(11) Número de Publicação: **PT 1926282 E**

(51) Classificação Internacional:
H04L 29/06 (2011.01) *H04L 29/14* (2011.01)

(12) **FASCÍCULO DE PATENTE DE INVENÇÃO**

(22) Data de pedido: **2000.10.13**

(30) Prioridade(s): **1999.10.14 US 159360 P**
2000.03.28 US 536639

(43) Data de publicação do pedido: **2008.05.28**

(45) Data e BPI da concessão: **2012.11.21**
018/2013

(73) Titular(es):
NOKIA CORPORATION
KEILALAHDENTIE 4 02150 ESPOO **FI**

(72) Inventor(es):
KHIEM LE **US**
CHRISTOPHER CLANTON **US**
HAIHONG ZHENG **US**
ZHIGANG LIU **US**

(74) Mandatário:
JOSÉ RAUL DE MAGALHÃES SIMÕES
RUA CASTILHO, 167 - 2.º 1070-050 LISBOA **PT**

(54) Epígrafe: **MÉTODO E SISTEMA PARA COMPRIMIR E DESCOMPRIMIR CABEÇALHOS DE PACOTES**

(57) Resumo:
A INVENÇÃO É UM SISTEMA E MÉTODO PARA SINCRONIZAR A TRANSMISSÃO DE CABEÇALHOS COMPRIMIDOS NOS PACOTES DE DADOS ENTRE UM TRANSMISSOR E UM RECETOR COM UMA APLICAÇÃO SEM FIOS PREFERIDA QUE É UM APERFEIÇOAMENTO DE RFC 2508. NUM SISTEMA COM UM TRANSMISSOR QUE TRANSMITE UMA DIVERSIDADE DE PACOTES, CADA UM CONTENDO UM CABEÇALHO, A UM RECETOR, UM MÉTODO DE SINCRONIZAÇÃO DA TRANSMISSÃO DE CABEÇALHOS COMPRIMIDOS ENTRE O TRANSMISSOR E O RECETOR DE ACORDO COM A INVENÇÃO INCLUI A TRANSMISSÃO DE UM PACOTE ATUAL A PARTIR DO TRANSMISSOR PARA O RECETOR COM INFORMAÇÕES A INDICAR QUE O TRANSMISSOR ESTÁ PREPARADO PARA ENVIAR PACOTES TRANSMITIDOS SUBSEQUENTEMENTE, EM QUE OS RESPETIVOS CABEÇALHOS DEVEM SER COMPRIMIDOS EM COMPARAÇÃO COM O CABEÇALHO INCLUÍDO NO PACOTE ATUAL; E A TRANSMISSÃO A PARTIR DO RECETOR PARA O TRANSMISSOR DE UM PACOTE DE RECONHECIMENTO A INDICAR QUE O RECETOR RECEBEU O PACOTE ATUAL.

RESUMO

MÉTODO E SISTEMA PARA COMPRIMIR E DESCOMPRIMIR CABEÇALHOS DE PACOTES

A invenção é um sistema e método para sincronizar a transmissão de cabeçalhos comprimidos nos pacotes de dados entre um transmissor e um recetor com uma aplicação sem fios preferida que é um aperfeiçoamento de RFC 2508. Num sistema com um transmissor que transmite uma diversidade de pacotes, cada um contendo um cabeçalho, a um recetor, um método de sincronização da transmissão de cabeçalhos comprimidos entre o transmissor e o recetor de acordo com a invenção inclui a transmissão de um pacote atual a partir do transmissor para o recetor com informações a indicar que o transmissor está preparado para enviar pacotes transmitidos subsequentemente, em que os respetivos cabeçalhos devem ser comprimidos em comparação com o cabeçalho incluído no pacote atual; e a transmissão a partir do recetor para o transmissor de um pacote de reconhecimento a indicar que o recetor recebeu o pacote atual.

DESCRIÇÃO

MÉTODO E SISTEMA PARA COMPRIMIR E DESCOMPRIMIR CABEÇALHOS DE PACOTES

A presente invenção refere-se à compressão e descompressão de cabeçalhos nas transmissões de pacotes de dados.

Relativamente à multimédia em tempo real baseada no Protocolo de Internet (IP – Internet Protocol), o Protocolo de Transferência em Tempo Real (RTP – Real-Time Transfer Protocol) é utilizado predominantemente em cima do Protocolo de Datagrama de Utilizador (UDP – User Datagram Protocol/IP). O RTP é descrito em detalhe em RFC 1889. O tamanho dos cabeçalhos IP/UDP/RTP combinados é de, pelo menos, 40 bytes para IPv4 e de, pelo menos, 60 bytes para IPv6. Um total de 40 a 60 bytes de *overhead* (espaço sem dados) por pacote pode ser considerado pesado nos sistemas (por ex., tal como redes celulares) em que a eficiência espetral é uma preocupação principal. Consequentemente, são necessários mecanismos de compressão de cabeçalhos IP/UDP/RTP adequados. Um esquema de compressão de cabeçalhos atual é descrito em RFC 2508, por S. Casner, V. Jacobson, "Compressing IP/UDP/RTP Headers for Low Speed Serial Links", *Internet Engineering Task Force* (IETP), fevereiro de 1999, e que é capaz de comprimir o cabeçalho IP/UDP/RTP de 40 a 60 bytes até 2 ou 4 bytes através de ligações ponto a ponto. Os algoritmos de compressão de cabeçalhos existentes baseiam-se na observação de que a maioria dos campos dos cabeçalhos de pacote IP permanece constante num fluxo de pacotes durante uma sessão. Deste modo, é possível comprimir as informações de cabeçalho estabelecendo um estado de compressão (as informações de cabeçalho completo) no descompressor e transportando

simplesmente a quantidade mínima de informações de cabeçalho do compressor para o descompressor.

Os esquemas de compressão de cabeçalhos IP/UDP/RTP, conforme descrito por exemplo em RFC 2508, tiram partido do facto de determinados campos de informação transportados no cabeçalho 1.) não serem alterados (campos de cabeçalho de 'Tipo 1') ou 2.) serem alterados de uma forma completamente previsível (campos de cabeçalho de 'Tipo 2'). Outros campos, denominados campos de cabeçalho de 'Tipo 3', variam de tal modo que têm de ser transmitidos de alguma forma em todos os pacotes (ou seja, não são compressíveis).

Os exemplos de campos de cabeçalho de Tipo 1 são o endereço IP, número de porta UDP, RTP SSRC (fonte de sincronização), etc. Estes campos só necessitam de ser transmitidos ao recetor/descompressor uma vez durante o curso de uma sessão (como parte dos pacotes transferidos no estabelecimento da sessão, por exemplo). Os campos de Tipo 1 são igualmente designados por campos 'inalteráveis'.

Os exemplos de campos de cabeçalho de Tipo 2 são os campos de carimbo de hora RTP, número de sequência RTP e IP ID. Todos têm uma tendência para incrementar em alguma quantidade constante de pacote(n) para pacote (n+1). Deste modo, não é necessário transmitir estes valores em todos os cabeçalhos. Só é preciso que o recetor/descompressor seja informado do valor de incremento constante, doravante referido como a diferença de primeira ordem (FOD – First Order Difference), associada a cada campo que apresente este comportamento. O recetor/descompressor utiliza estes FODs para regenerar os valores de campo de Tipo 2 atualizados ao reconstruir o cabeçalho original. Os campos de Tipo 2 fazem parte dos campos 'alteráveis'.

Convém realçar que, às vezes, os campos de Tipo 2 serão alterados de alguma forma irregular. A frequência desses eventos depende de diversos fatores, incluindo o tipo de multimédia que está a ser transmitido (por ex., voz ou vídeo), a fonte de multimédia efetiva (por ex., para voz, o comportamento pode variar entre altifalantes), e o número de sessões que partilham simultaneamente o mesmo endereço IP.

Um Exemplo de um campo de cabeçalho de Tipo 3 é o M-bit RTP (Marcador), que indica a ocorrência de alguma fronteira na multimédia (por ex., o fim de um fotograma de vídeo). Uma vez que a multimédia varia normalmente de formas imprevisíveis, estas informações não podem ser fielmente previstas. Os campos de Tipo 3 fazem parte dos campos 'alteráveis'.

O descompressor mantém as informações de contexto de descompressão que contêm todas as informações pertinentes relacionadas com a reconstrução do cabeçalho. Estas informações são essencialmente campos de Tipo 1, valores FOD e outras informações. Quando os pacotes se perdem ou ficam danificados, o descompressor pode perder a sincronização com o compressor, de modo a não poder mais reconstruir pacotes corretamente. A perda de sincronização pode ocorrer quando os pacotes se perdem ou ficam danificados durante a transmissão entre o compressor e o descompressor.

Dado o referido acima, o compressor necessita de transmitir três tipos diferentes de cabeçalhos durante o curso de uma sessão:

- Cabeçalho Completo (FH - Full Header): contém o conjunto completo de todos os campos de cabeçalho (Tipos 1, 2 e 3). Este tipo de cabeçalho é o menos

ideal para enviar devido ao seu tamanho grande (por ex., 40 bytes para IPv4). Em geral, é desejável enviar um pacote FH apenas no início da sessão (para estabelecer dados de Tipo 1 no recetor). A transmissão de pacotes FH adicionais tem efeitos adversos na eficiência do algoritmo de compressão. Quando o compressor transmite pacotes FH, é referido como estando no 'estado FH'.

• Primeira Ordem (FO - First Order): contém informações de cabeçalho mínimas (por ex. campos de Tipo 3), campos de controlo específicos de compressor/descompressor (específicos para o algoritmo de compressão em utilização), e informações que descrevem alterações nos campos FOD atuais. Um pacote FO é basicamente um pacote SO (descrito abaixo), com informações adicionais que estabelecem novas informações FOD para um ou mais campos de Tipo 2 no descompressor. Se a compressão de cabeçalhos estiver a ser aplicada num fluxo VoIP (voice over internet protocol - voz sobre protocolo de Internet), a transmissão de um pacote FO poderá ser acionada pela ocorrência de um segmento contínuo de voz após um intervalo de silêncio na voz. Esse evento resulta em alguma alteração inesperada no valor de carimbo de hora RTP e numa necessidade de atualizar o carimbo de hora RTP no recetor por um valor que não o FOD atual. O tamanho de pacotes FO depende do número de campos de Tipo 2 cuja diferença de primeira ordem foi alterada (e a quantidade do valor absoluto de cada alteração). Quando o compressor transmite pacotes FO, é referido como estando no 'estado FO'.

• Segunda Ordem (SO - Second Order): um pacote SO contém informações de cabeçalho mínimas (por ex. campos de Tipo 3), e campos de controlo específicos de

compressor e descompressor. O modo preferido de funcionamento para o compressor e descompressor é a transmissão e receção de pacotes SO, devido ao seu tamanho mínimo (na ordem de apenas 2 bytes ou até menos). Quando o compressor transmite pacotes SO, é referido como estando no 'estado SO'. Os pacotes SO só são transmitidos se o cabeçalho atual se adaptar ao padrão de um FOD.

O RFC 2508 baseia-se no conceito de que, na maior parte das vezes, os campos RTP que mudam de um pacote para o seguinte, tal como o carimbo de hora RTP, podem ser previstos através de extrapolação linear de pacotes SO transmitidos. Essencialmente, a única informação que tem de ser enviada é um número de sequência, utilizado para deteção de erro e perda de pacotes (bem como um ID de informações de contexto). Quando o transmissor determina que a extrapolação linear não pode ser aplicada no pacote atual relativamente ao pacote imediatamente anterior, é transmitido um pacote FO. Para iniciar a sessão, é transmitido um pacote FH. Além disso, quando o recetor determina que existe uma perda de pacotes (conforme detetado através de um número de sequência que é incrementado em mais de 1), o recetor solicita explicitamente ao transmissor para transmitir o cabeçalho completo de modo a permitir uma nova sincronização.

Contudo, a compressão de cabeçalhos definida em RFC 2508 não é perfeitamente adequada para determinados ambientes (tais como ambientes celulares ou sem fios), em que a largura de banda tem uma grande procura e os erros são comuns. No esquema de compressão de cabeçalhos RFC 2508, parte-se do princípio de que o carimbo de hora RTP tem, na maior parte das vezes, um padrão linearmente crescente. Quando o cabeçalho está em conformidade com o padrão, essencialmente só é necessário um número de

sequência abreviado enviado como SO no cabeçalho comprimido. Quando o cabeçalho não está em conformidade com o padrão, a diferença entre os carimbos de hora RTP do cabeçalho atual e do anterior é enviada no cabeçalho comprimido FO.

O requisito de largura de banda adicional pode manifestar-se de várias formas, dependendo do ambiente de funcionamento. Por exemplo, nos sistemas celulares é, no geral, bastante aconselhável limitar a utilização da largura de banda o mais possível, uma vez que é um recurso escasso.

O RFC 2508 sofre de falta de robustez para suportar perdas ou erros de cabeçalho, uma vez que a descompressão do cabeçalho atual só pode ser efetuada se o cabeçalho imediatamente anterior também tiver sido recebido sem erros e descomprimido. A compressão de cabeçalhos definida em RFC 2508 não é perfeitamente adequada para ambientes celulares, em que a largura de banda tem uma grande procura e não são invulgares grandes rajadas de erros. Em RFC 2508, quando é detetada uma perda de pacotes, os pacotes subsequentes com cabeçalhos comprimidos são invalidados, com um outro requisito que é necessário para enviar um cabeçalho de tamanho grande para recuperar do erro. Os cabeçalhos de tamanho grande consomem largura de banda e criam picos na procura de largura de banda que são mais difíceis de gerir.

A utilização apenas de um número de sequência abreviado (um com um número limitado de bits) para detetar a perda de pacotes não é robusta para uma rede propensa a erros, tal como numa rede sem fios em que a grande perda pode acontecer a qualquer momento. Neste caso, a grande perda é definida como perda do ciclo de sequência ou de pacotes em fila. Numa situação de grande perda, pode perder-se uma série de pacotes no número de pacotes do

ciclo de sequência com uma identificação de pacote definida por um número limitado de bits e, por consequência, o número de sequência no pacote recebido pelo descompressor (dispositivo de receção na ligação ascendente ou ligação descendente) do recetor é moldado (repete-se). Por exemplo, assumindo que o número de sequência consiste em bits k, o ciclo de sequência é igual a 2^k.

Conforme ilustrado na Fig. 1, o compressor (dispositivo de transmissão na ligação ascendente ou ligação descendente) enviou o pacote com a seq=n no momento t_0, e os seguintes 2^k pacotes, a começar no pacote com a seq=n+1 no momento t_1 e a terminar no pacote com a seq=n no momento t_2. No momento t_3, o compressor envia um pacote com um número de sequência igual a n+1 novamente. Partamos do princípio de que um pacote com um número de sequência igual a n+1 é enviado no momento t_1 até o pacote com o número de sequência igual a n ser enviado no momento t_2, em que todos se perdem devido a uma grande perda e, em seguida, o descompressor só recebe o pacote com o número de sequência igual a n enviado no momento t_0 e o pacote com o número de sequência igual a n+1 enviado no momento t_3. Com base no esquema de deteção de perda de pacotes atual definido em RFC 2508, o descompressor conclui que não existe nenhuma perda de pacotes e descomprime o pacote de uma forma errada. Isto não só afeta a exatidão da descompressão do pacote com um número de sequência igual a n+1, como também os pacotes subsequentes.

A presente invenção pode fornecer melhores transmissão e receção de pacotes em ambientes, tais como comunicações sem fios, que são propensos a interrupção periódica de receção de pacotes, tal como a causada pela diminuição de intensidade, etc. A invenção pode fornecer um melhor desempenho de transmissão e receção de pacotes em comparação com RFC 2508, incluindo a eliminação do problema

de moldagem (*wrap around*) do estado da técnica descrito acima na Fig. 1. A descodificação adequada de um cabeçalho comprimido num pacote atual de acordo com a invenção não depende da descompressão correta de um pacote imediatamente anterior, tal como com RFC 2508.

Esta invenção fornece um mecanismo que deteta uma grande perda ao nível da compressão de cabeçalhos, bem como um esquema de recuperação correspondente após a deteção da grande perda. Geralmente, a invenção é aplicável a protocolos de comunicação onde a sincronização de sequências tem de ser mantida entre o transmissor e o recetor, na presença de grandes rajadas de erros.

A compressão de cabeçalhos adaptável é uma estrutura geral para compressão de cabeçalhos robusta que pode ser parametrizada para explicar a existência/não-existência e as características de desempenho de um canal inverso. A estrutura inclui três modos básicos de funcionamento:

- Modo Determinista Bidirecional: este modo é utilizado no caso de existir um canal inverso 'bem definido', que pode ser utilizado para transportar vários tipos de informações de retorno do descompressor para o compressor. Um exemplo desse retorno do descompressor é um reconhecimento utilizado, por ex., para passar de um estado de compressão inferior para um estado de compressão superior.

- Modo Oportunista Bidirecional: este modo de funcionamento é utilizado no caso de existir um canal inverso, mas é definido 'vagamente', ou seja, o canal pode não estar sempre disponível ou pode ser lento/não fidedigno.

• Modo Unidirecional (Pessimista ou Otimista): este modo de funcionamento é utilizado quando não existe nenhum canal inverso de qualquer tipo.

A invenção é definida pelas reivindicações.

A Fig. 1 ilustra a deficiência da perda de pacotes com RFC 2508.

A Fig. 2 ilustra um exemplo de uma arquitetura de sistema que pode ser utilizada para praticar a presente invenção.

A Fig. 3 ilustra conceptualmente as informações de contexto de compressão.

A Fig. 4 ilustra conceptualmente as informações de contexto de descompressão.

A Fig. 6 ilustra a transição de um compressor a partir de cabeçalhos de transmissão com um número elevado de bits para cabeçalhos com um número inferior de bits utilizando reconhecimentos.

A Fig. 7 ilustra a transição de um compressor a partir de cabeçalhos de transmissão com uma primeira ordem de compressão para cabeçalhos com uma segunda ordem de compressão.

A Fig. 8 ilustra a deteção e recuperação de pacotes quando ocorre uma moldagem com um número de pacotes superior a 2^k, em que são utilizados bits k para codificar números de sequência n definidos pelos bits k.

A Fig. 9 ilustra o funcionamento de um compressor e descompressor utilizando reconhecimentos para controlar a transição entre números de sequência que contêm bits k e bits ℓ de extensão e novamente para bits k.

A Fig. 10 ilustra a redução de largura de banda alcançada através da transmissão de uma sequência de pacotes FH e FO antes de ser gerado um reconhecimento pelo descompressor significando a receção de um pacote FH.

As Figs. 14A a F ilustram formatos de pacotes que são transmitidos pela presente invenção.

A Fig. 15 ilustra a mudança de um estado de compressão de um compressor para um estado de compressão superior apenas depois de ser recebido um reconhecimento de um descompressor.

A Fig. 16 ilustra a mudança de um estado de compressão de um compressor para um estado de compressão superior antes de um número atual de pacotes chegar a uma descompressão quando é recebido um reconhecimento.

A Fig. 17 ilustra a mudança de um estado de compressão de um compressor para um estado de compressão superior depois de um número predefinido de pacotes chegar a um descompressor antes de chegar um reconhecimento.

A Fig. 18 ilustra a mudança de um estado de compressão de um compressor para um estado de compressão superior apenas depois de um número atual de pacotes chegar a uma descompressão.

A Fig. 19 ilustra graficamente uma comparação da invenção com RFC 2508.

A Fig. 20 ilustra uma análise gráfica do desempenho da presente invenção.

A Fig. 2 ilustra um sistema exemplar no qual a presente invenção pode ser praticada. Contudo, convém compreender que a presente invenção não está limitada a isso, com outras arquiteturas de sistema sendo igualmente aplicáveis na prática da invenção. Um terminal 102 está ligado a uma rede IP 108. O terminal 102 pode ser, sem limitação, um computador pessoal ou afins a executar RTP/UDP/IP e a fornecer amostras de voz em pacotes RTP para transmissão através da rede IP 108. O terminal 102 inclui um ponto final RTP 104 que identifica este terminal (por ex., incluindo endereço IP, número de porta, etc.) como uma fonte e/ou destino de pacotes RTP. Contudo, embora a rede IP 108 seja fornecida como um exemplo de uma rede de pacotes, podem ser utilizados outros tipos de redes de comutação de pacotes ou afins em vez disso. O terminal 102 inclui igualmente um temporizador local 103 para gerar um carimbo de hora.

As infraestruturas de rede de acesso (ANI - Access Network Infrastructures) 110 e 120, que podem residir num subsistema de estação de base (BSS - Base Station Subsystem), estão ligadas à rede IP 108. As ANIs são entidades de rede e nós de rede. Vários terminais móveis sem fios, que são entidades de rede e nós de rede e funcionam como compressores móveis e descompressores móveis (são ilustrados dois terminais sem fios 130 e 150), são acoplados através de ligações de radiofrequência (RF) 140 às ANIs 110 e 120. Quando um dos terminais móveis 130 e/ou 150 se desloca, é necessário que os terminais, de vez em

quando, como uma consequência do movimento fora do alcance da ligação de rádio a uma ANI, sejam entregues a outra ANI. Quando a compressão e a descompressão de cabeçalhos são utilizadas e se encontram na ANI, este processo também necessita da transferência de informações de contexto de compressão e descompressão de uma ANI (antiga) para outra ANI (nova) para obter continuidade, por ex. se os terminais móveis 130 e/ou 150 se deslocarem e forem entregues da ANI 110 à ANI 120. A transferência, conforme descrito abaixo, pode acontecer em vários momentos, mas para minimizar a interrupção deve estar concluída no momento em que a ANI nova assume o papel de compressão/descompressão de cabeçalhos da ANI antiga. O reposicionamento das funções de compressão/descompressão ocorre quando a nova entidade de rede assume o controlo num determinado momento. Por outro lado, a transferência de informações de contexto pode ser difundida através de um material de tempo e anteceder o reposicionamento. A ligação RF 140 inclui, conforme ilustrado, um tráfego de ligações ascendentes 142 (dos terminais móveis 130 e 150 para a ANI 110) e um tráfego de ligações descendentes 144 (da ANI 110 para os terminais móveis 130 e 150). Os terminais móveis 130 e/ou 150 são entregues a partir de uma ANI, tal como ANI 110, quando um ou mais dos terminais móveis se deslocam para outra ANI, por ex. ANI 120. Cada ANI funciona como interface com um ou mais dos terminais sem fios (ou radiofrequência) (incluindo o terminal 130) numa região para a rede IP 108, incluindo a conversão entre sinais fixos (fornecidos a partir da rede IP 108) e sinais sem fios ou RF (fornecidos a ou a partir dos terminais 130 e 150). Deste modo, cada ANI permite que os pacotes, tais como, mas não se limitando a, pacotes RTP transmitidos e recebidos a partir da rede IP 108, sejam enviados através da ligação RF 140 a, pelo menos, um dos terminais sem fios 130 e 150, e permite que a transmissão de pacotes, tais como pacotes RTP, mas não se limitando a pacotes RTP, seja efetuada a partir dos terminais 130 e 150

e através da rede IP 108 para outro terminal, tal como o terminal 102.

Cada ANI inclui uma diversidade de entidades. A representação e explicação mais detalhadas da ANI 110 são fornecidas para facilitar a compreensão da arquitetura e do funcionamento de todas as ANIs existentes na rede. Todas as ANIs podem pertencer à mesma arquitetura da ANI 110, mas não são ilustradas no mesmo grau de detalhe. A ANI 110 inclui um ou mais adaptadores ANI (ANI_AD), tais como ANI_AD 112 (ilustrado em detalhe) e ANI_AD 114, em que cada um inclui preferencialmente um temporizador 113 para fornecer um carimbo de hora. Cada ANI_AD efetua a compressão (antes do tráfego de ligações descendentes) e a descompressão (depois do tráfego de ligações ascendentes) de cabeçalhos. Os cabeçalhos (um ou mais campos de cabeçalho, tais como um carimbo de hora e número de sequência) para pacotes RTP recebidos a partir da rede IP 108 são comprimidos por ANI_AD 112 antes da transmissão para os terminais 130 e 150 através do tráfego de ligações descendentes 142, e os cabeçalhos de pacote recebidos a partir dos terminais móveis 130 e 150 são descomprimidos por ANI_AD 112 antes da transmissão para a rede IP 108. O ANI_AD 110 funciona como um transmissor/recetor (transcetor) e especificamente como um compressor/descompressor 115 com o compressor a comprimir pacotes de dados antes da transmissão e o descompressor a descomprimir pacotes de dados após a receção. O ANI_AD 110 funciona como interface com terminais localizados numa área específica ou diferente na região para a rede IP 108. O ANI_AD 112 inclui um temporizador 113 para implementar uma técnica de descompressão baseada no temporizador. O ANI_AD 112 inclui igualmente uma função de redução de instabilidade de sinal (JRF - Jitter Reduction Function) 116 que funciona para medir a instabilidade de sinal nos pacotes (ou cabeçalhos) recebidos através da rede 108 e

rejeitar quaisquer pacotes/cabeçalhos que tenham instabilidade de sinal excessiva.

Cada terminal inclui uma diversidade de entidades. A explicação mais detalhada do terminal móvel 130 é fornecida para facilitar a compreensão da conceção e do funcionamento de todos os terminais móveis 130 e 150 existentes na rede que têm uma conceção e um funcionamento semelhantes. Cada um dos terminais móveis também pode funcionar como um compressor/descompressor nas comunicações fora do alcance das ANIs 110 e 120 e, especificamente, com outras redes. O terminal móvel 130 inclui um ponto final RTP 132 que é uma fonte (transmissor) e/ou destino (recetor) para pacotes RTP e identifica o endereço IP, o número de porta, etc. do terminal. O terminal móvel 130 inclui um adaptador de terminal (MS_AD) 136 que efetua a compressão (pacotes para serem transmitidos através do tráfego de ligações ascendentes 142) e a descompressão (pacotes recebidos através do tráfego de ligações descendentes 144) de pacotes. Deste modo, o adaptador de terminal (MS_AD) 136 pode ser considerado como um compressor/descompressor (transcetor) de cabeçalhos 137, semelhante ao compressor/descompressor ANI_AD. A terminologia MS_AD tem o mesmo significado de AD. O MS_AD 136 inclui igualmente um temporizador 134 (um temporizador recetor) para calcular uma aproximação (ou estimativa) de um carimbo de hora RTP de um cabeçalho atual e para medir o tempo decorrido entre os pacotes recebidos sucessivamente para localizar a perda de pacotes durante a transmissão para o terminal através da degradação sem fios, tal como a diminuição de intensidade. O MS_AD 136 pode utilizar informações adicionais no cabeçalho RTP para aperfeiçoar ou corrigir a aproximação de carimbo de hora, conforme descrito no Pedido de Patente copendente n.º de série 091377,913. A aproximação de carimbo de hora pode ser corrigida ou ajustada com base no carimbo de hora comprimido fornecido no cabeçalho RTP.

Deste modo, é possível utilizar um temporizador local e um carimbo de hora comprimido para regenerar o carimbo de hora correto para cada cabeçalho RTP.

Os pacotes RTP, incluindo pacotes com cabeçalhos comprimidos e não comprimidos, são transmitidos na rede, tal como, mas sem limitação, a rede exemplar da Fig. 2, através de uma ligação de dados (tal como uma ligação sem fios 140), em que a largura de banda tem uma grande procura e os erros não são invulgares. A presente invenção não é limitada a uma ligação sem fios, mas é aplicável a uma grande variedade de ligações (incluindo ligações fixas, etc.).

A Fig. 3 ilustra conceptualmente as informações de contexto de compressão e os exemplos. As informações de contexto de compressão são um conjunto, um subconjunto ou uma representação de um subconjunto de informações que podem ser de qualquer tipo num cabeçalho utilizado pelo compressor como uma entrada para o algoritmo de compressão para produzir um cabeçalho comprimido e podem ser transmitidas de uma entidade para outra entidade. A outra entrada é a partir da fonte dos cabeçalhos a serem comprimidos.

A Fig. 4 ilustra conceptualmente as informações de contexto de descompressão e os exemplos. As informações de contexto de descompressão são um conjunto, um subconjunto ou uma representação de um subconjunto de informações que podem ser de qualquer tipo num cabeçalho utilizado pelo descompressor como uma entrada para o algoritmo de descompressão para produzir um cabeçalho descomprimido e podem ser transmitidas de uma entidade para outra entidade. A outra entrada é a partir da fonte dos cabeçalhos a serem descomprimidos.

Tanto as informações de contexto de compressão como de descompressão são dinâmicas, ou seja, podem ser atualizadas pelo compressor e descompressor respetivamente. A frequência de atualizações depende do esquema de compressão de cabeçalhos. Os eventos que podem resultar numa atualização das informações de contexto de compressão no compressor incluem a compressão de um cabeçalho de entrada ou a receção do retorno a partir do descompressor. Os eventos que podem resultar numa atualização das informações de contexto de descompressão no descompressor incluem a descompressão de um cabeçalho de entrada.

A Compressão de Cabeçalhos Adaptável (ACE - Adaptive Header Compression) é uma estrutura geral para compressão de cabeçalhos robusta que pode ser parametrizada para explicar a existência/não-existência e as características de desempenho de um canal de retorno. A estrutura inclui três modos básicos de funcionamento:

- Modo Determinista Bidirecional: este modo é utilizado no caso de existir um canal inverso 'bem definido', que pode ser utilizado para transportar vários tipos de informações de retorno do descompressor para o compressor. Um exemplo desse retorno do descompressor é o reconhecimento (ACK) utilizado, por ex., para passar de um estado de compressão inferior para um estado de compressão superior. Através da receção de ACKs por meio de um canal bem definido, o compressor tem o conhecimento de que foi recebido algum cabeçalho específico. O compressor tira partido desse conhecimento para comprimir de forma mais fidedigna e mais eficaz.

- Modo Oportunista Bidirecional: este modo de funcionamento é utilizado no caso de existir um canal inverso, mas é definido 'vagamente', ou seja, o canal

pode não estar sempre disponível ou pode ser lento/não fidedigno. Existem muitas aplicações importantes que são bidirecionais. Um exemplo principal é a voz ou o vídeo de conversação. Nesses casos, existe inerentemente um canal inverso, que pode transportar o retorno.

• Modo Unidirecional (Pessimista ou Otimista): este modo de funcionamento é utilizado quando não existe nenhum canal inverso de qualquer tipo. Uma vez que não existe qualquer retorno do descompressor relativamente ao respetivo estado atual, o compressor tem de enviar ocasionalmente algumas informações de atualização ao descompressor, que podem ser utilizadas para restabelecer o sincronismo no caso de algo ter corrido mal. Dependendo de vários fatores (por ex., condições do canal), que podem ser conhecidos do compressor, a abordagem neste modo pode ser pessimista (atualizações mais frequentes) ou otimista (atualizações menos frequentes). Além disso, existem eventos que podem acionar o compressor para enviar informações FH, atualizar o descompressor e reduzir a possibilidade de descompressão incorreta.

O compressor ACE pode ser caracterizado por estar a avançar através de uma série de estados. O compressor abandona um estado de compressão inferior e entra num estado de compressão superior quando tiver confiança suficiente de que o descompressor recebeu algumas informações.

No caso da compressão de cabeçalhos RTP, os estados são Cabeçalho Completo, Primeira Ordem e Segunda Ordem.

• Estado de Cabeçalho Completo (FH): o compressor entra neste estado quando está no momento da

inicialização ou quando ocorre algum evento excecional (avaria da CPU ou perda de memória). Neste estado, o compressor transmite essencialmente informações de cabeçalho FH ao descompressor. Um cabeçalho FH contém o conjunto completo de todos os campos de cabeçalho, mais algumas informações adicionais. Estas informações podem incluir dados específicos de compressor/descompressor, tais como o CID (Context Identifier – Identificador de Contexto, utilizado para discriminar múltiplos fluxos). O compressor permanece neste estado até ter adquirido confiança suficiente de que o descompressor recebeu as informações de cabeçalho FH. Os pacotes FH são os menos ideais para transmitir devido ao seu tamanho grande (por ex., pelo menos 40 bytes para IPv4, 60 bytes para IPv6). O compressor abandona este estado e entra no estado FO quando tiver confiança suficiente de que o descompressor recebeu corretamente um cabeçalho FH. Essa confiança pode advir, por ex., da receção de um ACK a partir do descompressor ou do envio de um número predefinido de FHs.

• Estado de Primeira Ordem (FO): o compressor entra neste estado quando é iniciada uma nova cadeia, depois de ter abandonado o estado FH. Neste estado, o compressor transmite essencialmente informações de cabeçalho FO. Um cabeçalho FO contém campos específicos de compressor/descompressor, e algumas informações que descrevem alterações irregulares que ocorreram nos campos alteráveis essenciais. O compressor permanece neste estado até ter adquirido confiança suficiente de que as informações de cabeçalho FO foram recebidas pelo descompressor. Essa confiança pode advir, por ex., da receção de Reconhecimentos a partir do descompressor ou do envio de um número predefinido de FOs.

• Estado de Segunda Ordem (SO): o compressor está neste estado quando o cabeçalho a ser comprimido está em conformidade com o padrão de uma cadeia, e o compressor está suficientemente confiante de que o descompressor adquiriu o padrão da cadeia. Neste estado, o compressor transmite cabeçalhos SO. Um cabeçalho SO contém essencialmente apenas um número de sequência. O modo preferido de funcionamento para o compressor/descompressor é a transmissão/receção de pacotes SO, devido ao seu tamanho mínimo (na ordem de apenas alguns bits).

Resumindo, uma sessão é iniciada com o compressor no estado FH. Nessa fase, o compressor envia um cabeçalho completo ao descompressor para estabelecer um contexto no descompressor. Isto inicia uma cadeia. Em seguida, o compressor entra no estado FO ou SO. No estado FO, envia as informações de atualização de campos alteráveis essenciais necessárias ao descompressor. No estado SO, envia as informações mínimas ao descompressor. O descompressor efetua uma extrapolação simples com base nas informações trocadas nos pacotes FH e FO anteriores até a cadeia terminar. Quando outra cadeia é iniciada, o compressor entra novamente no estado FO e o processo repete-se.

Os modos de transmissão bidirecionais utilizam reconhecimentos para várias funções:

• para informar o compressor de que as informações FH foram recebidas; nesse caso, o compressor sabe que o descompressor adquiriu as informações necessárias para descomprimir cabeçalhos FO e, por conseguinte, o compressor pode transitar de forma fidedigna para o estado de compressão superior, FO; este tipo de ACK é referido como FH-ACK;

• para informar o compressor de que as informações FO foram recebidas; nesse caso, o compressor sabe que o descompressor adquiriu as informações necessárias para descomprimir cabeçalhos SO e, por conseguinte, o compressor pode transitar de forma fidedigna para o estado de compressão superior, SO; este tipo de ACK é referido como FO-ACK;

• para informar o compressor de que o cabeçalho com um número específico n foi recebido; nesse caso, o compressor sabe que o descompressor pode determinar o número de sequência sem qualquer ambiguidade causada pela moldagem de contador até ao número de cabeçalho n + seq_ciclo, em que seq_ciclo corresponde ao ciclo de contador; o compressor pode utilizar de forma fidedigna bits k para o número de sequência de cabeçalhos, sem se preocupar com a descodificação de número de sequência ambígua ou incorreta no descompressor; este tipo de ACK é referido como SO-ACK.

O controlo de transição dos estados FH para FO para SO através de Reconhecimentos garante que não existe nenhuma propagação de erros. Ou seja, um cabeçalho comprimido que não seja recebido com erro pode ser sempre descomprimido corretamente, uma vez que a sincronização nunca se perde.

Existe muita flexibilidade relativamente a quando e com que frequência o descompressor envia os reconhecimentos. A ACE também é extremamente resistente aos ACKs que se perdem ou atrasam. O compressor adapta constantemente a respetiva estratégia de compressão com base nas informações atuais a serem comprimidas e nos ACKs recebidos. Por exemplo, a perda ou atraso de um FO-ACK pode fazer com que o compressor fique mais tempo no estado FO. A perda ou atraso de um SO-ACK pode fazer com que o

compressor envie mais bits para o número de sequência, de modo a impedir qualquer descompressão incorreta no descompressor, causada pela moldagem de contador.

Os ACKs podem ser transmitidos periodicamente ou não periodicamente. A frequência dos reconhecimentos não periódicos pode ser diminuída ou aumentada a partir de uma taxa periódica. Um ACK pode ser enviado com menos frequência, uma vez que os ACKs se perdem devido a erros ou congestionamento da rede, ou os ACKs não podem ser transmitidos devido à disponibilidade intermitente do canal inverso ou a algumas condições do descompressor. Um ACK também pode ser transmitido em intervalos mais curtos do que o ACK periódico tradicional. Por exemplo, quando o canal inverso está muito levemente carregado e disponível, o descompressor pode transmitir ACKs com mais frequência e, por consequência, o compressor pode funcionar de forma mais eficaz e fidedigna.

Consequentemente, o canal de retorno utilizado para transmitir os ACKs pode ter requisitos muito vagos. Isto deve-se ao facto de os ACKs só afetarem a eficiência de compressão, e não a exatidão. O atraso ou a perda de ACKs pode fazer com que o tamanho dos cabeçalhos comprimidos aumente, mas mesmo nesses casos o aumento é logarítmico.

No modo determinista bidirecional, a transição de FH/FO para FO/SO é baseada no reconhecimento. No modo unidirecional, nunca é enviado um ACK, pelo que o número de pacotes FH/FO que são enviados antes da transição para o estado FO/SO depende de um valor predefinido ou selecionado de forma dinâmica/adaptável. No modo oportunista bidirecional, o ACK pode atrasar-se, pelo que a transição de FH/FO para FO/SO não se baseia estritamente em ACK, mas depende do que acontecer primeiro entre 1) a transmissão de

um número de FH/FO predefinido ou selecionado de forma dinâmica/adaptável, e 2) a receção de, pelo menos, um ACK.

Resumindo, o número de pacotes FO/FH a enviar antes de mudar para o estado FO/SO depende do facto de ser necessário um ACK antes da mudança e/ou de um parâmetro *m* ajustável que pode ser predefinido ou selecionado de forma dinâmica/adaptável. Abaixo, são descritos quatro casos.

A Fig. 15 ilustra a mudança do estado de compressão do compressor com base apenas na chegada de um ACK apropriado. Uma sequência de, pelo menos, um cabeçalho de um estado específico que, conforme ilustrado sem limitação, corresponde a cabeçalhos FH ou FO é transmitida ao descompressor. O descompressor, ao receber o primeiro cabeçalho FH(n) ou FO(n) (pode ser um cabeçalho que não o primeiro cabeçalho), transmite novamente um Ack(n) que, na receção, faz com que o compressor transite para um estado de compressão superior que, conforme ilustrado, corresponde a FO(N+i+1) ou SO(n+i+1) que é o pacote transmitido imediatamente após a receção de ACK(n).

As Figs. 16 e 17 ilustram a mudança do estado de compressão do compressor com base num parâmetro *m* que corresponde a um número de cabeçalhos transmitidos ou ACK (mudar o acesso sempre que forem transmitidos *m* cabeçalhos FO/FH ou for recebido ACK). A forma de realização da Fig. 16 não é limitada à transmissão de cabeçalhos FH/FO/SO através do compressor. Na Fig. 16, um ACK(n) chega antes de o número de cabeçalhos FO/FH transmitidos atingir *m*, o que, como resultado do número predefinido de cabeçalhos FH/FO a serem transmitidos, faz com que o compressor mude para um estado de compressão superior e transmita o pacote FO(n+i) ou SO(n+i). A forma de realização da Fig. 17 não é limitada à transmissão de cabeçalhos FH/FO/SO através do compressor.

Na Fig. 17, um ACK chega depois de *m* cabeçalhos serem transmitidos.

A Fig. 18 ilustra que a mudança do estado de compressão do compressor ocorre depois de um número definido de cabeçalhos ser enviado sem quaisquer reconhecimentos.

Em modos diferentes, a estratégia de funcionamento do compressor e descompressor é diferente.

Modos de Funcionamento do Compressor

- Baseado estritamente em ACK: neste modo, o compressor depende estritamente da receção dos ACKs. Se um ACK não tiver chegado a tempo devido a perda, disponibilidade do canal, condições do descompressor, etc., o compressor muda para um modo menos comprimido, por exemplo, aumentando o comprimento dos campos de codificação, e só pode regressar a um modo mais comprimido depois de receber ACKs apropriados.

- Baseado vagamente em ACK: neste modo, o ACK ajuda a melhorar a eficiência e fiabilidade quando é recebido, mas o compressor não depende estritamente da receção do ACK. Se o compressor receber um ACK apropriado, muda de um estado menos comprimido para um estado mais comprimido, ou mantém-se no estado atual se já se encontrar no estado mais comprimido. Se não tiver recebido um ACK apropriado, muda de um estado menos comprimido para um estado mais comprimido com base noutros critérios, por ex., o envio de um determinado número de cabeçalhos menos comprimidos, em vez da receção de ACK.

- Não baseado em ACK: o compressor funciona normalmente neste modo quando não está disponível nenhum canal inverso. Neste modo, o critério de transição de um estado menos comprimido para um estado mais comprimido não é baseado em ACK. De preferência, o compressor pode mudar de um estado menos comprimido para um estado mais comprimido depois de ter sido transmitido um determinado número de cabeçalhos menos comprimidos. Esse número pode ser um parâmetro ajustável. Além disso, existem eventos que podem acionar o compressor para enviar informações menos comprimidas, de modo a atualizar o descompressor e reduzir a possibilidade de descompressão incorreta.

Modos de Funcionamento do Descompressor

- Envio de ACK determinista: neste modo, o descompressor envia ACKS periodicamente e quando recebe pacotes FH/FO. Além disso, o descompressor pode enviar ACKs de volta para o compressor mais frequentemente do que periodicamente quando o canal inverso está levemente carregado e disponível.

- Envio de ACK oportunista: neste modo, o descompressor não envia estritamente ACKs periodicamente. Só envia um ACK quando tem uma oportunidade para o enviar, por ex., quando o canal inverso está disponível para transportar o ACK.

- Nenhum ACK: neste modo, o compressor não envia quaisquer ACKs.

Uma aplicação de exemplo em que o esquema de compressão e descompressão de cabeçalhos é útil é aquela em que os pacotes VoIP (Voice over IP) (ou telefonia IP) são transmitidos através de sistemas celulares. Quando VoIP é

aplicado nos sistemas celulares, é importante minimizar o *overhead* do cabeçalho IP/LUDP/RTP devido à largura de banda limitada da interface sem fios ou aérea (RF). Num sistema desses, por exemplo, ANI_AD funciona como interface entre a rede IP e um terminal de computador que executa RTP/UDP/IP (por ex., terminal 130) com uma interface celular ou RF para receber pacotes RTP através da ligação sem fios ou RF. Esta é meramente uma aplicação de exemplo da técnica de compressão/descompressão da presente invenção.

Definições

n: número de sequência atribuído pelo compressor e transportado nos cabeçalhos. O número n é sempre incrementado em 1 para cada novo pacote e independentemente do número de sequência RTP. Note que n pode ser codificado em bits k (= (CD_SN)k) ou bits ℓ_de_extensão (= (CD_SN)ℓ_de_extensão).

CD_SN 139: contador interno correspondente a n. O compressor e o descompressor mantêm os respetivos contadores. É possível escolher um tamanho suficientemente grande para contadores internos, de modo a evitar qualquer ambiguidade durante a sessão, por ex., 32 bits.

(CD_SN)k: bits k menos significativos de CD_SN. (CD_SN)k é normalmente enviado no cabeçalho comprimido.

(CD_SN)ℓ_de_extensão: bits ℓ_de_extensão menos significativos de CD_SN. (CD_SN)ℓ_de_extensão é enviado no cabeçalho comprimido no modo de extensão.

S_RFH: CD_SN do pacote cujo cabeçalho é conhecido por ser corretamente reconstruído pelo descompressor;

S_RFH é continuamente atualizado pelo compressor com base no retorno do descompressor. S_RFH é enviado na forma de bits k ou ℓ_de_extensão menos significativos.

N_decorrido: um contador mantido pelo compressor para controlar o número de pacotes enviados desde o último pacote reconhecido. N_decorrido = CD_SN atual - S_RFH, se S_RFH for sempre definido como igual ao último pacote reconhecido pelo compressor quando recebe um ACK. R_RFH: CD_SN do pacote de referência no descompressor, que é definido como igual a S_RFH quando um pacote FO(n, S_RFH) é recebido.

SN(n): o número de sequência RTP do último pacote enviado pelo compressor. Se o compressor não efetuar uma reordenação, SN(n) não é necessariamente uma sequência crescente de forma monótona, em que n é definido por valores de bit dos bits k ou ℓ de extensão.

TS(n): o carimbo de hora RTP do último pacote enviado pela compressão.

CFO(n): diferença de primeira ordem atual no pacote n. Note que se trata de um vetor, com cada um dos respetivos componentes individuais igual à diferença entre o campo correspondente no pacote n e no pacote (n-1); por exemplo, o componente de carimbo de hora de CFO(n) é calculado como TS(n) - TS(n-1).

FO_DIF(n_2, n_1): diferença de primeira ordem no pacote n_2, relativamente ao pacote n_1. Cada um dos respetivos campos individuais é igual à diferença entre o campo correspondente no pacote n_2 e no pacote n_1; por exemplo, o campo de carimbo de hora de FO_DIF(n_2, n_1) é calculado como TS(n_2) - TS(n_1).

N_Última_Interr: o CD_SN correspondente à interrupção mais recente (ou seja, alteração não linear). É atualizado (pelo compressor) para n sempre que CFO(n) != CFO(n-1). S_DFOD: diferença de primeira ordem predefinida no remetente. S_DFOD é um vetor que especifica o padrão atual. S_DOD é utilizado pelo compressor para determinar se o cabeçalho atual está em conformidade com o padrão. O cabeçalho atual n está em conformidade com o padrão se cabeçalho(n) = cabeçalho(n-1) + S_DFOD. Quando o padrão não muda de uma cadeia para a outra, S_DFOD é estático. Caso contrário, o compressor tem de determinar S_DFOD numa base dinâmica.

TS_DFOD e SN_DFOD: os componentes no S_DFOD correspondentes ao carimbo de hora e número de sequência RTP, respetivamente.

R_DFOD: diferença de primeira ordem predefinida no compressor. R_DFOD é um vetor que especifica o padrão atual. R_DOD é utilizado pelo descompressor para descomprimir cabeçalhos SO. Quando o padrão não muda de uma cadeia para a outra, R_DFOD é estático. Caso contrário, o descompressor tem de determinar R_DFOD numa base dinâmica. Intencionalmente, S_DFOD e R_DFOD são *sempre* iguais durante o estado SO.

Sinalizador_de_extensão: um sinalizador mantido pelo compressor. Se for VERDADEIRO, serão utilizados bits *l*_de_extensão para o CD_SN e serão enviados outros parâmetros de sequência nos cabeçalhos. Caso contrário, será enviado (CD_SN)k. Note que este próprio sinalizador é igualmente transportado nos cabeçalhos, de modo a que o descompressor saiba que codificação de CD_SN é utilizada.

Cabeçalho(n): um termo utilizado genericamente para referir as informações de cabeçalho enviadas pelo compressor. O cabeçalho(n) pode ser enviado de várias formas, FH(n), FO(n, S_RFH), SO(n), dependendo do estado do compressor. Note que, no cabeçalho, n está efetivamente codificado como (CD_SN)k ou (CD_SN)ℓ_de_extensão, dependendo do sinalizador de extensão.

Dif(n_2, n_1): a *verdadeira* distância entre n_2 e n_1, que estão codificados como (CD_SN)k ou (CD_SN)ℓ_de_extensão. Difl (n_2, n_1) = CD_SN(n_2) - CD_SN(n_1), em que CD_SN(n_2) e CD_SN(n_1) são o CD_SN correspondente a n_2 e n, respetivamente. Por exemplo, se o primeiro pacote transportar (CD_SN)k = 14 e o segundo transportar (CD_SN)k = 1, a verdadeira distância é (16 + 1 - 14) = 3, e não (1 -14) = -13.

FH_Req: enviado pelo descompressor para solicitar ao compressor para funcionar no estado FH. Isto é utilizado, por exemplo, quando o descompressor acabou de recuperar de uma avaria e já não tem informações fidedignas sobre o estado.

ACK(n): enviado em resposta ao cabeçalho(n). Um ACK significa que o pacote (n) é recebido corretamente. Seq_ciclo: seq_ciclo -1 é o número máximo atingido pelo número de sequência antes da moldagem e do regresso a 0. Seq_ciclo = 2^k.

SEQ Ext_ciclo: = $2^{\ell\_de\_extensão}$.

HSW 117: o compressor mantém uma janela deslizante de cabeçalhos enviados ao descompressor (HSW): cabeçalho (n), cabeçalho (n+1), cabeçalho (n+2), etc. Os cabeçalhos podiam ter sido enviados na forma de FH, FO

ou SO. Quando o compressor recebe um ACK(n), irá eliminar qualquer Cabeçalho(p), em que p $<n_2-n_1$, e mover igualmente Cabeçalho(p=n) para Cabeçalho(S_RFH). Os campos estáticos não necessitam de ser armazenados como entradas múltiplas em HSW. Apenas é necessária uma única cópia dos campos estáticos. Note que, em HSW, cada cabeçalho é marcado com uma cor, ou verde ou vermelho.

Cor de um cabeçalho (pacote): é verde se o CD_SN transportado no cabeçalho (pacote) estiver codificado em bits k. Caso contrário, é vermelho, por ex. bits ℓ de extensão. Na implementação, pode ser utilizado um sinalizador de 1 bit para armazenar a cor. A cor é utilizada para ajudar o compressor a identificar exclusivamente um cabeçalho quando receber um ACK.

OAW 135: o descompressor mantém uma janela deslizante de cabeçalhos que descomprimiu e reconheceu com êxito: cabeçalho (n1), cabeçalho (n2), cabeçalho (n3). Note que os ACKs não são certamente conhecidos por serem recebidos pelo compressor. A janela será referida como a janela Ack excelente (OAW - Outstanding Ack Window). Os campos estáticos não necessitam de ser armazenados como entradas múltiplas em OAW. Apenas é necessária uma única cópia dos campos estáticos.

R_Último_Descomp: o CD_SN do último pacote reconstruído pelo descompressor. O descompressor mantém o cabeçalho completo correspondente que é referido como FH(Último_Descomp). Nota:

Os cabeçalhos IP, UDP e RTP originais são transferidos, exceto para as alterações descritas abaixo.

• O campo Comprimento Total do cabeçalho IP (2 bytes) é substituído por sinalizador_de_extensão (1 bit), núm_seq (4 ou 8 bits), e outras informações opcionais. O sinalizador de extensão é igual a 1, núm_seq tem um comprimento de 8 bits, ou seja, o pacote é enviado no modo de extensão.

• O campo Comprimento no cabeçalho UDP (2 bytes) é igualmente substituído pelo ID de contexto para compressão de cabeçalhos, referido como CID. O compressor pode comprimir em simultâneo múltiplos fluxos RTP independentemente uns dos outros. A cada fluxo é atribuído um CID exclusivo. Na implementação, o CID pode ter um comprimento de 1 byte ou 2 bytes, dependendo do número máximo de fluxos RTP num determinado momento.

Uma primeira forma de realização da invenção, que pode ser praticada com o sistema da Fig. 2, baseia-se nos campos IP/UDP/RTP que, na maior parte das vezes, são constantes ou podem ser extrapolados de uma forma linear. Nesses casos, o cabeçalho comprimido apenas transporta um número de sequência de não-extensão múltiplo com bits k que fornece informações suficientes para extrapolação linear. Tal como RFC 2508, com a invenção em que a extrapolação linear resulta na reconstrução de cabeçalho incorreta, o transmissor envia informações de diferença FO. Ao contrário de RFC 2508, as informações de diferença são calculadas relativamente a um pacote de referência conhecido por ser recebido corretamente, em vez do que antecede imediatamente o pacote atual. Esta funcionalidade garante que o cabeçalho atual pode ser reconstruído de forma fidedigna, mesmo em caso de perda de um ou mais pacotes antigos. Uma vez que o cabeçalho pode ser reconstruído de forma fidedigna dessa maneira, não é necessário enviar um cabeçalho completo. A referência é determinada e atualizada pelo compressor do

recetor de acordo com os reconhecimentos recebidos a partir do descompressor.

O compressor utiliza como cabeçalho de referência um cabeçalho que é conhecido por ser descomprimido corretamente com base nos reconhecimentos (ACKs) recebidos, conforme ilustrado geralmente na Fig. 6. O compressor envia uma série de pacotes de cabeçalho completo FH(n)...FH (n+i+1)...que abrange uma sequência de tempo mesmo antes de t2, momento esse em que é recebido o reconhecimento ACK(n) produzido pelo compressor que recebe o pacote de cabeçalho completo FH(n). A transição de FH para FO (FO(n+j) conforme ilustrado) é síncrona. A sequência de tempo para t2 depende do tempo de transmissão de rádio de ida e volta.

A Fig. 7 ilustra a transição do compressor relativamente à transmissão de pacotes de primeira ordem FO (FO(n) para FO (n+i+1)... com, pelo menos, um reconhecimento ACK(n) e, opcionalmente, ACK(n+1) sendo gerado pelo descompressor antes de o compressor transitar sincronamente para os pacotes de segunda ordem SO (SOn+j) com um atraso de rádio de ida e volta para o momento t2 necessário para a sincronização. O número de ACKs necessário pelo compressor antes da transição do estado FO para o estado SO depende das variações entre pacotes. Por exemplo, se a variação entre pacotes for linear com parâmetros constantes, apenas é necessário um ACK para a transição do estado FO para o estado SO. Para poder descomprimir o cabeçalho atual, o descompressor só necessita de saber o cabeçalho de referência em vez do cabeçalho imediatamente anterior, tal como com RFC 2508. Neste esquema, o compressor envia cabeçalhos completos (cabeçalhos de tipo FH) na inicialização. Envia informações de diferença de primeira ordem (cabeçalhos de tipo FO) quando a extrapolação linear não se aplica. Contudo, o compressor não transmite cabeçalhos de diferença SO até o

FH ou FO anterior ser reconhecido e a extrapolação linear ser aplicada.

No compressor, o pacote de referência é definido como sendo o último cujo ACK foi recebido. Especificamente, quando o transmissor recebe ACK(n), um reconhecimento para o pacote n, irá obter o pacote n armazenado anteriormente na memória e guardá-lo como o pacote de referência (o pacote n foi armazenado na memória quando foi enviado). Subsequentemente, toda a compressão de cabeçalhos é efetuada relativamente a esse pacote, até ser recebido um novo ACK, ponto esse em que o pacote que acabou de ser reconhecido se torna o novo pacote de referência. A memória do transmissor onde os potenciais pacotes de referência são armazenados é referida como a janela de cabeçalho enviado (HSW - Header Sent Window) representada como entidade 117 na Fig. 2. Se tiverem de ser enviadas informações de diferença FO no cabeçalho comprimido, o compressor irá incluir explicitamente o número de sequência do pacote de referência (número de sequência de referência).

No descompressor, quando um ACK(n) é enviado, o cabeçalho(n) é armazenado na janela de reconhecimento excelente (OAW) representada como 135 na Fig. 2. Subsequentemente, quando um cabeçalho de diferença FO é recebido, o descompressor irá determinar o pacote de referência a partir do número de sequência de referência, obter o pacote de referência correspondente a partir da OAW 135 e utilizá-lo para reconstruir o cabeçalho completo.

A invenção utiliza a linearidade dos carimbos de hora RTP gerados no compressor que seguem rigorosamente um padrão linear como uma função de relógio. Com base no temporizador 134 mantido no descompressor do recetor e através da utilização de um número de sequência de extensão para pacotes FO definido por bits ℓ de extensão, é possível

detetar e recuperar toda uma grande perda dentro de um limiar.

Com a voz sendo assumida, se o intervalo de tempo entre amostras de voz e pacotes consecutivos for *t* ms, o carimbo de hora RTP do cabeçalho n (gerado no momento n* t ms) é igual ao carimbo de hora RTP do cabeçalho 0 (gerado no momento 0) mais TS_passo * n, em que TS_passo corresponde a uma constante dependente de um *codec* de voz da fonte de voz do transmissor. Consequentemente, o carimbo de hora RTP nos cabeçalhos recebidos pelo descompressor segue um padrão linear como uma função de tempo, mas menos rigorosamente do que o compressor, devido à instabilidade de sinal de atraso entre o compressor e o descompressor. No funcionamento normal (ausência de avarias ou falhas), a instabilidade de sinal de atraso é limitada, para cumprir os requisitos de tráfego em tempo real de conversação.

Uma cadeia ocorre como uma sequência de cabeçalhos, todos em conformidade com um determinado padrão. Especificamente, o número de sequência (SN - Sequence Number) RTP é incrementado em 1 de um cabeçalho para o outro. O carimbo de hora (TS - Time Stamp) RTP não é decrescente e segue algum padrão previsível: se os cabeçalhos n_1 e n_2 estiverem na mesma cadeia, o TS do cabeçalho n_2 pode derivar do TS do cabeçalho n_1 e da função de padrão. Os outros valores de campo, exceto talvez a soma de teste UDP e o IP Id, não são alterados na cadeia. Deste modo, depois de um cabeçalho, por ex. n_1, ser corretamente descomprimido, os cabeçalhos subsequentes na mesma cadeia podem ser descomprimidos através de extrapolação de acordo com o padrão. Assim que o compressor determina que um cabeçalho foi descomprimido com êxito e que o padrão foi adquirido pelo descompressor, só tem de enviar um número de sequência de bits k, indicado por (CD_SN)k, como um cabeçalho comprimido, para os pacotes subsequentes na mesma

cadeia. (CD_SN)k corresponde aos bits k menos significativos de um número de sequência de descompressor (compressor) maior (CD_SN).

Neste esquema, o descompressor utiliza o temporizador 134 ou outro temporizador não ilustrado na Fig. 2 para obter o tempo decorrido entre dois pacotes recebidos consecutivamente. Com base nessas informações de temporização e na regra de compressão específica que é utilizada pelo compressor juntamente com o número de sequência, o descompressor pode detetar e/ou recuperar da grande perda.

Suponhamos que

- HDR(i) é o cabeçalho com o número de sequência abreviado i

- HDR(n_1) é o último cabeçalho que foi descomprimido com êxito

- HDR(n_2) é o cabeçalho que foi recebido imediatamente após HDR(n_1)

- TS_passo é o incremento TS RTP a cada *t* ms

- SEQ_CICLO é o ciclo do número de sequência utilizado em HDR

- DIF(n_2, n_1) é a diferença entre pacotes com o número de sequência n_2 e n_1.

É definido conforme apresentado em seguida:

- DIF(n_2,n_1) = n_1 quando n_2>n_1

- $DIF(n_2, n_1) = n_2+2^k-n_1$ quando $n_2 \leq n_1$

Ao receber $HDR(n_2)$ imediatamente após receber $HDR(n_1)$, o descompressor determina se aconteceu ou não uma grande perda entre estes dois pacotes, ou seja, se existem ou não pacotes $DIF(n_2, n_1)$ perdidos ou se existem pacotes $SEQ\_CICLO * p + DIF(n_2, n_1)$ ($p>1$) perdidos. O esquema de deteção também depende do tipo de $HDR(n_2)$. Com base nos três tipos de cabeçalho definidos nos esquemas de compressão de cabeçalhos, são listados abaixo 2 casos.

Caso 1: $HDR(n_1)$, $SO(n_2)$

Caso 2: $HDR(n_1)$, $FO(n_2)$

Recuperação e Deteção de Grande Perda para o Caso 1

Para detetar se existe ou não uma grande perda no caso 1, um temporizador (por ex. temporizador 134) é mantido no compressor, e é verificado e atualizado sempre que um pacote é recebido. Suponhamos que

- T é o momento em que $HDR(n_1)$ é recebido

- $T+\Delta T$ é o momento em que $HDR(n_2)$ é recebido

O compressor só envia pacotes SO se um ou mais pacotes FH ou FO anteriores tiverem sido reconhecidos, bem como se a extrapolação linear se aplicar a partir do cabeçalho reconhecido. Por conseguinte, se $HDR(n_2)$ for SO independentemente do tipo de $HDR(n_1)$, a extrapolação linear aplica-se de $HDR(n_1)$ a $HDR(n_2)$. Isto significa basicamente que o carimbo de hora RTP e o número de sequência RTP para os pacotes n_1 a n_2, quando gerados no compressor, seguem rigorosamente um padrão linear como uma função de relógio. Consequentemente, os cabeçalhos relacionados com o

descompressor também seguem um padrão linear como uma função de tempo, mas com oscilação devido à instabilidade de sinal entre o transmissor e o recetor.

Partindo do princípio de que o limite superior de instabilidade de sinal é sempre inferior a (SEQ_CICLO * t) ms, aplica-se a seguinte regra para detetar uma grande perda:

[Regra 1]

- se (DIF(n_2,n_1) * t ms ≤ ΔT < (DIF(n_2,n_1) + SEQ_CICLO) * t ms), apenas se perdem os pacotes DIF(n_2,n_1);

- se ((DIF(n_2,n_1) + SEQ_CICLO) * t ms ≤ ΔT < (DIF(n_2,n_1) + 2 * SEQ_CICLO) * t ms), perdem-se os pacotes (SEQ_CICLO + DIF (n_2,n_1));

- Em geral, se ((DIF(n_2,n_1) + *i** SEQ_CICLO) * *t* ms ≤ ΔT < (DIF(n_2,n_1) + *(i* +1) * SEQ_CICLO) * *t* ms), perdem-se os pacotes (*i* *SEQ_CICLO + DIF(n_2,n_1));

Para recuperar o carimbo de hora RTP e o número de sequência RTP de uma grande perda, apenas é necessário o número de pacotes perdidos.

Suponhamos que

- N_{perda} é o número de pacotes perdidos entre o pacote n_1 e o pacote n_2 que pode ser obtido com base na regra 1

- TS(i) e SEQ(i) são o carimbo de hora RTP e o número de sequência RTP do pacote i

O carimbo de hora RTP e o número de sequência RTP podem derivar do carimbo de hora RTP e do número de sequência RTP do pacote n_1, bem como N_{perda}, que são ilustrados na seguinte fórmula.

$$TS(n_2) = TS(n_1) + N_{perda} * TS\_PASSO$$

$$SEQ(n_2) = SEQ(n_1) + N_{perda}$$

A Fig. 8 ilustra uma forma de realização da invenção que deteta e recupera de uma grande perda no caso 1 para detetar uma grande perda quando ocorre uma modelação na perda, onde se perdem mais pacotes do que 2^k, em que k corresponde ao número de bits no número de sequência SN(n) dos pacotes. Partindo do princípio de que o descompressor recebe HDR(n), que é enviado no momento t0, e todos os pacotes enviados entre t1 e t3 se perderam, o descompressor recebe SO(n+1), que é enviado no momento t3. Uma vez que o pacote n+1 enviado no momento t3 é um pacote SO que indica que todos os pacotes enviados de t0 a t3 pertencem à mesma cadeia, ou seja, o número de sequência RTP Sn(n) destes pacotes segue um padrão linear como uma função de relógio, o número de pacotes perdidos entre t0 e t3 pode ser detetado pelo temporizador 134 mantido no descompressor. O procedimento é conforme apresentado em seguida:

Partindo do princípio de que o intervalo de tempo entre amostras de voz e pacotes consecutivos é de t ms, o descompressor inicia o respetivo temporizador local 134 (com o valor t_s) quando recebe HDR(n) enviado no momento t0 e, em seguida, o temporizador aumenta com base no relógio. Quando SO(n+1), enviado no momento t3, é recebido no descompressor, o temporizador atingirá um valor aproximadamente igual a $t_s+(2^k+1)t$ devido à instabilidade de sinal. Uma vez que a diferença de tempo entre t_s e t_e (à

volta de $(2^k+1)t$) é superior a t ms, o pacote n enviado no momento t0 e o pacote n+1 enviado no momento t3 não serão pacotes enviados consecutivamente e acontecerá uma grande perda entre estes dois pacotes. Além disso, uma vez que a diferença de tempo entre t_s e t_e é inferior a $(2^{k+1}+1)t$, não haverá mais de um ciclo de sequência de perda de pacotes. Por conseguinte, o descompressor pode concluir que existem 2^k pacotes perdidos.

Recuperação e Deteção de Grande Perda para o Caso 2

O esquema de recuperação e deteção de grande perda não pode ser aplicado no caso 2, em que o cabeçalho FO é recebido após um cabeçalho SO, FO ou FH. Neste caso, mesmo que a diferença de tempo Δt entre os pacotes n_2 e n_1 seja superior a (DIF (n_2,n_1) + SEQ_CICLO) * *t* ms, não significa que ocorreu uma grande perda, uma vez que pode dever-se ao período de silêncio do compressor. Neste caso, com base nas informações fornecidas no cabeçalho FO, o TS RTP pode ser regenerado com êxito, ao contrário do número de sequência RTP, uma vez que não é possível distinguir uma grande perda ou o silêncio apenas com base na temporização. Para resolver este problema, é utilizado um número de sequência de extensão com bits ℓ_de_extensão (bits ℓ_de_extensão>k de não extensão) para os primeiros pacotes FO numa série FO, em que todos os pacotes pertencem à mesma cadeia, até ser enviado novamente ACK para FO a partir do descompressor. Este esquema deteta e recupera de uma grande perda partindo do princípio de que o limite superior de grande perda é inferior a $2^{\ell\_de\_extensão}$ * *t* ms.

Para implementar este esquema, um contador interno CD_SN 139 para pacotes é mantido pelo compressor. O CD_SN 139 conta todos os pacotes enviados a partir do compressor do transmissor. O número de sequência enviado no cabeçalho completo modificado e no cabeçalho comprimido é apenas um

instantâneo dos bits k de não-extensão ou ℓ_de_extensão menos significativos do CD_SN 139. Com base na regra do esquema de compressão/descompressão de cabeçalhos, o descompressor é capaz de reconstruir o número absoluto atual dos pacotes recebidos.

Suponhamos que

CD_SN_ÚLTIMO é o número de pacote absoluto do último pacote recebido, ou seja, pacote n; CD_SN_ATUAL é o número de sequência absoluto para o pacote FO atual, ou seja, pacote n_2; e DIF(CD_SN_ATUAL, CD_SN_ÚLTIMO) é definido como (CD_SN_ATUAL)-(CD_SN_ÚLTIMO). Com base na seguinte regra, é possível detetar com êxito uma grande perda no limite superior de $2^{\ell\_de\_extensão} * t$ ms.

[Regra 2]

se (DIF(CD_SN_ATUAL, CD_SN_ÚLTIMO) > 2^k,

é detetada uma grande perda.

O número de pacotes perdidos N_{perda} pode ser calculado com a seguinte fórmula:

$$N_{perda} = DIF(CD\_SN\_ATUAL, CD\_SN\_ÚLTIMO) = (CD\_SN\_)-(ATUAL-CD\_SN\_ÚLTIMO).$$

Com base em N_{perda}, o número de sequência RTP pode ser regenerado pelo descompressor segundo as fórmulas seguintes, e o carimbo de hora RTP pode ser regenerado com base no cabeçalho de referência e no delta correspondente.

$$SEQ(n_2) = SEQ(n_1) + N_{perda}$$

Uma vez que o número de sequência de extensão utiliza mais largura de banda do que o número de sequência normal, tendo em conta o respetivo número superior de bits, não convém que o utilize frequentemente, mas apenas para os primeiros pacotes FO até o ACK para esta série de FO regressar ao compressor.

Convém referir que este esquema não pode detetar e recuperar de uma grande perda que seja superior a $2^{\ell\_de\_extensão} * t$ ms; por conseguinte, ℓ_de_extensão deve ser selecionado cuidadosamente, de modo a poder detetar a grande perda que não pode ser indicada a partir da camada inferior da pilha de protocolos. Se a grande perda for superior a $2^{\ell\_de\_extensão} * t$ ms, é necessário que uma camada inferior possa fornecer a indicação dessa perda extremamente grande, e a desconexão na camada inferior ou outros métodos de recuperação de uma grande perda, por ex. o envio de um pedido ao compressor, são aplicados.

Quando o compressor necessitar de enviar uma série de pacotes de tipo FO, em que todos os pacotes pertencem à mesma cadeia, utilizará o número de sequência com bits ℓ_de_extensão até, pelo menos, um ACK para esta série de pacotes regressar a partir do descompressor. Contudo, quando for necessário enviar um pacote SO, o compressor apenas utiliza um número de sequência de bits k.

No compressor, o temporizador 134 (ou outro temporizador não ilustrado na Fig. 2) é iniciado sempre que chega um pacote. O temporizador é executado até chegar o pacote seguinte. Se o pacote de entrada for do tipo FH, não são necessárias quaisquer informações de temporização, e o temporizador deve ser reposto.

Se o pacote de entrada for do tipo FO, a regra 2 é aplicada para verificar se aconteceu ou não uma grande perda e qual deve ser o esquema de recuperação correspondente para transmissão de pacotes FO.

Outra forma de realização da invenção, praticada com o sistema da Fig. 2, utiliza o facto de os campos IP/UDP/RTP serem constantes ou poderem ser extrapolados a partir de um padrão previsível. Nesses casos, o cabeçalho comprimido apenas transporta um número de sequência de não-extensão com bits k que fornece informações suficientes para extrapolação. Quando a extrapolação resultar na descompressão incorreta de um ou mais campos, o compressor envia informações adicionais necessárias para esses campos (informações de atualização).

Para fornecer robustez aos erros e às rajadas de erros, o compressor codifica as informações de atualização relativamente a um cabeçalho de referência conhecido por ser corretamente descomprimido. O compressor sabe que um cabeçalho é corretamente descomprimido quando recebe o reconhecimento correspondente. Um mecanismo ACK garante que o cabeçalho atual pode ser reconstruído de forma fidedigna, mesmo em caso de perda de um ou mais cabeçalhos antigos.

Uma cadeia é definida como uma sequência de cabeçalhos, todos em conformidade com um determinado padrão. O número de sequência (SN) RTP é incrementado em 1 de um cabeçalho para o outro. O carimbo de hora (TS - Time Stamp) RTP não é decrescente e segue algum padrão previsível. Se os cabeçalhos n e n_2 estiverem na mesma cadeia, o TS do cabeçalho n_2 pode derivar do produto do desvio de número de sequência p entre n_2 e n_1 e TS e a função de padrão. Os outros valores de campo, exceto talvez uma soma de teste UDP e o IP Id, não são alterados na cadeia. Deste modo, depois de um cabeçalho, n_1, ser

corretamente descomprimido, os cabeçalhos subsequentes na mesma cadeia podem ser descomprimidos através de extrapolação de acordo com o padrão. Assim que o compressor é informado de que um cabeçalho foi descomprimido com êxito e de que o padrão foi adquirido pelo descompressor (conforme demonstrado pelos ACKs), apenas envia um número de sequência, como um cabeçalho comprimido, para os pacotes subsequentes na mesma cadeia.

No caso da voz, o TS tem um padrão linearmente crescente. Deste modo, o TS do cabeçalho (n_2) pode ser calculado como: $TS(n_2) = TS(n_1) + TS\_passo * p$, em que TS_passo corresponde ao incremento de carimbo de hora entre dois cabeçalhos consecutivos e p corresponde ao desvio de número de sequência entre os pacotes n_2 e n_1. Um intervalo silencioso quebra a relação linear e faz com que uma cadeia termine. Uma nova cadeia é iniciada com um novo segmento contínuo de voz.

Deste modo, o compressor passa por três fases diferentes: inicialização, atualização e extrapolação. Uma sessão é iniciada com uma fase de inicialização. Nessa fase, o compressor envia os cabeçalhos completos ao descompressor até ser recebido um ACK. Depois de concluída a inicialização, quando uma cadeia é iniciada, o compressor do transmissor entra numa fase de atualização, na qual envia as informações de atualização necessárias ao descompressor. Assim que o compressor recebe um ACK ou ACKs a indicar que o descompressor adquiriu as informações necessárias para efetuar a extrapolação, o compressor transita para a fase de extrapolação. Na fase de extrapolação, o compressor só envia um número de sequência como cada cabeçalho comprimido, até a cadeia terminar. Quando é iniciada outra cadeia, o compressor do transmissor entra noutra fase de atualização, e todo o processo é repetido. Os cabeçalhos enviados nas fases de

inicialização, atualização e extrapolação são FH, FO e SO. FH, FO e SO transportam um número de sequência incrementado em 1 em cada cabeçalho enviado pelo compressor SO, que consiste essencialmente apenas nesse número de sequência. No seguimento, os ACKs enviados em resposta a FH e FO denominam-se FH ACK e FO ACK respetivamente.

Grande rajada de erros e perda de sincronização - ACKs Periódicos

Se o número de sequência acima estiver codificado com bits k, será moldado em cada cabeçalho seq_ciclo (seq_ciclo = 2^k). Por conseguinte, se uma rajada de erros durar mais do que os cabeçalhos seq_ciclo, o descompressor não pode determinar claramente o número de cabeçalhos decorridos apenas a partir do número de sequência e, consequentemente, não pode efetuar uma descompressão adequada. Para abordar este problema de grande rajada e moldagem, são utilizados reconhecimentos (ACKs) periódicos. A Fig. 9 ilustra o funcionamento da invenção utilizando os números de sequência de bits k e os números de bits ℓ de extensão com a colaboração de reconhecimentos (ACKs) periódicos que são gerados sincronamente pelo descompressor em todos os pacotes recebidos 2^k detetados. Partindo do princípio de que o compressor recebe o ACK(n) para o pacote n antes de enviar o pacote SO n+i com um número de sequência de bits k, o compressor está a espera de outro ACK para o pacote (n+2^k+N_RT) a partir do descompressor antes de enviar o pacote (n+i+2^k+N_RT). Partindo do princípio de que ACK(n+2^k+N_RT) se perde durante a transmissão, o compressor é transferido para o estado de extensão utilizando bits ℓ e envia o pacote (n+i+2^k+N_RT) com um número de sequência de bits ℓ de extensão. Quando o descompressor recebe o pacote (n+i+2^k+N_RT) com um número de sequência de ℓ de extensão, envia um ACK (n+i+2^k+N_RT) de volta para o compressor.

Quando o compressor recebe o ACK(N+1+2^k+N_RT), é transferido novamente para o estado de não extensão, e envia o pacote SO(n+j) com apenas um número de sequência de bits k. É esperado que o descompressor envie um ACK em intervalos regulares, curtos o suficiente para que, normalmente, o compressor receba um ACK pelo menos uma vez em cada 2^k cabeçalhos seq_ciclo. O compressor mantém um contador, N_decorrido, para controlar o número de pacotes decorridos desde o último ACK recebido. Se N_decorrido > 2^k cabeçalhos seq_ciclo, o compressor funciona no modo de extensão, em que são utilizados bits ℓ_de_extensão, em vez de um número menor de bits k de não-extensão, para o número de sequência, com o número de bits ℓ_de_extensão > o número de bits k de não-extensão. O número de sequência de bits ℓ_de_extensão e o número reduzido de bits k no número de sequência de não-extensão podem ser vistos como os bits menos significativos do número de sequência de não-extensão e os bits ℓ_de_extensão são os bits menos significativos do contador CD_SN 139 do transmissor. São indicados por (CD_SN) k e (CD_SN) ℓ_de_extensão respetivamente. N_decorrido será incrementado em 1 para cada pacote enviado pelo compressor. O número de sequência SN só diminui quando o compressor recebe um ACK do descompressor, e permite que o compressor regresse ao modo normal, ou seja, utilizar o número atual dos bits menos significativos de CD_SN. Estes ACKs são referidos como ACKs periódicos. Os ACKs podem ser não periódicos conforme descrito abaixo quando, por exemplo, o canal no qual os ACKs são enviados está ocupado, o que requer o atraso do ACK de uma forma não periódica.

Para explicar o atraso de ida e volta, o descompressor do recetor necessita de antecipar o momento do envio de um ACK periódico. O descompressor tem de enviar um ACK periódico cedo o suficiente para que o compressor receba normalmente ACKs pelo menos uma vez em cada seq_ciclo. Levando em consideração o tempo de ida e volta, o

descompressor tem de enviar ACKs pelo menos uma vez em cada cabeçalho (seq_ciclo - N_RT). A quantidade N_RT é calculada como EST_RTT/T_H, arredondada para cima até ao número inteiro mais elevado seguinte. A quantidade EST_RTT é uma estimativa do atraso de ida e volta atual calculado pelo descompressor e pode ser avaliada de forma dinâmica com base em medições recentes ou definida simplesmente para RTT_n (definido abaixo). Na prática, T_H pode derivar de características de *codec* do transmissor ou do espaçamento efetivo observado.

Suposições

Para garantir a exatidão e obter um desempenho elevado, é necessário fazer algumas suposições sobre o canal de comunicações entre o compressor e o descompressor (CD-CC). O canal pode ser uma ligação ou uma concatenação de ligações (rede ou redes).

Os pacotes transferidos através do canal inverso e de avanço podem ficar danificados ou perder-se, mas as respetivas ordens são mantidas (ou seja, canal FIFO). A quantidade MAX_EB é definida como o número máximo de pacotes consecutivos que podem perder-se através de CD-CC. Na prática, para as ligações celulares, MAX_EB é reforçado pelas camadas inferiores da pilha de protocolos que decidem perder a ligação quando é atingido um limiar de pacotes perdidos consecutivos.

O canal pode efetuar a fragmentação e a nova montagem no descompressor, mas mantém e fornece o comprimento dos pacotes que estão a ser transferidos. Note que esta fragmentação é diferente da fragmentação IP.

Este esquema presume que existe um mecanismo para detetar erros entre o compressor e o descompressor. Parte-se do princípio de que o canal fornece essa deteção de

erros. Se não existir nenhuma deteção de erros disponível a partir do canal, o esquema pode ser alargado de uma forma simples através da adição de um código de deteção de erros ao nível do compressor-descompressor. A correção de erros é benéfica, mas opcional.

O atraso de ida e volta entre o compressor e o descompressor é definido como o tempo para enviar e processar um cabeçalho(n), de o processar e devolver um ACK(n). Para evitar qualquer ambiguidade na mensagem original que está a ser reconhecida, o ACK não pode sofrer nenhum atraso de ida e volta muito longo que o (CD_SN)ℓ_de_extensão na direção de avanço tenha moldado. É razoável assumir que o tempo para enviar e processar o cabeçalho (n) é limitado, devido aos requisitos de tráfego em tempo real. O tempo para devolver ACK(n) depende do canal inverso utilizado para transmiti-lo. Por exemplo, se o canal for baseado na contenção, pode sofrer um atraso devido à fila de espera. Parte-se do princípio de que as camadas inferiores impõem um limite de atraso na transmissão de ACK, se necessário através da rejeição desses ACKs que ficaram na fila durante muito tempo. Com base nestas considerações, parte-se do princípio de que existe um limite superior do atraso de ida e volta: RTT_UB. Além disso, existe um atraso de ida e volta nominal, indicado por RTT_n, que é o atraso de ida e volta mais provável durante o funcionamento normal. Obviamente, RTT_n < RTT_UB. A estimativa de RTT_n deve ser efetuada antes da implementação, uma vez que é utilizado RTT_n para determinar o valor ideal de k (para obter uma explicação, consulte abaixo). Note que, no tempo de execução, o recetor necessita de estimar o atraso de ida e volta efetivo. Os detalhes são apresentados abaixo.

Com base nas suposições 1 e 4, para garantir a exatidão do esquema proposto, o valor de ℓ_de_extensão deve satisfazer as seguintes condições:

1. $2^{\ell\_de\_extensão} * T\_H \geq RTT\_UB$, em que T_H corresponde ao espaçamento temporal entre dois cabeçalhos consecutivos; isto é necessário para evitar a ambiguidade no ACK

2. $2^{\ell\_de\_extensão} > MAX\_EB$; isto é necessário para manter a sincronização de sequências mesmo quando ocorrem grandes rajadas

Existe alguma flexibilidade para escolher o valor de ℓ. Contudo, para obter um desempenho ideal, o mesmo deve ser ajustado com base na distribuição de rajadas de erros de canal (tanto na direção inversa como de avanço) e no atraso de ida e volta. Em seguida, são apresentadas algumas considerações:

3. $2^k * T\_H \geq RTT\_n$, para reduzir a possibilidade de o compressor mudar para o modo de extensão devido a ACKs em atraso.

4. $2^k >$ o número mais provável de perdas de pacotes consecutivos. Isto é necessário para reduzir a possibilidade de o compressor mudar para o modo de extensão devido a grandes rajadas de erros.

5. K não deve ser demasiado pequeno. Caso contrário, serão enviados demasiados ACKs periódicos do descompressor para o compressor, provocando a inundação do canal inverso e a diminuição da eficiência de compressão. Por outro lado, um grande valor de k irá resultar num *overhead* transportado em cada cabeçalho.

O conceito básico é que, quando a condição do canal se deteriorar, o compressor e o descompressor recuam utilizando bits ℓ_de_extensão (CD_SN) para garantir a exatidão. Por outro lado, serão utilizados bits k (CD_SN) mais pequenos durante as condições de canal normais para obter a melhor eficiência. Os detalhes da mudança entre os dois modos são descritos abaixo.

A Fig. 10 ilustra uma forma de realização de redução da largura de banda que envia, pelo menos, uma sequência de pacotes de dados, que transita em cada sequência entre estados diferentes de consumo de cabeçalhos, do compressor para o descompressor antes de um reconhecimento (ACK), gerado pelo descompressor, ser recebido pelo compressor para fazer com que o compressor transite a compressão de cabeçalhos para um grau mais elevado de compressão de cabeçalhos. Uma vez que o compressor, antes de receber qualquer reconhecimento, transmite periodicamente cabeçalhos com, pelo menos, alguma compressão, o número mais pequeno de bits resultante transmitido antes de receber um reconhecimento poupa largura de banda. Na fase de inicialização, o compressor pode, sem limitação, alternar entre cabeçalhos completos (FH) e cabeçalhos de primeira ordem (FO), ou seja, um conjunto de FH, depois um conjunto de FO, depois um conjunto de FH, depois um conjunto de FO, e assim sucessivamente, até ser recebido um ACK. Cada conjunto pode incluir, pelo menos, um cabeçalho. O descompressor transmite um ACK quando recebe corretamente um FH. Na Fig. 10, os cabeçalhos FH e FO alternados são transmitidos um a seguir ao outro, ou seja, FH(0), FO(1), FH(2), FO(3), até o compressor receber o ACK (0) para o pacote 0 e utiliza o FH(0) como cabeçalho de referência a partir dessa altura para a descompressão.

As Figs. 14A a F ilustram um exemplo do formato de pacotes SO, ACK, FO, FH, FO EXT e FH REQ que podem ser

utilizados com a prática da presente invenção. Aplicam-se as seguintes abreviaturas: PT corresponde ao tipo de pacote, C_RTP_SN corresponde ao número de sequência RTP comprimido, C_RTP_TS corresponde ao carimbo de hora RTP comprimido e C_IP_ID corresponde ao IP_ID comprimido. Contudo, convém compreender que a presente invenção não é limitada a isso. O campo PT para o pacote SO pode ser codificado como 0, o pacote ACK como 10, o pacote FO como 110, o pacote FH como 1110, o pacote FO_EXT como 11110 e o pacote FH_REQ como 111110. Nos pacotes FO e FO_EXT, M corresponde a um marcador de bits no cabeçalho RTP. No pacote FO, T corresponde a um sinalizador de um bit 1 que é definido para 1 se C_RTP_TS estiver presente e, caso contrário, para zero, e I corresponde a um sinalizador de um bit que é definido para 1 se C_IP_ID estiver presente e, caso contrário, para zero. Nos pacotes FH, os cabeçalhos IP e UDP podem ser comprimidos se o comprimento do pacote for fornecido por uma camada inferior no descompressor. O pacote FO_EXT só é transmitido se vários campos não essenciais tiverem sofrido alterações; a máscara de bits é utilizada para indicar que campos estão presentes, e C_RTP_TS e C_IP_ID estão sempre presentes, tornando os sinalizadores de bits T e I desnecessários. Finalmente, o pacote FH_REQ só é enviado em circunstâncias excecionais, tal como uma avaria no sistema.

Talvez seja necessário adicionar um campo de identificador de contexto (CID) a cada um dos cabeçalhos acima se forem comprimidos múltiplos fluxos RTP e a camada inferior não fornecer diferenciação entre fluxos. O CID pode apenas ser necessário para uma direção, tal como num sistema celular, quando a estação móvel (MS) tiver só um fluxo RTP em cada direção, e o CID não é necessário para o tráfego de ligações descendentes (incluindo ACKs). O CID de quantidade tem de ser incluído para o tráfego de ligações

ascendentes (incluindo ACKs), uma vez que a descompressão no lado da rede lida sempre com múltiplos fluxos.

Em seguida, é apresentado um exemplo de pseudocódigo que pode ser utilizado para escrever o código para o compressor.

Este exemplo ilustra o caso em que são necessários dois ACKs para a transição da fase de atualização para a fase de extrapolação. Para facilitar, a alternância de pacotes FH e FO, conforme ilustrado na Fig. 8, e a compensação de número de sequência não são ilustradas no pseudocódigo.

Neste exemplo, parte-se do princípio de que S_DFOD e R_DFOD são não estáticos. Por conseguinte, os mesmos são determinados pelo compressor e descompressor numa base dinâmica, conforme apresentado em seguida:

- O S_DFOD de quantidade é calculado como CFO(m) quando o compressor recebe ACK(n) e ACK(n-p) e (n-p) ≥ N_Última_Interr. Note que p não é necessariamente igual a 1.

- O descompressor calcula R_DFOD quando recebe o primeiro pacote SO após um pacote não SO. O R_DFOD de quantidade é calculado através da utilização de uma extrapolação linear baseada nos últimos dois cabeçalhos reconhecidos armazenados em OAW 135.

O comportamento do compressor pode ser modelado como uma máquina de estado, especificado pela tabela abaixo.

Para abordar o problema de moldagem de contador e grande rajada de erros, o compressor espera receber um ACK pelo menos uma vez em cada cabeçalho seq_ciclo, e mantém um

sinalizador de extensão. Se o sinalizador for verdadeiro, o compressor deverá funcionar no modo de extensão, ou seja, enviar (CD-SN)ℓ_de_extensão. Caso contrário, envia (CD-SN)k. O sinalizador de extensão é definido como verdadeiro sempre que N_decorrido > seq_ciclo. Caso contrário, é definido como falso. Note que N_decorrido continua a aumentar, a menos que o transmissor receba um ACK (para obter detalhes, consulte o pseudocódigo). No modo de extensão, se ext_ciclo tiver decorrido sem um reconhecimento, o transmissor transita para o estado FH.

O compressor entra no estado SO quando, pelo menos, dois pacotes com CD_SN≥ N_Última_Interr tiverem sido reconhecidos. Em seguida, define S_DFOD como igual ao CFO mais recente.

Inicialmente, o compressor inicia a sessão no estado FH. A HSW 117 está vazia. O N_decorrido de quantidade é definido como zero. O sinalizador_de_extensão é definido como falso.

É necessário executar procedimentos adicionais no caso de entrega. Para facilitar, os mesmos não são incluídos aqui.

Estado FH

Evento	Ação
receber ACK(n) para FH(n)	• *ver* o compressor a processar o ACK(n) *pseudocódigo* • estado ← ESTADO_FO;

No estado FH, o procedimento para enviar o cabeçalho(n) é

```
{
      calcular CFO(n) e atualizar N_Última_Interr;
      enviar como FH(n);
       armazenar o cabeçalho(n) em HSW, cor B vermelha; /* n
em FH é codificado em
       bits k_de_extensão */
       }
```

Estado FO

Evento	Ação
receber ACK(n) para FO(n,m)	• *ver* o compressor a processar o ACK(n) *pseudocódigo*
receber FH_Req	• estado ← ESTADO_FH;

No estado FO, o procedimento para enviar o cabeçalho(n) é

```
{
      calcular CFO(n) e atualizar N_Última_Interr;
      se N_decorrido >= seq_ciclo
            sinalizador_de_extensão B VERDADEIRO;
      mais
            sinalizador_de_extensão B FALSO;
      se {N_decorrido >= ext_ciclo
      {
            enviar FH(n), armazenar o cabeçalho(n) em SHW, cor B vermelha;
            estado β ESTADO_FH;
            N_decorrido B 0;
      }
      se tiver recebido mais de dois ACKs, E os dois mais recentes CD_SNs
forem reconhecidos >=
      N_Última_Interr
      {
            S_DFOD B CFO(n);
```

```
            enviar SO(n), armazenar o cabeçalho(n) em HSW, cor B cor_atual();
/* ver
            função abaixo */
            estado B ESTADO_SO;
      }
      mais
            enviar FO(n, S_RFH); armazenar o cabeçalho(n) em HSW, cor B
cor_atual();
            } N_decorrido B N_decorrido + 1;
            }
            cor_atual() {
      se sinalizador_de_extensão = VERDADEIRO
            devolver vermelho;
      mais
            devolver verde;
            }
```

Estado SO

Evento	Ação
receber ACK(n)	• *ver* o compressor a processar o ACK(n) *pseudocódigo*
receber FH_Req	• estado ← ESTADO_FH;

No estado SO, o procedimento para enviar o cabeçalho(n) é

```
{
      calcular CFO(n) e atualizar N_Última_Interr;
      se N_decorrido >= seq_ciclo
            sinalizador_de_extensão B VERDADEIRO;
      mais
            sinalizador_de_extensão B FALSO;
      se N_decorrido >= ext_ciclo
       {
            enviar FH(n), armazenar o cabeçalho(n) em SHW, cor = vermelha;
```

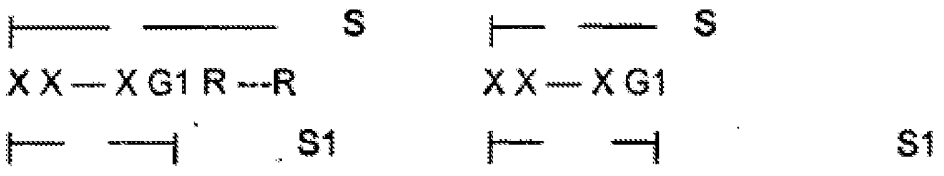
```
            estado B ESTADO_FH;
            N_decorrido B 0;
      }
      se CFO(n) = S_DFOD
            enviar SO(n); armazenar o cabeçalho(n) em HSW, cor B cor_atual();
      mais
      {
            enviar FO(n, S_RFH); armazenar o cabeçalho(n) em HSW, cor B
cor_atual();
            estado B ESTADO_FO;
      }
      N_decorrido BN_decorrido + 1;
      }
      Compressor a processar ACK(n)
      {
      se a cor de ACK(n) for verde /* n é codificado em bits k */
            h_ack β um cabeçalho verde em HSW 117 cujo (CD_SN)k = n; /*
consultar abaixo para obter
            detalhesr */
      mais /* n é codificado em k_de_extensão
      bits */
            h_ack β um cabeçalho vermelho em HSW 117 cujo
(CD_SN)k_de_extensão = n;
      S_RFH β CD_SN de h_ack;
      Eliminar todos os cabeçalhos em HSW anteriores a (mais antigos do que)
h_ack;
      Mover h_ack para Cabeçalho(S_RFH);
      N_decorrido β Dif(atual|CD_SN, S_RFH): |
      }
```

É possível provar que, no procedimento acima, só e apenas um cabeçalho em HSW 117 pode ser corretamente identificado como o cabeçalho que está a ser reconhecido, ou seja, não existirá qualquer ambiguidade do ACK. Se o ACK(n) for vermelho, ou seja, n for codificado utilizando bits ℓ_de_extensão, apenas um cabeçalho vermelho pode

corresponder ao ACK, uma vez que existem, no máximo, $2^{\ell\_de\_extensão}$ cabeçalhos em HSW 117. Caso contrário, se o ACK(n) for verde, será demonstrado que o mesmo ainda pode ser mapeado exclusivamente para um cabeçalho verde em HSW 117.

Partamos do princípio de que é sempre tirado um instantâneo da HSW 117 depois de o compressor enviar um pacote, e o mesmo é representado com uma cadeia de letras Rs (para cabeçalhos vermelhos) e Gs (para cabeçalhos verdes). Suponhamos que S é a cadeia correspondente a um instantâneo arbitrário. Note que S começa com o pacote mais antigo enviado pelo transmissor e termina com o mais novo. Além isso, entre o envio de dois pacotes consecutivos pelo compressor, a cadeia S não sofre alterações, a menos que um ACK chegue durante esse tempo, que corresponderá a algumas letras do início do S.

Agora, suponhamos que G1 indica o G mais à direita (mais novo) no S, e S1 como o prefixo de S até (incluindo) G1. Em seguida, existem apenas dois casos possíveis, conforme ilustrado abaixo.



Caso 1 Caso 2

Suponhamos que len(S1) indica o comprimento de S1. No caso 1, uma vez que existe um R a seguir a G1, len(S1) tem de ser igual a seq_ciclo ($=2^k$). Caso contrário, o compressor não terá enviado o pacote a seguir a G1 como um vermelho. No caso 2, len (S1) s seq_ciclo tem de ser

verdadeiro. Caso contrário, o compressor terá enviado G1 como um vermelho. Por conseguinte, em qualquer dos casos, len(S1) é igual ou inferior a seq_ciclo.

Uma vez que G1 é a letra verde mais à direita em S, é demonstrado que, no máximo, 2^k cabeçalhos verdes podem existir em HSW 117 a qualquer momento. Deste modo, quando um ACK verde é recebido pelo compressor, o CD_SN de bits k no ACK pode ser utilizado para identificar exclusivamente um cabeçalho verde em HSW.

Note que o descompressor tem de cooperar com o compressor para garantir que a sincronização de CD_SN é mantida durante a transição entre os dois modos. Primeiro, se o descompressor receber um pacote vermelho e decidir reconhecer esse pacote, tem de enviar um ACK vermelho que transporta (CD_SN)ℓ_de_extensão.

Segundo, se o descompressor receber um pacote FO, FO(n, m), o cabeçalho de referência correto tem de ser o cabeçalho mais novo (mais recente) em OAW 135, cujos bits k (se m for bit k) ou ℓ_de_extensão (se m for bit k_de_extensão) menos significativos correspondem a m. Note que isto se baseia na suposição de que, em cada direção, o canal se comporta como um FIFO.

A Fig. 19 ilustra a segunda condição. Neste exemplo, NT0 e NT2 são os valores de CD_SN nos momentos T0 e T2, respetivamente. Suponhamos que, em T1, o compressor envia o pacote ACK(NT0), em que NT0 é codificado em bits ℓ_de_extensão. Em T2, o compressor recebe o ACK(NT0). Em seguida, calcula N_decorrido igual a (NT2-NT0) e descobre que N_decorrido<seq_ciclo. Ao mesmo tempo, um pacote RTP chega ao compressor e o compressor decide enviá-lo como um pacote FO, utilizando o cabeçalho(NT0) como referência. Uma vez que N_decorrido<seq_ciclo(=2^k), o NT2 e NT0 no pacote

FO são codificados em bits k. Em T3, o FO chega ao descompressor. Para obter o cabeçalho de referência correto, o descompressor procura simplesmente na respetiva OAW de uma ponta (o mais recente) à outra (o mais antigo), e encontra o primeiro cabeçalho cujos bits k menos significativos do respetivo CD_SN correspondem a (NT0)k.

Note que, em T3, a OAW 135 do descompressor pode conter mais de 2^k cabeçalhos. Contudo, o funcionamento acima fornece sempre o cabeçalho de referência correto. Devido à propriedade FIFO do canal de avanço, tudo o que for recebido (e, deste modo, reconhecido) pelo descompressor entre T1 e T3 tem de ser enviado pelo compressor entre T0 e T2. Por outras palavras, se A indicar o conjunto de cabeçalhos em OAW 135 que foram adicionados a seguir ao cabeçalho (NT0), e B indicar o conjunto de cabeçalhos em HSW 117 em T2, mantém-se sempre A $\subseteq$ B. Uma vez que $|A| < 2^k$, temos $|B| < 2^k$. Por conseguinte, não existem dois cabeçalhos no conjunto B para que os bits k menos significativos dos respetivos CD_SNs correspondam a (NT0)k no pacote FO(NT2, NT0).

Em seguida, é apresentado um exemplo de pseudocódigo para o descompressor:

O descompressor é, em geral, acionado por aquilo que é recebido pelo compressor (ou seja, FH, FO ou SO).

A seguir, "recebido corretamente" significa que não foi detetado nenhum erro no cabeçalho recebido (FH, FO ou SO). Além das informações de estado acima mencionadas, o descompressor mantém igualmente uma cópia do último cabeçalho reconstruído, ou seja, cabeçalho (R_Último_Descomp). Ao receber um pacote FO, o descompressor irá utilizar o procedimento descrito acima

relativamente ao pseudocódigo do compressor para obter o cabeçalho de referência correto.

Se FH(n) for recebido corretamente

```
{
      reconstruir o cabeçalho(n) a partir de FH(n);
      enviar ACK(n);
      R_Último_Reconhecido←n;
   armazenar cabeçalho(n) na OAW 135 e igualmente o
cabeçalho(R_Último_Descomp);
   }
   se FO(n, m) for recebido corretamente
   {
   se não for possível encontrar o cabeçalho(m) na OAW 135 /*, tal só pode
acontecer devido a falhas no sistema
  */
                Enviar FH_Req;
   mais
            obter o cabeçalho (m) a partir da OAW 135 ou o cabeçalho (R_RFH);
            eliminar os cabeçalhos em OAW que são mais antigos do que o
cabeçalho (R_RFH);
            reconstruir o cabeçalho(n)-FO_DIF(n, m) + cabeçalho(m);
            se R_RFH!=m
                  R_RFH-m e armazenar o cabeçalho(m) como o cabeçalho(R_RFH);
            se FO(n, m) for um dos primeiros dois pacotes FO recebidos ou
                  os pacotes N_RT FO foram recebidos desde o último pacote FO
reconhecido
            {
                  Enviar ACK(n);
                  R_Último_Reconhecido←n; armazenar o cabeçalho(n) na OAW:
              }
              armazenar o cabeçalho reconstruído (n) no
cabeçalho(R_Último_Descomp);
}
Se SO(n) for recebido corretamente
```

```
{
      se for o primeiro pacote SO a seguir a um pacote não SO
      {
            encontrar os dois cabeçalhos reconstruídos mais recentemente em
OAW 135;
            se não for encontrado /*, tal só pode acontecer devido a uma
falha
no sistema */
                  Enviar FH_Req;
            mais /*suponhamos que os dois cabeçalhos são o cabeçalho (p) e
cabeçalho (q), p<q*/
                  R_DFOD+FO_DIF(q,p)/Dif(q,p);
      }
      reconstruir o cabeçalho(n)←R_DFOD * Dif(n, R_Último_Descomp) +
      cabeçalho(R_Último_Descomp);
      armazenar o cabeçalho(n) no cabeçalho(R_Último_Descomp);
      se 1) os pacotes (seq_ciclo - N_RT) tiverem decorrido desde
R_Último_Reconhecido, ou,
      /* tempo para enviar um ack periódico */
        2) o sinalizador_de_extensão em SO estiver ON e este for o primeiro
pacote desses, ou,
            /* o compressor mudar para o modo de extensão; enviar ack para
que o compressor regresse ao modo normal */
        3) tiverem sido recebidos mais de N_RT pacotes com
sinalizador_de_extensão ON desde R_Último_ACK
            /* aparentemente o ack anterior não foi recebido; enviar outro
ack */
      {
            Enviar ACK(n); n é codificado no modo de extensão se for cumprida
a condição 2 ou 3
          R_Último_Reconhecido←n:
          armazenar o Cabeçalho (n) na OAW;
      }
}
      armazenar cabeçalho(n) na OAW 135 e igualmente o
cabeçalho(R_Último_Descomp);
```

```
}
se FO(n, m) for recebido corretamente
{
      se não for possível encontrar o cabeçalho(m) na OAW 135  5 /*, tal só
pode acontecer devido a falhas no sistema
      */
                  Enviar FH_Req;
      mais
       {
            obter o cabeçalho (m) a partir da OAW 135 ou o cabeçalho (R_RFH);
            eliminar os cabeçalhos em OAW que são mais antigos do que o
cabeçalho (R_RFH);
            reconstruir o cabeçalho(n)←FO_DIF(n, m) + cabeçalho(m);
            se R_RFH!=m
                  R_RFH←m e armazenar o cabeçalho(m) como o cabeçalho(R_RFH);
            se FO(n, m) for um dos primeiros dois pacotes FO recebidos ou
                  os pacotes N_RT FO tiverem sido recebidos desde o último
pacote FO reconhecido
            {
                  Enviar ACK(n);
                  R_Último_Reconhecido-n; armazenar o cabeçalho(n) na OAW:
            }
            armazenar      o      cabeçalho      reconstruído      (n)      no
cabeçalho(R_Último_Descomp);
}
Se SO(n) for recebido corretamente
{
      se for o primeiro pacote SO a seguir a um pacote não SO
      {
            encontrar os dois cabeçalhos reconstruídos mais recentemente em
OAW 135;
            se não for encontrado /*, tal só pode acontecer devido a uma
falha
no sistema */
                  Enviar FH_Req;
            mais /*suponhamos que os dois cabeçalhos são o cabeçalho (p) e
```

```
cabeçalho (q), p<q*/
                    R_DFOD+FO_DIF(q,p)/Dif(q,p);
          }
          reconstruir o cabeçalho(n)←R_DFOD * Dif(n, R_Último_Descomp) +
          cabeçalho(R_Último_Descomp);
          armazenar o cabeçalho(n) no cabeçalho(R_Último_Descomp);
          se 1) os pacotes (seq_ciclo - N_RT) tiverem decorrido desde
R_Último_Reconhecido, ou,
          /* tempo para enviar um ack periódico */
            2) o sinalizador_de_extensão em SO estiver ON e este for o primeiro
pacote desses, ou,
                /* o compressor mudar para o modo de extensão; enviar ack para
que o compressor regresse
               ao modo normal */
          3)   tiverem   sido   recebidos   mais   de   N_RT   pacotes   com
sinalizador_de_extensão ON desde R_Último_ACK
               /* aparentemente o ack anterior não foi recebido; enviar outro
ack */
        {
             Enviar ACK(n); n é codificado no modo de extensão se for cumprida
a condição 2 ou 3
             R_Último_Reconhecido←n:
          armazenar o Cabeçalho (n) na OAW;
       }
       }
```

HSW 117 e OAW 135

No pior dos casos, em que o atraso de ida e volta é efetivamente igual a RTT_UB, pode ser necessário que a OAW 135 ou HSW 117 mantenha $2^{\ell\_de\_extensão}$ cabeçalhos. Contudo, isso é muito improvável de acontecer. Na maioria dos casos, é necessário manter menos de 2^k entradas na HSW 117 ou OAW 135. Na prática, isto significa um número muito pequeno de entradas para OAW e HSW. Por exemplo, 16 (k=4) entradas

irão fornecer 320 ms de tempo de ida e volta, assumindo um espaçamento de 20 ms por pacote.

Os campos estáticos não necessitam de ser armazenados como entradas múltiplas em HSW 117 ou OAW 135. Apenas é necessária uma única cópia dos campos estáticos.

Em RFC 2508, cada cabeçalho comprimido transporta um número de sequência. Na maioria dos casos, o número de sequência é suficiente para reconstruir o cabeçalho completo através de extrapolação linear. Para esses pacotes em que a extrapolação linear resulta na reconstrução incorreta de cabeçalhos, o compressor envia informações de uma diferença de primeira ordem relativamente ao pacote imediatamente anterior. Deste modo, a perda de um pacote irá invalidar os pacotes subsequentes com cabeçalhos comprimidos, uma vez que o pacote perdido pode estar a transportar informações de diferença FO. O RFC 2508 depende unicamente do número de sequência de 4 bits para detetar perdas de pacotes. O número de sequência é moldado a cada 16 pacotes. Quando ocorre uma rajada de erros superior a 16 pacotes, existe uma probabilidade de 1 em 16 de não serem detetados erros, o que é inaceitavelmente elevado. Além disso, mesmo que o descompressor fosse capaz de detetar erros e recuperar de erros, o descompressor tem de solicitar ao compressor para enviar um cabeçalho de tamanho grande através do envio de uma mensagem ESTADO_CONTEXTO. Deste modo, foi provocado um atraso de ida e volta antes de o cabeçalho solicitado chegar ao recetor. No caso de tráfego de conversação em tempo real, este atraso é convertido numa interrupção da conversação. Além disso, o envio de um cabeçalho de tamanho grande é dispendioso em termos de largura de banda.

Uma forma de realização da presente invenção utiliza um número de sequência de bits k (k pode ser definido igual

a 4) para extrapolação linear. Tal como RFC 2508, quando a extrapolação linear resulta na reconstrução incorreta de cabeçalhos, o compressor envia informações de uma diferença FO. Ao contrário de RFC 2508, a diferença é calculada relativamente a um pacote de referência conhecido por ser recebido corretamente. Esse pacote não é necessariamente o que antecede imediatamente o pacote atual. Esta funcionalidade garante que o cabeçalho atual pode ser reconstruído de forma fidedigna, mesmo em caso de perda de um ou mais pacotes antigos. Uma vez que o cabeçalho pode ser reconstruído de forma fidedigna dessa maneira, não é necessário enviar um cabeçalho completo. As informações de diferença de primeira ordem podem, na maior parte das vezes, ser codificadas com menos bits do que o valor absoluto do cabeçalho completo. O cabeçalho de diferença FO tem um campo adicional que transporta o número de referência, ou seja, o número de sequência do pacote de referência. Para garantir que os erros serão detetados mesmo na presença de grandes rajadas de erros, o descompressor envia um ACK com frequência suficiente, de modo a que o compressor receba um ack, pelo menos, uma vez em cada pacote seq_ciclo. Na ausência desse ACK, o compressor irá presumir que pode existir uma grande rajada de erros. Em seguida, na maioria dos casos, basta que o compressor mude simplesmente para um número de sequência de bits ℓ_de_extensão, em que ℓ_de_extensão é suficientemente grande para evitar qualquer ambiguidade. Aconteça o que acontecer, a perda de um pacote não irá invalidar pacotes subsequentes com cabeçalhos comprimidos. Por conseguinte, quando o descompressor deteta uma perda de pacotes, não tem de solicitar a retransmissão.

A Fig. 20 abaixo ilustra resultados comparativos do RFC 2508 do estado da técnica com a invenção. É assumido um atraso unidirecional (fixo) de 60 ms neste teste. O espaçamento intermédio entre pacotes RTP é de 30 ms. O

modelo de erro aleatório é utilizado com taxas de erro de pacote de média diferente. A relação de compressão é definida como a relação entre o tamanho médio de cabeçalhos comprimidos e o tamanho dos cabeçalhos IP/UDP/RTP originais. Note que, com a invenção, o tamanho dos pacotes ACK é incluído no cálculo do tamanho de cabeçalho comprimido médio. A invenção supera o RFC 1508 assim que a taxa de erro de pacote for superior a 0,4%.

O esquema robusto da invenção necessita que o compressor e o descompressor mantenham as filas HSW 117 e OAW 135, respetivamente. Assumindo que as idas e voltas são inferiores a 320 ms, o tamanho das filas é de 16 entradas + uma cópia dos campos estáticos.

	Ipv4	Ipv6
Tamanho dos campos estáticos (em bytes)	18	40
Tamanho de uma entrada (em bytes)	22	20
Tamanho total de HSW ou OAW para uma sessão bidirecional (em bytes)	(16*22 + 18)* 2 = 740	(16*20 + 40)*2 = 720

Cerca de 1 megabyte de memória irá permitir lidar com mais de 1400 sessões em simultâneo. A carga de processamento para gerir as filas é muito moderada, uma vez que envolve a manipulação de ponteiro.

Embora a invenção tenha sido descrita em termos das respetivas formas de realização preferidas, convém compreender que podem ser efetuadas diversas modificações da invenção sem sair do âmbito da mesma. É pretendido que essas modificações façam parte do âmbito das reivindicações em anexo.

Lisboa, 15 de Janeiro de 2013

REIVINDICAÇÕES

1. Um método de funcionamento de um descompressor (115, 137) num sistema com um transmissor (110, 120) que transmite uma diversidade de pacotes, em que cada um contém um cabeçalho para um recetor (130, 150), o método caracterizado por compreender a descompressão de um cabeçalho comprimido contido num pacote atual recebido pelo recetor através:

da determinação com um contador (134) no recetor de um tempo decorrido Δt entre pacotes recebidos consecutivamente, o tempo decorrido Δt sendo a diferença entre os tempos nos quais o pacote atual e um pacote recebido imediatamente antes foram recebidos;

da determinação sobre se o tempo decorrido Δt é igual ou superior a um lapso de tempo que indica que falta, pelo menos, um pacote entre o pacote atual e o pacote recebido imediatamente antes;

do processamento do tempo decorrido Δt que indica que falta, pelo menos, um pacote para determinar um número de pacotes em falta entre o pacote recebido imediatamente antes e o pacote atual;

da adição do número de pacotes em falta a um número do pacote recebido imediatamente antes para atualizar um número do pacote atual numa sequência de transmissão da diversidade de pacotes; e

da descompressão do cabeçalho comprimido do pacote atual utilizando o número atualizado do pacote atual.

2. Um método de acordo com a reivindicação 1, caracterizado por o cabeçalho do pacote atual ser um cabeçalho comprimido de segunda ordem.

3. Um método de acordo com a reivindicação 1 ou 2, caracterizado por:

o número de pacotes em falta ser calculado como i*SEQ_CICLO+DIF (n_2, n_1) se (DIF(n_2, n_1) + i*SEQ_CICLO) *(unidades de tempo t) ≤ Δt < (DIF(n_2, n_1) + (i+1)*SEQ_CICLO)*(unidades de tempo t), em que i corresponde a um número inteiro igual ou superior a zero, n_2 corresponde a um número de sequência do pacote atual numa sequência de transmissão dos pacotes, n_1 corresponde a um número de sequência do pacote recebido imediatamente antes na sequência de transmissão dos pacotes, SEQ_CICLO é igual a 2^k, em que k corresponde ao número de bits do número de sequência, DIF(n_2, n_1) corresponde à diferença no número de sequência entre os pacote atual e o pacote recebido imediatamente antes e em que DIF(n_2, n_1)=n_2-n_1 quando n_2>n_1 e DIF(n_2, n_1) = n_2+2^k-n_1 quando n_2 ≤n_1, e em que as unidades de tempo t correspondem ao intervalo de tempo entre pacotes consecutivos.

4. Um método de acordo com qualquer uma das anteriores reivindicações, caracterizado por a descompressão do cabeçalho comprimido compreender a utilização da extrapolação linear.

5. Um método de acordo com a reivindicação 4, caracterizado por a extrapolação linear compreender o cálculo de um carimbo de hora TS RTP do pacote atual como TS(n_2) = TS(n_1) + N_{perda}*(TS_PASSO) e o cálculo de um número de sequência SEQ RTP do pacote atual como SEQ(n_2) = SEQ(n_1) + N_{perda}, em que TS_PASSO corresponde a um incremento TS RTP

em cada unidade de tempo t, e N_{perda} corresponde ao número de pacotes em falta.

6. Um método de acordo com qualquer uma das anteriores reivindicações, caracterizado por compreender a reposição do contador (134) após a determinação do tempo decorrido Δt.

7. Um método de acordo com qualquer uma das anteriores reivindicações, caracterizado por k corresponder a quatro.

8. Um recetor (130, 150) que é organizado para receber uma diversidade de pacotes transmitidos, em que cada um contém um cabeçalho; caracterizado por o recetor compreender um contador (134) que é organizado para determinar um tempo decorrido Δt entre pacotes recebidos consecutivamente, o tempo decorrido Δt sendo a diferença de tempo entre um momento de receção no recetor de um pacote atual e um momento de receção no recetor de um pacote recebido imediatamente antes, o recetor sendo organizado para determinar se o tempo decorrido Δt é igual ou superior a um lapso de tempo que indica que falta, pelo menos, um pacote entre o pacote atual, para processar o tempo decorrido Δt, para adicionar o número de pacotes em falta a um número do pacote recebido imediatamente antes para atualizar um número do pacote atual numa sequência de transmissão da diversidade de pacotes, e para descomprimir o cabeçalho comprimido do pacote atual utilizando o número atualizado do pacote atual.

9. Um recetor de acordo com a reivindicação 8, caracterizado por o cabeçalho do pacote atual ser um cabeçalho comprimido de segunda ordem.

10. Um recetor de acordo com a reivindicação 8 ou 9, caracterizado por:

o recetor (130, 150) ser organizado para calcular o número de pacotes em falta como i*SEQ_CICLO+DIF (n_2, n_1) se (DIF(n_2, n_1) + i*SEQ_CICLO) *(unidades de tempo t) ≤ Δt < (DIF(n_2, n_1) + (i+1)*SEQ_CICLO)*(unidades de tempo t), em que i corresponde a um número inteiro igual ou superior a zero, n_2 corresponde a um número de sequência do pacote atual numa sequência de transmissão dos pacotes, n_1 corresponde a um número de sequência do pacote recebido imediatamente antes na sequência de transmissão dos pacotes, SEQ_CICLO é igual a 2^k, em que k corresponde ao número de bits do número de sequência, DIF(n_2, n_1) corresponde à diferença no número de sequência entre os pacote atual e o pacote recebido imediatamente antes e em que DIF(n_2, n_1)=n_2-n_1 quando n_2>n_1 e DIF(n_2, n_1) = n_2+2^k-n_1 quando n_2 =n_1, e em que as unidades de tempo t correspondem ao intervalo de tempo entre pacotes consecutivos.

11. Um recetor de acordo com a reivindicação 10, caracterizado por o recetor (130, 150) ser organizado para calcular um carimbo de hora TS RTP do pacote atual como TS(n_2) = TS(n_1) + N_{perda}*(TS_PASSO) e calcular um número de sequência SEQ RTP do pacote atual como SEQ(n_2) = SEQ(n_1) + N_{perda}, em que TS_PASSO corresponde a um incremento TS RTP em cada unidade de tempo t, e N_{perda} corresponde ao número de pacotes em falta.

12. Um recetor de acordo com qualquer uma das reivindicações 8 a 11, caracterizado por o recetor (130, 150) ser organizado para repor o contador (134) após a determinação do tempo decorrido Δt.

Lisboa, 15 de Janeiro de 2013

FIG. 1
(ESTADO DA TÉCNICA)

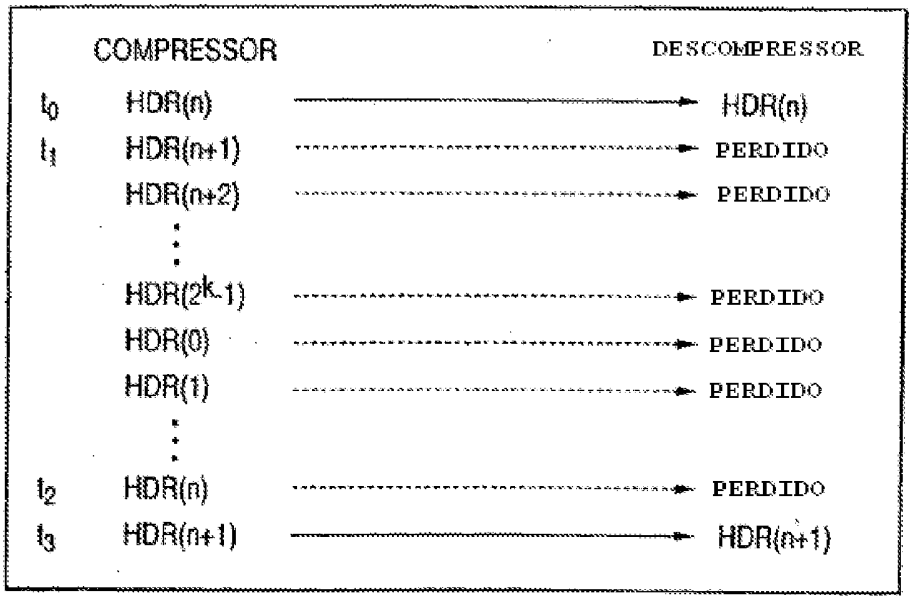


FIG. 19

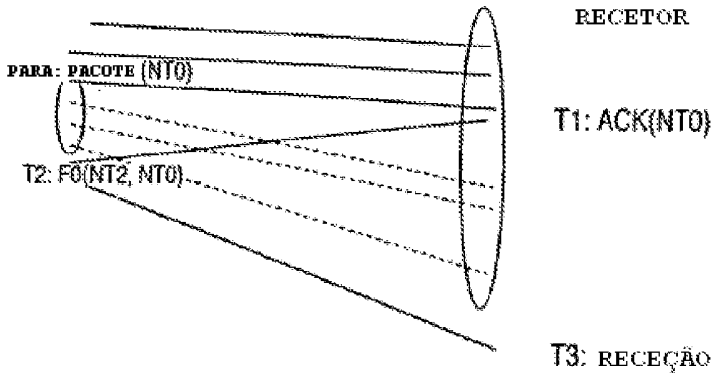


FIG. 2

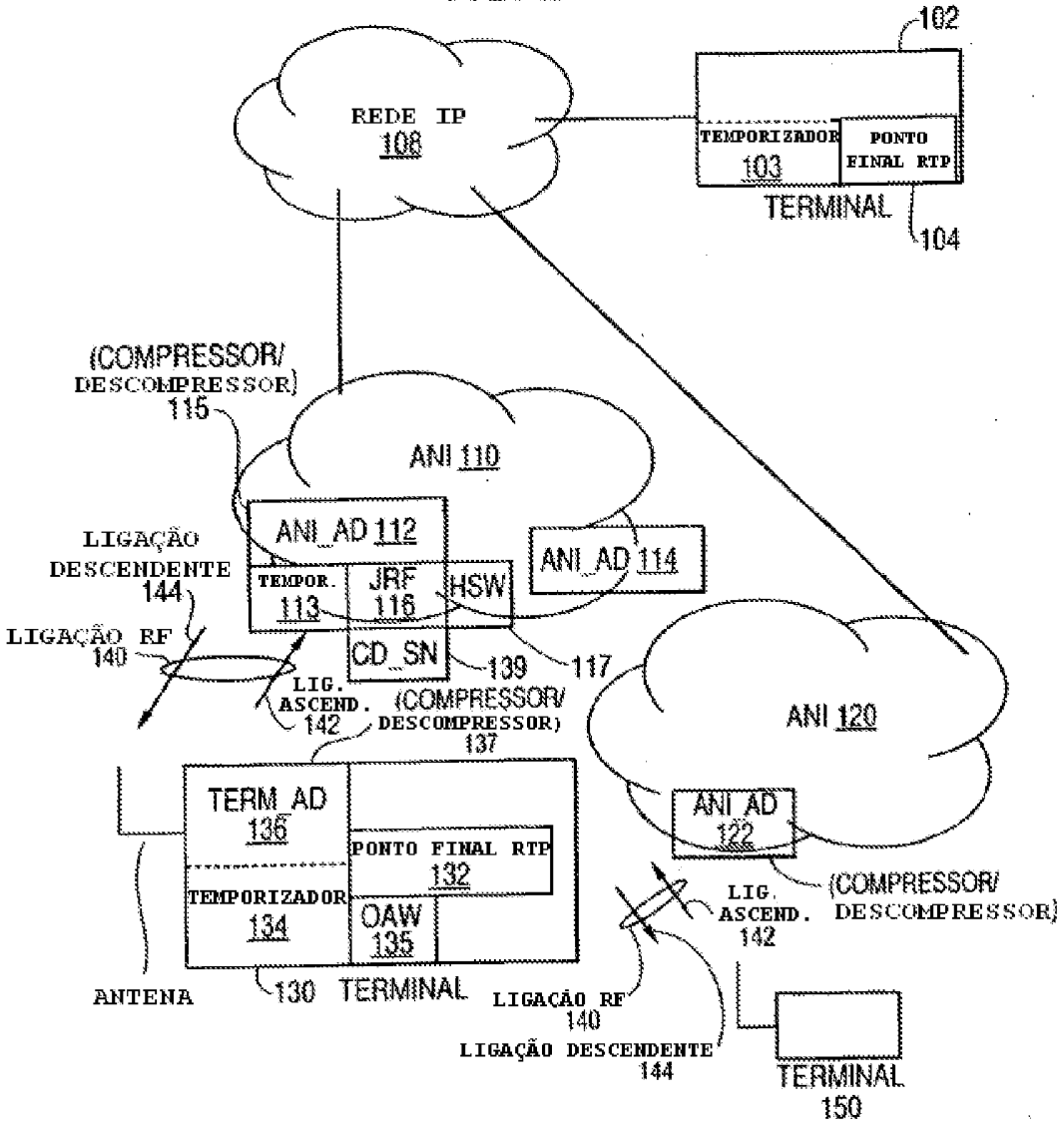

102
REDE IP
108
TEMPORIZADOR
103
PONTO FINAL RTP
TERMINAL
104
(COMPRESSOR/ DESCOMPRESSOR)
115
ANI 110
ANI_AD 112
TEMPOR.
113
JRF
116
HSW
ANI_AD 114
LIGAÇÃO DESCENDENTE
144
LIGAÇÃO RF
140
CD_SN
139
117
LIG. ASCEND.
142
(COMPRESSOR/ DESCOMPRESSOR)
137
ANI 120
TERM_AD
136
PONTO FINAL RTP
132
TEMPORIZADOR
134
OAW
135
ANI_AD
122
LIG. ASCEND.
142
(COMPRESSOR/ DESCOMPRESSOR)
ANTENA
130 TERMINAL
LIGAÇÃO RF
140
LIGAÇÃO DESCENDENTE
144
TERMINAL
150

FIG. 3

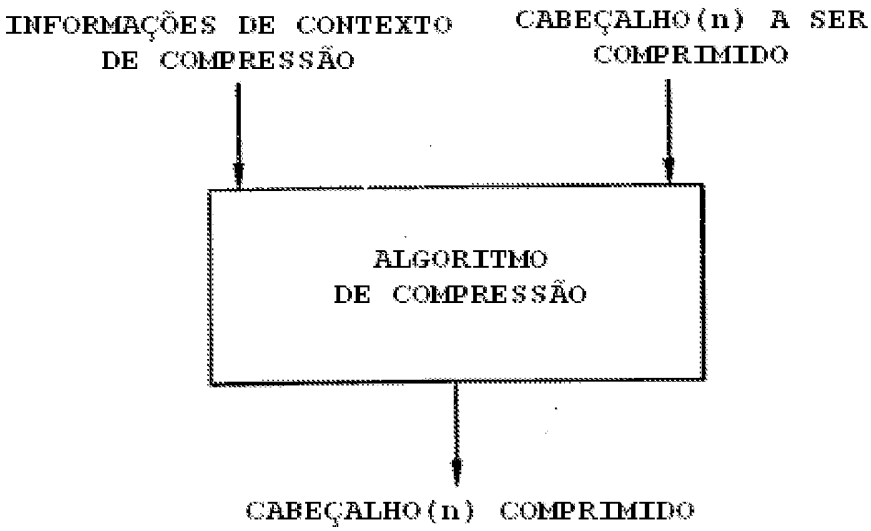

INFORMAÇÕES DE CONTEXTO DE COMPRESSÃO
CABEÇALHO(n) A SER COMPRIMIDO
ALGORITMO DE COMPRESSÃO
CABEÇALHO(n) COMPRIMIDO

FIG. 4

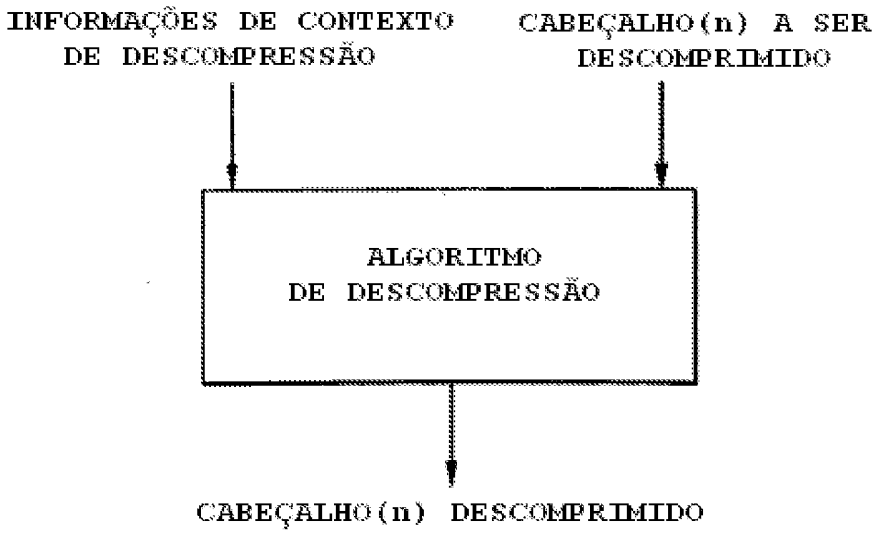

INFORMAÇÕES DE CONTEXTO DE DESCOMPRESSÃO
CABEÇALHO(n) A SER DESCOMPRIMIDO
ALGORITMO DE DESCOMPRESSÃO
CABEÇALHO(n) DESCOMPRIMIDO

FIG. 6

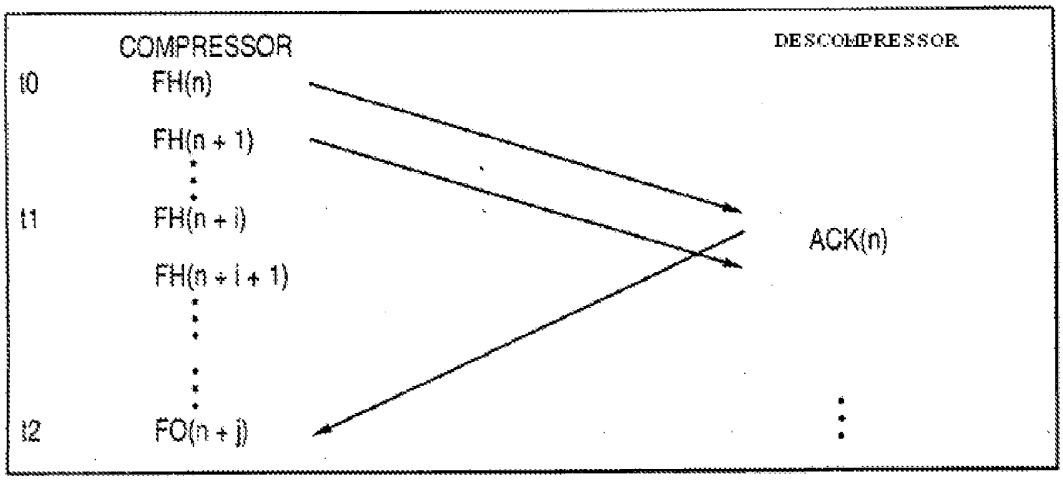

COMPRESSOR
DESCOMPRESSOR
t0
FH(n)
FH(n + 1)
t1
FH(n + i)
FH(n + i + 1)
t2
FO(n + j)
ACK(n)

FIG. 7

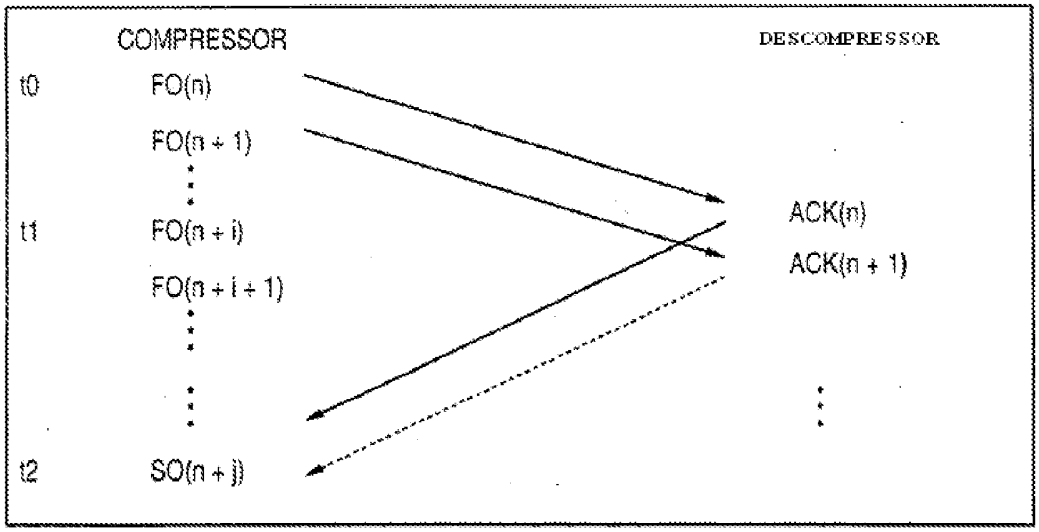

COMPRESSOR
DESCOMPRESSOR
t0
FO(n)
FO(n + 1)
t1
FO(n + i)
FO(n + i + 1)
t2
SO(n + j)
ACK(n)
ACK(n + 1)

FIG. 8

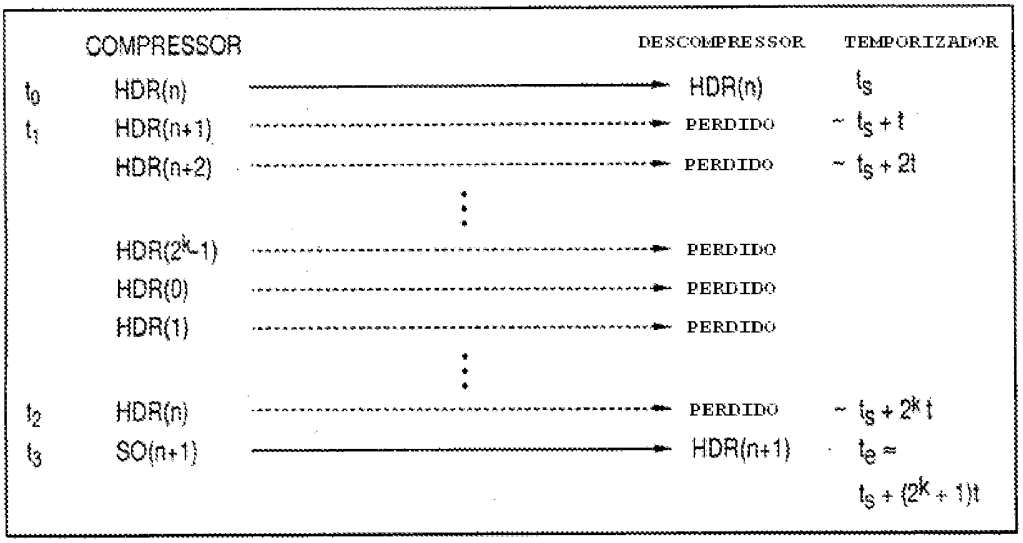

COMPRESSOR
DESCOMPRESSOR
TEMPORIZADOR
t_0 HDR(n) HDR(n) t_s
t_1 HDR(n+1) PERDIDO ~ $t_s + t$
HDR(n+2) PERDIDO ~ $t_s + 2t$
HDR(2^k-1) PERDIDO
HDR(0) PERDIDO
HDR(1) PERDIDO
t_2 HDR(n) PERDIDO ~ $t_s + 2^k t$
t_3 SO(n+1) HDR(n+1) $t_e \approx$ $t_s + (2^k + 1)t$

FIG. 9

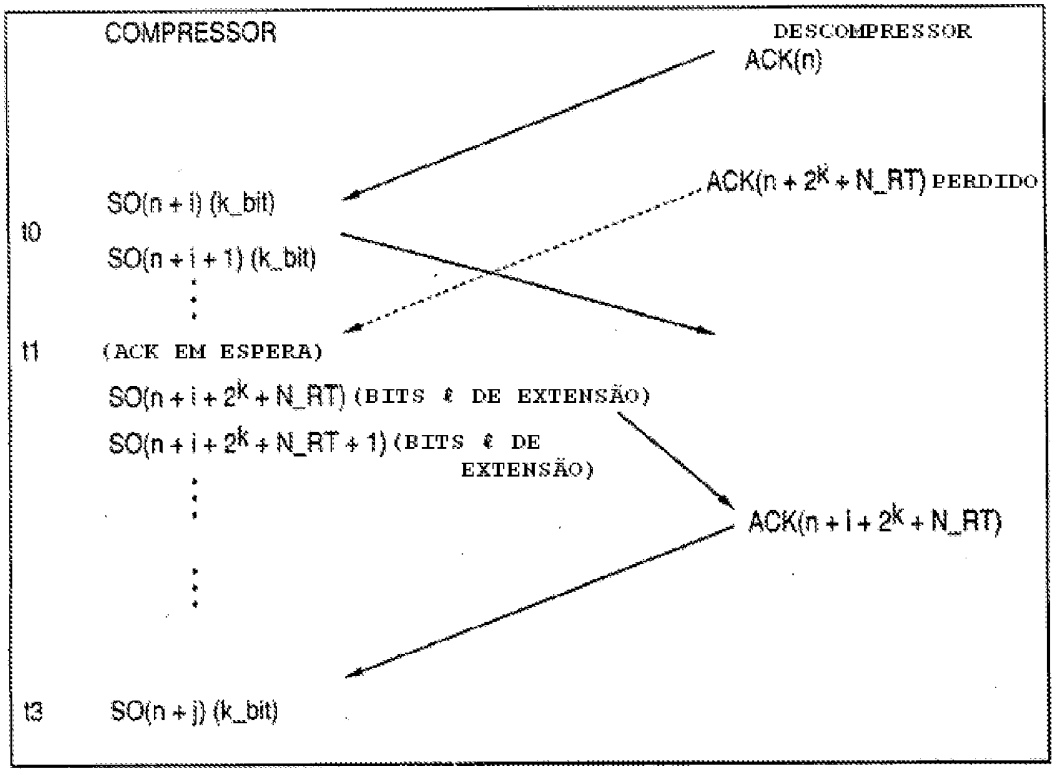

COMPRESSOR
DESCOMPRESSOR
ACK(n)
SO(n + i) (k_bit)
ACK(n + 2^k + N_RT) PERDIDO
t0
SO(n + i + 1) (k_bit)
t1
(ACK EM ESPERA)
SO(n + i + 2^k + N_RT) (BITS ℓ DE EXTENSÃO)
SO(n + i + 2^k + N_RT + 1) (BITS ℓ DE EXTENSÃO)
ACK(n + i + 2^k + N_RT)
t3
SO(n + j) (k_bit)

FIG. 10

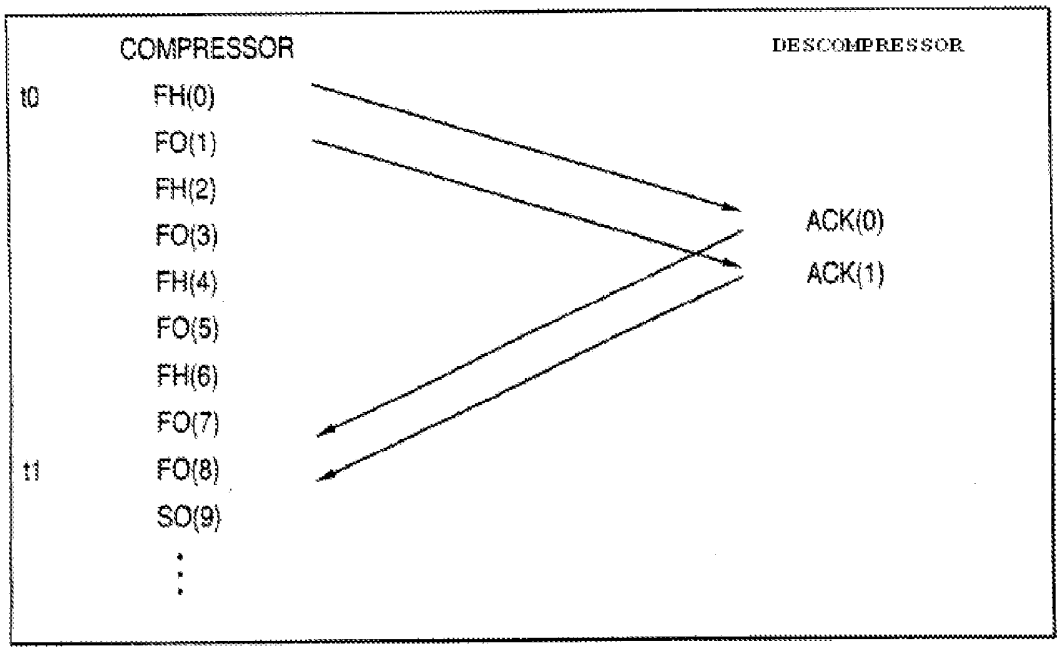

COMPRESSOR
DESCOMPRESSOR
t0
FH(0)
FO(1)
FH(2)
FO(3)
FH(4)
FO(5)
FH(6)
FO(7)
t1
FO(8)
SO(9)
ACK(0)
ACK(1)

FIG. 14A 1. Pacote SO (PT = 0)

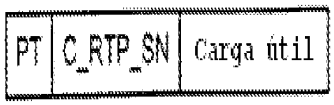


FIG. 14B 2. Pacote ACK (PT = 10)

FIG. 14C 3. Pacote FO (PT = 110)

M – Bit de Marcador no Cabeçalho RTP (1 bit)

T – Sinalizador definido como 1 se C_RTP_TS estiver presente, caso contrário como 0 (1 bit)

I – Sinalizador definido como 1 se C_IP_ID estiver presente, caso contrário como 0 (1 bit)

FIG. 14D 4. Pacote FH (PT = 1110)

* Os campos de comprimento nos cabeçalhos IP e UDP dos pacotes FH podem ser substituídos pelas informações de compressão de cabeçalhos, partindo do princípio de que o comprimento do pacote é fornecido pela camada inferior no lado do descompressor.

FIG. 14E

5. Pacote FO_EXT (PT = 11110)

- O pacote FO_EXT só será transmitido se um ou vários campos não essenciais tiverem sofrido alterações A Máscara de Bits é utilizada para indicar que campos estão presentes neste pacote.
- C_RTP_TS e C_IP_ID estarão sempre presentes num pacote FO_EXT. Por conseguinte, o sinalizador de bits T e I não são necessários.

IG. 14F

6. Pacote FH_REQ (PT = 111110)

O pacote de pedido de cabeçalho completo só será enviado em situações excecionais, por ex. avaria no sistema

FIG. 15

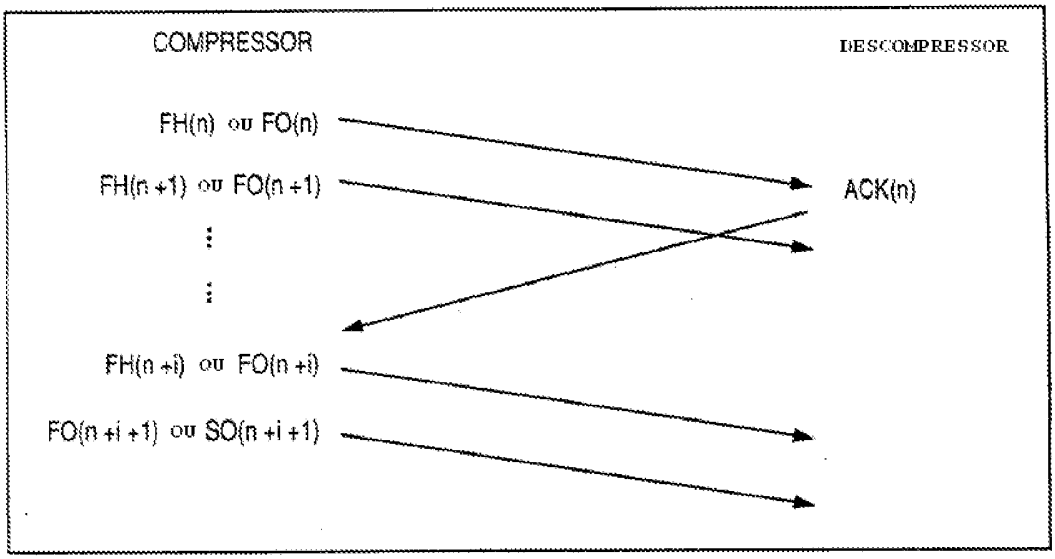

COMPRESSOR
DESCOMPRESSOR
FH(n) ou FO(n)
FH(n +1) ou FO(n +1)
ACK(n)
FH(n +i) ou FO(n +i)
FO(n +i +1) ou SO(n +i +1)

FIG. 16

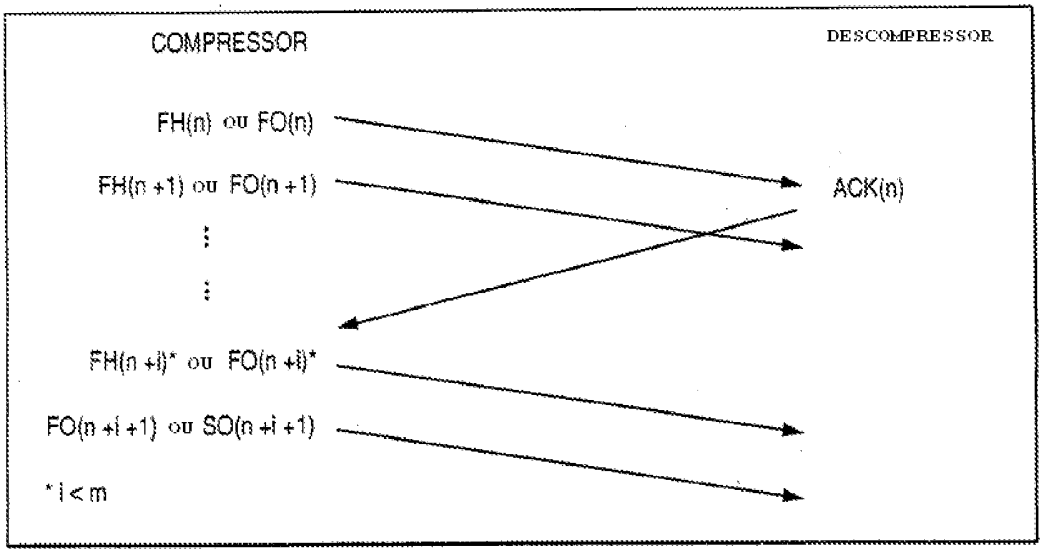

COMPRESSOR
DESCOMPRESSOR
FH(n) ou FO(n)
FH(n +1) ou FO(n +1)
ACK(n)
FH(n +i)* ou FO(n +i)*
FO(n +i +1) ou SO(n +i +1)
* i < m

FIG. 17

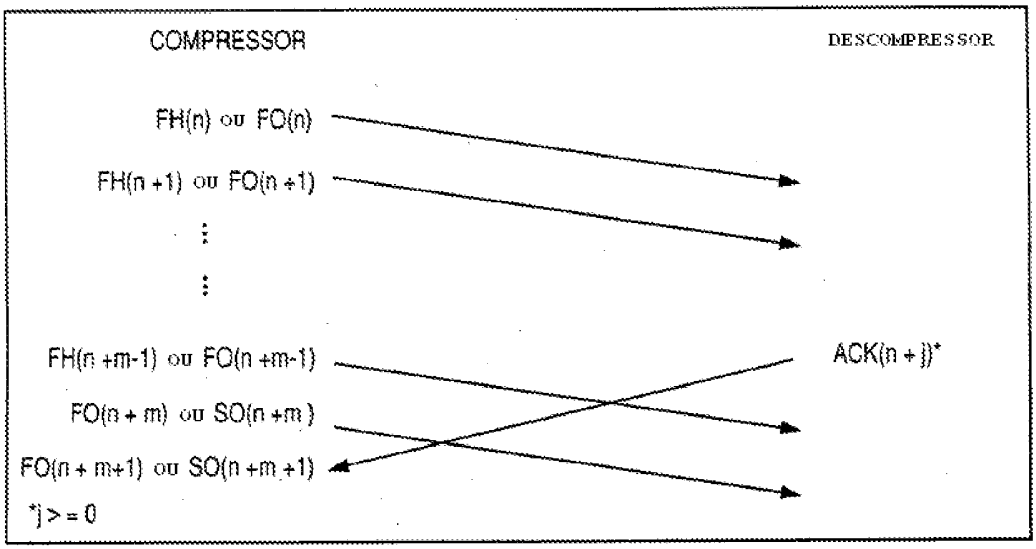

COMPRESSOR
DESCOMPRESSOR
FH(n) ou FO(n)
FH(n +1) ou FO(n +1)
FH(n +m-1) ou FO(n +m-1)
FO(n + m) ou SO(n +m)
FO(n + m+1) ou SO(n +m +1)
ACK(n + j)*
*] > = 0

FIG. 18

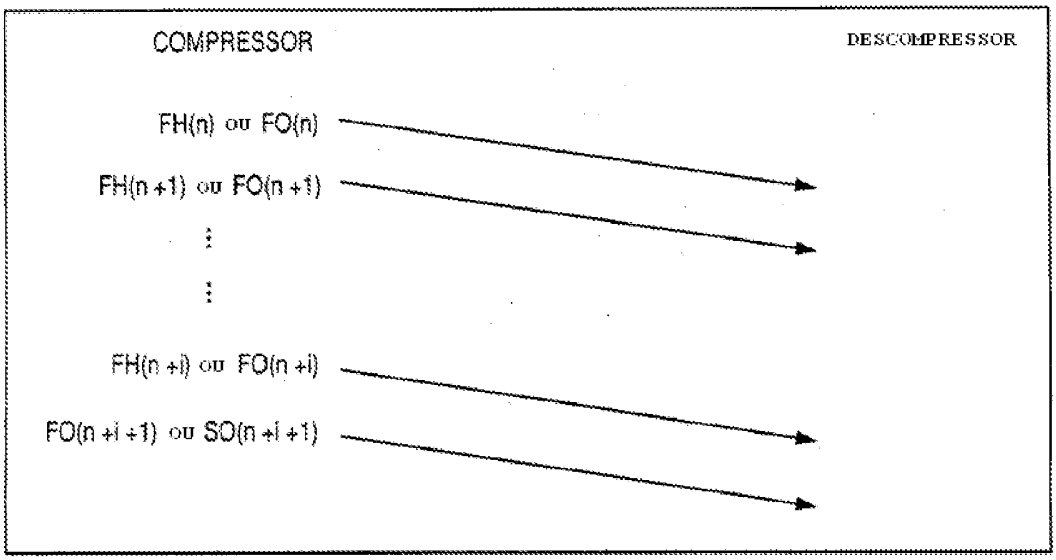
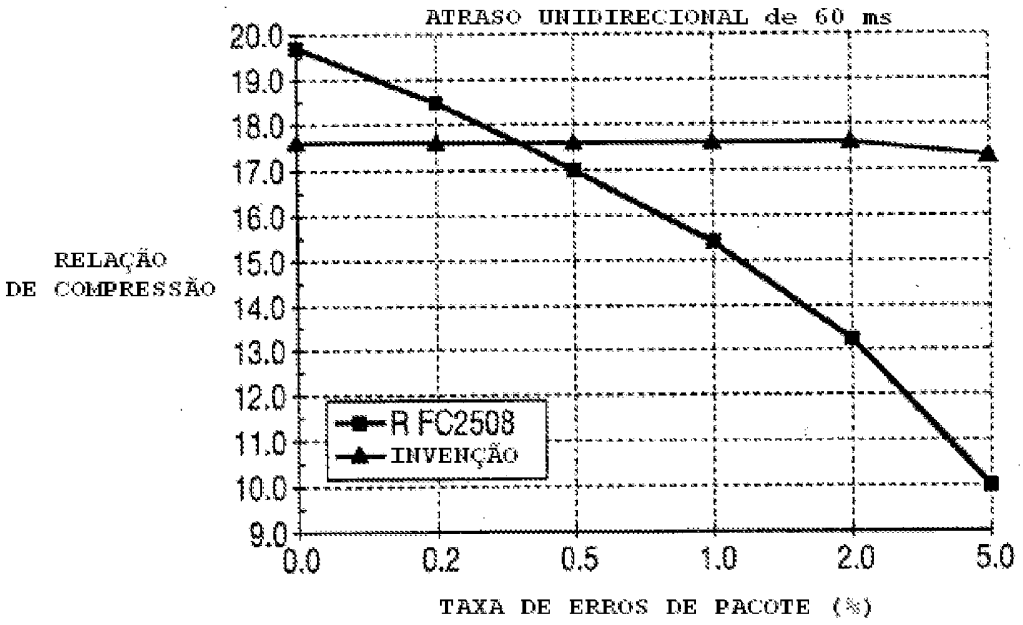

COMPRESSOR
DESCOMPRESSOR
FH(n) ou FO(n)
FH(n+1) ou FO(n+1)
FH(n+i) ou FO(n+i)
FO(n+i+1) ou SO(n+i+1)

FIG. 20


ATRASO UNIDIRECIONAL de 60 ms
RELAÇÃO DE COMPRESSÃO
20.0
19.0
18.0
17.0
16.0
15.0
14.0
13.0
12.0
11.0
10.0
9.0
0.0
0.2
0.5
1.0
2.0
5.0
R FC2508
INVENÇÃO
TAXA DE ERROS DE PACOTE (%)