

Überlegungen zur Entwicklung komplexer Grid-Anwendungen mit Globus Toolkit

Andreas Walter¹, Klemens Böhm², Stephan Schosser²

¹FZI Forschungszentrum Informatik, IPE, Haid-und-Neu-Straße 10-14, 76131 Karlsruhe
awalter@fzi.de

²Universität Karlsruhe (TH), IPD, Am Fasanengarten 5, 76131 Karlsruhe
{boehm, schosser}@ipd.uka.de

Abstract: Verteilte Anwendungen mit einem hohen Ressourcenbedarf lassen sich heutzutage mit Hilfe von Grid-Techniken ohne den Einsatz großer Rechenzentren erstellen. Stattdessen nutzt man freie Ressourcen auf Computern, die im Internet verteilt sind. Grid Middleware ist hierfür eine Grundlage. Hauptinteresse dieser Arbeit ist die Analyse der am weitesten verbreiteten Middleware für Grid-Anwendungen, dem Globus Toolkit. Uns geht es insbesondere um die Erstellung von komplexen Anwendungen mit hohem Ressourcenbedarf. Für unsere Untersuchung haben wir als geeignete Anwendung Web-Crawling ausgewählt, aufgrund eines sehr hohen Daten- und Kontrollaufwands über viele verteilt arbeitende Knoten. Wir zeigen, welche Anforderungen dabei an Globus Toolkit gestellt werden, und wie gut sich diese Plattform für solche Anwendungen eignet.

1 Einleitung

Existierende Grid-Anwendungen haben einen großen Bedarf an Rechenleistung, oder sie dienen zur Übertragung von großen Datenmengen [FK03] zwischen entfernten Standorten. Sie nutzen überschüssige Ressourcen auf Computern, die im Internet verteilt sind. Dies führt zu einer Reihe von neuen Anforderungen. Grid-Middleware vereinfacht die Erstellung eines "Grids" für verteilte Anwendungen. Sie stellt Komponenten zur Koordination, Nutzung und Verwaltung von Ressourcen in einem Grid bereit. Fokus dieser Arbeit ist die Grid-Middleware Globus Toolkit¹, der de-facto Standard zur Erstellung von Grid-Anwendungen. Globus Toolkit hilft bei der Erstellung von Grid-Anwendungen in zweierlei Hinsicht: Es stellt sowohl oft benötigte Funktionen auf der Anwenderschicht, zum Beispiel eine Ressourcen-Verwaltung, zur Verfügung, wie auch betriebs-systemspezifische Funktionen, zum Beispiel Sicherheitsfunktionalität.

Die Steuerkomponenten von Globus Toolkit haben die Aufgabe, die anfallende Arbeitslast gleichmäßig auf die vorhandenen Ressourcen in einem Grid durch Vergabe von Arbeitsaufträgen (Jobs) zu verteilen. Die hierbei vergebenen Jobs an Knoten des Grids enthalten meist einfache, aber zeitaufwendige Berechnungsaufgaben. In diesem Papier untersuchen wir, wie gut diese Komponenten die Erstellung von Anwendungen unterstützen, die eine Vielzahl von Ressourcen benötigen und aus komplexen Jobs bestehen.

¹ Globus Toolkit; <http://www.globus.org>

Wir untersuchen exemplarisch, wie gut Globus Toolkit zum Betrieb eines verteilt arbeitenden Web-Crawlers geeignet ist. Eine Web-Crawler Anwendung basierend auf Globus Toolkit ist im Vergleich zu anderen Grid-Anwendungen sehr komplex: Ein Crawler lädt Webseiten aus dem Internet und benötigt hierfür viel Bandbreite und Speicherplatz. Zusätzlich ist ein Kontrollsystem erforderlich, das eine Liste von zu ladenden Webseiten verwalten kann. Ein Crawler fragt bei diesem Kontrollsystem nach solchen Webseiten (Jobs) an. Das Kontrollsystem muss dafür sorgen, dass einzelne Webseiten nur einmal geladen werden. Insgesamt führen diese Anforderungen zu einem erheblichen Koordinationsaufwand und zu einem hohen Ressourcenbedarf. Anhand einer geeigneten Architektur für verteilte Web-Crawler werden wir prüfen, ob sich Globus Toolkit als Grundlage für die Erstellung einer solchen Anwendung eignet, um daraus allgemeine Rückschlüsse für die Erstellung komplexer Anwendungen mit Globus Toolkit zu ziehen.

Kapitel 2 gibt einen Überblick über die Komponenten von Globus Toolkit. In Kapitel 3 beschreiben wir, basierend auf verwandten Arbeiten, vorhandene Erweiterungen für Globus Toolkit, die geeignet sind, um umfangreiche Grid-Anwendungen zu erstellen. Eine geeignete Referenzarchitektur für verteilte Web-Crawler stellen wir in Kapitel 4 vor. Diese Architektur wird verwendet, um Globus Toolkit für die Erstellung komplexer Grid-Anwendungen zu bewerten. In Kapitel 5 wird Globus Toolkit im Hinblick auf die Einsetzbarkeit zum Betrieb von Web-Crawlern evaluiert. Kapitel 6 fasst die Ergebnisse zusammen und gibt einen Ausblick.

2 Globus Toolkit

Grids ermöglichen die Erstellung von verteilten Anwendungen (Grid-Anwendungen) mit hohem Ressourcenbedarf, wie zum Beispiel Rechenleistung, ohne hierfür große Rechenzentren betreiben zu müssen. Stattdessen möchte man überschüssige Ressourcen auf Rechnern nutzen, die im Internet verteilt sind. Teilnehmer stellen für solche Grid-Anwendungen freiwillig Ressourcen wie CPU-Leistung oder Speicherplatz zur Verfügung. Da die Knoten einer Grid-Anwendung über das Internet verteilt sind, entstehen hierbei eine Reihe von weiteren Anforderungen im Vergleich zu einer verteilten Anwendung in einem lokalen Netzwerk. Grid-Middleware Techniken sollen die Erstellung solcher Anwendungen vereinfachen und stellen hierzu eine Reihe von Komponenten bereit, die häufig in einer Grid-Umgebung benötigt werden. Im Folgenden werden die wichtigsten Anforderungen und ihre Umsetzung in Globus Toolkit, dem de-facto Standard im Umfeld von Grid-Middleware Techniken [FK01], beschrieben.

2.1 Anforderungen an Grid-Middleware

Globus Toolkit soll Teilnehmern einer Grid-Anwendung das sichere Verfügbarmachen von Ressourcen wie zum Beispiel Rechenleistung, Datenbanken und Speicherplätzen über das Internet ermöglichen. Die Hauptanforderung der Teilnehmer ist dabei, dass die lokale Kontrolle über die zur Verfügung gestellten Ressourcen nicht verloren geht. Foster et al. [FK02, FK01] haben diese Anforderungen und ihre Umsetzung durch Standardkomponenten in Globus Toolkit wie folgt definiert:

- **Sicherheitsanforderungen:** Die Sicherheitsanforderungen umfassen die Authentifizierung und Autorisierung von zugangsberechtigten Nutzern eines Grids sowie die Verschlüsselung sämtlicher über das Internet übertragenen Daten. Globus Toolkit stellt hierfür ein Sicherheitscenter zur Erstellung von SSL-Zertifikaten für berechnete Teilnehmer bereit [Th99].
- **Fairness:** Mit Fairness meint man die Anforderung, alle Anfragen an eine Grid-Anwendung gerecht und somit gleichmäßig über die vorhandenen Ressourcen verteilen zu können. In Globus Toolkit gibt es hierfür eine Steuerkomponente „Resource Management“, die diese Anforderung erfüllt.
- **Kontrolle:** Die Teilnehmer eines Grids sollen bestimmen können, wie umfangreich die Ressourcen, die sie zur Verfügung gestellt haben, genutzt werden dürfen. Hierzu enthält Globus Toolkit die Komponente „Resource Allocation“. Mit der Komponente „Monitoring“ können die Teilnehmer zusätzlich beobachten, wie intensiv ihre Ressourcen im Grid genutzt werden.
- **Flexibilität:** Der Inhaber einer Ressource kann die damit verbundenen Systeme zu jeder Zeit ein- bzw. ausschalten. Eine Grid-Anwendung muss somit flexibel mit wegfallenden und hinzukommenden Ressourcen umgehen können. Hierfür stellt Globus Toolkit die Komponente „Directory“, also ein Verzeichnis, zur Verfügung. Diese Komponente listet alle aktuell vorhandenen Ressourcen auf.
- **Einheitliche Laufzeitumgebung:** Die Entwicklung von Grid-Anwendungen erfordert die Unterstützung von Programmiersprachen und Bibliotheken. Globus Toolkit ermöglicht die Nutzung der Programmiersprachen JAVA, C und Python und stellt entsprechende Bibliotheken bereit.
- **Heterogenität:** Ein Grid besteht mit steigender Größe aus einer Vielzahl von heterogenen Ressourcen. Die Anforderung, die sich hierbei stellt, ist, dass den Nutzern des Grids und den Grid-Anwendungen diese Heterogenität soweit wie möglich verborgen bleibt. Um dies zu ermöglichen, erlaubt Globus Toolkit einen dienstbasierten Ansatz zur Entwicklung von Komponenten. Dienste in Globus Toolkit heißen Grid-Services, basieren auf XML-Standards [Gu06, Ch06] und können zusätzlich die gerade beschriebenen Standardkomponenten von Globus Toolkit nutzen.

Neben der Sicherheitsinfrastruktur sind Grid-Services „das wichtigste Konzept in Globus Toolkit“ [Fo05].

2.2 Rolle und Chancen von Grid-Middleware in der Industrie

In diesem Abschnitt gehen wir kurz auf die aktuellen und zukünftigen Möglichkeiten von Grid-Middleware in der Industrie ein. Das Ausnutzen von Rechenleistung auf Rechnern freiwilliger Teilnehmer ermöglicht umfangreiche kostengünstige Berechnungen [GS02]. Hierzu können Firmen ihre vorhandenen Systeme als Grid zusammenschließen und gemeinsam umfangreiche Berechnungen durchführen. So genannte „Access Grids“,

wie zum Beispiel das vom amerikanischen Bundesstaat Louisiana geförderte „Louisiana Grid“², sollen auch kleineren und mittelständischen Unternehmen ermöglichen, kostengünstig Berechnungen durchzuführen. Für eine bestimmte Zeit können Ressourcen angemietet und genutzt werden. In Zukunft können Grids auch neue Einnahmequellen für Unternehmen darstellen. Projekte wie Sorma³ setzen sich zum Ziel, in einer selbstorganisierten und automatisierten Auktionsplattform Ressourcen zu versteigern. Unternehmen können dann Ressourcen in Zeiten geringer Auslastung, zum Beispiel nachts oder an Wochenenden, an andere Unternehmen vermieten und somit zusätzliche Einnahmen generieren.

3 Verwandte Arbeiten

Hier stellen wir zuerst Erweiterungen der Globus Toolkit Plattform vor, die bei der Entwicklung von komplexen Grid-Anwendungen hilfreich sein können, und anschließend einige interessante Grid-Anwendungen, die auf Globus Toolkit basieren.

3.1 Erweiterungen für Globus Toolkit

Aktuelle Grid-Anwendungen beschäftigen sich hauptsächlich mit der Verwaltung und Zuteilung von Rechenleistung. Unser Hauptaugenmerk bezüglich der im Folgenden vorgestellten Erweiterungen gilt der Einsetzbarkeit im Umfeld komplexer Anwendungen, speziell zur Erstellung eines gridbasierten Web-Crawlers. Ein Web-Crawler benötigt hauptsächlich die Ressourcen Bandbreite und Speicherplatz. Dies erfordert entsprechende Steuerkomponenten in einer solchen Grid-Anwendung, die mit diesen Ressourcen umgehen können.

Die Standardkomponenten für die Steuerung von Ressourcen in Globus Toolkit sind nicht besonders flexibel. Sie sind nicht geeignet zur Verwaltung von ganzen Clustern von Computersystemen. Ferner ermöglichen sie nicht die Definition von komplexen Zuteilungsregeln für Ressourcen [Fo05].

Für diese Zwecke gibt es die Erweiterungen *Condor*⁴ und *Sun Grid Engine*⁵. *Condor* ermöglicht die Verwaltung und die Zuteilung von Jobs innerhalb ganzer Computer-Cluster. Die *Sun Grid Engine* erweitert die Ressourcenverwaltung in Globus Toolkit um die Möglichkeit, Limits für die Nutzung von Rechenzeit einzelner Systeme basierend auf Budgets zu definieren. Für einen Web-Crawler wäre es zusätzlich erforderlich, dass eine Steuerkomponente auch die Ressource Bandbreite, zum Beispiel den Verbrauch insgesamt oder die maximale Download-Rate, limitieren könnte.

² <http://dmrl.latech.edu/Access-Grid/>

³ Sorma-Projekt - www.sorma-project.org

⁴ Condor – High Throughput Computing; <http://www.cs.wisc.edu/condor>

⁵ Sun N1 Grid Engine, <http://www.sun.com/software/gridware/>

Die Middleware *OGSA DAI*⁶ ermöglicht die Integration von heterogenen Datenbanksystemen in ein Grid. Diese kann man, beispielsweise zur Speicherung von Daten, die Web-Crawler liefern, einheitlich in ein Grid integrieren.

3.2 Grid-Anwendungen basierend auf Globus Toolkit

Im Folgenden gehen wir auf zwei interessante Grid-Anwendungen ein, die mit Hilfe von Globus Toolkit erstellt wurden. In beiden Fällen handelt es sich, wie auch bei einer Web-Crawler Anwendung, um Anwendungen mit sehr hohem Ressourcenbedarf. Der Einsatz von Globus Toolkit hat jeweils geholfen, diesen Ressourcenbedarf durch Nutzung überschüssiger Ressourcen auf anderen Rechnern abzudecken.

Virtual Screening on a Desktop [Fo03] ist eine Grid-Anwendung mit einem hohen Bedarf an Rechenleistung. Hierzu nutzt die Anwendung Rechenleistung auf Desktop-Computern, um die Struktur von Molekülen zu berechnen. Zur Verwaltung und Zuteilung von Jobs wurde *Condor* eingesetzt. Innerhalb der durchgeführten Evaluierung wurde hierdurch eine 72-fache Beschleunigung der Gesamtrechenleistung auf 94 im Internet verteilt liegenden Desktop-Rechnern, im Vergleich zu einem einzelnen Rechner, erreicht.

Eine erfolgreiche Grid-Lösung zum Umgang mit sehr großen Datenbeständen und der Integration von heterogenen Ressourcen ist die Grid-Anwendung *The World Wide Desktop* [GS02]. Dort wurden mit Hilfe von Globus Toolkit fünfzig Institutionen aus dem Bereich der Astronomie in ein Grid zur Analyse von astronomischen Daten integriert.

4 Referenzarchitektur eines verteilt arbeitenden Web Crawlers

Das World Wide Web besteht aus Milliarden sich häufig ändernder Web-Seiten [SS06]. Eine leistungsfähige Crawlerarchitektur sollte daher Web-Seiten so schnell wie möglich verarbeiten und einzelne Webseiten regelmäßig wieder besuchen können. Grosse Rechenzentren, beispielsweise von Google, erfüllen diese Anforderungen [BP98], der Betriebsaufwand ist aber sehr hoch. Eine Grid-basierte Web-Crawler Anwendung könnte diese Anforderungen ebenfalls erfüllen und gleichzeitig auf den teureren und aufwendigen Betrieb von Rechenzentren verzichten, da eine steigende Anzahl von Web-Crawlern in einem Grid die Verarbeitung von mehr Web-Seiten pro Zeiteinheit ermöglicht.

4.1 Referenz-Architektur für verteilte Web-Crawler

Abbildung 1 zeigt die Referenzarchitektur für einen verteilten Web-Crawler, die Shkapnyuk und Suel vorgeschlagen haben [SS02]. Web-Crawler kontaktieren Web-Server im Internet und laden die dort vorhandenen Web-Seiten. Anschließend übertragen die Web-Crawler diese Web-Seiten zur Speicherung an Speicherplätze.

⁶ The OGSA-DAI Project, <http://www.ogsadai.org.uk>

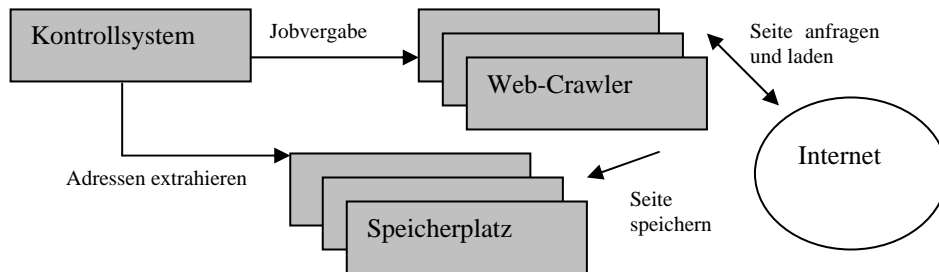


Abbildung 1: Komponentenansicht eines Crawlersystems, vgl. [SS02]

Das Kontrollsystem hat zwei wichtige und gleichzeitig sehr umfangreiche Aufgaben. Zum einen muss es neue Adressen zu Web-Seiten aus den vorhandenen Web-Seiten in den Speicherplätzen extrahieren und bereits bearbeitete Web-Seiten filtern, um das mehrfache Verarbeiten gleicher Web-Seiten in kurzer Zeit zu vermeiden. Die gefilterten Adressen müssen dann als Jobs an die vorhandenen Web-Crawler weitergegeben werden. Hierbei muss das Kontrollsystem dafür sorgen, dass die Jobs gleichmäßig an die Web-Crawler weitergegeben werden. Ansonsten kann es zur Überlastung einiger Web-Crawler kommen. In ähnlicher Weise muss das Kontrollsystem dafür sorgen, dass die Web-Crawler die geladenen Web-Seiten gleichmäßig zur Speicherung an die vorhandenen Speicherplätze weitergibt. Beide Aufgaben zusammen erfordern somit ein Kontrollsystem, das die Erstellung neuer Jobs und die anschließende komplexe Jobzuteilung durchführen kann.

Shkapenyuk und Suel haben diese Referenzarchitektur innerhalb eines lokalen Netzwerks installiert. Während der Evaluation sahen sie als Engpässe hauptsächlich die Ressourcen Bandbreite, die die acht eingesetzten Web-Crawler für den Download von Web-Seiten benötigen, und die Ressource Speicherplatz, die die Speicherplätze für das Ablegen der Web-Seiten benötigen. Das Kontrollsystem benötigte hauptsächlich Rechenleistung zur Extraktion neuer Adressen sowie Hauptspeicher, um über bereits bearbeitete Web-Seiten Buch führen zu können.

4.2 Umsetzung der Referenzarchitektur mit Globus Toolkit

Diese Referenzarchitektur für verteilte Web-Crawler stellt, wie eben beschrieben, umfangreiche Anforderungen an ein Kontrollsystem. Möchte man die Referenzarchitektur mit Hilfe von Globus Toolkit auf Basis eines Grids umsetzen, liegt es nahe, möglichst viele der vorhandenen Standardkomponenten von Globus Toolkit wieder zu verwenden. Auf den Rechnern von freiwilligen Teilnehmern können dann die Speicherplätze und Web-Crawler betrieben werden. Weiterhin wird ein zentrales Kontrollsystem benötigt. Die Sicherheitskomponenten von Globus Toolkit ermöglichen die sichere Kommunikation, Authentifizierung und Autorisierung zwischen allen vorhandenen Komponenten. Für diese Anforderungen sind also keine weiteren Eigenentwicklungen nötig. Flexibilität im Hinblick auf hinzukommende und wegfallende Komponenten ist mit Hilfe der Verzeichnis-Komponente von Globus Toolkit ebenfalls gegeben. Dadurch kann das Kon-

trollsystem immer die aktuell vorhandene Menge an Web-Crawlern und Speicherplätzen im Grid ermitteln.

Mit den Steuer- und Kontrollkomponenten von Globus Toolkit, ebenso mit den im Kapitel "Verwandte Arbeiten" beschriebenen Erweiterungen, ist es allerdings nicht möglich, (a) die Grid-Anforderungen Fairness und Kontrolle zu ermöglichen und (b) das Kontrollsystem für einen Web-Crawler zu realisieren. Zwar ermöglichen diese Komponenten die Verwaltung und Zuteilung von Rechenleistung und den Teilnehmern die Kontrolle der freigegebenen Rechenleistung eines Systems, nicht aber die Verwaltung und Kontrolle von Bandbreite. Weiterhin ist die Definition von komplexen Jobzuteilungen, wie dies bei Web-Crawlern erforderlich ist, nicht möglich. Um einen verteilt arbeitenden Web-Crawler mit Globus Toolkit zu erstellen, muss man somit sämtliche Steuerkomponenten selbst erstellen. Weiterhin muss man Komponenten erstellen, mit denen die Teilnehmer bestimmen können, wie viel Speicherplatz und Bandbreite sie zur Verfügung stellen.

4.3 Erweiterungen für einen hochgradig verteilt arbeitenden Web-Crawler

Um die Anforderung an eine Crawlerarchitektur, möglichst viele Web-Seiten pro Zeiteinheit zu verarbeiten, mit Hilfe eines Grids zu erfüllen, benötigt man sehr viele Teilnehmer, die entweder einen Web-Crawler oder eine Speicherkomponente auf ihren Systemen betreiben. Dies erfordert die Erstellung eines hochgradig verteilt arbeitenden Web-Crawlers. Shkapenyuk and Suel [SS02] haben eine hierfür eine erweiterte Referenzarchitektur skizziert (aber nicht umgesetzt). Sie enthält kein zentrales, sondern ein verteilt arbeitendes Kontrollsystem. So wird die Verwaltung einer sehr großen Zahl von Web-Crawlern möglich, allerdings kommen eine Reihe weiterer Anforderungen hinzu.

Die Knoten eines verteilten Kontrollsystems müssen untereinander kommunizieren, um zu ermitteln, welche Web-Seiten andere Knoten bereits bearbeitet haben. Zusätzlich wird noch ein Mechanismus benötigt, der hinzukommende Crawler und Speicher gleichmäßig auf die Knoten des verteilten Kontrollsystems aufteilt, um Fairness zu gewährleisten.

Idealerweise wird diese Aufteilung in ein Verzeichnis integriert. Diese Zuteilungsmechanismen sind nicht in der Verzeichniskomponente von Globus Toolkit enthalten. Ein solches Verzeichnis muss daher selbst erstellt werden.

Damit anfragende Web-Crawler in einem Grid immer mit Jobs versorgt werden können, muss mindestens ein Knoten des Steuersystems stets verfügbar sein. Nach Möglichkeit stellt man hierfür einige dedizierte Systeme bereit. Diese sollten möglichst wenig rechenintensive Aufgaben haben, idealerweise nur eine, nämlich eine Liste von gefilterten Adressen zu Web-Seiten für anfragende Web-Crawler bereitzuhalten. Im Unterschied zur Referenzarchitektur bedeutet dies, dass nicht diese Kontrollsysteme die Arbeit der Extraktion neuer Adressen zu Webseiten durchführen. Hierzu kann man die Rechenleistung, Bandbreite und auch den Speicherplatz der freiwilligen Teilnehmer im Grid ausnutzen und diese Aufgaben den Web-Crawlern und Speicherplätzen übertragen.

Zusätzlich zum Anfragen und Laden von Web-Seiten sollen daher die Web-Crawler selbst geladene Web-Seiten lesen und neue Adressen von Web-Seiten extrahieren. Anschließend übertragen die Crawler diese Adressen an die Speicherplätze. Die Speicherplätze sollen zusätzlich die Ermittlung bereits bearbeiteter und unbearbeiteter Web-Seiten übernehmen. Wir sehen dabei eine Möglichkeit darin, jedem Speicherplatz eine Menge von Domains (Web-Sites) zuzuteilen, für die sie verantwortlich sind. So können die Speicherplätze die Adressen dieser Web-Sites verwalten. Die Speicherplätze übertragen dann nur unbearbeitete Adressen an die Steuersysteme. Die Steuersysteme müssen bei dieser Verteilung lediglich noch Koordinationsaufgaben übernehmen. Die verteilten Komponenten des Steuersystems merken sich, welche ihrer angebotenen Speicherplätze welche Domains speichern und verwalten, und kommunizieren untereinander, um zu ermitteln, ob bereits ein anderer Speicherplatz eine Domain verwaltet. Mit dieser Verteilung der Arbeit kann man erreichen, dass ein schlankes Steuersystem entsteht, das lediglich noch Koordinationsaufgaben hat und von den Speicherplätzen gefilterte Adressen zu unbearbeiteten Web-Seiten bekommt. Diese gibt es dann an anfragende Web-Crawler weiter. Mit diesem schlanken Steuersystem haben wir weiterhin eines der Hauptprinzipien in einem Grid umgesetzt, nämlich die Verlagerung von ressourcenintensiven Aufgaben weg von zentralen Stellen [FK03]. In der von uns konzeptionierten, hochgradig verteilten Web-Crawler Anwendung übernehmen die Knoten von freiwilligen Teilnehmern eines Grids nun alle Aufgaben, die viel Rechenleistung, Bandbreite und Speicherplatz benötigen.

5 Evaluation von Globus Toolkit

Wie im vorigen Kapitel gezeigt, sind eine Reihe eigener Entwicklungen nötig, um einen gridbasierten Web-Crawler mit Hilfe von Globus Toolkit erstellen zu können. Vorteile von Globus Toolkit im Vergleich zu anderen Grid-Middleware Lösungen wie *BOINC*⁷ oder *UNICORE*⁸ sind die umfangreichen Lösungen zu allen Sicherheitsfragen wie auch umfangreiche Lösungen zur Einbindungen heterogener Systeme auf Basis von Diensten. Dies macht Globus Toolkit zu der am besten geeigneten Grid-Middleware für die Erstellung eines Grids zum Betrieb einer hochgradig verteilten Web-Crawler Anwendung.

Wir sind daher der generellen Frage nachgegangen, wie robust die Globus Toolkit Plattform selbst ist. Da eine Web-Crawler Anwendung zumeist viele Web-Seiten parallel lädt und verarbeitet, gehen wir von einer hohen Belastung der darunter liegenden Systeme, speziell der Grid-Middleware, aus. Unter Robustheit verstehen wir hier das Verhalten von Globus Toolkit bei starker Auslastung. Wir erwarten, dass die Plattform auch bei einer hohen Prozessorauslastung und Hauptspeichernutzung stabil weiterläuft.

Zur Evaluation dieser Annahme haben wir einen Web-Crawler erstellt und innerhalb der Globus Toolkit Plattform installiert. Dieser Web-Crawler bearbeitet parallel acht ver-

⁷ <http://boinc.berkeley.edu>

⁸ <http://www.unicore.com>

schiedene Jobs, indem er die entsprechenden Web-Seiten lädt, neue Webadressen extrahiert und die Web-Seite anschließend auf der Festplatte speichert.

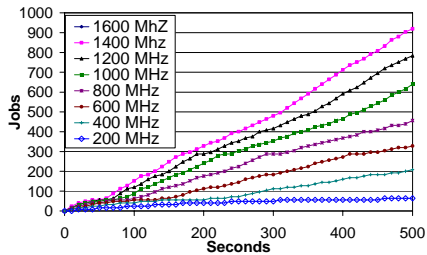


Abbildung 2 : Verhalten bei limitierter CPU

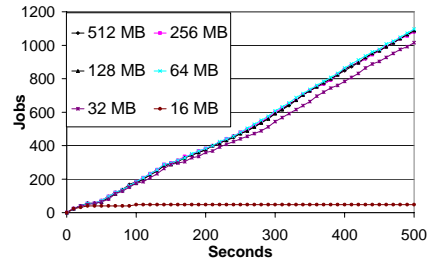


Abbildung 3 : Verhalten bei limitiertem RAM

Abbildung 2 zeigt die Ergebnisse der ersten Evaluierung. Hierbei wurde die Prozessorleistung schrittweise von 1600 MHz auf 200 MHz reduziert, um somit eine Vollausslastung des Systems zu simulieren. Auch bei geringer Prozessorleistung hat die Plattform stabil weitergearbeitet, und es kam zu keinen Fehlermeldungen. Abbildung 3 zeigt die Ergebnisse der zweiten Evaluierung. Hierbei wurde die Menge an verfügbarem RAM für die Grid-Middleware und somit auch für die Crawleranwendung schrittweise von 512 MB auf 16 MB reduziert. Wie man sieht, hat die Menge an Hauptspeicher kaum Einfluss auf die Leistung des Web-Crawlers. Globus Toolkit verhält sich somit auch bei einer hohen Systemauslastung robust und scheint daher auch für komplexe und ressourcenhungrige Anwendungen wie Web-Crawling geeignet zu sein.

6 Zusammenfassung und Ausblick

Basierend auf dem umfangreichsten Framework für Grid-Middleware – Globus Toolkit – gibt es bereits eine Reihe leistungsfähiger Anwendungen, die Rechenleistung auf Rechnern von freiwilligen Teilnehmern im Internet nutzen. Mit dieser Arbeit sind wir der Frage nachgegangen, wie gut Globus Toolkit geeignet ist, um mit Hilfe der vorhandenen Standardkomponenten komplexe Anwendungen mit einem hohen Ressourcenbedarf zu realisieren. Hierbei hat sich gezeigt, dass diese Komponenten und auch verfügbare Erweiterungen hauptsächlich auf die Verwaltung und Steuerung von Rechenleistung ausgelegt sind. Dadurch sind die Komponenten für die Realisierung von komplexen Grid-Anwendungen mit einem komplexen Ressourcenbedarf zu unflexibel.

Wie gezeigt, benötigt eine Web-Crawler Anwendung sehr viel Bandbreite, Speicherplatz und Rechenleistung. Zur Kontrolle dieser Ressourcen wie auch zur Steuerung der darin vorhandenen Web-Crawler und Speicherplätze müssen daher eigene Komponenten entwickelt werden, um eine gridbasierte Web-Crawler Anwendung zu erstellen.

Die Evaluierung hat die Robustheit von Globus Toolkit auch bei hohen Belastungen gezeigt. Dies motiviert die vollständige Umsetzung der in diesem Papier konzeptionierten umfangreichen Web-Crawler als Anwendung in einem Grid mit Hilfe von Globus

Toolkit. Diese Anwendung enthält ein verteilt arbeitendes Steuer- und Kontrollsystem wie auch ein verteilt arbeitendes Verzeichnis. Unser Hauptziel ist dabei zu zeigen, wie zukünftige Standardkomponenten von Globus Toolkit durch eine höhere Flexibilität die Erstellung von komplexen Grid-Anwendungen vereinfachen können.

Eine weitere Frage betrifft den dienstorientierten Ansatz von Globus Toolkit. Globus Toolkit erlaubt die Erstellung von Komponenten als Dienste. Mit der Erstellung des gridbasierenden Web-Crawlers auf Basis von Grid-Services wollen wir daher auch der Frage nachgehen, wie aufwendig die Erstellung von Grid-Services mit Hilfe von Globus Toolkit ist, und wie performant Grid-Services im Vergleich zu anderen Dienstplattformen wie zum Beispiel Tomcat⁹ sind.

7 Literaturverzeichnis

- [BP98] *Brin, S.; Page, L.*: The anatomy of a large-scale hyper textual Web search engine; In: Computer Networks and ISDN Systems, Volume 30, 1998.
- [Ch06] *Chinnici, R et. al.*: Web Services Description Language (WSDL) Version 2.0, W3C Whitepaper, March 2006; <http://www.w3.org/TR/2006/CR-wsdl20-20060327/>; last visited 2006-07-24.
- [FK01] *Foster, I.; Kesselman, C.*: The Anatomy of the Grid; Lecture Notes in Computer Science, Volume 2150; 2001.
- [FK02] *Foster, I.; Kesselman, C.; Nick, J.; Tuecke, S.*; The Physiology of the Grid, Global Grid Forum, June 2002.
- [FK03] *Foster, I.; Kesselman, C.*: The Grid. Blueprint for a New Computing Infrastructure, 2nd Edition, Morgan Kaufmann Publishers, 2003.
- [Fo05] *Foster, I.*: Globus Toolkit Version 4: Software for Service-Oriented Systems. IFIP Conference on Network and Parallel Computing, Lecture Notes in Computer Science, Volume 3779; 2005.
- [GS02] *Gray, J.; Szalay, A.*: The World Wide Telescope, Science Bd.293, 2002.
- [Gu06] *Gudgin et al.*; Web Services Addressing 1.0 – SOAP Binding, W3C Whitepaper, March 2006.
- [Na06] *Nadalin, A. et al.* WS-Security, OASIS Standard Specification, February 2006, <http://docs.oasis-open.org/wss/v1.1/>, last visited 2006-07-24
- [SS02] *Shkapenyuk, V.; Suel, T.*: Design and implementation of a high-performance distributed Web crawler; In: Proceedings of the 18th International Conference on Data Engineering, San Jose; Page 357-368; 2002.
- [SS06] *Sullivan, D.; Sherman, C.*: The search engine watch; www.searchenginewatch.com; last visited 2006-07-24.
- [Th99] *Thompson, M.*: Certificate-based Access Control for Widely Distributed Resources; In: Proceedings of the Eighth USENIX, Washington, D.C; August 1999.

⁹ www.apache.org