Balanceamento de carga utilizando kd-trees para particionar dinamicamente o ambiente virtual de MMOGs

Carlos Eduardo B. Bezerra*
Grupo de Processamento Paralelo e Distribuído
Instituto de Informática - UFRGS

Resumo

MMOGs (*massively multiplayer online games*, ou jogos online maciçamente multijogador), são aplicações que requerem conexões com grande largura de banda para funcionarem adequadamente. Essa demanda por largura de banda é maior principalmente nos servidores que hospedam o jogo. Como nesse tipo de jogo costuma haver milhares a dezenas de milhares de jogadores simultâneos, sendo que a interação entre cada par de jogadores é intermediada pelo servidor, é sobre este que recai o maior custo no que se refere a uso de largura de banda para realizar o envio de atualizações de estado do ambiente do jogo para os jogadores. Para contornar este problema, são propostas arquiteturas com vários servidores, onde cada um deles gerencia uma região do ambiente virtual, e cada jogador conecta-se somente ao servidor que gerencia a área onde ele está jogando. No entanto, para distribuir a carga entre os servidores, é necessário um algoritmo de particionamento do ambiente virtual que, para poder reajustar o balanceamento de carga durante o jogo, seja dinâmico. Alguns trabalhos nesse sentido podem ser citados, como [De Vleeschauwer et al. 2005; Ahmed and Shirmohammadi 2008; Bezerra and Geyer 2009], mas, utilizando um algoritmo geométrico mais adequado, pode-se alcançar um nível melhor de granularidade da distribuição, sem comprometer o tempo de rebalanceamento, ou mesmo reduzindo-o. Neste trabalho, é proposta a utilização de uma KD-Tree para dividir o ambiente virtual do jogo em regiões, cada uma das quais sendo designada a um dos servidores. As coordenadas de corte das regiões são ajustadas dinamicamente de acordo com a distribuição dos avatares ao longo do ambiente virtual.

Palavras-chave: MMOGs, balanceamento de carga, servidor distribuído, kd-trees

1 Introdução

Os MMOGs têm como principal característica a grande quantidade de jogadores interagindo simultaneamente, chegando a ter dezenas até centenas de milhares de participantes simultâneos [Schiele et al. 2007]. Ao se usar uma arquitetura cliente-servidor para que os jogadores se comuniquem entre si, é necessário que esse servidor intermedie a comunicação entre cada par de jogadores.

Para permitir que os jogadores interajam, o servidor recebe os comandos de cada jogador, calcula o estado resultante do jogo e envia o novo estado para os jogadores que estejam interagindo com ele. É fácil perceber que o número de mensagens trocadas entre os jogadores que passam pelo servidor será proporcional ao quadrado no número de jogadores, se todos estiverem interagindo com todos. Obviamente, dependendo desse número de jogadores, o custo de manutenção de uma infra-estrutura centralizada como essa se torna muito alto, restringindo esse mercado de jogos MMOG para grandes empresas com muitos recursos.

Buscando reduzir esse custo, tem-se buscado soluções descentralizadas. Algumas destas utilizam redes par-a-par: [Schiele et al. 2007; Rieche et al. 2007; Hampel et al. 2006; El Rhalibi and Merabti 2005; Iimura et al. 2004; Knutsson et al. 2004]. Outras propõem a utilização de um sistema distribuído composto por nodos servidores de baixo custo, conectados através da Internet, como é proposto em [Ng et al. 2002; Chertov and Fahmy 2006; Lee and Lee 2003; Assiotis and Tzanov 2006]. De qualquer forma, o "mundo", ou ambiente virtual do jogo, é dividido em regiões e, para cada região, é designado um servidor – ou um grupo de pares para administrarem-no em conjunto, no caso das redes par-a-par. Cada uma dessas regiões deve possuir um conteúdo tal que a carga imposta sobre o servidor não seja maior que a sua capacidade. Quando um servidor se torna sobrecarregado, devido a uma mudança na distribuição dos objetos do jogo, a divisão do ambiente virtual deve ser recalculada para aliviar aquele servidor.

Neste trabalho, é proposto o uso de uma kd-tree para realizar o particionamento do ambiente virtual. Quando o movimento das entidades do jogo no ambiente virtual faz com que uma determinada região se sobrecarregue, o servidor daquela região dispara o balanceamento de carga, reajustando os limites de sua região com a ajuda da estrutura de dados kd-tree. O texto está organizado da seguinte maneira: na seção 2, são apresentados alguns trabalhos relacionados; na seção 3, é apresentada em detalhes o algoritmo proposto aqui; nas seções **??** e **??** são apresentadas as simulações e os resultados alcançados e, na seção **??**, são apresentadas algumas conclusões e considerações finais deste trabalho.

2 Trabalhos relacionados

ahmed bezerra luque

Uma idéia inicial poderia ser a de distribuir os jogadores entre servidores, de maneira que o número de jogadores em cada servidor fosse proporcional à largura de banda daquele servidor. No entanto, essa distribuição não funcionaria, pelo fato de que a carga causada pelos jogadores depende também do quanto os jogadores estão interagindo entre si. Por exemplo, se os avatares de dois jogadores estiverem muito distantes um do outro, provavelmente não haverá interação entre eles e, portanto, o servidor precisará apenas atualizar cada um a respeito de suas próprias ações. No entanto, se estes avatares estiverem próximos, cada jogador deverá ser atualizado não apenas a respeito do resultado de suas próprias ações, como também das ações do outro jogador.

Percebe-se, então, que quando os avatares estão distantes uns dos outros, o tráfego cresce linearmente com o número de jogadores (Figura 2). Porém, se eles estão próximos uns dos outros, o tráfego cresce quadraticamente (Figura 2). Por fim, ambas as funções de crescimento do número de mensagens podem estar presentes no mesmo jogo se, em alguns lugares do ambiente virtual, os avatares estiverem próximos e, em outros lugares, eles estiverem distantes.

Essa característica de localidade, no que diz respeito à distribuição dos avatares no ambiente virtual, está presente na grande maioria dos jogos multijogador. Existem algumas exceções, como GuildWars [ArenaNet 2005], em que apenas grupos com um número limitado de jogadores podem iniciar uma partida. Este tipo de jogo é baseado no modelo de instâncias, onde todos os avatares dos jogadores se encontram em um espaço social, de interação li-

*e-mail: carlos.bezerra@inf.ufrgs.br

mitada, menos dinâmica e, portanto, com tráfego de rede algumas ordens de grandeza menor [Vilanova et al. 2008]. Quando pretende-se iniciar uma partida "real", os jogadores requisitam ao servidor que seja criado um grupo de ação. Dessa forma, impede-se que um número teoricamente ilimitado de jogadores interajam entre si, sobrecarregando o servidor.

Normalmente, porém, os jogadores podem mover seus avatares livremente através do mundo do jogo. Isso torna possível a formação de pontos de interesse – também conhecidos como *hotspots* [Ahmed and Shirmohammadi 2008] – ao redor dos quais os jogadores se concentram mais do que em outras regiões do ambiente virtual (Figura 2). Aliás, muitos jogos de RPG online maciçamente multijogador não só permitem como também estimulam, até certo ponto, a formação destes pontos de interesse. Nestes mundos dos MMORPGs, existem cidades inteiras, onde os jogadores se encontram para conversar, trocar mercadorias virtuais do jogo e/ou duelar, assim como existem também zonas desérticas, sem muitos atrativos para os jogadores, e onde o número de avatares presentes é relativamente pequeno, se comparado com outros lugares no jogo.

Por esta razão, não é suficiente apenas dividir os jogadores entre os servidores, mesmo que proporcionalmente aos recursos de cada um destes. Em primeiro lugar, em alguns casos o uso de largura de banda do servidor é quadrático ao número de jogadores, enquanto é linear em outros. Essa razão por si só já é suficiente para buscar outro critério para o balanceamento de carga. Além disso, há outra questão importante: a existência de pontos de interesse. Esta última característica motiva à criação de um esquema de balanceamento de carga para jogos que impeça que a presença de pontos de interesse degrade a qualidade do jogo além do tolerável.

Outra questão é que se os jogadores em um mesmo ponto de interesse forem divididos entre diferentes servidores, cada um destes precisará não apenas enviar o estado do mundo resultante das ações para os jogadores conectados a ele, como também deverá enviá-lo para o servidor ao qual os outros jogadores estão conectados. Este, por sua vez, encaminhará este resultado para seus jogadores. Percebe-se, então, que cada estado deverá ser enviado duas vezes, para cada par de jogadores que se comunicam através de servidores diferentes (Figura 2). Esse overhead não apenas causa o desperdício de recursos dos servidores, como também aumenta o atraso para atualização de estado das réplicas do jogo nas máquinas dos jogadores. Isto faz com que o tempo entre o envio de uma ação por um jogador conectado a um servidor e o recebimento do estado resultante por outro jogador, conectado a outro servidor, seja maior, prejudicando a interação entre eles.

Assim sendo, jogadores que estão interagindo entre si devem, idealmente, estar conectados ao mesmo servidor. Contudo, é possível que todos os jogadores estejam ligados entre si através de relações transitivas de interação. Por exemplo, dois avatares, de dois jogadores diferentes, podem estar distantes um do outro, porém ambos interagindo com um terceiro avatar, entre os dois (Figura 2). Contudo, ainda assim será necessário dividí-los entre servidores. A questão é quais pares de jogadores estarão divididos em servidores diferentes. É necessário, portanto, decidir um critério para agrupar jogadores em um mesmo servidor.

3 Proposta

definições: avatar, coordenada de corte, região sweep para calculo do peso de cada avatar kd-tree balanceamento automatico (2^{lallaa}) corte dinâmico recursao para balanceamento (irmaos, primos, pri-primos etc.) ajuste dos limites entidade central para balanceamento (necessario calculo dos pesos dos avatares)

A proposta para balanceamento de carga dinâmico para MMOGs apresentada aqui tem como base dois critérios: primeiramente, deve-se considerar a possibilidade do sistema ser heterogêneo, ou seja, de que cada servidor tenha uma quantidade diferente de recursos. Aqui se define como "recursos" a largura de banda envio que aquele servidor tem disponível para enviar atualizações de estado para os jogadores.

Essa escolha se deve ao fato de que cada jogador envia comandos para o servidor a uma taxa constante, logo o número de mensagens recebidas pelo servidor por unidade de tempo cresce linearmente em relação ao número de jogadores, enquanto que, como foi discutido, o número de atualizações de estado enviadas pelo servidor pode ser quadrático, no pior caso.

Como foi dito na introdução, para dividir o ambiente do jogo em regiões propõe-se utilizar uma estrutura de dados conhecida, que é a kd-tree. A grande maioria dos MMOGs, como World of Warcraft [Blizzard 2004], Ragnarok [**?**] e Lineage [**?**], apesar de ter gráficos em 3D, o mundo simulado – cidades, florestas, pântanos e pontos de interesse em geral – nestes jogos é mapeado em duas dimensões. Por esse motivo, propõe-se o uso de uma kd-tree com k = 2.

A cada nodo da árvore corresponde uma região do espaço e, além disso, neste nodo é armazenada um valor correspondente a uma coordenada de corte. Cada um dos dois filhos daquele nodo representarão uma subdivisão da região representada pelo nodo pai, sendo que um deles representa a sub-região com coordenada menor do que a coordenada de corte, e o outro, a região com coordenada maior ou igual à coordenada de corte. Cada nó folha também representa uma região do espaço, porém não armazenam nenhuma coordenada de corte. A cada nível da árvore, alterna-se o eixo da coordenada de corte (no caso de duas dimensões, os eixos x e y).

3.1 Construção da kd-tree

A cada nodo folha da kd-tree corresponde um servidor do sistema servidor. Dessa forma, a região representada por aquele nodo folha será administrada pelo servidor associado a ele. Para se dividir o espaço inicialmente, é construída uma kd-tree balanceada (tanto quanto possível, pois depende do número de servidores, ou nós folha, ser igual a alguma potência de 2). Para isto foi utilizada a seguinte função recursiva para criação da árvore:

void KDTree::buildTree(int nodeId, int treeLvl) {

if (nodeId + $2^{treeLvl}$ >= NUM_SERVERS) {

smallerChild = biggerChild = NULL;

return;

}

else {

smallerChild = new KDTree();

smallerChild.parent = this;

smallerChild.buildTree(nodeId, treeLvl + 1);

biggerChild = new KDTree();

biggerChild.parent = this;

biggerChild.buildTree(nodeId + $2^{treeLvl}$, treeLvl + 1);

}

}

FIGURA DE UMA KD-TREE COM OS IDS ASSOCIADOS

No algoritmo acima, o nodeId serve para calcular quantos filhos cada nodo deve ter e, nos nodos folha, pode ser usado para determinar qual o servidor associado à região representada por aquela folha da árvore. O objetivo com isso é tentar criar uma árvore balanceada, onde cada nodo tem duas sub-árvores com pesos semelhantes.

3.2 Cálculo do peso das regiões

3.3 Balanceamento dinâmico

Uma vez criada a árvore, cada servidor é associado a um nó folha, que determina um subespaço. Todas as atualizações de estado que devem ser enviadas a jogadores cujos avatares estejam naquela região deverão ser enviadas pelo servidor correspondente. Quando um servidor fica sobrecarregado, ele tem a possibilidade de transferir parte da carga designada a ele para algum outro servidor. Para fazer isso, é utilizada a kd-tree, através do reajuste das coordenadas de corte dos nodos das regiões.

Acknowledgements

To Robert, for all the bagels.

Referências

AHMED, D., AND SHIRMOHAMMADI, S. 2008. A Microcell Oriented Load Balancing Model for Collaborative Virtual Environments. In *Proceedings of the IEEE Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems, VECIMS*, Piscataway, NJ: IEEE, Istanbul, Turkey, 86–91.

ARENANET, 2005. Guild wars. 2005. Disponível em: ¡http://www.guildwars.com/¿. Acesso em: 26 jun. 2009.

ASSIOTIS, M., AND TZANOV, V. 2006. A distributed architecture for MMORPG. In *Proceedings of the ACM SIGCOMM workshop on Network and system support for games, NetGames, 5.*, New York: ACM, Singapore, 4.

BEZERRA, C. E. B., AND GEYER, C. F. R. 2009. A load balancing scheme for massively multiplayer online games. *Massively Multiuser Online Gaming Systems and Applications, Special Issue of Springer's Multimedia Tools and Applications*.

BLIZZARD, 2004. World of warcraft. 2004. Disponível em: ¡http://www.worldofwarcraft.com/¿. Acesso em: 26 jun. 2009.

CHERTOV, R., AND FAHMY, S. 2006. Optimistic Load Balancing in a Distributed Virtual Environment. In *Proceedings of the ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV, 16.*, New York: ACM, Newport, USA, 1–6.

DE VLEESCHAUWER, B., ET AL. 2005. Dynamic microcell assignment for massively multiplayer online gaming. In *Proceedings of the ACM SIGCOMM workshop on Network and system support for games, NetGames, 4.*, New York: ACM, Hawthorne, NY, 1–7.

EL RHALIBI, A., AND MERABTI, M. 2005. Agents-based modeling for a peer-to-peer MMOG architecture. *Computers in Entertainment (CIE) 3*, 2, 3–3.

HAMPEL, T., BOPP, T., AND HINN, R. 2006. A peer-to-peer architecture for massive multiplayer online games. In *Proceedings of the ACM SIGCOMM workshop on Network and system support for games, NetGames, 5.*, New York: ACM, Singapore, 48.

IIMURA, T., HAZEYAMA, H., AND KADOBAYASHI, Y. 2004. Zoned federation of game servers: a peer-to-peer approach to scalable multi-player online games. In *Proceedings of the ACM SIGCOMM workshop on Network and system support for games, NetGames, 3.*, New York: ACM, Portland, USA, 116–120.

KNUTSSON, B., ET AL. 2004. Peer-to-peer support for massively multiplayer games. In *Proceedings of the IEEE Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM, 23.*, [S.l.]: IEEE, Hong Kong, 96–107.

LEE, K., AND LEE, D. 2003. A scalable dynamic load distribution scheme for multi-server distributed virtual environment systems with highly-skewed user distribution. In *Proceedings of the ACM symposium on Virtual reality software and technology*, New York: ACM, Osaka, Japan, 160–168.

NG, B., ET AL. 2002. A multi-server architecture for distributed virtual walkthrough. In *Proceedings of the ACM symposium on Virtual reality software and technology, VRST*, New York: ACM, Hong Kong, 163–170.

RIECHE, S., ET AL. 2007. Peer-to-Peer-based Infrastructure Support for Massively Multiplayer Online Games. In *Proceedings of the 4th IEEE Consumer Communications and Networking Conference, CCNC, 4.*, [S.l.]: IEEE, Las Vegas, NV, 763–767.

SCHIELE, G., ET AL. 2007. Requirements of Peer-to-Peer-based Massively Multiplayer Online Gaming. In *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid, CCGRID, 7.*, Washington, DC: IEEE, Rio de Janeiro, 773–782.

VILANOVA, F. J., BEZERRA, C. E. B., CRIPPA, M. R., CECIN, F. R., AND GEYER, C. F. R. 2008. P2PSE - A Peer-to-Peer Support for Multiplayer Games. In *Proceedings of the Brazilian Symposium on Computer Games and Digital Entertainment - Computing Track, SBGames, 7.*, [S.l.]: Sociedade Brasileira de Computação - SBC, Belo Horizonte, 47–53.