Leistungsanalyse von Anwendungen mit Intel VTune

Martin Pelzer, Moritz Vieth

Stand: 27.11.2003

Seminararbeit zur Vorlesung
„Verteilte und parallele Systeme 2“
bei Prof. Rudolf Berrendorf

Fachbereich Informatik
Fachhochschule Bonn-Rhein-Sieg

Inhaltsverzeichnis

1. Einleitung

Diese Seminararbeit wurde im Rahen der Lehrveranstaltung „Verteilte und parallele Systeme 2" im Wintersemester 2003/2004 zur Erreichung der Prüfungszulassung angefertigt. Sie beschäftigt sich mit Softwareoptimierung mittels des Tools VTune der Firma Intel. Sie gibt eine Einführung in die Softwareoptimierung, beschäftigt sich eingehend mit den einzelnen Komponenten von VTune, zeigt anschließend Grenzen der Software auf, gibt einen kurzen Überblick über Konkurrenzprodukte und versucht schließlich, eine Bewertung zu Intel VTune abzugeben.

1.1 Einführung in die Softwareoptimierung

Viele Softwareprodukte haben einige wenige Stellen im Code, die den Großteil der Rechenzeit ausmachen, zum Beispiel bestimmte Methoden, die von einer Schleife aus aufgerufen werden. Diese Stellen im Code nennt man „Hotspots".
Diese Hotspots „auf dem Papier", also durch reines Analysieren des Codes von Hand, zu finden ist sehr schwierig und zeitaufwendig. Daher gibt es Programme, wie zum Beispiel das hier besprochene VTune von Intel, die diese Analyse automatisieren bzw. den Programmierer dabei unterstützen, Hotspots ausfindig zu machen und diese zu „entschärfen". So können diese Programme, nachdem ein Hotspot, beispielsweise eine oft aufgerufene Schleife, mittels diverser vom Programm zur Verfügung gestellter Methoden ausfindig gemacht wurde, Vorschläge unterbreiten, wie sich die gefundene Stelle im Code anderweitig strukturieren ließe, um weniger rechenaufwendig zu sein oder seltener aufgerufen zu werden.
Wichtig bei der Optimierung von Software ist jedoch nicht nur, eine möglichst gute Performance zu erzielen, sondern auch Stabilität zu gewährleisten, besonders bei Programmen, die eine Mehrzahl von Threads benutzen.
Kapitel 2 beschäftigt sich mit den oben genannten Methoden, um Hotspots und mögliche Problempunkte im Bezug auf Stabilität zu lokalisieren und Verbesserungsvorschläge zu unterbreiten.

1.2 Versionen von VTune

Intel VTune wird in zwei Versionen angeboten. Zum Einen gibt es die Version 7.0 für Windows. Sie enthält sämtliche in dieser Seminararbeit vorgestellten Komponenten, darunter auch die Remote Data Collection, die es unter anderem ermöglicht, auch auf Linux-Systemen eine Performance-Analyse über ein Remote-Login durchzuführen. Hierbei stehen allerdings nicht alle Funktionen zur Verfügung. Zudem lässt sich diese Version vollständig in eine .NET-Entwicklungsumgebung von Microsoft einbetten [Intel].

Des weiteren gibt es die Version 1.1 für Linux. Sie enthält lediglich die Sampling- und Call Graph-Funktionen, ist dafür aber nicht auf ein Windows-System angewiesen, von dem mittels Remote Data Collection eine Performance-Analyse durchgeführt wird [Intel].
Für Betriebssysteme wie z.B. MacOS X oder Solaris gibt es selbstverständlich keine Versionen, da diese nur auf nicht zum Intel-Befehlssatz kompatiblen Prozessoren laufen. Allerdings lassen sich durch die Unterstützung von Java bedingt (Java-)Programme für diese Systeme optimieren, allerdings nur auf programmlogischer Ebene, also ohne prozessorspezifische Eigenheiten auszunutzen.
Beide Versionen von VTune lassen sich prinzipiell auf jedem System benutzen. Allerdings bleiben die prozessorspezifischen Optimierungen, die VTune bietet, den CPUs der Firma Intel vorbehalten. Diese werden allerdings vollständig unterstützt: VTune kann Programme sowohl für die Desktop- (Pentium 3, Pentium 4), als auch die Server- (Itanium, Itanium2, Xeon), als auch die Mobilprozessoren (Pentium 4-m, Pentium M) des weltgrößten Prozessorherstellers optimieren. Auch die im Pentium 4 enthaltene Hyper-Threading-Technologie wird unterstützt.
Intel VTune unterstützt die Programmiersprachen C, C++, Java, Assembler und Fortran und lässt sich in Microsofts .NET-Technologie einbinden [Intel].

2. Komponenten von VTune

VTune besteht aus mehreren Komponenten, die verschiedene Sichten auf das Programm bieten und dadurch verschiedene Optimierungsmöglichkeiten aufzeigen. Dieses Kapitel stellt die wichtigsten dieser Komponenten vor und erklärt ihre Funktionsweise. Außerdem wird in Kapitel 2.6 ein Plugin für VTune vorgestellt, durch welches sich Threadfehler in parallel angelegten Programmen auffinden lassen.
Auch wenn in den nachfolgenden Ausführungen teilweise erläutert wird, wie man aus den graphischen Darstellungen in die Quellcodedarstellung gelangt, ist der Quellcode des zu untersuchenden Programms zur Ausführung von VTune nicht unbedingt erforderlich. Einige Komponenten funktionieren auch nur mit den kompilierten Dateien. Zwar kann man dann nicht direkt Optimierungen vornehmen, kann aber den Programmierer über bestehende Hotspots informieren [Intel].

2.1 Intel Tuning Assistant

Die wohl wichtigste Komponente des Programms ist der Intel Tuning Assistant. Er ist in der Lage, ein gegebenes Stück Code zu analysieren und selbständig Optimierungsvorschläge zu erarbeiten. Dies kann er aufgrund der Ergebnisse eines oder auch des Vergleiches mehrerer Testläufe. Hierbei kann der Assistent sowohl Vorschläge zielgerichtet auf einen bestimmten Prozessor machen, wobei hier nur Intel-Prozessoren berücksichtigt werden, als auch die spezifischen Merkmale mehrerer Programmiersprachen berücksichtigen. Die Komponente unterstützt Java, Fortran, Assembler, C und C++ und optimiert Code für sämtliche aktuellen Intel-Prozessoren (Server-, Desktop- und Mobile-Modelle).

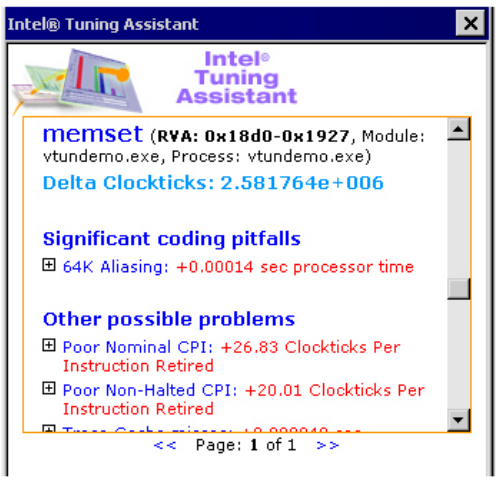

Abbildung 1: Tuning Advice Report für eine Methode

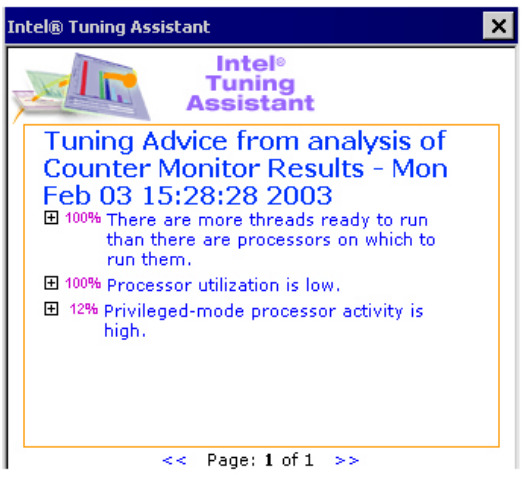

Abbildung 2: systemweiter Tuning Advice Report

Die Ansicht, in der die Erkenntnisse des Tuning Assistant präsentiert werden, nennt sich „Tuning Advice Report". Hier werden zu einzelnen vom Benutzer gewählten Methoden

mögliche Probleme im Code aufgelistet (sogenannte „Insights“) (siehe Abb. 1). Durch einen Klick lassen sich weitere Informationen zu diesem Problem anzeigen, nebst eines Vorschlags zur Lösung des Problems. Weiterhin kann der Benutzer sich eine umfassende Beschreibung des Problems geben lassen, in dem er auf die Beschreibung klickt. Zusätzlich zum Anzeigen eines Problemlösungsvorschlages lässt sich auch vom Tuning Advice Report direkt zur Methode im Quellcode springen. Neben den Insights zu den einzelnen Methoden lässt sich auch aufgrund einer Analyse des in Kapitel 2.4 beschriebenen Counter Monitors eine Übersicht über die programmweiten Probleme geben, nebst einer Sortierung nach Relevanz (siehe Abb 2.) [Tutorial].
Zu den Funktionen des Tuning Assistant gehören neben algorithmischen Verbesserungen wie dem Auffinden redundanter Befehle, ineffizienter Typumwandlungen und zeitaufwendigen Schleifenkonstrukten auch das Auffinden von Stellen, an denen spezielle Intel-Optimierungen wie die Benutzung von Befehlssatzerweiterungen wie MMX, SSE und SSE2 sinnvoll sein könnten. Der Assistent stellt auch Beispiele zur Implementierung dieser Merkmale in den Quellcode zur Verfügung. Zusätzlich erkennt der Tuning Assistant Engpässe, die auf bestimmten Prozessoren auftreten könnten. Ein Beispiel hierfür ist die Verwendung von 64-Bit-Strukturen auf einem 32-Bit-Prozessor [Flash].

2.2 Sampling

Der erste Schritt hin zu einem effizienter laufenden Programm mittels VTune ist das sogenannte Sampling. Hierbei unterscheidet man Time-Based- und Event-Based-Sampling. Beiden Methoden liegt dasselbe Prinzip zu Grunde: der Prozessor wird von Zeit zu Zeit unterbrochen und die Prozess-ID, die Thread-ID sowie der Instruction Pointer werden gespeichert. Beim Time-Based-Sampling geschieht diese Unterbrechung in

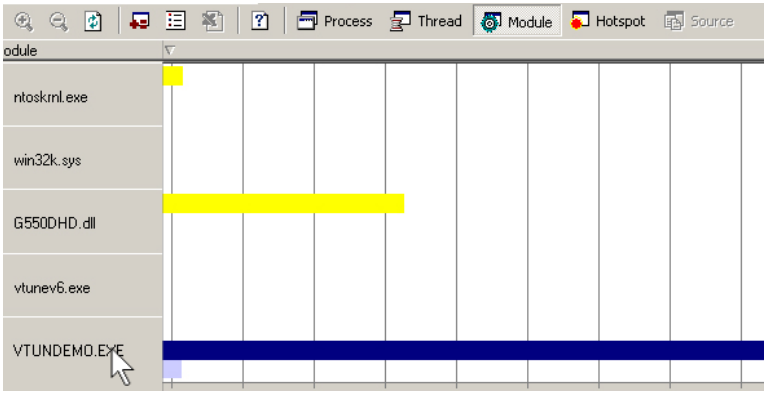

Abbildung 3: Darstellung der Ergebnisse des Sampling im „Modules View“

regelmäßigen Zeitabständen, beim Event-Based-Sampling geben hingegen gibt die Anzahl der ausgeführten Prozessoroperationen den Unterbrechungsrhythmus an.
Die mittels Sampling gesammelten Daten geben Auskunft über die Verteilung der Rechenzeit und helfen somit beim Auffinden von sogenannten „Hotspots“, also besonders

rechenintensiven Programmteilen. Da Sampling systemweit arbeitet beziehen sich die gesammelten Daten nicht nur auf das zu optimierende Programm.

Nach dem Sammelprozess bietet VTune verschiedene Sichten auf die Daten. Alle Sichten bis auf den „Source View" werden als Balkendiagramm realisiert. Neben der einfachen Anzeige des Quellcodes („Source View") stehen ein Überblick über alle während der Aufzeichnungsperiode vorhandenen Prozesse („Process View"), ein Blick in einen Prozess, der die in diesem enthaltenen Threads vergleicht („Thread View") sowie der voreingestellte „Modules View", der eine Sicht auf mehrere Module bietet (siehe Abb. 3). Diese Modelle können ausgewählte Prozesse und Threads sowie weitere Teilkomponenten von diesen Sein. Als letzte Sicht steht der sogenannte „Hotspots View" zur Verfügung (siehe Abb.4). Dieser stellt die einzelnen Methoden/Funktionen/Prozeduren eines

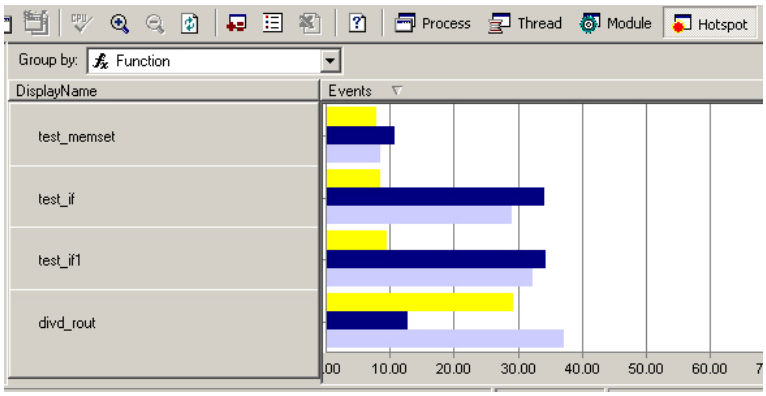

Abbildung 4: Laufzeit einzelner Methoden im „Hotspots View"

Prozesses oder Threads dar. Durch diese Sichten lässt sich mit wenigen Klicks durch immer detaillierte Betrachtung schnell ein Hotspot lokalisieren. Durch einen Doppelklick auf den Balken des Hotspots gelangt man dann an die zugehörige Stelle im „Source View". Nun kann einem der bereits besprochene Intel Tuning Assistant weiterhelfen oder man nimmt eigene Optimierungen vor.

Es ist weiterhin möglich, die gesammelten Samples verschiedenen CPUs in Multiprozessorsystemen zuzuordnen. Dabei wird in der aktuellen Sicht jeder Balken durch so viele ersetzt, wie es Prozessoren im System gibt. Man kann also erkennen, wie die Threads auf die CPUs verteilt wurden. Hier lassen sich auch schwächen im Scheduling-Algorithmus des Systems erkennen.

Der große Vorteil des Sampling ist der geringe Overhead, der durch die Messung entsteht. Es sind keine Änderungen am Quellcode für Debugausgaben oder Ähnliches notwendig. Intel gibt den entstehenden Overhead mit nur etwa einem Prozent der Laufzeit an. Dadurch wird der normale Programmablauf so unverändert wie möglich im Testszenario wiedergespiegelt [Intel][Tutorial].

2.3 Call Graph

Diese Komponente analysiert die Aufrufhierarchie des Programms. Als Ergebnis stellt sie die Methoden eines Programms als Baum dar. Jede Methode ist Kind der Methode, von der aus sie aufgerufen wurde (siehe Abb. 5). Zu den einzelnen Methoden werden Informationen angeboten, wie zum Beispiel die Anzahl der Aufrufe und die Zeit, die die Methode zur Ausführung benötigt hat. Die Kanten in diesem Baum werden verschieden dick dargestellt, je nach der Zeit, die für die aufgerufene Funktion benötigt wurde. Der Weg durch den Baum, der die meiste Zeit benötigt hat, wird „Critical Path" genannt. Dieser wird zusätzlich rot hervorgehoben. Diesem Critical Path sollte das Hauptaugenmerk bei der Optimierung gehören. Weiterhin werden die Methoden (die Knoten des Baumes) je nach Laufzeit eingefärbt. Hierbei ist allerdings nur die „eigene" Laufzeit relevant, also die Zeit, die die Methode benötigt hat, um den eigenen Code auszuführen; die Laufzeit der aufgerufenen Methoden wird hier nicht berücksichtigt. Ist eine Methode grau eingefärbt, bedeutet dies, dass sie kaum Laufzeit benötigt hat, während andere Methoden orange eingefärbt werden. Hier bedeutet eine dunklere Färbung (bis hin zum rot), dass die jeweilige Methode sehr lange zur Ausführung gebraucht hat.

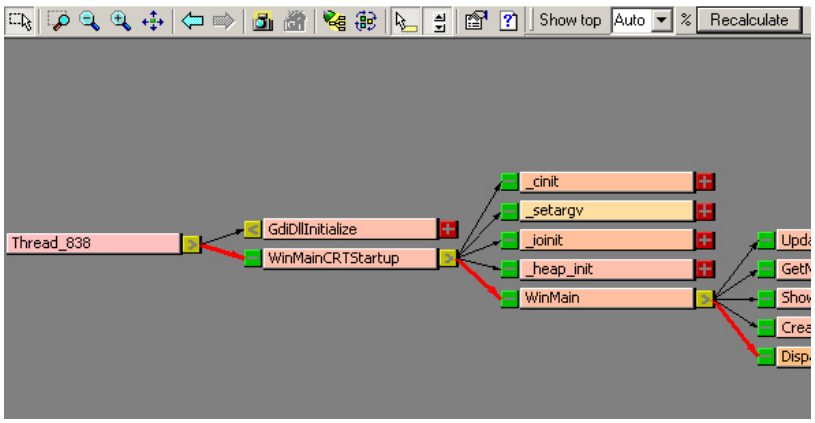

Abbildung 5: die gesammelten Ergebnisse werden als Baum dargestellt

Durch Ziehen der Maus über die einzelnen Elemente des Graphen lassen sich zu den jeweiligen Elementen genaue Informationen abrufen, wie zum Beispiel die Anzahl der Aufrufe oder die Rechenzeit, die diese Methode verbraucht hat.

Mit einem Rechtsklick auf eine Methode kann man direkt in der Quellcode dieser Methode gelangen. Hier kann der eingangs genannte Intel Tuning Assistant, wie auch schon beim Sampling, Vorschläge zur Optimierung des Codes unterbreiten [Tutorial].

Um die beschriebenen Informationen zu erhalten, führt VTune eine modifizierte Version des zu optimierenden Programms durch, bei der vor und nach sämtlichen Methodenaufrufen eine Nachricht an VTune gesendet wird. Aus diesen Daten baut VTune den Baum auf. Die nötigen Modifikationen des Quellcodes werden von VTune automatisch

durchgeführt. Da durch die Programmmodifikationen ein gewisser Overhead entsteht, ist das Testszenario hier nicht mehr so realistisch wie beim Sampling, auch wenn die Zeiten zum Mitprotokollieren der Daten aus den gemessenen Methodenlaufzeiten herausgerechnet werden. Allerdings gewinnt man durch den Call Graph wesentlich mehr Informationen als durch Sampling (s.o.) [Flash].

2.4 Counter Monitor

Jedes Windows-System besitzt eine Vielzahl von Zählern, die den aktuellen Zustand des Systems wiederspiegeln. Unter Windows 2000 zum Beispiel sind das standardmäßig Zähler für die als „ready" markierten Threads in der Prozessor-Queue und die Kontextwechsel pro Sekunde, die zur Verfügung stehenden Bytes im Hauptspeicher, der vom System benötigte Anteil der Prozessorzeit (die sogenannte „Privileged Time) [WinNet] und gesamte Prozessorzeit (also die Auslastung) in Prozent und wie viele Netzwerkfehler pro Sekunde auftreten. Dies wird im Zähler für den Redirector gespeichert und ist relevant, wenn aus dem eigenen Programm ein Client-Rechner angesprochen wird [Microsoft][Tutorial]. Ob und welche Zähler es unter einem Linux-

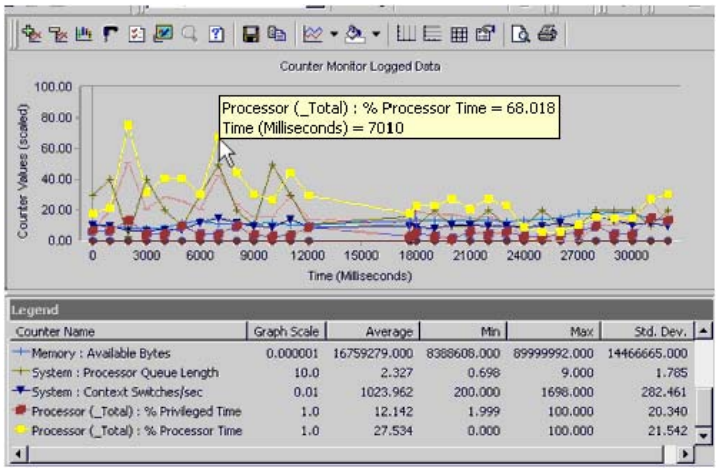

Abbildung 6: Counter Monitor

System gibt, ist hier nicht relevant, da diese Komponente nicht von der Linux-Version unterstützt wird.

Der Counter Monitor arbeitet im Normalfall mit dem oben genannten Sampling zusammen. Wird ein Sample genommen, so werden auch die Werte der Systemzähler abgefragt und gespeichert. Der Verlauf dieser Werte lässt sich nun nach Beenden des Testlaufes in verschiedenen Ansichten darstellen; üblicherweise als Graph, der sämtliche Werte aller Counter enthält (siehe Abb. 6). Durch Überfahren eines der Punkte mit der Maus werden die genauen Werte dieses Zählers zum ausgewählten Zeitpunkt angezeigt. Wie bereits erwähnt, lassen sich die Werte der einzelnen Zähler in verschiedenen Ansichten darstellen, unter anderem als Statistik über die Durchschnittswerte oder in tabellarischer Form.

Mit Hilfe dieser Ansichten lassen sich Stellen im Programm ausfindig machen, in denen bestimmte Bereiche des Systems besonders stark belastet werden, zum Beispiel Stellen, an denen nur wenig Hauptspeicher zur Verfügung steht oder an denen der Prozessor sehr stark in Anspruch genommen wird. Diese Stellen lassen sich markieren, um anschließend z.B. zu den entsprechenden Sampling-Daten zu springen oder sich vom bereits vorgestellten Intel Tuning Assistant Hinweise geben lassen, wie sich die relevanten Stellen im Code verbessern lassen [Tutorial].
Da diese Komponente lediglich bestimmte, ohnehin gemessene Werte des Systems zu bestimmten Zeitpunkten ausliest, wirkt sich diese Art der Analyse, genau wie Sampling, das im Normalfall parallel hierzu eingesetzt wird, so gut wie gar nicht auf den normalen Programmverlauf aus. Auch müssen keine Eingriffe in den Quellcode gemacht werden.

2.5 Remote Data Collection

VTune unterstützt das Sammeln von Daten zu einem Programm auf einem anderen Rechner. Dies bedeutet, dass VTune nicht auf jedem System installiert werden muss, auf dem das Programm getestet werden soll, sofern alle Rechner über ein Netzwerk verbunden sind. Zum Sammeln der Daten muss lediglich ein relativ kleines Programm ohne grafische Benutzeroberfläche auf jedem Rechner installiert werden. Diese Sammelsoftware, die die Daten an den VTune-Rechner sendet, benötigt weitaus weniger Ressourcen als das komplette VTune, belegt also auch weniger Hauptspeicher. Dadurch wird das Testszenario noch realistischer als wenn die komplette VTune-Software im Speicher liegen würde. Die Ressourcenersparnis kommt hauptsächlich durch die nicht vorhandene (aber auch nicht notwendige) GUI der Sammelsoftware.
Bevor Intel in diesem Jahr eine Version von VTune für Linux auf dem Markt brachte, wurde Remote Data Collection genutzt, um Programme auf Linux-Umgebungen zu testen. Dies heißt aber nicht, dass per Remote Data Collection sämtliche Betriebssysteme unterstützt werden, da für Linux eine spezielle Sammelsoftware zur Verfügung stand.
Für Remote Data Collection muss eine TCP/IP-Verbindung zwischen den beiden Rechnern bestehen. Außerdem lassen sich über diesen Weg ausschließlich Event-Based-Sampling sowie Call Graph durchführen [HOC1].

2.6 Intel Thread Checker

Der „Thread Checker" ist standardmäßig nicht in VTune integriert, sondern ist ein kostenpflichtiges Plugin, welches ebenfalls von der Firma Intel stammt. Die Software erweitert die Funktionalität von VTune um die Möglichkeit, sogenannte Threadfehler aufzufinden sowie um die Möglichkeit mit OpenMP-Pragmas umzugehen.
Threadfehler sind Fehler, die durch fehlende oder fehlerhafte Synchronisation von Threads entstehen können. Hierzu zählen zum Beispiel Race Conditions (mehrere Threads greifen in undefinierter Reihenfolge auf eine Speicherstelle zu) oder auch

Deadlocks (alle Threads befinden sich in einem Zustand, in dem sie auf eine Aktion eines anderen Threads warten). Die Schwierigkeit beim Auffinden dieser Fehler liegt darin, dass sie nicht immer auftreten. Ein fehlerhaftes multithreaded Programm kann durchaus wie vorgesehen ablaufen, nämlich dann, wenn die Zugriffe der Threads auf die gemeinsamen Speicherstellen zufällig in der vom Entwickler gewollten Reihenfolge ablaufen. Es ist aber nicht garantiert, dass beim nächsten Start die Zugriffsreihenfolge identisch sein wird.

Der „Thread Checker" analysiert die Threads eines Programms im Hinblick auf ihre Zugriffe auf gemeinsame Speicherstellen und zeigt mögliche Fehler auf. Zu jedem gefundenen Fehler werden die verursachende Variable sowie ihre Benutzung in den einzelnen Threads aufgezeigt.

Im Gegensatz zu den bisher vorgestellten Komponenten ist das primäre Ziel des „Thread Checkers" nicht die Steigerung der Ablaufgeschwindigkeit des untersuchten Programms, sondern die Optimierung hinsichtlich der fehlerfreien Zusammenarbeit der einzelnen Threads. An dieser Komponente zeigt sich, dass der Begriff Softwareoptimierung neben der Geschwindigkeit auch noch andere Aspekte wie eben die Fehlerfreiheit beinhalten kann.

Im Lieferumfang des „Thread Checkers" enthalten ist ein weiteres Plugin namens „Thread Profile". Dieses arbeitet nach demselben Prinzip wie Sampling und stellt die verbrauchte CPU-Zeit für jeden Thread einer OpenMP-basierten Anwendung graphisch dar.

Der „Thread Checker" liegt zur Zeit in der Version 1.0 für Microsoft Windows vor, die Version 2.0 befindet sich im Beta-Status [Thread][HOC2].

3. Grenzen der Software

Trotz seiner recht umfangreichen Funktionen hat auch VTune seine Grenzen. Dieses Kapitel soll diese aufzeigen und somit den Anwendungsrahmen für die Software eingrenzen.

Die wohl wichtigste Grenze der Software ist die Beschränkung auf Prozessoren der Firma Intel (und damit auf Prozessoren des Herstellers von VTune). Zwar lässt sich die Software auch auf anderen Systemen starten und auch Tests ausführen, allerdings stehen dort nicht alle Komponenten zur Verfügung und die Optimierung verläuft aufgrund der Unkenntnis der Software über die prozessorinternen Abläufe nicht so effizient wie auf Intel-Systemen, auf die die Software abgestimmt ist. Optimierung hinsichtlich prozessorspezifischer Befehlssatzerweiterungen wie SSE, SSE2 und MMX durch VTune ist also Intel-Prozessoren vorbehalten. Alternativen zu VTune für andere Prozessorarchitekturen zeigt das Kapitel „Konkurrenzprodukte" auf.

Eine weitere Grenze ist die Beschränkung auf bestimmte Programmiersprachen und Betriebssysteme, die bereits in Kapitel 1 aufgezeigt wurde. Der Intel Tuning Assistant kann nur Programme, die in einer der in Kapitel 1 aufgezählten Programmiersprachen geschrieben wurden, analysieren und demzufolge auch nur für diese Programme Verbesserungsvorschläge anbieten. Intel erweitert die Palette bekannter Programmiersprachen allerdings mit der Zeit immer weiter. So ist in der aktuellen Version 7 die Unterstützung der .NET-Welt von Microsoft hinzugefügt worden [Intel].

Die Beschränkung auf Microsoft-Windows-Betriebssysteme ist in diesem Jahr durch Intel aufgehoben worden. Mit der Einführung von VTune für Linux steht nun auch für den UNIX-Abkömmling eine (fast) gleichwertige Version von VTune zur Verfügung, wodurch der manchmal mühsame Umweg über die Remote Data Collection entfällt. Durch diesen Schritt ist VTune nun für die beiden auf Intel-Systemen am weitesten verbreiteten Betriebssysteme verfügbar [Intel].

Als Fazit lässt sich festhalten, dass Intel bemüht ist, die aufgezeigten Grenzen von VTune abzubauen und das Programm so vielfältig einsetzbar wie möglich zu machen. Ausnahme hiervon ist die Beschränkung auf Intel-Prozessoren. Diese ist zwar verständlich, da VTune die Leistung der hauseigenen Prozessoren optimieren soll und Intel die Konkurrenz nicht unterstützen will, allerdings wird der Anwender, will er VTune verwenden, einer Beschränkung hinsichtlich seiner Systeme unterworfen, die rein softwaretechnisch gesehen eigentlich nicht erforderlich ist.

4. Kurzer Überblick über Konkurrenzprodukte

Wie bereits erwähnt, ist VTune ausschließlich für Intel-Prozessoren entwickelt worden und daher nicht in der Lage, Software auf andere Architekturen hin mit der Effizienz zu optimieren, wie sie es für Intel-CPUs kann. Diese Eingrenzung ist mit einem Seitenblick auf den Hersteller natürlich verständlich. Für den Entwickler heißt dies aber, dass er, möchte er seine Software bezüglich Geschwindigkeit auf Prozessoren anderer Hersteller hin optimieren, auf Alternativen zurückgreifen muss. Dieses Kapitel soll einen kurzen Überblick über diese Alternativen bieten.

Neben Intel stellt auch der weltweit zweitgrößte Prozessorhersteller, AMD, für seine Prozessoren eine eigene Optimierungssoftware bereit: den „AMD CodeAnalyst". Im Gegensatz zu VTune ist dieser kostenlos, ist aber, genauso wie VTune, auf Prozessoren des Herstellers beschränkt. Der „CodeAnalyst" ist sowohl für Windows 2000 und XP als auch für Linux aktuell in der Version 2.1 verfügbar und lässt sich sowohl auf AMD x86-Prozessoren als auch auf die neue AMD64-Architektur anwenden. Seine Funktionen umfassen Timeline-Profiling, Event-Based-Profiling sowie Pipeline Simulation. Das Profiling entspricht hierbei weitestgehend dem Sampling bei Intel VTune. Insgesamt ist der Funktionsumfang des AMD-Produkts eine Untermenge des Funktionsumfang des Intel-Konkurrenzprodukts. Erwähnenswert ist, dass auch die AMD-Software Multiprozessorsysteme (mit bis zu acht CPUs) unterstützt. Als Fazit kann man festhalten, dass der AMD Code Analyst als Pendant zu VTune für die AMD-Welt gedacht ist, er allerdings noch nicht an die Funktionsvielfalt und -tiefe des Vorbilds heranreicht [AMD].

Die Firma Apple hat mit der Einführung des G5-Prozessors eine neue Software namens „XCode" vorgestellt. Sie vereint eine Entwicklungsumgebung, ein Paket verschiedener Compiler, diverse Entwicklungstools für die Kommandozeile sowie auch einige Tools zur Softwareoptimierung in einem Paket. Es finden sich einige Programme, die Komponenten von VTune ähnlich sind, so zum Beispiel „Sampler", das ähnlich wie CallGraph arbeitet und „Shark", dass man entfernt mit dem Intel Tuning Assistant in Verbindung bringen könnte. Tools zur Softwareoptimierung bilden allerdings, wie gesagt, nur einen kleinen Teil in „XCode". Einen Funktionsumfang, wie VTune ihn bietet, darf man wohl nicht erwarten [Apple].

Als weiteres Tool soll hier „Morph" vorgestellt werden. „Morph" ist ein Projekt an der Harvard University. Die Entwickler versuchen eine Performancesteigerung durch sogenannte „Late Code Modification" zu erzielen. Dies bedeutet, dass das Tool erst eingreift, nachdem das zu optimierende Programm übersetzt wurde. „Morph" schlägt also einen anderen Weg ein als VTune und richtet sich daher nicht an exakt dieselbe Zielgruppe. Über den Funktionsumfang von „Morph" konnten wir leider nichts Konkretes herausfinden [Morph].

Abschließend soll hier noch ein Verweis auf die anderen Ausarbeitungen dieses Seminars erfolgen. Diese Arbeiten beschäftigen sich mit weiteren Optimierungstools wie zum Beispiel „gcov", „gprof", JVMPI und Hpjmeter [berre]. Auch diese Tools sind Konkurrenzprodukte zu VTune, arbeiten allerdings weitestgehend rein textbasiert und bieten daher nicht die graphischen Auswertungsmöglichkeiten wie VTune.

Zusammenfassend kann festgehalten werden, dass es zwar eine Vielzahl an Optimierungstools auf dem Markt gibt, die zum Teil sogar frei erhältlich sind, VTune aber wohl in Sachen Funktionsumfang und besonders hinsichtlich der Intuitivität der Bedienung (graphische Auswertung) einen der vordersten Plätze einnimmt. Entwickler, die für andere Systeme als solche mit Intel-Prozessoren eine Optimierungssoftware suchen, können VTune durchaus für prozessorunabhängige Optimierungen einsetzen, sollten sie die Vorteile der Software nutzen wollen. Dieses Kapitel zeigt aber auch, dass es durchaus eine Reihe an Alternativen gibt.

5. Zusammenfassung und Bewertung von VTune

Dieses Kapitel fasst die Erkenntnisse dieser Seminararbeit kurz zusammen und versucht, VTune hinsichtlich Funktionsumfang und Bedienbarkeit zu bewerten.
VTune umfasst eine Vielzahl von Komponenten, die durch unterschiedliche Methoden eine große Anzahl an Informationen über ein Programm liefern können. Aufgrund dieser Datenbasis lassen sich direkt im Programm Optimierungen am Quellcode vornehmen. Als zentrale Komponente gibt der Intel Tuning Assistant Tipps zur Optimierung gefundener Hotspots.
Im Gegensatz zu vielen anderen Optimierungstools präsentiert sich VTune also als vollständige Optimierungssoftware, die mehrere Untersuchungsverfahren sowie die anschließende Optimierung von gefundenen Hotspots integriert. Der Anwender muss während des gesamten Optimierungsverfahrens keine andere Software benutzen oder auf die Konsole zurückgreifen.
Besonders hervorzuheben ist der Intel Tuning Assistant. Diese Programmkomponente unterstützt den Entwickler bei der Optimierung und kann dadurch die Qualität der Ergebnisse verbessern sowie den Aufwand der Optimierung verringern. Der Assistent macht den Hauptunterschied zu anderen, kostenlosen, Produkten aus, die meist ähnliche oder gleiche Funktionen wie VTune anbieten (wobei hier die Darreichungsform meist etwas komplizierter ausfällt), da der Entwickler bei diesen Produkten eigene Schlüsse aus den gewonnen Ergebnissen ziehen muss. Dies ist ohne tiefgehende Kenntnisse z.B. des Prozessors fast unmöglich, auch geschieht es leicht, dass einige nicht ganz offensichtliche Problemstellen nicht erkannt werden.
Im Kontext der Vorlesung „Verteilte und parallele Systeme 2" ist die Unterstützung von Multiprozessorsystemen erwähnenswert. Diese kann dem Entwickler helfen, herauszufinden, wie viele Prozessoren für seine Anwendung sinnvoll sind. Die Möglichkeit, durch ein (kostenpflichtiges) Plugin Unterstützung bei der Suche nach Threadfehlern zu erhalten, zeigt, dass die VTune-Umgebung neben der klassischen Leistungsoptimierung auch für andere Aspekte sinnvoll erweiterbar ist.
Die Lösung als Gesamtpaket steht, im Gegenteil zu vielen anderen Optimierungstools, nicht umsonst zur Verfügung. Somit eignet sich die Software nur für Unternehmen, die häufig mit der Optimierung von Software zu tun haben und für die sich die Investition dadurch rentieren kann. Für den Hobbyprogrammierer ist VTune sicherlich viel zu überdimensioniert.
Als abschließendes Fazit kann festgehalten werden, dass die Firma Intel mit ihrer Optimierungssoftware ein sowohl leistungsstarkes und umfangreiches als auch, dank der graphischen Oberfläche, übersichtliches und relativ leicht verstehbares Softwareoptimierungsprogramm vertreibt.

6. Literatur

- [AMD] AMD Inc., http://www.amd.com/us-en/Processors/DevelopWithAMD/0,,30_2252_3604,00.html. Web-Site des Herstellers zum Produkt „AMD Code Analyst".
- [Apple] Apple Computer Inc., http://developer.apple.com/tools/. Web-Site des Herstellers zum Produkt „Apple XCode".
- [Berre] Berrendorf, Prof. Dr. R., http://www2.inf.fh-bonn-rhein-sieg.de/~rberre2m/ lehre/ws0304/vups2/vups2.html. Web-Site zum Seminar, in dessen Rahmen diese Arbeit entstand.
- [Morph] Chen, J. Bradley, The Morph Project, http://www.eecs.harvard.edu/morph.
- [Flash] Intel Corporation, http://www.intel.com/software/products/vtune/downloads/VTune_V6.htm. Flashpräsentation zu "VTune".
- [Tutorial] Intel Corporation, https://shale.intel.com/SoftwareCollege/CourseDetails.asp?courseID=17. Tutorial zu „VTune".
- [Intel] Intel Corporation, http://www.intel.com/software/products/vtune/. Web-Site des Herstellers zum Produkt „VTune".
- [Thread] Intel Corporation, http://www.intel.com/software/products/threading/tcwin/. Web-Site des Herstellers zum Produkt „ThreadChecker".
- [HOC1] ho-Computer Software GmbH, www.hocomputer.de/vtune_performance_analyzer.htm. Web-Site eines deutschen Intel-Vertriebspartners zum Produkt „VTune".
- [HOC2] ho-Computer Software GmbH, http://www.hocomputer.de/body_thread_checker.htm. Web-Site eines deutschen Intel-Vertriebspartners zum Produkt „ThreadChecker".
- [Microsoft] Microsoft Corporation, Monitoring Network Activity, http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/winxppro/proddocs/SAG_MPmonperf_21.asp.
- [WinNet] Windows Network & .NET Magazine, More Windows NT Performance Monitor, http://www.winnetmag.com/Articles/ArticleID/280/pg/2/2.html.

7. Abbildungsverzeichnis

- Abbildungen 3, 4 und 5: Intel Corporation, http://www.intel.com/software/products/vtune/downloads/VTune_V6.htm. Flashpräsentation zu „VTune".
- Abbildungen 1, 2 und 6: Intel Corporation, https://shale.intel.com/SoftwareCollege/CourseDetails.asp?courseID=17. Tutorial des Herstellers zu VTune.