

Proposta de Middleware Para Compressão Adaptativa de Dados em Ambientes Android

Angelo F. Assis¹, Ricardo A. R. Oliveira¹

¹Departamento de Computação – Universidade Federal de Ouro Preto (UFOP)
Ouro Preto – MG – Brasil

{angeloassis13, rrabelo}@gmail.com

Abstract. *The evolution of mobile devices, combined with increased access to the Internet through wireless networks results in a situation in which users stay online longer, using all available network. The Android operating system for mobile devices, is an example of a system that explores all this development resources, and is the system used in this work. On mobile devices, there are several limitations to interface, battery time, processing power and general limitations of the hardware configuration. These constraints present challenges in developing mobile applications. Thus, we present a solution to save energy, the adaptability, that is to adjust the application's behavior according to its environment awareness, in order to minimize consumption of scarce resources such as energy and bandwidth. The adaptation discussed is to compress data to be sent through the wireless channel, which reduces transmission time, but increases the processing time for compression / decompression. This adaptation introduces a research that evaluates when compression should occur, for several parameters of the device context must be evaluated and combined with a decision model included in the middleware.*

Resumo. *A evolução dos dispositivos móveis, associado a um maior acesso à Internet através de redes sem fio resulta em uma situação na qual os usuários permanecem mais tempo online, usando toda a cobertura disponível. O sistema operacional Android, para dispositivos móveis, é um exemplo de sistema que explora toda essa evolução de recursos, e é o sistema utilizado no trabalho. Em dispositivos móveis, existem diversas limitações para interface, tempo de baterias, capacidade de processamento e limitações gerais da configuração de hardware. Estas restrições apresentam desafios no desenvolvimento de aplicações móveis. Assim, apresentaremos uma solução para economia de energia, a adaptabilidade, que consiste em ajustar o comportamento do aplicativo de acordo com sua consciência do ambiente, com o objetivo de minimizar o consumo de recursos escassos, como a energia e largura de banda. A adaptação discutida é a compressão dos dados a serem enviados pela rede sem fio, que diminui o tempo de transmissão, porém aumenta o tempo de processamento para compressão / descompressão. Essa adaptação introduz um estudo que avalia quando a compressão deve ocorrer, pois vários parâmetros do contexto do dispositivo devem ser avaliados e combinados em um modelo decisório incluso no middleware.*

1. Introdução

Os aparelhos celulares muitas vezes não são mais usados somente para conversa entre pessoas, mas sim com várias outras funções com a de GPS, acesso à internet, para uma

possível conferência de email, entre outros. Não só os celulares, mas diversos dispositivos móveis com comunicação sem fio já estão presentes em nossas atividades diárias como PDAs (Personal Digital Assistant), notebooks, netbooks, smartphones. Além da evolução dos dispositivos e aplicativos, existe a evolução das tecnologias de rede sem fio, que estão cada vez mais poderosas com maior largura de banda, maior alcance, etc.

1.1. Problema

A transmissão de dados e o acesso a internet em um ambiente de comunicação sem fio a partir de um dispositivo móvel é uma tarefa diferenciada se comparada a computadores de mesa, uma vez que os recursos dos dispositivos móveis são inerentemente escassos. Existem várias limitações para os dispositivos móveis como interface, tempo de baterias, capacidade de processamento e limitações gerais da configuração de hardware. As principais restrições no ambiente de computação móvel são os dispositivos, as redes sem fio, e a mobilidade [Gupta 2008]. Essas restrições apresentam desafios reais no desenvolvimento de aplicações para esse tipo de ambiente. Quanto à limitação do hardware, um dos maiores problemas é o gasto de bateria. Primeiramente, várias aplicações sensíveis ao contexto executam continuamente em background coletando informações ou esperando por uma condição de ativação e assim, uma pequena necessidade de energia tem o potencial para afetar o dispositivo mais que outras aplicações mais pesadas, porém que executam em menor tempo. Além disso, várias aplicações são consideradas úteis por um grande número de usuários. Com isso, o impacto de consumo dessas aplicações deve ser minimizado. A complexidade desses dispositivos torna difícil para os desenvolvedores compreender o consumo de energia das aplicações.

1.2. Motivação

Para superar as limitações dos dispositivos móveis, podemos utilizar informações do contexto no qual o dispositivo está inserido afim de realizar adaptações dinâmicas na aplicação. A noção de contexto se refere à situação na qual um dispositivo está inserido. Computação ciente de contexto é um paradigma computacional no qual aplicações podem descobrir e se aproveitar de informações contextuais, tal como a localização do usuário, nível de bateria do dispositivo, tipos de rede sem fio disponíveis, etc. O objetivo de uma aplicação ciente de contexto é elaborar uma maneira de obter informações do contexto do usuário, ambiente e dispositivo computacional, considerando tanto suas características de hardware, como também de software e de comunicação para prover serviços apropriados para um determinado usuário em um certo momento. Adicionar a ciência de contexto em uma aplicação móvel pode incrementar sua usabilidade de forma a aumentar a satisfação do usuário, além de permitir que sejam adicionadas novas funcionalidades à aplicação.

Considerando que os recursos dos dispositivos que mais consomem energia são as redes sem fio, geralmente 3G, Bluetooth e Wifi [Rice and Hay 2010], é necessário introduzir algum processo de adaptação do conteúdo da Web, fazendo com que a comunicação e transmissão dos dados economize energia. Uma solução para superar esse problema é a adaptabilidade. A adaptação consiste em alterar ou ajustar o comportamento do aplicativo de acordo com o seu contexto. Essa adaptação é mais eficaz quando ocorre de forma dinâmica, isto é, quando o processo de adaptação analisa o ambiente e decide que ação deve ser tomada durante o tempo de execução. O objetivo é economizar ou minimizar o consumo de recursos escassos, como a energia e largura de banda, ou otimizar as propriedades que são mais visíveis para os usuários - como o tempo de resposta e qualidade

dos dados. O processo de adaptação pode ser realizado na aplicação, no sistema, ou em um middleware específico. Geralmente, um middleware projetado para um ambiente sem fio é dividido em dois componentes - um no dispositivo e outro em um ponto entre o dispositivo e seu par na rede com fio (por exemplo, a estação de base).

Assim, supõe-se que para amenizar os problemas com limitação de dispositivos móveis, podemos usar as informações do contexto do dispositivo e propor adaptações como a compressão de dados a serem transmitidos pela rede sem fio. A partir do estudo das condições que definem quando e como a compressão deve ocorrer, o objetivo é o desenvolvimento de um modelo decisório, implementado em Android, que define se um arquivo deve ser comprimido ou não antes de ser transmitido para um dispositivo móvel por uma rede sem fio. O foco inicial do trabalho é para arquivos de texto, mas a técnica desenvolvida pode ser estendida para outros tipos de arquivo, por exemplo, multimídia.

1.3. Contribuição

Esse trabalho propõe uma nova técnica para decisão de quando um arquivo deve ser comprimido antes de sua transmissão em um ambiente sem fio. Ao contrário de alguns dos trabalhos anteriores, o modelo proposto foi testado com dispositivos reais, não sendo validado apenas com estimativas e simulações, além de ter sido implementado com base em estudos de quais os parâmetros que realmente influenciam na transferência de arquivos e principalmente no gasto de tempo e energia.

O restante deste artigo está organizado da seguinte forma: a seção 2 apresenta alguns trabalhos relacionados com o assunto, além de críticas e considerações sobre cada um. A seção 3 fornece uma base teórica do trabalho, com conceitos importantes para o entendimento de toda a implementação e do modelo desenvolvido. Na seção 4 são descritos detalhes da implementação do modelo, além da descrição dos experimentos. Finalmente, na seção 5, apresentam-se os resultados obtidos, as conclusões alcançadas e estimativas de trabalhos futuros.

2. Trabalhos Relacionados

O trabalho de [Couto 2003] avalia quando e como deve ocorrer a compressão de arquivos HTML para uma possível transmissão por uma rede sem fio. Seus resultados foram obtidos através de testes simulados considerando diferentes situações e arquivos HTML. Foram estudados vários algoritmos de compressão, com objetivo de avaliar a desempenho de cada um com arquivos HTML e XML. Assim, foi proposto um modelo adaptativo que define quando a compressão deve ser usada e considera alguns fatores do contexto para tomar essa decisão. Esses fatores são considerados em dois módulos, um módulo de *tempo* e outro de *energia*. Cada um desses módulos passa um valor V_t e V_e respectivamente para um terceiro módulo, D_e , responsável pela decisão final. Os valores de V_t e V_e são definidos a partir de algumas propriedades como o tamanho do arquivo, taxa de compressão, largura de banda, etc. Um fator de peso, W_t e W_e é associado a cada módulo, representando sua confiança, sendo:

$$W_t, W_e \in [0, 1] \quad (1)$$

A decisão final é tomada a partir do uso da seguinte fórmula:

$$D_c = \begin{cases} 0 & , \text{ se } W_t = W_e = 0 \\ \frac{W_t \cdot V_t + W_e \cdot V_e}{2.0} & , \text{ senão} \end{cases} \quad (2)$$

Um arquivo é enviado em sua forma original quando $D_c < 0.5$ e é comprimido se $D_c \geq 0.5$. Isso significa que a compressão ocorre apenas se a decisão dos módulos for de alta confiança e concordarem com a compressão ou quando um valor de confiança é alto o bastante para compensar o outro. O problema nesse caso é o valor de D_c que deve ser definido para tomar ou não a decisão. O trabalho que será desenvolvido estudará esse valor, que atualmente é 0.5, podendo ajustar de acordo com testes e estudos realizados a partir desse modelo. A mudança desse valor influencia diretamente no desempenho do sistema, podendo prejudicar ou melhorar bastante o resultado, por isso, deve ser bem estudado antes de ser alterado. Vários testes com diferentes valores são fundamentais para definir se esse valor será alterado e ainda se será o mesmo em todos os casos. Dependendo da situação pode-se definir de acordo com o tipo do arquivo, conexão, ou outros fatores, fazendo com que exista uma definição dinâmica ciente do contexto.

Os estudos de [Xiao et al. 2010] mostraram que a eficiência da economia depende do equilíbrio entre a energia gasta na compressão/descompressão do dado e na energia economizada na transmissão. A economia na transmissão depende da redução dos dados, que por sua vez depende da taxa de compressão, que varia de acordo com o algoritmo de compressão e o tipo do dado. Foi proposto um framework de adaptação de energia baseado em Proxy que utiliza compressão de dados com o objetivo de economizar energia em dispositivos móveis quando estes estão recebendo dados de uma rede sem fio. Para melhorar a eficiência da compressão, o framework leva em consideração as características do dado, ambiente de transmissão da rede, e informações de contexto do cliente. O framework mostrou uma economia de energia através do caso de estudo de serviço de email adaptado.

O trabalho de [Rice and Hay 2010] resultou em um framework de medição que fornece com detalhes o consumo de energia de cada recurso e é projetado para desenvolver um entendimento de como alguns aspectos particulares de uma aplicação consomem energia. O estudo foi realizado nos dispositivos Android, G1 e Magic. Um resultado interessante desse trabalho é que foi constatado que entre as redes 2G, 3G e WiFi, a rede WiFi é a que possui menor custo de energia para o dispositivo, seguida por 3G e então 2G. Esse tipo de análise realça a importância da consideração não só da velocidade de transmissão, mas também do tipo de rede em uso entre os parâmetros que definem o contexto do dispositivo. Dessa forma, trabalhos como o de [Steinberg and Pasquale 2002] podem ser facilmente melhorados, pois os autores utilizam um modelo adaptativo simples para compressão, tanto para imagens (compressão com perda) quanto para texto (compressão sem perda) que considera somente a largura de banda do meio de transmissão para determinar se a compressão deve ser realizada.

3. Contexto do Dispositivo

O modelo decisório desse trabalho é implementado como um serviço que será oferecido ao usuário. Nesse caso, serviços são definidos como uma maneira adaptada de acessar recursos ou funcionalidades remotas utilizando a infraestrutura de rede disponível. Todas

as condições que serão analisadas para a definição da decisão do modelo definem o contexto do dispositivo móvel. Um fator relevante no desenvolvimento desses serviços é o uso dessas informações de contexto no qual o serviço é requisitado. O contexto pode ser dividido em duas partes: as características físicas, como capacidade do dispositivo, sua localização física ou rede na qual está conectado no momento e características lógicas, como preferências do usuário relacionadas às aplicações que são utilizadas neste dispositivo [Salber et al. 1999].

Como parâmetros do contexto, consideraremos nesse trabalho as informações relativas ao tipo e velocidade da rede na qual está conectado o dispositivo, o tamanho do arquivo a ser baixado e uma estimativa da taxa de compressão desse arquivo. Dentro desse ambiente, o serviço terá acesso a um servidor com os arquivos disponíveis para serem baixados. A comunicação entre o dispositivo e o servidor será através de requisições HTTP realizadas pelo serviço, que recebe como resposta um download do servidor. Entretanto, antes do serviço realizar a requisição HTTP, deve executar o modelo decisório, o que inclui coletar todas as informações do contexto, analisar a viabilidade da compressão e a partir da resposta do modelo requisitar o arquivo no servidor, comprimido ou não. Consideraremos que o servidor já possui os arquivos requisitados em sua forma normal e comprimidos.

4. Compressão Adaptativa

Em geral, o formato comum de documentos na Web é o HTML e suas variantes (ASP, JSP, PHP, XML, DHTML e outros). Compactar um arquivo desse tipo antes da sua transmissão através de um ambiente sem fio parece ser uma solução óbvia de adaptação se a intenção for apenas economizar a largura de banda. Porém a compressão de dados a serem enviados pelo canal de rede sem fio diminui o tempo de transmissão e aumenta o tempo de processamento para compressão e descompressão. Esse tipo de adaptação introduz um estudo que avalia quando a compressão deve ocorrer.

A compressão dos arquivos com certeza diminui o tempo de transmissão, mas por outro lado, os arquivos compactados precisam ser comprimidos e descomprimidos, o que pode aumentar o tempo de resposta total e pode consumir mais energia do que as transmissões sem compressão. Suponha o seguinte cenário: um servidor Web acessado por diferentes clientes, que requisitam downloads, através de um canal sem fio. Como cada aparelho tem suas características próprias e estas características variam muito neste ambiente heterogêneo, a decisão de comprimir um arquivo não é uma tarefa simples e deve ser feita com cuidado. Para algumas aplicações e dispositivos a compressão pode ser útil, enquanto que o mesmo aplicativo em um dispositivo diferente poderia ter um melhor desempenho sem compressão. Tomando a decisão correta de comprimir ou não o arquivo pode-se aumentar o tempo de bateria nos dispositivos, diminuir o tempo de transmissão dos dados, e aumentar o desempenho geral do sistema.

O primeiro passo para avaliar se o arquivo deve ser comprimido é definir em quais cenários a compressão deve ser utilizada, ou seja, quando a compressão reduzir o tempo de resposta ou consumo de energia. O segundo passo é selecionar os parâmetros que interferem na escolha, como: largura de banda, tamanho do arquivo a ser transmitido, taxa de compressão, tipo do dispositivo onde a descompressão é feita, consumo de energia para transmissão e processamento, entre outros. Em terceiro, todos esses parâmetros

devem ser combinados para alcançar o objetivo descrito: economizar tempo e/ou energia [Couto et al. 2003].

O middleware que realiza essa adaptação tem que ser compatível com o sistema operacional (SO) usado no dispositivo. Cada um desses dispositivos possui um SO diferente, que gerencia os recursos do sistema e fornece uma interface entre o dispositivo e o usuário, possibilitando uma interação entre eles. Cada SO trabalha de uma forma diferente no gerenciamento de memória, processos, etc., e aborda recursos específicos de hardware de acordo com o dispositivo. O SO para dispositivos móveis, chamado Android, inicialmente desenvolvido pelo Google, é um exemplo de sistema que explora toda a evolução de recursos desenvolvidos e adaptados para esse tipo de dispositivo, principalmente a conexão à internet. Devido a essa característica do Android de explorar bastante a conexão com internet, este foi o sistema operacional escolhido para desenvolver um middleware que realizasse a compressão adaptativa dos dados a serem transmitidos pela rede sem fio.

5. Android

A plataforma Android foi concebida inicialmente pelo Google. Atualmente, a plataforma está sendo mantida pelo Open Handset Alliance, que é um grupo formado por mais de 30 empresas (de tecnologias de dispositivos móveis, provedoras de serviços móveis, fabricantes, etc.) as quais se uniram para inovar e acelerar o desenvolvimento de aplicações e serviços, trazendo aos consumidores uma experiência mais rica em termos de recursos e menos custosa em termos financeiros para o mercado móvel. Pode-se dizer que a plataforma Android é a primeira plataforma móvel completa, aberta e livre, além de ter a característica peculiar de não ser dependente do hardware, possibilitando sua instalação em praticamente qualquer modelo de aparelho celular. Esta plataforma foi desenvolvida utilizando kernel do sistema operacional Linux. Sendo assim, todas as características intrínsecas deste sistema foram incorporadas, como, por exemplo, o sistema de arquivos, gerenciamento de memória, etc.

Como o sistema operacional Android foi inicialmente desenvolvido pela empresa Google, a maioria dos seus recursos e aplicativos são direcionados a serviços disponibilizados pelo Google. Como exemplo, podemos citar Gmail, Gtalk, Google Maps, Google Agenda, e outros webservices que sejam considerados úteis pelo usuário. Além dos aplicativos que fazem uso dos serviços do google, alguns dispositivos Android já são vendidos com certos aplicativos instalados, como twitter, facebook, canais de notícias, etc. O Android ainda conta com vários desenvolvedores, que estão cada vez mais interessados em desenvolver aplicativos para Android. Esse interesse ocorre devido ao grande investimento de empresas em Android, que contribuem com a popularização e com o crescimento acelerado de dispositivos sendo comprados mundialmente. Milhares de aplicativos estão disponíveis no Android Market, que seria um equivalente à conhecida AppStore, pertencente a Apple. Grande parte desses aplicativos são gratuitos, ou então tem preços muito baixos, contribuindo para que os usuários Android adquiram com maior facilidade os programas de sua preferência.

Todos esses aplicativos contribuem para que o Android seja cada vez mais usado por usuários que podem personalizar seu aparelho da forma que preferir. Porém, um detalhe importante a ser observado é que a maioria desses aplicativos utilizam a conexão à

internet. Todos os exemplos citados são aplicativos que utilizam conexão HTTP para o acesso aos serviços e informações. O Android ainda possui o recurso de sincronização, que mantém todos os aplicativos sempre atualizados. Para isso é preciso que exista sempre uma conexão disponível, e assim o dispositivo utiliza essa conexão para cada aplicativo que esteja em execução solicitando sincronização. A utilização desse recurso gera uma grande concorrência no recurso de rede sem fio do dispositivo, pois várias são as aplicações que utilizam a conexão e requisitam informações na internet.

Outra característica da plataforma Android é a alta taxa de resposta às requisições do usuário, além de fornecer uma API completa para a criação de aplicações cientes do contexto. Essa característica é refletida inclusive no estilo de programação dos dispositivos, uma vez que os desenvolvedores devem criar aplicativos que sejam responsivos à interação com o usuário, caso contrário, o sistema operacional finaliza a execução do aplicativo.

Cada aplicativo Android pode ser subdividido ainda mais em unidades funcionais distintas:

- Atividades: uma atividade apresenta uma interface visual focada nas atividades do usuário;
- Serviços: os serviços não têm uma interface visual do usuário, e executam em segundo plano por tempo indeterminado;
- Intents: respondem às solicitações de serviço de outra aplicação.

Atividades são os componentes de um aplicativo Android que estendem a classe base *Activity* e definem uma interface que consiste em uma visualização que responde a eventos. Se um aplicativo consiste em três janelas (por exemplo, uma janela de login, uma janela de visualização de texto e uma janela de visualização de arquivo), cada uma é geralmente representada por uma diferente classe *Activity*. O Android mantém uma pilha de histórico para cada execução de aplicativo a partir da página inicial e é possível clicar no botão Voltar para rolar de volta por esse histórico de atividades.

Um serviço é um aplicativo executado em um nível baixo e sem um monitor ou interface gráfica. É geralmente um aplicativo que deve ser executado por muito tempo em segundo plano. É possível se conectar a um serviço já em execução ou iniciar o serviço se ele não estiver em execução, possibilitando o uso de chamadas de procedimento remoto como mostrado na figura 1. Em suma, o mecanismo funciona da seguinte forma: é declarada uma interface RPC que se pretende implementar através de um simples IDL (linguagem de definição de interface). A partir dessa declaração, a ferramenta AIDL gera uma definição de interface Java que deve ser colocado à disposição de ambos processos, local e remoto.

Um exemplo de serviço é um programa reproduzidor de mídia reproduzindo uma lista de músicas. Apesar de um aplicativo de reprodução de mídia apresentar uma interface que permite que os usuários definam suas listas de reprodução, o programa passa o controle para o serviço para realmente reproduzir as músicas da lista de reprodução fornecida.

Um dos primeiros passos para a realização desse trabalho foi verificar o algoritmo de compressão disponível na plataforma de desenvolvimento Android. Essa plataforma

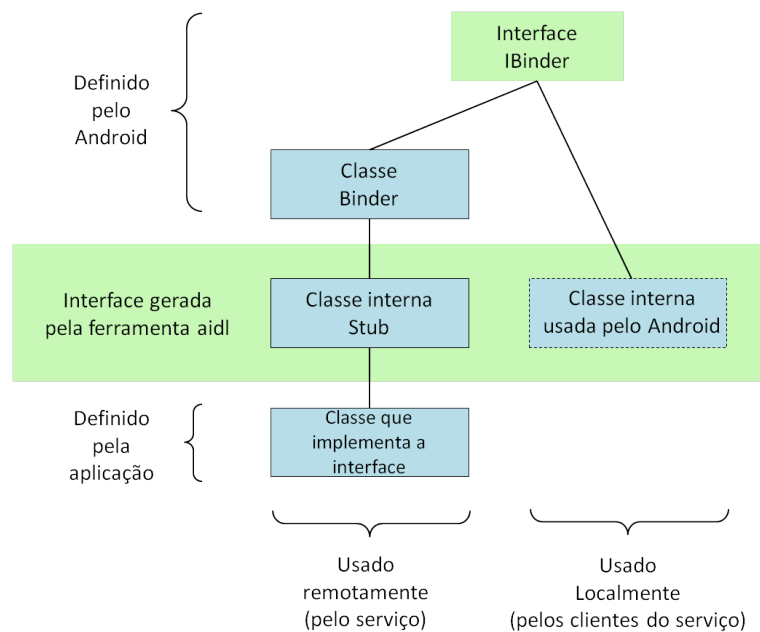


Figura 1. Chamadas de Procedimento Remoto [Android Developers 2010]

conta com métodos que implementam a tecnologia de compressão GZIP, criada por Jean-Loup Gailly e Mark Adler [Gailly and Adler 2003], [Delorie 2007]. Essa funcionalidade está disponível no pacote `java.util.zip`, que está presente no Android. A classe `GZIPInputStream` deste pacote é usada para ler dados armazenados no formato GZIP, sendo capaz de realizar leitura e descompressão dos dados. Esse método foi escolhido por já ser implementado nas bibliotecas padrões do Java, além de ser considerado eficiente e confiável.

6. Middleware para Compressão Adaptativa

Considerando as características do Android, foi criada uma aplicação com uma atividade e um serviço, mostrada na figura 2.

Essa aplicação simula um usuário navegando na internet e baixando arquivos aleatórios. Devido à característica do serviço de ser executado em segundo plano e à característica da atividade de ter que ser responsiva ao usuário, é trivial que em uma aplicação como essa, o download seja realizado no serviço e a atividade fique por conta apenas de interações com usuários. A atividade tem a função de coletar a URL que contém o endereço de um arquivo a ser baixado e acionar o serviço. Após o serviço ser iniciado, o mesmo é vinculado à atividade e recebe a URL coletada. A partir de centenas de testes realizados com essa aplicação, foi possível montar uma base de dados considerando o tempo de transmissão e o gasto de energia para diferentes tamanhos de arquivos e velocidades de rede. Essa base de dados será útil para o treinamento do modelo decisório que foi implementado a partir de uma rede neural Multilayer Perceptron (MLP) completamente conectada, onde foram considerados quatro fatores:

1. Tamanho do arquivo;
2. Taxa de compressão;
3. Tipo de rede sem fio conectada;
4. Velocidade atual da rede sem fio;



Figura 2. Integração - Atividade e Serviço

A rede neural utilizada pode ser melhor visualizada na figura 3. Os quatro parâmetros do contexto do dispositivo formam a entrada da rede. A camada intermediária possui 20 neurônios com função de ativação sigmoidal. A camada de saída possui dois neurônios, onde cada um representa uma classe: baixar o arquivo comprimido ou não comprimido.

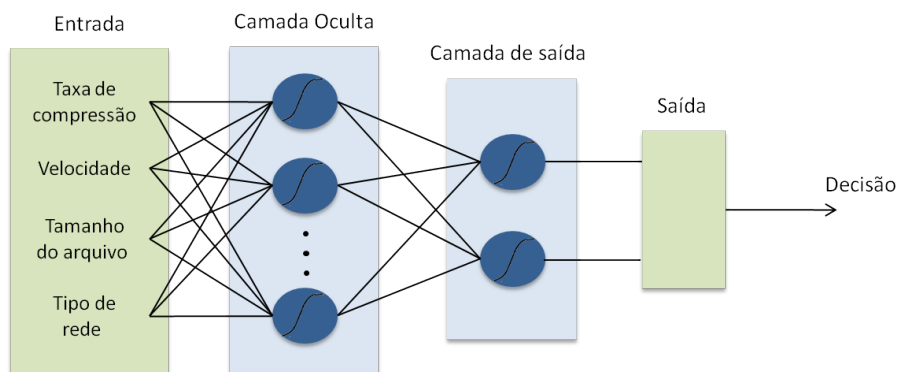


Figura 3. Rede Neural - Modelo Implementado

Com a base de treinamento pronta, foi aplicado o algoritmo *backpropagation*, que é um algoritmo de aprendizado supervisionado. Esse treinamento é comprovadamente eficiente na atualização dos pesos das conexões [Rumelhart et al. 1988]. Uma vantagem da utilização da rede neural nesse caso foi a possibilidade de realizar um treinamento antecipado que forneceu os valores dos pesos de cada conexão. Com esses pesos salvos, basta transferir a rede para o dispositivo e executar o cálculo de decisão com base nas funções de cada neurônio e pesos de cada conexão.

Então, a aplicação utilizada sofre uma pequena alteração, para que tome a decisão

de compressão. Como antes, a atividade coleta a URL que contém o endereço de um arquivo a ser baixado e aciona o serviço. Após o serviço ser iniciado, o mesmo é vinculado à atividade e recebe a URL coletada. Então começa uma sequência de passos de execução do modelo. Em primeiro lugar, o serviço se conecta à URL através de HTTP. Em seguida coleta todas as informações necessárias para definir o contexto do dispositivo. A partir desses parâmetros, executa o modelo decisório que indica se deve baixar o arquivo comprimido ou não. Por fim, de acordo com a resposta do modelo, o serviço realiza efetivamente o download e realiza uma chamada de volta para a atividade indicando que o download foi realizado. Caso seja baixado o arquivo comprimido, antes de chamar a atividade novamente o serviço realiza a descompressão.

7. Experimentos e Resultados

Com toda a estrutura e aplicação já descrita em 6, foi utilizado para experimentos um dispositivo com Android 2.1 desenvolvido pela empresa HTC, chamado Nexus. Esse dispositivo possui conexões 2G, 3G, Wifi 802.11 b/g e Bluetooth 2.0, sendo útil para futuros testes em diversificados tipos de redes, considerando diferentes situações. Possui 512MB de memória RAM, um processador de 1GHz e uma bateria de 1400mAh, que, segundo sua especificação, dura mais de 290 horas em *standby*. Sua memória é expansível até 32GB, com cartão de memória.

Utilizando esse dispositivo, foi conduzida uma série de experimentos utilizando a rede Wifi. Nesses experimentos houve grande variação da velocidade de transmissão da rede, entre 1 e 72Mbps. Essa variação foi intencional, com o propósito de verificar a influência da velocidade de transmissão no tempo de download e gasto de bateria, fatores que são fundamentais para que o modelo decisório tome a decisão certa. Nesses testes foram baixados arquivos de texto (.txt) de 1, 5, 10, 500, 1000 e 2000Kb. Todos esses arquivos também foram comprimidos e disponibilizados no servidor, para que pudessem ser baixados também em sua forma comprimida, comparando-se os tempos de transmissão e gasto de bateria para concluir qual download seria mais vantajoso. Os arquivos comprimidos atingiram em média 90% de taxa de compressão, o que significa que arquivos de 1Mb passam a ter apenas 100Kb.

Após a realização dos experimentos e análise dos dados, uma das primeiras considerações a ser feita é que usar arquivos de texto (.txt) e a compressão GZIP é uma situação em que a compressão se mostra realmente vantajosa, pois foram alcançadas taxas de compressão de em média 90%. Considerando essa informação e o trabalho de Rice e Hay (2010) (detalhado na seção 2), conclui-se que para esses dispositivos, sempre que se economizar tempo na transmissão, também estará economizando bateria. Sendo assim, arquivos com alta taxa de compressão tendem à decisão de que devem ser comprimidos, pois dessa forma o tempo de transmissão será bem menor e o usuário perceberá esse menor tempo, assim como a economia no gasto de bateria.

Após os testes e uma rigorosa análise dos dados obtidos, foi possível montar uma base de testes para uma rede neural, especificando suas entradas e saída esperada. As entradas consideradas são os valores de tamanho de arquivo, taxa de compressão, tipo de rede sem fio e velocidade da rede sem fio. A saída esperada é 1 se o download do arquivo comprimido é melhor e 0 se o download do arquivo não comprimido é melhor. Com essa base de dados pronta, foi utilizada a toolbox para redes neurais do matlab (Neural

Network Tollbox™ 7) para testar a utilização da rede neural [Beale et al. 2010]. A base foi dividida entre 70% para treinamento, 15% para validação e 15% para testes, sendo executada 30 vezes na toolbox. A métrica de avaliação utilizada foi a taxa de acertos da porção da base selecionada para testes. Essa taxa alcançou uma média de 95,28% com um desvio padrão de 7,15%. Isso significa que apesar de um desvio padrão relativamente alto, a taxa de acerto mostra um resultado satisfatório, pois a operação de execução da rede para o cálculo da saída é rápida e sua influência no tempo de resposta ao usuário e no gasto de bateria não chega a ser considerável. Já a operação de treinamento pode ser custosa, devido às limitações de processamento de dispositivos móveis, sendo preferível então que seja realizada anteriormente.

Como o Android permite várias aplicações rodando ao mesmo tempo, foram instaladas cinco aplicações no dispositivo que realizam a mesma tarefa, baixar um arquivo de algum servidor usando conexão HTTP. O arquivo a ser baixado foi o de tamanho 1Mb, também utilizado em todos os outros experimentos. As cinco instâncias foram executadas ao mesmo tempo, e todas elas baixavam 10 vezes o arquivo. Entre as cinco instâncias em execução, três delas baixavam o arquivo sem ser compactado e as outras duas baixavam o arquivo compactado, para depois descomprimir e salvar no cartão de memória. Esse experimento serviu para estressar o dispositivo, considerando que houve uma grande concorrência de recursos. Todas as instâncias precisavam da conexão o tempo todo, e duas delas precisavam de um maior tempo de processamento para descomprimir os arquivos. Com esse teste notou-se que as duas instâncias que baixavam os arquivos comprimidos acabaram bem antes das outras três. Comparando-se com os outros testes, é possível perceber que com várias instâncias rodando simultaneamente, o tempo de download é aproximadamente três vezes maior, como mostrado na tabela 1.

Tabela 1. Média dos tempos (em segundos) de download com arquivo de 1Mb

	Cinco instâncias simultâneas	Apenas uma instância
Não Comprimido	41,48	14,34
Comprimido	23,21	7,86

8. Conclusão

Neste trabalho investigamos o uso da compressão em um dispositivo com o sistema operacional android, considerando os cenários nos quais ela deva acontecer. Devido aos avanços da arquitetura, outros parâmetros influenciam no desempenho do download do arquivo comprimido.

O teste com cinco instâncias simultâneas nos faz concluir que a maneira como o Android trabalha, com diversas aplicações que fazem uso da conexão HTTP contribui para uma alta competição por recursos. Sendo assim, é viável que em alguns casos sejam baixados arquivos compactados, fazendo com que exista uma menor concorrência para o uso da rede. Mas também pode ser que em alguns casos seja preferível baixar o arquivo não compactado, considerando que outras aplicações em execução fazem mais uso de processador do que de conexão. Esse fator introduz mais um parâmetro no contexto do dispositivo, relativo ao número de aplicações sendo executadas no momento, que utilizam os mesmos recursos da aplicação de interesse do usuário. Quanto mais aplicações estiverem em execução utilizando HTTP, maior será o tempo de download.

Assim, um possível trabalho futuro é analisar a influência de outras aplicações sobre a conexão com a internet, verificando o impacto causado sobre sua aplicação.

Referências

- Android Developers (2010). Android developers. <http://developer.android.com/index.html>.
- Beale, M., Hagan, M., and Demuth, H. (2010). Matlab neural network toolbox user's guide version 7.
- Couto, R. R., Rabelo, R. A., and Loureiro, A. A. F. (2003). Compressão adaptativa de arquivos html em ambientes de comunicação sem fio. In *Proceedings of the XXI Simpósio Brasileiro de Redes de Computadores*, pages 313–328.
- Couto, R. R. P. (2003). Compressão adaptativa de arquivos html em ambientes de comunicação sem fio. Dissertação de mestrado, UFMG - Universidade Federal de Minas Gerais, PPGCC - Programa de Pós-graduação em Ciência da Computação.
- Delorie, D. (2007). Gzip user's manual. http://www.delorie.com/gnu/docs/gzip/gzip_toc.html.
- Gailly, J. and Adler, M. (2003). The gzip home page. <http://www.gzip.org>.
- Gupta, A. K. (2008). Challenges of mobile computing. In *2nd National Conference on Challenges & Opportunities in Information Technology (COIT-2008)*, Mandi Gobindgarh.
- Rice, A. C. and Hay, S. (2010). Decomposing power measurements for mobile devices. In *PerCom*, pages 70–78.
- Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1988). *Learning representations by back-propagating errors*, pages 696–699. MIT Press, Cambridge, MA, USA.
- Salber, D., Dey, A. K., and Abowd, G. D. (1999). The context toolkit: Aiding the development of context-enabled applications. In *CHI*, pages 434–441.
- Steinberg, J. and Pasquale, J. (2002). A web middleware architecture for dynamic customization of content for wireless clients. In *Proceedings of the 11th international conference on World Wide Web, WWW '02*, pages 639–650, New York, NY, USA. ACM.
- Xiao, Y., Siekkinen, M., and Ylä-Jääski, A. (2010). Framework for energy-aware lossless compression in mobile services: the case of e-mail. In *IEEE International Conference on Communications*, pages 1–6.