Titulo: Modeling GPU-CPU workloads and systems / Bibliografia: The ACM Digital Library (http://portal.acm.org/results.cfm?coll=ACM&dl=ACM&CFID=87091431&CFTOKEN=16714906) / Autores: Andrew Kerr, Gregory Diamos, Sudhakar Yalamanchili / Aluno: Vilmar Travassos RA078272.

Este resumo apresenta uma emulação e tradução de infraestrutura, e seu uso, na caracterização de cargas de trabalho em GPUs (Graphics Processing Units) da NVIDIA, líder mundial em tecnologias de computação visual e criadora da GPU. Em particular, técnicas padrão de análise de dados foram empregadas (*benchmarks*) para caracterizar padrões de referência, suas relações com a máquina e com parâmetros de aplicação, além da construção de modelos preditivos para a escolha da implementação entre CPU ou GPU, do *Kernel* baseado nos resultados de tradução do Ocelot. Em novembro de 2006, a NVIDIA CUDA, apresentou uma proposta genérica de arquitetura de computação paralela, como um novo modelo de programação paralela e conjunto de instruções, que otimiza o processador de computação paralela em GPUs NVIDIA, para resolver vários problemas computacionais complexos de uma maneira mais eficiente que em uma CPU. CUDA vem com um ambiente de *software* que permite aos desenvolvedores usar a linguagem de programação "C" como sendo uma linguagem de alto nível; outras interfaces de linguagens de programação de aplicações são suportadas, tais como *CUDA FORTRAN*, *OpenCL* e *DirectCompute*. O *NVIDIA's Parallel Thread eXecution* (PTX), é uma arquitetura de instrução virtual, com execução semântica explícita de dados em paralelo, que estão bem adaptados às GPUs da NVIDIA. O PTX é composto por um conjunto de instruções de RISC-like para tipos explícitos de cálculos aritméticos computacionais, *loads* e *stores* para um conjunto de endereços, instruções para paralelismo e sincronização, e variáveis construídas. As funções implementadas em PTX, conhecidas como *Kernell* destinam-se a ser executadas por um grande número de *threads* organizadas hierarquicamente em uma matriz de segmento cooperativo (CTAs). A infraestrutura do compilador Ocelot se esforça para dissociar aplicações CUDA de GPUs, envolvendo o *CUDA Runtime API*, analisando o *Kernel* armazenado como bytecode da aplicação em uma representação interna, para execução desse *Kernel* em dispositivos presentes e mantendo uma lista completa de alocações CUDA armazenados em memória. Ao desvincular uma aplicação CUDA, a partir do driver CUDA, o Ocelot oferece uma estrutura para simulação de GPUs colhendo métricas de desempenho, traduzindo *Kernels* para outras arquiteturas GPUs, executando instrumentação e otimização o *Kernel* para a execução em GPUs. O *framework* de tradução Ocelot fornece uma eficiente execução de *Kernels CUDA* em CPUs multicore, já na primeira tradução do Kernel de PTX para um conjunto de instruções nativas de infraestrutura Low-Level Virtual Machine (LLVM), aplicando uma série de transformações que implementam o modelo de execução PTX, com controle de estrutura dos dados disponíveis para o tipo escalar de microprocessadores. Esse *insight* fornece uma clara necessidade de se aprofundar mais na caracterização e refinamento de modelos preditivos, um trabalho bem mais extenso.