Workshop do Núcleo de Apoio à Pesquisa em Software Livre
São Carlos, 16 a 17 de Outubro de 2014.

Paralelização Automática de Código em Plataformas Heterogêneas

Rogério A. Gonçalves e Alfredo Goldman

Laboratório de Sistemas de Software
DCC – IME – USP
{rag,gold}@ime.usp.br

Introdução

- O hardware tem evoluído rapidamente e as plataformas tornam-se cada vez mais heterogêneas.
- A busca por alto desempenho tem-se voltado para a exploração dispositivos aceleradores.
- Ponto de vista do software.
- No cenário atual é evidente a grande distância entre o potencial de desempenho disponibilizado pelas plataformas e as aplicações que precisam fazer uso desse potencial.
- Plataformas vs. Aplicações.

Introdução

- Os *kits* de desenvolvimento ainda não estão maduros o suficiente.
- Programar novas aplicações ou traduzir aplicações de código legado não é uma tarefa trivial.
- Não há suporte direto para as aplicações existentes serem reescritas.
- O processo de reescrever o código é oneroso e em muitas situações impraticável.
- São necessárias soluções que deem suporte à execução de código de novas aplicações e as de código legado.
- O ideal é que a ligação entre contextos seja feita por ferramentas: ***compiladores paralelizantes.***
- Abordagens: *Bindings*, Diretivas de Compilação e **Paralelização Automática**.

Abordagens que se destacam

- ***Diretivas de Compilação***
  - Anotações são inseridas no código para guiar o compilador no processo de tradução e paralelização.

```
#pragma acc data copy(b[0:n*m]) create(a[0:n*m])
{
 for (iter = 1; iter <= p; ++iter) {
 #pragma acc kernels
 {
  for (i = 1; i < n-1; ++i)
   for (j = 1; j < m-1; ++j) {
    /* Suprimido. */
   }
  for( i = 1; i < n-1; ++i )
   for( j = 1; j < m-1; ++j )
   /* Suprimido. */
  }
 }
}
```

Exemplos mais conhecidos:

- OpenMP → *multicore*
- OpenACC → GPU

Abordagens que se destacam

- ***Paralelização Automática***
  - Código sem modificações (nem anotações).
  - A ferramenta deve detectar as regiões paralelizáveis e gerar uma versão paralela.

Algumas ferramentas que utilizam Paralelização Automática:

- Par4All
- PPCG
- KernelGen

Proposta

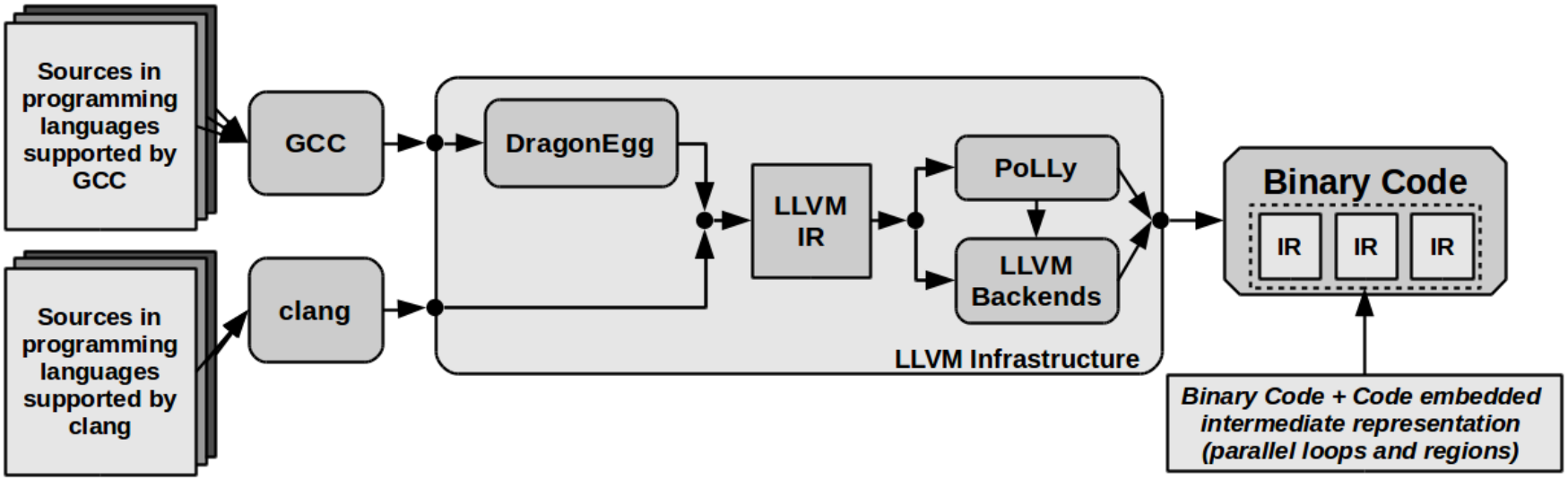
- Estudo de técnicas e ferramentas para o uso de paralelização automática em sistemas de computação paralela.
- Explorar o paralelismo latente das aplicações.
- Uso de eficiente de recursos → ***multicore e multiGPU.***
- Execução combinada de ***múltiplas aplicações*** → explorar o paralelismo entre tarefas de aplicações distintas.
- Recursos disponíveis nas arquiteturas *Kepler* e *Maxwell* podem favorecer soluções híbridas com múltiplas GPUs.
  - Paralelismo dinâmico, o acesso pareado e o mapeamento e compartilhamento de memória.

Ferramentas utilizadas

- ***LLVM (llvm.org)***
  - Projeto que fornece uma infraestrutura de compilação.
  - ***Desenvolvimento:*** *LLVM Team - University of Illinois at Urbana-Champaign.*
  - **Licença:** *University of Illinois/NCSA Open Source License*
  - Certificada pela OSI.
- ***Projetos (***DragonEgg, PoLLy e NVPTX)
  - ***DragonEgg:*** *Plugin* para GCC que transforma GIMPLE para LLVM-IR.
  - ***PoLLy:*** Implementação do Modelo *Polyhedral* para LLVM.
  - ***NVPTX:*** *Backend* LLVM para gerar código PTX (intermediário das GPUs da NVIDIA).

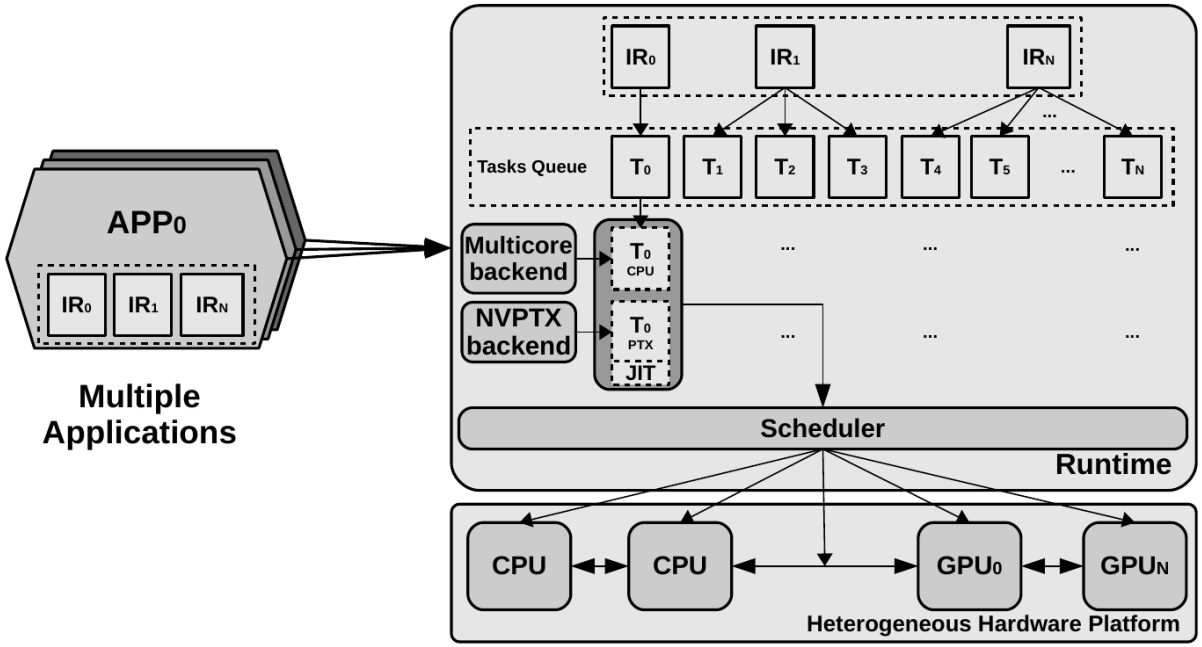
Proposta

- ***Ferramenta de Compilação:***
  - Detecção de regiões paralelizáveis que possam ter sua execução acelerada.
  - Preparar o código de entrada para o ambiente de execução.
  - Regiões mantidas em código intermediário (embarcadas).



Proposta

- ***Ambiente de Execução:***
  - Explorar elementos heterogêneos em cenários híbridos *multicore* e multiGPU.
  - Escalonamento de tarefas de múltiplas aplicações.



Obrigado!

Rogério Aparecido Gonçalves
Doutorando IME-USP
rag@ime.usp.br