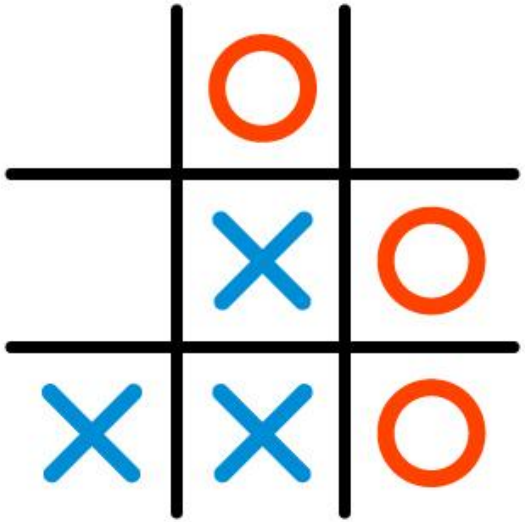
Sistema distribuído

MAC0352 - Redes de Computadores e Sistemas Distribuídos
Instituto de Matemática e Estatística · USP

Arthur Font
Lucas Pires

Jogo da Velha - Regras

- Um jogador joga com o ‘O’ e outro com o ‘X’
- O jogo acaba quando algum jogador forma uma linha, seja ela na horizontal, vertical ou diagonal ou quando não há mais espaços vazios
- Pontuação:
  - Vitória: 2 pontos
  - Empate: 1 ponto



Arquitetura híbrida

- Cliente/servidor: os clientes se comunicam com o servidor para algumas ações, como por exemplo: fazer o login
- P2P: os clientes comunicam entre si para realizarem a partida

O protocolo de rede permite:

- Conexão de vários clientes simultaneamente
- Mensagens criptografadas entre servidor e clientes
- Heartbeat entre servidor e clientes
- Verificação periódica de latência entre clientes durante partidas
- Troca de mensagens em modo texto (ASCII) entre cliente e servidor e entre clientes

Protocolo - Considerações

- O protocolo TCP foi escolhido porque garante que os pacotes sejam transmitidos de maneira integral e ordenada
- O heartbeat foi implementado da seguinte forma:
  - O cliente envia um ‘ping’ a cada 5 segundos
  - O servidor espera por um ‘ping’ em um intervalo máximo de 10 segundos. Caso o limite seja ultrapassado, o servidor fecha a conexão deste cliente.
- Desta maneira, o servidor economiza recursos de rede e CPU, pois não precisará enviar uma resposta a cada cliente [1,2]

Fluxo do código:

- Servidor:
  - Thread principal: Espera por novas conexões
  - Thread secundária: Comunicação com cada cliente
- Cliente:
  - Thread principal: Interação com o servidor (Slide 7)
  - Thread secundária: Heartbeat
  - Thread terciária: Jogo da Velha (P2P)

Comandos implementados (via prompt):

- adduser <usuario> <senha>
- passwd <senha antiga> <senha nova>
- login <usuario> <senha>
- leaders
- list
- begin <oponente>
- logout
- exit
- send <linha> <coluna>
- delay
- end: encerra uma partida antes da hora

Servidor - Suporte a falhas

O sistema tolera as seguintes falhas do servidor, limitadas a um intervalo de 3 minutos:

- Processo do servidor foi finalizado por um ‘kill -9‘
- Rede do servidor foi desconectada por um ‘ifdown‘

Para isso, utilizamos:

- Arquivo de log: reconstrói o estado do servidor
- Pandas (.csv): banco de dados persistente às falhas

Análise de desempenho

3 cenários:

- 1. Somente o servidor iniciado
  - Rede: 0.000 KB/sec Sent/Received
  - CPU: Inicia por volta de 5,0% e lentamente cai até 0,0%
    - Queda explicada por período de startup do Python
    - Servidor fica apenas esperando por novas conexões

Análise de desempenho

3 cenários:

- 2. Servidor com 2 clientes conectados sem jogar
  - Rede: 0.026 KB/sec Sent | 0.037 KB/sec Received
  - CPU servidor: 0,1%
  - CPU clientes: 0,0% + 0,0%
    - Trabalho apenas de manter os sockets TLS abertos e clientes enviam periodicamente um heartbeat

Análise de desempenho

3 cenários:

- 3. Servidor com 2 clientes conectados e jogando
  - Rede: 0.026 KB/sec Sent | 0.037 KB/sec Received
  - CPU servidor: 0,0%
  - CPU clientes: 0,0% + 0,0%
    - Por conta da arquitetura usada, existe pouca diferença da lógica de plano de fundo quando o cliente esta em partida, adicionando apenas o cálculo de delay

Análise de desempenho

Metodologia para avaliação da carga na rede e na CPU:

- CPU:
  - Comando: *$ps -p $(pgrep -d',' -f ./main.py) -o %cpu,cmd*
  - Este comando mostra a porcentagem de uso de cada processo com nome ./main.py, assim abrangendo clientes e o servidor
- Rede:
  - Comando: $sudo nethogs -a
  - Este comando exibe a carga da rede utilizada por cada processo. A flag -a é essencial para exibir os processos de todas as interfaces da rede, incluindo loopbacks

Ambientes utilizados

- Computacional
  - OS: Arch Linux
  - Processador: Intel(R) Core(TM) i5-4790k CPU @ 4.40GHz
  - CPU(s): 4 Cores + 4 (hyper-threading)
  - Memória RAM: 16 GB
- Rede (no momento do teste)
  - Largura de banda contratada: 240 Mbps
  - Velocidade de Download ≈ 250 Mbps
  - Velocidade de Upload ≈ 20 Mbps

Referências

1. https://www.oreilly.com/library/view/python-cookbook/0596001673/ch10s13.html
2. https://docs.oracle.com/cd/E19206-01/816-4178/6madjde6e/index.html
3. https://pandas.pydata.org/

Obrigado pela atenção!