo jogador após ter terminado de enviar suas mensagens. A mensagem de ataque tem o formato:

`1_[atacante]_[atacado]_[POSICAO X]_[POSICAO Y]_0`

O tipo "1"indica uma mensagem de ataque e a situação "0"indica que a mensagem ainda não foi lida. As posições x e y são as posições que o atacante deseja atacar no tabuleiro do atacado. Quando um jogador recebe uma mensagem de ataque, caso ela esteja dentro do padrão esperado, ele altera a situação da mensagem para "1"que significa ACK e a envia pelo anel. O ataque é validado através da função “checaAtaque"que retorna o resultado do ataque. O atacado então coloca mais uma mensagem no anel, que deve chegar ao atacante e indica o resultado do ataque. Essa mensagem de resultado do ataque terá o formato:

`[2, 3, 4 ou 5]_[atacante]_[atacado]_[POSICAO X]_[POSICAO Y]_0`

O atacado também aguarda um ACK desta mensagem, da mesma maneira que acontece com a mensagem de ataque.

Se o atacante recebe uma mensagem do tipo 2 ou 3, ele deve colocar uma nova mensagem no anel do tipo 6 ou 7 respectivamente, que deve ser lida por todos os outros jogadores. Caso esta mensagem volte para ele com situação diferente de 3 (no caso de 4 jogadores), ele deve reenviar a mensagem pois significa que algum jogador ainda não a leu.

2.6 Fim de jogo

Um jogador perde quando todos os seus navios tiverem afundado, porém ele deve continuar conectado para repassar as mensagens dos outros jogadores que ainda estão no jogo. O jogo acaba quando resta apenas um jogador, o que é checado no laço principal através da função "acabouJogo"

2.7 Bugs conhecidos

- Se um jogador se desconectar por um tempo e se reconectar, as mensagens que não chegaram a ele e foram perdidas vão chegar através do mecanismo de timeout. Porém se o jogador desconectado receber o bastão, o bastão será perdido já que não existe timeout ou ACK/NACK para o bastão. Este bug é esperado e normal devido a proposta do trabalho.
- O fim de jogo e término dos programas nas máquinas não está acontecendo como o esperado