Packet Transactions: High-Level Programming for Line-Rate Switches

Grupo 1
Francisco Caeiro, **47823**
Bruno Andrade, **47829**
António Estriga, **47839**

Qual é o problema que os autores tentam resolver?

Com o aparecimento de chips programáveis com bom desempenho no *data plane* surgiu um novo problema: como programar estes chips. A solução encontrada antes deste artigo foi o P4, uma linguagem de programação de baixo nível, baseada em *match-action tables*, em que é possível programar algoritmos no data plane. Apesar desta solução cumprir os requisitos de desempenho e de ser possível programar os algoritmos necessários, esta linguagem é uma DSL com uma sintaxe totalmente diferente das linguagens que estamos habituados (C, Python, Java…). Esta sintaxe de baixo nível dificulta a conversão de algoritmos escritos em linguagens de alto nível para P4.

Para melhor compreensão futura, é de mencionar que *line-rate switches* são switches que conseguem processar pacotes à velocidade da rede disponível, isto é, sem atrasar pacotes.

Este problema é relevante?

Este é um problema de elevada relevância pois são cada vez mais utilizados chips programáveis. Estes chips permitem personalizar e acrescentar algoritmos sem ser necessário comprar novos switches ou esperar que estes algoritmos sejam implementados pelas empresas que fabricam os switches. No entanto, apesar dos chips programáveis terem estas vantagens, é sempre necessário programadores especialistas em, por exemplo, P4 para programarem o data plane.

Qual é a sua solução? Que novas técnicas foram usadas?

Para facilitar o trabalho de um programador de algoritmos de data plane foi criada uma nova linguagem, o Domino, o seu respetivo compilador e um *machine model*, o Banzai, para permitir correr o código compilado nos switches programáveis.

O Domino é a primeira DSL de alto nível para *line-rate* switches. A linguagem tem uma sintaxe similar à da linguagem C e é baseada em transações de pacotes. Uma transação de pacotes é um bloco de código que é executado atomicamente, isto é, é executado até ao fim ou não é executado. A transação de pacotes é uma abstração que permite ao programador idealizar que cada pacote passa pela função de forma sequencial, um de cada vez. Isto permite que o programador apenas tenha de se preocupar com o algoritmo e não com a paralelização do mesmo.

O compilador do Domino traduz o código para configurações Banzai. Este apenas compila se for possível correr o programa a *line-rate*. A compilação do código tem 3 fases: Pré-Processamento, Pipelining e Geração do Código. Na fase de Pré-Processamento, o código é simplificado de forma a transformar-se em código de 3 moradas, isto é, todas as instruções transformam-se em leituras e escritas para variáveis de estado ou operações sobre campos de pacotes, que guardam o resultado em variáveis de estado como `pkt.f1 = pkt.f2 op pkt.f3`. No caso da fase de Pipelining o código sequencial é transformado em *codelets* de pipeline, para permitir a paralelização do código. Na fase de Geração do Código, se forem respeitadas as limitações de uma máquina Banzai, os *codelets* de *pipeline* são transformados numa configuração.

O modelo Banzai é um modelo para *line-rate* switches que é baseado nas arquiteturas já existentes, como o FlexPine. O modelo é constituído por duas pipelines, uma *ingress pipeline*, que recebe os pacotes executa operações e coloca-os na fila, e uma *egress pipeline*, que recebe os pacotes da fila,

executa operações e transmite os pacotes. A grande novidade do Banzai são as unidades de processamento chamadas átomos, onde cada átomo corresponde a uma operação atómica suportada pelo modelo. Cada átomo pode ser representado por um bloco de código sequencial e é implementado diretamente em hardware, permitindo uma performance a *line-rate*. Para manter o processamento à velocidade desejada, é necessário respeitar algumas regras, como por exemplo, o estado entre átomos não pode ser partilhado (limitação de hardware), cada operação é atómica e apenas demora um ciclo do relógio e o número de átomos por etapa e o número de etapas é limitado. Este modelo é bastante eficiente, mas tem dificuldades em lidar com grandes quantidades de dados e com operações que apenas necessitam de ser executadas em alguns pacotes.

Para analisar a solução, foi realizada uma avaliação onde foram desenhadas algumas máquinas Banzai para testar o código. Conclui-se que todas as operações testadas corriam à velocidade necessária para manter a velocidade de ligação e que os átomos ocupam pouco espaço em relação ao tamanho do chip.

Como é que se destaca de trabalhos anteriores?

Para comparar o Domino com o P4, foram executados vários algoritmos nas duas linguagens com o objetivo de os comparar em termos de linhas de código. Ficou demonstrado que os algoritmos em Domino são mais fáceis de escrever, ajudando o facto de já existirem versões dos algoritmos em linguagens de alto nível ou em pseudocódigo, para além de necessitarem de menos linhas de código.

Quais são os pontos mais fortes deste artigo? E os seus pontos fracos?

O artigo em si está bastante bem escrito e estruturado, apresentando uma introdução em todas as secções, com um resumo do seu conteúdo e a maneira como esse conteúdo é abordado.

Considerando agora os pontos fracos: os investigadores mencionam quais os átomos mais expressivos para a implementação dos vários algoritmos, mas não é mencionado uma lista de quais seriam os átomos a implementar num produto em produção.

É também mencionado que a implementação em P4 de apenas um dos algoritmos é disponibilizada publicamente. Neste artigo não é mencionado se, no caso de serem publicadas mais implementações, vale a pena mudar o hardware de uma rede para poder, utilizando o Domino, programar em mais alto-nível.

Apesar das limitações referidas nos pontos anteriores, o trabalho desenvolvido por estes investigadores é bom para a área de programação de algoritmos de data-plane, pois a implementação dos mesmos é mais leve em carga de trabalho e os seus resultados são equivalentes a trabalhos anteriores.

Como seria uma extensão deste trabalho?

É falado em vários pontos do artigo onde é que os investigadores gostariam de investir no futuro, tais como transações de múltiplos pacotes (pois apenas é estudado transações de pacotes unitários), mas talvez o ponto mais importante seja a otimização do compilador, para que este possa auxiliar no processo de desenho de átomos, tendo em conta que este processo atualmente é feito manualmente e *ad hoc*.