Kurzfassung

Die heutigen Hochleistungsrechner bedienen sich einer stetig wachsenden Anzahl von Prozessorkernen. Damit eine parallel ausgeführte Anwendung diese steigende Anzahl gut ausnutzen kann, muss das Programm eine entsprechende Skalierung besitzen. Die Kommunikation zwischen Prozessen oder Threads in der parallelen Anwendung hat einen großen Einfluss auf das Skalierungsverhalten. Neue PGAS-basierte Bibliotheken oder Spracherweiterungen zur Parallelisierung versprechen eine höhere Skalierung durch das Bereitstellen von asynchronen einseitigen Kommunikationsoperationen. GASPI ist ein Standard für solch eine Bibliothek und trennt die Synchronisierung von der Datenübertragung. Damit werden Methoden zur Prozesssynchronisierung und zu einseitigen Kommunikationsroutinen bereitgestellt. Der Entwickler muss selbst entscheiden, wann der Datentransfer und die dazugehörige Synchronisierung erfolgen soll. Um die Entwicklung solcher Anwendungen zu unterstützen, wurde in dieser Arbeit ein Werkzeug entwickelt, das Optimierungsvorschläge zu einseitigen Kommunikationsoperationen anhand einer aufgezeichneten Ausführungssequenz von Ereignissen erzeugt. Die Optimierungsvorschläge enthalten neue Positionen für den Aufruf der jeweiligen Kommunikationsoperation, um das Kommunikationsverhalten einer Anwendung zu verbessern. Damit die Erzeugung solcher Vorschläge möglich ist, müssen die Speicherzugriffe innerhalb der Anwendung identifiziert werden, denn die jeweilige Kommunikationsoperation steht in direkter Abhängigkeit zum referenzierten Speicher. In der Arbeit wurde ein *Tracing-Werkzeug* genutzt, das Funktionsaufrufe sowie Speicherzugriffe einer parallelen Anwendung aufzeichnet. Basierend auf dem erstellten *Trace* wird der kritische Pfad ermittelt, so dass nur die für die Skalierung und Performance relevanten Ereignisse der Anwendung betrachtet werden. Entlang des kritischen Pfades werden Kommunikationsoperationen und deren Optimierungen gesucht. Für die jeweilige Operation werden dem Nutzer Vorschläge zur Optimierung unterbreitet. Diese werden durch eine dynamische Analyse generiert und beziehen sich auf das erstellte *Trace*. Der konkrete Quelltext wird nicht mit in die Analyse einbezogen und der Nutzer muss die Realisierbarkeit der Vorschläge selbst prüfen. Durch die Umsetzung der generierten Optimierungen soll die Laufzeit der Anwendung verbessert werden. Das Verfahren wurde für GASPI-Programme implementiert und mit einem *stencil code* evaluiert. Die Auswertung der Laufzeit hat gezeigt, dass die Skalierung der Anwendung verbessert wurde.