## Kurzfassung

Die aktuell dominierende Herausforderung im Hochleistungsrechnen ist die Entwicklung paralleler Algorithmen, welche bis in den Petaflop Bereich skalieren. Ein etablierter Ansatz ist die Verwendung massiv-paralleler Grafikprozessoren (GPUs), häufig unter Verwendung von NVIDIAs Compute Unified Device Architecture (CUDA). CUDA bietet eine umfangreiche Programmierschnittstelle und erzielt hohe Rechenleistung je GPU. Damit Programme die verfügbaren Ressourcen auf GPU und CPU effizient nutzen, müssen diese jedoch häufig mehrere Parallelisierungsebenen gleichzeitig verwenden. Aus dieser steigenden Programmkomplexität ergibt sich die Notwendigkeit für leistungsfähige Werkzeuge, sowohl für die Programmentwicklung als auch für deren Leistungsanalyse.

Beitrag dieser Arbeit zum Stand der Technik ist die Bestimmung des kritischen Pfades von heterogenen CUDA- und MPI-Programmen sowie die Unterstützung des Entwicklers mittels Optimierungsvorschlägen, basierend auf einem Modell der Interaktionen und Abhängigkeiten zwischen CUDAs CPU-und GPU-Funktionen und MPI Operationen. Auf der Grundlage dieses regelbasierten Modells und der Aufzeichnung einer Programmausführung kann ein Abhängigkeitsgraph konstruiert werden. In diesem lässt sich der kritische Pfad bestimmen, welcher die Laufzeit des Programms dominiert. An Punkten an denen eine Programmspur auf Grund von Synchronisation auf eine andere warten muss und verzögert wird, können Wartestellen in den Graph eingefügt werden.

Aus den in der Analyse ermittelten Daten können hypothetische Programmabläufe vorhergesagt und verglichen werden. Schlussendlich wird dem Entwickler eine Liste mit vielversprechenden Optimierungsstellen, zu erwartenden Leistungsgewinnen und detaillierten Informationen zu Programmabhängigkeiten angeboten. Während der Entwicklung der Arbeit wurden zufällig generierte Eingabedaten sowie praktische Beispiele genutzt, um die berechneten Abhängigkeiten zu verifizieren und vorhergesagte Leistungssteigerungen zu validieren.