HOCHSCHULE OSNABRÜCK
UNIVERSITY OF APPLIED SCIENCES

7. März 2017

Stelle für eine studentische oder wissenschaftliche Hilfskraft

Wir bieten im Rahmen des HiPReP-Projekts [1] zum nächstmöglichen Zeitpunkt eine Stelle für eine studentische oder wissenschaftliche Hilfskraft an. Die Stelle umfasst ca. 10 - 20 Stunden/Woche und ist zunächst auf sechs Monate befristet.

Der High-Performance Reconfigurable Processors (HiPReP) ist ein Co-Processor, der für High-Performance Computing optimiert ist. Gemäß der Faustregel „90% der Ausführungszeit werden in 10% des Codes verbracht“ sollen hier besonders Schleifen effizient ausgeführt werden können. Software-Compiler bieten schon diverse Techniken, um Schleifen zu analysieren und entsprechende Optimierungen des Codes durchzuführen. Ein Ziel des Projektes ist, diese Techniken auch für die Abbildung auf Hardware nutzbar zu machen.

Aufgaben

Im Zusammenhang mit der Entwicklung des HiPRePs sollen eine oder mehrere der folgenden Aufgaben durchgeführt und ausführlich dokumentiert werden:

- **Erzeugen eines Datenfluss-Graphen (DFG) aus C-Code mittels LLVM**
  Dieses Programm soll auf der Compiler-Infrastruktur LLVM [2] aufbauen und eine Graph-Datenstruktur (DFG) für innere Schleifenrümpfe erzeugen. Die Graphen werden dann in einem geeigneten Format zur Visualisierung und Weiterverarbeitung gespeichert.
- **Optimierung von Datenfluss-Graphen**
  In diesem Teilprojekt sollen DFGs eingelesen, optimiert und wieder gespeichert werden. Als mögliche Optimierungen kommen z. B. algebraische Vereinfachungen infrage, die den DFG verkleinern.
- **Schleifen-Optimierungen**
  Weitergehende Optimierungen erfordern Schleifen-Transformationen (vgl. [3,4]), z. B. Loop-Invariant Code Motion. Diese sollen - falls noch nicht implementiert - in den LLVM-Compiler integriert werden.

In diesem Bereich kann auch ein **Thema für eine Bachelor- oder Master-Thesis** abgesprochen werden.

```
for (int i = 0; i < n; i++) do
   a[i] = a[i] + 5 * x;
end for
```

Loop-Invariant Code Motion $\longrightarrow$

```
int c = 5 * x;
for (int i = 0; i < n; i++) do
   a[i] = a[i] + c;
end for
```

Beispiel für eine Compiler-Optimierung

Voraussetzungen

- Programmiererfahrungen in Java oder C/C++

Ansprechpartner

Philipp Käsgen	(SB 0216)	p.kaesgen@hs-osnabrueck.de
Professor Weinhardt	(SB 0223)	m.weinhardt@hs-osnabrueck.de

Literaturverzeichnis

[1] https://www.hs-osnabrueck.de/hiprep

[2] llvm.org

[3] D. F. Bacon, S. L. Graham, and O. J. Sharp, “Compiler transformations for high-performance computing,” ACM Comput. Surv., vol. 26, no. 4, pp. 345–420, 1994.

[4] K. D. Cooper and L. Torczon, Engineering a compiler. 2011.