Integração do Flight Simulator 2002 com um protocolo de difusão epidémica*

M. João Monteiro†
Universidade de Lisboa
mjmonteiro@di.fc.ul.pt

José Pereira
Universidade do Minho
jop@di.uminho.pt

Luís Rodrigues
Universidade de Lisboa
ler@di.fc.ul.pt

Resumo

Os jogos multi-utilizador são aplicações que têm alcançado uma enorme popularidade na Internet actual. Este crescente interesse leva a que estas aplicações possuam hoje em dia um elevado número de participantes. Isto coloca problemas de escalabilidade e, consequentemente, de desempenho, cuja resolução é determinante para o sucesso destas aplicações.

Neste artigo estamos interessados em procurar soluções de middleware *para suportar aplicações multi-utilizador em grande-escala. Em particular, estamos interessados em estudar a viabilidade de utilizar protocolos de difusão probabilista para suportar a disseminação de informação em jogos multi-utilizador recorrendo a uma arquitectura entre-pares (*peer-to-peer*). Para isso, concretizámos a integração de um jogo multi-utilizador comercial, o Flight Simulator 2002 da Microsoft, com um protocolo de difusão probabilista concebido para este fim, o NEEM. O artigo descreve o trabalho realizado e o resultado da avaliação do protótipo desenvolvido.*

Palavras Chave: Difusão epidémica, arquitectura entre-pares (*peer-to-peer*), jogos multi-utilizador.

1 Introdução

As aplicações multi-utilizador estão cada vez mais presentes na Internet actual. Estas aplicações têm gerado o interesse, não só dos utilizadores, mas também dos

*Este trabalho foi parcialmente suportado pelo projecto RUMOR (POSI/40088/CHS/2001) e pela Microsoft Research Grant (2001-39).

†Contacto: Faculdade de Ciências da Universidade de Lisboa, Dep. de Informática. Bloco C5, Campo Grande 1749-016 Lisboa.

prestadores de serviço, que encontram nestas aplicações fontes de receitas e um meio eficaz de publicidade. De todas estas aplicações, os jogos multi-utilizador são um dos casos de maior popularidade. Dado o grande número de utilizadores que os jogos podem atrair, estas aplicações levantam interessantes problemas de escalabilidade. Note-se que os utilizadores destes jogos não se restringem aos jogadores, uma vez que os jogos *on-line* estão a tornar-se jogos de espectadores [2], que participam activamente no jogo através de conversas entre eles ou mesmo através de apostas[1].

Um requisito frequente dos jogos multi-utilizador é a necessidade de difundir informação por todos os utilizadores de forma periódica. Infelizmente, é extremamente complexo, senão impossível, conciliar a fiabilidade da difusão com a escalabilidade do sistema. Alguns protocolos de difusão fiável, como o RMTP [11], geram um número muito grande de confirmações que têm de ser recebidas e processadas, uma tarefa que pode rapidamente sobrecarregar o emissor. Mesmo utilizando mecanismos escaláveis de confirmações, as mensagens têm de ser guardadas em *buffers* e retransmitidas até que todos os destinatários confirmem a sua recepção ou sejam apontados como falhados [7]. Assim, quando um receptor está mais lento, as mensagens acumulam-se no emissor, e este começa, inevitavelmente, a transmitir ao ritmo deste receptor, tornando, desta forma, todo o grupo mais lento [17].

Os protocolos de difusão probabilista, também designados de difusão epidémica, ultrapassam as limitações de escalabilidade atrás mencionadas. Estes protocolos suportam a disseminação eficiente de dados por um elevado número de nós, oferecendo garantias probabilistas de entrega [8, 1, 9, 3], apresentando um compromisso interessante entre a fiabilidade e a escalabilidade necessárias aos jogos multi-utilizador. Usando estes protocolos, é possível assegurar um débito elevado, mesmo em grupos com um elevado número de membros com desempenho heterogéneo, uma vez que estes protocolos distribuem a carga por todos os membros do grupo. Dentro destes protocolos, destaca-se o NEEM [15]. Este, através da utilização do TCP, ao nível do transporte, e de uma gestão de *buffers* eficaz que permite o descartar de mensagens já obsoletas, é particularmente adaptado ao suporte de jogos multi-utilizador em grande escala.

Este artigo descreve um trabalho de integração de um jogo multi-utilizador comercial, o Microsoft Flight Simulator 2002, com o protocolo de difusão epidémica NEEM. Com este trabalho pretendemos validar a utilidade deste tipo de protocolos para resolver os problemas de escalabilidade que se colocam quando se suporta um número muito elevado de participantes.

O resto do artigo está estruturado da seguinte forma: As Secções 2 e 3 apresentam os componentes utilizados no trabalho descrito neste artigo. A Secção 4

[1] Exemplos em `http://youplaygames.com/`.

descreve em detalhe a integração dos componentes apresentados e a Secção 5 apresenta a avaliação desta integração. Finalmente, a Secção 6 conclui o artigo.

2 Microsoft Flight Simulator 2002

O jogo escolhido para realizar e validar a integração de jogos multi-utilizador com protocolos de difusão probabilista foi o Microsoft Flight Simulator 2002 (FS2002). Nas secções seguintes é descrito o funcionamento geral do FS2002 e também certas particularidades que são relevantes para a posterior descrição da integração. Os problemas de escalabilidade dos jogos multi-utilizador são também discutidos no contexto concreto do FS2002.

2.1 Descrição do Jogo

O FS2002 é um simulador de voo que pode funcionar em modo multi-utilizador sobre redes locais ou na Internet. Cada jogador é responsável por controlar um avião num mundo virtual partilhado entre todos os jogadores. A cada jogador são disponibilizados vários tipos de aviões e vários cenários de voo. Numa sessão multi-utilizador, existe um nó coordenador que selecciona o cenário em que a sessão irá decorrer e que é responsável pela manutenção da filiação do grupo. No caso de o coordenador falhar, outro nó assume este papel e o jogo prossegue. Os utilizadores podem ainda interagir através de mensagens textuais num ambiente típico de "*chat*".

A concretização do FS2002, no sistema operativo Windows, utiliza a API DirectPlay [12]. Este é um protocolo de transferência de mensagens muito utilizado para concretizar jogos multi-utilizador no ambiente Windows. O DirectPlay disponibiliza também um protocolo de filiação. O FS2002 utiliza uma arquitectura entre-pares em que o estado do jogo é replicado por todos os participantes. Assim, cada jogador mantém o estado real do avião que controla e, periodicamente, actualiza os outros participantes com a informação sobre a sua posição e velocidade [13]. O pacote que leva as actualizações do estado está representado na Figura 1. O vector velocidade é representado pelas três componentes $(vel\_lat, vel\_lon, vel\_alt)$. Os campos correspondentes à posição do avião são os seis últimos. Para a posição também se indica a latitude, a longitude e a altitude, mas agora divididas em bits mais significativos e bits menos significativos, que se encontram identificados, respectivamente por $\_M$ e $\_m$, na Figura 1.

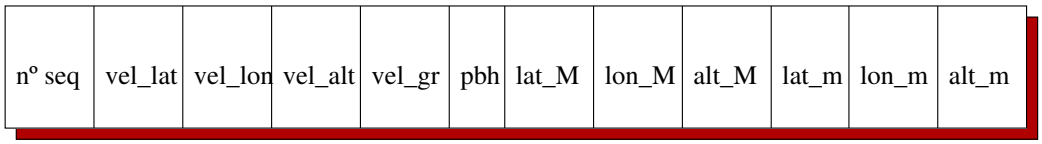

Figura 1: Pacote das actualizações.

2.2 Limitações de Desempenho

Actualmente, os utilizadores de FS2002 acedem à Internet através de ADSL, de cabo ou ainda de modems analógicos. Qualquer uma destas ligações tem capacidades muito limitadas de *uplink*. Isto restringe a implantação de uma arquitectura de comunicação entre-pares pois não existe largura de banda suficiente para um nó disseminar informação para todos os outros. A utilização de algoritmos de difusão probabilista atenua esta limitação, uma vez que cada nó só necessita de contactar um número reduzido de participantes para efectuar a difusão. No entanto, mesmo neste caso, a pouca largura de banda disponível pode gerar situações de sobrecarga, levando à perda de mensagens. Note-se que uma solução centralizada coloca também problemas de escalabilidade do servidor.

De seguida é feita uma análise aos requisitos de comunicação de um jogo como o FS2002, num cenário com capacidades de comunicação reais. Cada jogador difunde 4 actualizações por segundo de 60 bytes cada. Num cenário onde os jogadores estão ligados através de modems V.90 (56 kbps *downlink*/33.6 kbps *uplink* de largura de banda) isto traduz-se num máximo de 17 destinos para cada actualização do estado. Como no FS2002 cada jogador é responsável por enviar as suas actualizações também para todos os espectadores, isto significa que utilizando um protocolo fiável não é possível com esta arquitectura suportar mais de 17 participantes (jogadores mais espectadores). Num cenário de banda larga, o *uplink* continua a ser substancialmente inferior ao *downlink* disponível, o que limita também o número de destinos possíveis. Este problema pode ser atenuado pela utilização de um servidor central com uma ligação dedicada com maior largura de banda. No entanto, como é expectável que um servidor acolha mais de uma instância do jogo, o total de tráfego continua proporcional ao número de espectadores e o custo desse serviço aumentaria linearmente com a largura de banda utilizada.

Neste artigo, propomos uma arquitectura alternativa que consiste em recorrer a um protocolo de difusão probabilista que permita aos espectadores colaborarem na distribuição descentralizada das actualizações provenientes dos jogadores, dispensando o servidor central e tirando o máximo partido da arquitectura entre-pares suportada pelo DirectPlay.

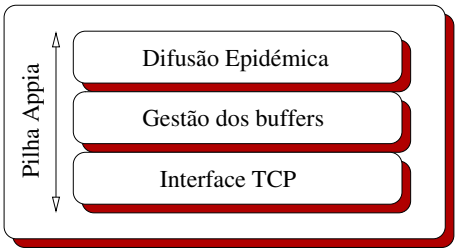

Figura 2: NEEM.

3 O Protocolo de Difusão Probabilista NEEM

Este trabalho utiliza um novo protocolo de difusão probabilista, o NEEM [15], que foi concebido para operar sobre redes com as características das utilizadas pelos jogos multi-utilizador como o FS2002. Os parágrafos seguintes descrevem a arquitectura do NEEM e os seus componentes fundamentais: a difusão epidémica, a gestão eficaz dos *buffers* e a utilização do TCP.

O NEEM foi concretizado utilizando o sistema Appia [14]. O Appia é uma moldura de objectos de suporte à composição e execução de micro-protocolos. Cada módulo do Appia corresponde a um micro-protocolo responsável por garantir uma determinada propriedade. Estes módulos são independentes e podem ser combinados, constituindo uma pilha de protocolos configurada com as propriedades desejadas. A Figura 2 apresenta a pilha de micro-protocolos que caracteriza o NEEM. Cada camada da pilha Appia apresentada na Figura 2 corresponde a um constituinte fundamental do NEEM, que descrevemos de seguida.

Difusão epidémica A camada de topo da pilha corresponde à difusão epidémica [4] que, no caso do NEEM, funciona da seguinte maneira: Cada mensagem é inicialmente etiquetada com o número máximo de retransmissões (r) e é enviada para f nós escolhidos ao acaso. Na recepção da mensagem, o número de retransmissões é decrementado. Quando o número de retransmissões atingir zero, a mensagem é descartada. Senão, a mensagem é novamente enviada para outros f nós. A entrega de uma mensagem dá-se apenas quando uma mensagem nova é recebida. As garantias do protocolo dependem da configuração de r e f [8, 9].

A filiação é mantida também de forma probabilista com cada nó a guardar uma lista dos nós que conhece [3, 5]. Esta lista corresponde a uma vista parcial do grupo que pode ser bastante mais pequena que o tamanho de todo o grupo, sendo no entanto suficiente para assegurar a fiabilidade da difusão. Em cada retransmissão, a lista dos nós conhecidos localmente é enviada juntamente com as mensagens de dados. Quando uma mensagem é recebida, a lista local é actualizada com

o conteúdo da lista que vem na mensagem. Se o tamanho da lista ultrapassar um limite pré-estabelecido, os elementos são removidos aleatoriamente.

Note-se que para um grupo de n participantes, tanto o tamanho da lista de filiação mantida por cada nó, bem como os parâmetros f e r são de ordem de grandeza $\log n$, tornando possível a utilização de grupos extremamente numerosos.

Gestão dos *buffers* A camada responsável pela gestão de *buffers* efectua o armazenamento das mensagens que vão ser enviadas. No caso de existirem poucas mensagens a serem enviadas, o facto de as mensagens serem guardadas em *buffers* permite que a carga sobre a rede seja distribuída no tempo evitando eventuais perdas de mensagens. No caso em que a rede está congestionada, o protocolo de difusão epidémica não pode esperar que haja espaço nos *buffers* para enviar mensagens pois assim permitiria que apenas um nó lento afectasse o grupo inteiro. A única opção é seleccionar mensagens a descartar. Esta selecção é feita por combinação de três estratégias:

Selecção semântica É seleccionada uma mensagem que foi identificada como estando obsoleta. Esta estratégia, ao contrário das outras, pode ser aplicada mesmo quando o *buffer* ainda não está cheio. Assim, potencialmente, a ocupação média do *buffer* diminui e, consequentemente, a latência diminui.

Selecção por idade É seleccionada a mensagem que foi retransmitida mais vezes. Esta é a estratégia a seguir caso não tenham sido encontradas mensagens obsoletas.

Selecção aleatória Como último recurso, é seleccionada uma mensagem ao acaso do *buffer* para arranjar espaço para a mensagem acabada de chegar. Esta estratégia introduz poucas melhorias no sistema face a situações de congestão.

Um aspecto fundamental da interface do NEEM consiste no modo de fornecer ao protocolo informação semântica sobre mensagens, em particular a que permite identificar mensagens obsoletas, sem comprometer a generalidade do protocolo. A concretização utilizada associa a cada mensagem um pequeno *bitmap*. Se o i-ésimo dígito do *bitmap* da mensagem com número de sequência n estiver activado, então a mensagem com número de sequência $n - i$ é considerada obsoleta [16].

Interface TCP A última camada surge como a interface da pilha Appia com o protocolo TCP integrado no sistema operativo. A escolha da utilização do TCP como protocolo de transporte baseou-se na existência do mecanismo de controlo de congestão que permite uma utilização mais adequada da largura de banda disponível. Uma vez que a largura de banda disponível em cada ligação da rede é

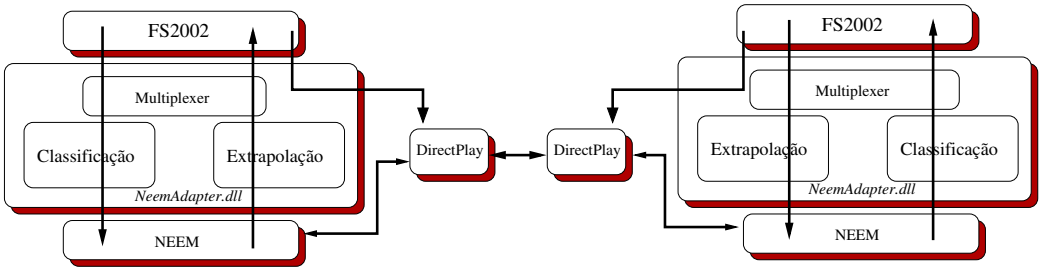

Figura 3: Arquitectura da integração.

partilhada por várias ligações TCP (i.e. uma para cada elemento da lista local de filiação), é possível reduzir o tamanhos dos *buffers* usados, reduzindo assim a latência introduzida bem como os recursos necessários [6].

4 Integração

A integração do FS2002 com o NEEM explorou o facto de o jogo assentar na API DirectPlay. Assim, foi desenvolvida uma camada de adaptação, designada por *NeemAdapter*, materializada por uma biblioteca dinâmica que intercepta as chamadas à biblioteca original *dplayx.dll*. A NeemAdapter intercepta todas as mensagens mas encaminha as mensagens de controle para a biblioteca original sem processamento adicional. As mensagens que contêm as actualizações das posições dos aviões são sujeitas a um processamento quer no envio quer na recepção. No envio são sujeitas a uma classificação e na recepção são armazenadas num componente de extrapolação. A NeemAdapter é ainda responsável por transformar a difusão entre-pares das actualizações do FS2002 numa invocação do protocolo NEEM. Estes componentes estão descritos nas secções seguintes. Na Figura 3 é apresentada a arquitectura de toda a integração.

O NEEM foi complementado com uma camada de interface, no topo da pilha Appia, responsável pela ligação entre a concretização em Java do NEEM e a concretização da NeemAdapter em C++. Outra alteração feita ao NEEM com vista à integração do protocolo com o FS2002 foi a substituição da camada inferior da pilha, a camada do TCP, por uma camada que faz de interface com o DirectPlay. Este por sua vez, invoca o TCP/IP integrado no sistema operativo. Estas alterações ao protocolo NEEM estão representadas na Figura 4.

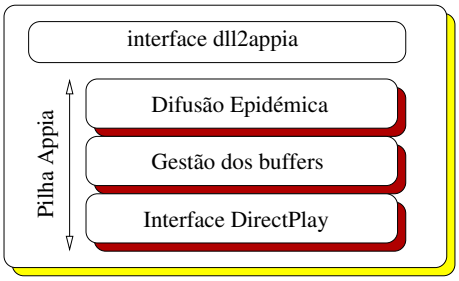

Figura 4: NEEM ajustado.

4.1 Multiplexer

O FS2002 baseia-se numa arquitectura entre-pares onde cada jogador é responsável por enviar a actualização da sua posição e velocidade a todos os outros intervenientes (jogadores e espectadores). Sem a camada de adaptação, este envio seria feito ponto-a-ponto para cada um dos destinatários. Para beneficiar da disponibilidade do protocolo de difusão probabilista, a NeemAdapter transforma cada conjunto de envios ponto-a-ponto, referentes a uma dada actualização, numa única invocação ao protocolo NEEM. Este, por sua vez, concretiza a difusão de forma epidémica o que, como foi referido anteriormente, é possível sem que cada nó tenha de comunicar directamente com todos os restantes.

4.2 Classificação

Uma das questões essenciais na utilização do NEEM é a codificação da semântica das mensagens. Assim, cada actualização a enviar por um jogador é submetida a um processo de classificação em relação às n actualizações anteriores[2]. Para tal, a NeemAdapter armazena uma lista com os n últimos pacotes de actualização. Através da comparação de uma dada actualização com as actualizações anteriores, a camada de classificação identifica as mensagens anteriores que se tornam obsoletas e constrói o *bitmap* associado a essa actualização. Por sua vez, a camada de gestão de *buffers* só precisa de interpretar estes *bitmaps* sem necessitar de conhecer o conteúdo das mensagens do FS2002.

A regra utilizada para determinar as relações de obsolescência entre estas mensagens é a seguinte. A mensagem m_j torna m_i obsoleta se:

[2]O parâmetro n depende da dimensão escolhida para o *bitmap*.

- $j > i$ e
- $\forall k : i < k \leq j$, o vector de velocidade de m_k diferir do de m_i em menos do que um factor F pré-definido, mais concretamente:

$$\sqrt{(vel\_lat_k - vel\_lat_i)^2 + (vel\_lon_k - vel\_lon_i)^2 + (vel\_alt_k - vel\_alt_i)^2} \leq F * \sqrt{vel\_lat_i^2 + vel\_lon_i^2 + vel\_alt_i^2}$$

Com esta regra, e analisando o tráfego do FS2002, observámos que com $F = 0.01$, cerca de 46% das mensagens ficam obsoletas pouco depois de terem sido enviadas. Pode-se também observar que usando $F = 0.02$ marcam-se 72% de mensagens como obsoletas. Estes valores só têm em conta as mensagens de actualizações de estado.

4.3 Extrapolação

A NeemAdapter intercepta as mensagens que são recebidas pelo DirectPlay e que têm como destino o FS2002. A chegada de actualizações tem um intervalo relativamente constante e que corresponde a cerca de 320ms. Assim, a NeemAdapter mantém um alarme para cada um dos aviões presentes na sessão (excluindo o próprio) de forma a mascarar atrasos ou até perdas de mensagens através da geração, por extrapolação, das mensagens em falta.

A NeemAdapter armazena o último pacote de actualização recebido de cada avião e, se o alarme disparar sem que nova actualização tenha sido recebida, dá-se o processo de extrapolação baseado na informação guardada. Este processo extrapola uma nova posição tendo como base a posição armazenada e assumindo que o vector velocidade se mantém constante. Assim, é construído um novo pacote de actualização com a informação extrapolada que é enviado para o FS2002. Este pacote extrapolado é armazenado na NeemAdapter em substituição do anterior, como qualquer actualização original recebida da rede.

5 Avaliação

Uma vez que não dispomos de recursos suficientes para construir uma experiência controlada com centenas de espectadores, foi recolhido tráfego do FS2002 sendo posteriormente utilizado para simulação. Este tráfego foi recolhido interceptando a comunicação para a API do DirectPlay. A utilização de simulação tem ainda a vantagem de permitir comparar o desempenho de diferentes protocolos quando submetidos exactamente ao mesmo tráfego.

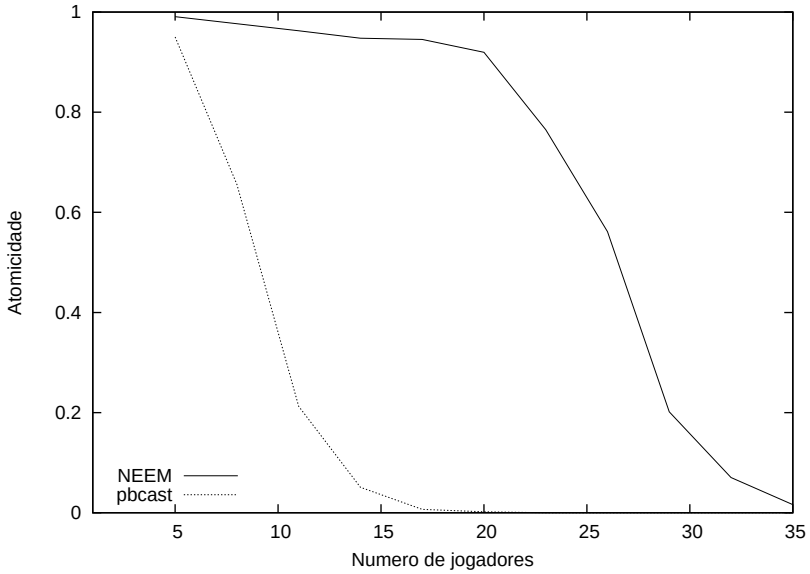

Figura 5: Desempenho do NEEM.

5.1 Modelo de simulação

O modelo usado simula o comportamento do NEEM permitindo assim saber quais as mensagens que são perdidas quando o sistema é parametrizado com o tráfego recolhido do FS2002. Este modelo é composto por um conjunto de processos que executam o protocolo NEEM, retransmitindo, armazenando e entregando mensagens. O mecanismo de filiação não é simulado, sendo as listas de processos conhecidos por cada nó geradas previamente de maneira aleatória, verificando-se apenas que todos os nós são mutuamente atingíveis. A rede é modelada por um conjunto de filas ligando os nós participantes. A largura de banda configurada para cada nó é então dividida por um conjunto de ligações que pertencem à lista de processos conhecidos e controlada usando um mecanismo simples de "balde furado".

O modelo foi então validado de três formas distintas. Em primeiro lugar, pela simulação e análise de sistemas elementares para os quais é possível prever o comportamento analiticamente. A validação de sistemas complexos foi feita comparando os resultados obtidos na simulação do protocolo pbcast com outros resultados de simulação obtidos por terceiros [8, 3, 10]. Finalmente, os resultados da simulação do protocolo NEEM foram comparados com resultados obtidos experimentalmente [4].

Os resultados apresentados foram obtidos usando 500 nós. A cada ligação ponto-a-ponto é associado um *buffer* com capacidade para armazenar 10 mensa-

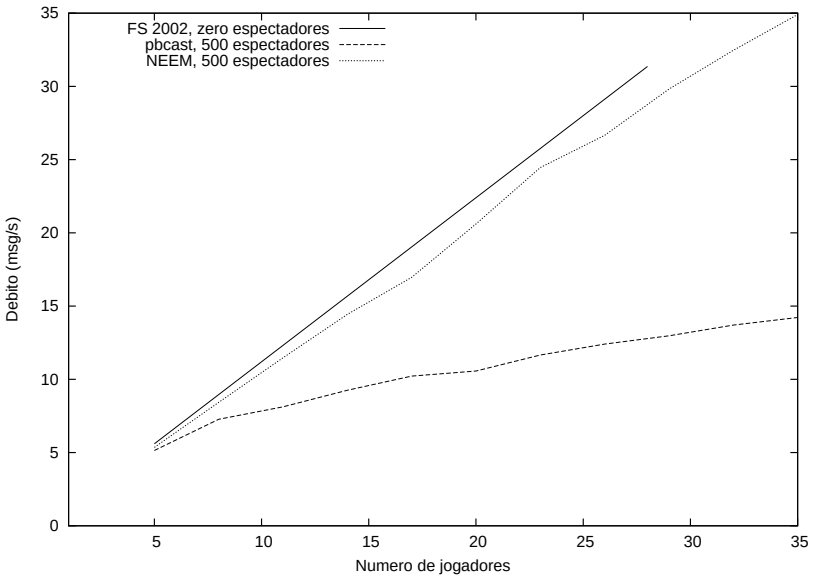

Figura 6: Desempenho do FS2002.

gens na camada de gestão dos *buffers*. Na difusão epidémica, a vista local tem 12 entradas, as mensagens são disseminadas para 6 destinos e no máximo cada mensagem é retransmitida 6 vezes. A rede foi configurada com 56kbps de largura de banda quer no *downlink* quer no *uplink* e com 25ms de latência, no pior caso. A justificação destes valores encontra-se em [8]. Cada simulação durou 300s, dos quais os primeiros 100s e os últimos 50s foram retirados para evitar estados de transição.

O padrão de tráfego retirado do FS2002 pode ser dividido em duas partes: uma parte do tráfego que nunca fica obsoleta e o restante tráfego que pode ser dividido em sequências, tantas quantos os aviões no ar. Podemos afirmar que cada mensagem numa sequência é considerada obsoleta pela sua sucessora com 46% de probabilidade – tendo $F = 0.01$.

5.2 Resultados

A análise do desempenho do NEEM e do desempenho do FS2002 resultante da integração apresentada foi feita por comparação a um protocolo de difusão epidémica bastante conhecido, denominado pbcast [8].

A Figura 5 apresenta a atomicidade da entrega de mensagens, isto é, a figura apresenta a contagem das mensagens que foram entregues a mais de 95% dos processos nas diferentes situações consideradas com um número crescente de jo-

gadores. Da análise desta figura podemos concluir que a gestão eficaz dos *buffers* presente no NEEM aumenta a utilidade do protocolo de difusão probabilista de 5 para 20 jogadores em simultâneo.

Por seu lado, a Figura 6 considera o impacto do NEEM quanto ao número de espectadores suportados pelo FS2002. Assim, a Figura 6 compara o desempenho do FS2002 original sem espectadores com o NEEM com 500 espectadores, tendo $F = 0.02$, ilustrando o débito de mensagens que nunca ficam obsoletas. Com o protocolo centralizado original do FS2002 e com a largura de banda assumida, todas as mensagens podem teoricamente ser transmitidas com sucesso até um máximo de 28 jogadores. Note-se que, como a transmissão é feita por cada um dos jogadores, este número incluirá também os espectadores. O número máximo de espectadores suportado será pois de 27 e apenas no caso extremo em que há apenas um jogador.

Com o NEEM a difusão das actualizações é feita de forma distribuída. Desta forma é possível suportar um grande número de espectadores independentemente do número de jogadores. A Figura 6 mostra como com 500 espectadores é possível entregar a quase totalidade das mensagens que nunca ficam obsoletas. Em contraste, um protocolo de difusão epidémica tradicional - pbcast - perderá mensagens que nunca ficam obsoletas.

O impacto destas perdas é visível quando se compara a rota extrapolada a partir de ambos os protocolos epidémicos com a rota original do avião. A Figura 7 apresenta a comparação da rota original do avião com a rota gerada com o recurso à extrapolação. Como se pode ver no gráfico, com o NEEM a disparidade raramente ultrapassa os 100m, e a maioria dos pacotes que sofrem extrapolação apresentam desvios na ordem dos 20m resultando numa diferença média de 32m. Em contraste, a rota extrapolada a partir do pbcast implica erros substancialmente maiores, resultando numa média de 49m.

6 Conclusões

Este artigo apresenta a integração de um jogo multi-utilizador, em particular, do Microsoft Flight Simulator 2002 com o protocolo NEEM, que combina difusão epidémica com aproveitamento da semântica das mensagens. A avaliação desta integração é feita recorrendo a uma concretização e a um simulador. Os resultados obtidos mostram que, em contraste com o protocolo utilizado originalmente pelo jogo que suporta apenas poucos jogadores ou espectadores, esta integração torna possível a difusão do estado do jogo para centenas de espectadores. A utilização da semântica das mensagens possibilita o suporte de centenas de espectadores não sendo para isso necessário restringir o número de jogadores em simultâneo como aconteceria com um protocolo epidémico tradicional.

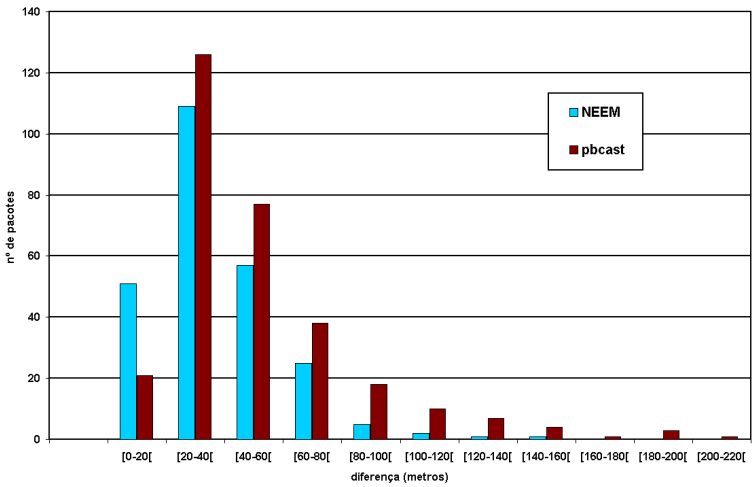

Figura 7: Comparação de rotas.

Referências

[1] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2), 1999.

[2] S. Drucker, L. He, M. Cohen, C. Wong, and A. Gupta. Spectator games: A new entertainment modality of networked multiplayer games. Technical report, Microsoft Research, 2002.

[3] P. Eugster, R. Guerraoui, S. Handrukande, A.-M. Kermarrec, and P. Kouznetsov. Lighweight probabilistic broadcast. In *IEEE International Conference on Dependable Systems and Networks (DSN)*, 2001.

[4] S. Formigo, J. Pereira, and L. Rodrigues. Difusão probabilista com fiabilidade semântica. In *Actas da 5^a Conferência sobre Redes de Computadores*, Faro, Portugal, 2002.

[5] A.J. Ganesh, A.-M. Kermarrec, and L. Massoulie. Peer-to-peer membership management for gossip-based protocols. *IEEE Transactions on Computers*, 2003.

[6] A. Goel, C. Krasic, K. Li, and J. Walpole. Supporting low latency TCP-based media streams. In *International Workshop on Quality of Service (IWQoS'2002)*, 2002.

[7] K. Guo. *Scalable Message Stability Detection Protocols*. PhD thesis, Cornell University, Computer Science Department, 1998.

[8] M. Hayden and K. Birman. Probabilistic broadcast. Technical Report TR96-1606, Cornell University, Computer Science, 1996.

[9] A.-M. Kermarrec, L. Masssoulié, and A. Ganesh. Reliable probabilistic communication in large-scale information dissemination systems. Technical Report 2000-105, Microsoft Research, 2000.

[10] P. Kouznetsov, R. Guerraoui, S. Handurukande, and A.-M. Kermarrec. Reducing noise in gossip-based reliable broadcast. In *IEEE Symposium on Reliable Distributed Systems (SRDS)*, 2001.

[11] J. Lin and S. Paul. RMTP: A reliable multicast transport protocol. In *IEEE Conference on Computer Communications (INFOCOM)*, 1996.

[12] Microsoft Corp. *DirectPlay 8.1*, 2002.

[13] Microsoft Corp. *Microsoft Flight Simulator 2002 Software Development Kit – Multiplayer/Flight instructor*, 2002.

[14] H. Miranda, A. Pinto, and L. Rodrigues. Appia, a flexible protocol kernel supporting multiple coordinated channels. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2001.

[15] J. Pereira, L. Rodrigues, M.João Monteiro, R. Oliveira, and A.-M. Kermarrec. NEEM: Network-friendly Epidemic Multicast. In *International Symposium on Reliable Distributed Systems (SRDS'2003)*, 2003.

[16] J. Pereira, L. Rodrigues, and R. Oliveira. Semantically reliable multicast: Definition, implementation and performance evaluation. *IEEE Transactions on Computers, Special Issue on Reliable Distributed Systems*, 2003.

[17] R. Piantoni and C. Stancescu. Implementing the Swiss Exchange Trading System. In *IEEE International Symposium on Fault-Tolerant Computing (FTCS)*, 1997.