Trabalho II de Redes

Copas utilizando Rede em Anel

Thiago José Barzotto

Prof. Albini, Prof. Todt

1. Introdução

A proposta do trabalho era criar um jogo de Copas através de uma Rede em Anel (Token Ring) com passagem de bastão implementada com sockets DGRAM. O trabalho foi inteiramente desenvolvido em Python.

2. O Jogo

Em Copas, todos os jogadores receberão 13 cartas. O primeiro jogador deverá jogar uma carta de sua mão e o restante dos jogadores deve jogar uma carta do mesmo naipe daquela jogada pelo primeiro jogador (se o jogador não possuir a carta do mesmo naipe, ele poderá jogar qualquer carta). Ao fim da rodada, o jogador que jogou a maior carta vencerá (na mesma ordem de jogo do Poker). Cada carta de copas vale 1 ponto e a Dama de Paus vale 10 pontos. Caso as cartas das mãos acabem e algum jogador possuir 33 pontos ou mais, o jogo acaba. Caso contrário, as cartas são redistribuidas. O jogador que possuir o menor número de pontos ao fim da partida é o vencedor.

3. O Protocolo

O protocolo segue bases simples. As informações gerenciais são guardadas em um cabeçalho de 3 bytes logo antes dos dados, que podem ter de 0 a 15 bytes.

Todas as mensagens da rede possuem as seguintes sessões, com seus seguintes tamanhos:

```
1    INICIO | DESTINO | ORIGEM |  TIPO  |  ACK
2    1 BYTE | 2 BITS  | 2 BITS | 3 BITS | 1 BIT
3
4    | TAMANHO | CHECKSUM | DADOS
5    |  4 BITS |  4 BITS  | 0-15 BYTES
```

Code 1. Formato de mensagens no protocolo.

- **Inicio: O primeiro byte** guarda o Marcador de Início (01111110) das mensagens. É utilizado por segurança extra e qualquer mensagem que não possue este início é ignorada pela rede.
- **Destino: 2 bits** guardam o ID do destinatário. Como a rede possue apenas 4 pontos de acesso, 2 bits são o suficiente para diferenciar os diferentes destinos.
- **Origem: 2 bits** guardam o ID da origem da mensagem. Quando um ponto recebe uma mensagem cuja origem seja ele mesmo, significa que a mensagem deu a volta pela Rede.
- **Tipo: 3 bits** guardam o tipo da mensagem, o que faz com que a Rede possua 8 tipos diferentes que são detalhados na próxima seção.
- **Ack: 1 bit** guarda o ack da mensagem. Caso a pessoa que recebeu a mensagem seja o destino, ela utiliza dos dados da mensagem e a repassa "flipando" este bit que será utilizado pela origem para checar se a mensagem foi recebida.
- **Tamanho: 4 bits** guardam o tamanho da mensagem, o que significa que cada mensagem tem um tamanho máximo de dados de 15 bytes, que serão suficientes.
- **Checksum: 4 bits** guardam o checksum da mensagem. Este campo é utilizado por segurança e, caso alguma checagem mostre um erro, a mensagem chegará à origem, que a retransmitirá em uma segunda tentativa.

4. Tipos de Mensagens

As mensagens foram divididas em 8 tipos, sendo eles os seguintes:

4.1. Tipo 0 - Passagem de Bastão

O bastão é passado através deste tipo de mensagem. Caso destino e origem da mensagem sejam os mesmos, a passagem de bastão deve ser feita ao próximo jogador. Caso seja diferente, ela deve ser feita ao destino.

4.2. Tipo 1 - Conexão e início de rodadas

Este tipo de mensagem inicia a conexão do anel e é enviada todos os inícios de rodada.

4.3. Tipo 2 - Distribui 13 cartas para o destino

Ao ocorrer a distribuição, esta mensagem viajará ao destino com 13 cartas.

4.4. Tipo 3 - Rodada

Este tipo de mensagem armazena as cartas da rodada e envia ao próximo à jogar.

4.5. Tipo 4 - Impressões na tela

Este tipo é utilizado apenas para impressões. Caso a mensagem possua tamanho 2, o primeiro dado é o Id do jogador e o segundo dado a carta jogada por ele. Caso a mensagem possua tamanho 1, o dado é o Id do jogador vencedor da rodada.

4.6. Tipo 5 - Soma de pontos ao vencedor da rodada

Este tipo armazena a quantidade de pontos feitos na rodada e envia ao vencedor da mesma.

4.7. Tipo 6 - Checagem de pontos

Este tipo da mensagem checa a pontuação de todos os jogadores, a fim de procurar algum que tenha 33 pontos ou mais.

4.8. Tipo 7 - Fim de Jogo

Este tipo de mensagem termina a conexão da rede e imprime na tela quais foram os ganhadores do jogo.

5. Bibliotecas Implementadas

3 bibliotecas auxiliares foram implementadas para que o trabalho fosse modularizado e tivesse um código compreensível.

5.1. Message

Nesta biblioteca foram implementadas funções de manipulação de mensagem seguindo o protocolo como montar e desmontar a mensagem, retransmitir com ou sem ack e receber e enviar mensagens. Também são armazenados "defines" para facilitar o entendimento do código principal.

5.2. Game

Nesta biblioteca foram implementadas funções de jogo como iniciar a conexão, jogar cartas, distribuir cartas, checar naipes, checar vencedor e somar pontos ao vencedor da rodada.

5.3. Printing

Nesta biblioteca foram implementadas funções que formatam a impressão do jogo como imprimir naipe e número das cartas, imprimir quem jogou qual carta, quem venceu a rodada e quem venceu o jogo.