Entwicklung der grob granularen Parallelisierung für den Neurosimulator FAUN 1.0 und Anwendungen in der Wechselkursprognose

Diplomarbeit

zur Erlangung des Grades eines Diplom-Ökonomen des Fachbereichs Wirtschaftswissenschaften der Universität Hannover

vorgelegt von

von Mettenheim, Hans-Jörg

[redacted]

Erstprüfer: Prof. Dr. M. H. Breitner

Hannover, den 18. Juli 2003

Für meinen Bruder Walter

Danksagung

Dank gebührt zuerst meiner Familie, die mich in allem was ich tue, stets unterstützt. Auch wenn dies bedeutet, dass sich in “guten” Zeiten zahlreiche Rechner in den Zimmern türmen und Kilometer an Kabel die Flure verzieren.

Ohne die Ideen und Denkanstöße von Herrn Prof. Dr. Michael H. Breitner hätte diese Arbeit so nicht entstehen können. Danken möchte ich vor allem auch für ermutigende Worte, wenn die eine oder andere Prognose nicht die gewünschten Ergebnisse brachte.

Regelmäßig verlassen konnte man sich auf die Hilfe von Herrn Dipl.-Math. Frank Köller (allgemeine Fragen rund um FAUN und Maple), Herrn Dipl.-Ök. Patrick Bartels (Softwareagent PISA) und Herrn Simon König (bei Fragen zu den Institutsrechnern). Herr Dipl.-Math. Patrick Mehmert war stets bereit, mir bei Fragen zur Parallelisierung zu helfen. Herr Dr. Werner Riedel vom Universitätsrechenzentrum der Technischen Universität Chemnitz war mir bei allen Fragen rund um den Cluster CLIC immer und schnell behilflich. Ihm, den Administratoren des CLICs und der Technischen Universität Chemnitz sei herzlich gedankt.

Oftmals nachgefragt, wenn es um Probleme rund um Linux ging, habe ich bei meinem langjährigen guten Freund Jan Christoph Nordholz. Stets konnte er helfen. Danke!

Inhaltsverzeichnis

1 Einleitung

Die vorliegende Diplomarbeit deckt zwei unterschiedliche Gebiete ab. Zunächst wird für den Neurosimulator FAUN 1.0[1] die grob granulare Parallelisierung entwickelt. Anschließend werden Anwendungen vorgestellt, die die zur Verfügung gestellte Rechenleistung ausnutzen. Die erste Anwendung stellt die kurzfristige[2] Prognose des Wechselkurses des Euro-US-Dollars dar. Die zweite Anwendung ist hingegen in der vorliegenden Ausführung theoretisch und dient in erster Linie als Machbarkeitsstudie zur Überprüfung der Leistung der neuronalen Netze. Hierzu wird ein dynamisches Spiel gewählt.

Ein Neurosimulator kann dann zum Einsatz kommen, wenn zwischen Ein- und Ausgabegrößen ein Zusammenhang hergestellt werden soll, andere mathematische Methoden jedoch versagen. Diese mathematischen Methoden können z. B. eine lineare Regression oder ein anderes Gleichungssytem sein. Sinnvoll ist der Einsatz eines Neurosimulators auch dann, wenn das Problem zwar theoretisch lösbar ist, die Lösung jedoch zu kompliziert ist.

Ein Neurosimulator versucht, neuronale Netze zu finden die möglichst gut zu den zur Verfügung stehenden Daten passen. Ein neuronales Netz besteht beispielsweise aus drei Schichten[3], die jeweils sog. Neuronen[4] enthalten. Die erste Schicht wird als Eingabe-

[1] FAUN = Fast approximation with universal neural networks, etwa "Schnelle Approximation mit universellen neuronalen Netzen". Die FAUN Namensrechte liegen vollständig bei der FAUN AG (Fahrzeugwerke Ansbach und Nürnberg, s. auch http://www.faun.de und http://www.faun.com), der hiermit gedankt sei, die Bezeichnung "Neurosimulator FAUN" verwenden zu dürfen.

[2] Der Begriff "kurzfristig" beschreibt in dieser Arbeit einen Zeitraum zwischen 5 und 45 Minuten.

[3] Es sind auch mehr Schichten möglich, z. B. vier Schichten. Zusätzliche Schichten machen jedoch das neuronale Netz komplizierter und bringen selten eine signifikant Verbesserung der Güte des neuronalen Netzes.

[4] Dieser Begriff ist aus der Biologie entlehnt und stellt eine Anlehnung an die Neuronen im (menschlichen)

schicht bezeichnet und enthält so viele Neuronen wie Eingabegrößen[5]. Die zweite Schicht ist die verdeckte Schicht und enthält eine gewisse, vom Benutzer zu wählende Anzahl an Neuronen. Die Zahl der verdeckten Neuronen[6] liegt meist zwischen 1 und 5. Die dritte Schicht, die Ausgabeschicht, enthält im Falle einer Ausgabegröße genau ein Neuron. Alle Neuronen sind über sog. Gewichte gekoppelt. Die Gewichte sind die vom Neurosimulator zu optimierenden Parameter. Eine ausführlichere Beschreibung der Funktionsweise von neuronalen Netzen mit engem Zusammenhang zur vorliegenden Arbeit finde sich z. B. in [2, 10].

Es ist zweckmäßig, eine größere Zahl (ca. 100 bis 10000) neuronaler Netze vom Neurosimulator berechnen zu lassen, um anschließend das beste Netz auswählen zu können. Das beste Netz ist dasjenige, welches den kleinsten Fehler aufweist, sich also den Daten am besten annähert. Der Neurosimulator FAUN[7], der seit 1996 von Herrn Prof. Dr. M. H. Breitner[8] entwickelt wird, kann in der im Jahre 2002 von Herrn Dipl.-Math. F. Köller[9] fertiggestellten Version 1.0 die neuronalen Netze unkompliziert berechnen. Es ist allerdings so, dass die Berechnungen auf einem einzelnen Rechner[10] für eine akzeptable Anzahl von Netzen u. U. sehr lange dauern können. Einige Rechnungen dieser Arbeit dauern, wenn sie auf einem einzelnen Rechner ausgeführt werden, in etwa 40 Stunden.

Gehirn dar. Keinesfalls jedoch kann ein neuronales Netz die Leistung des menschlichen Gehirns erbringen, wie dies in der Euphorie der frühen Anfangsphase der Computertechnik in den 1950er und 60er Jahren erwartet wurde, s. auch [16]. Diese Visionen sind längst einer realistischen Sichtweise gewichen, die den Einsatz neuronaler Netze auf klar umrissene Anwendungsgebiete und Aufgabenstellungen beschränkt. Ein neuronales Netz wird heute nicht mehr als Universallösung gesehen. Die Parallelen zur Biologie liegen nun lediglich in den Bezeichnungen und (ansatzweise) in der Verschaltung der Neuronen untereinander.

[5] Hinzu kommt i. allg. ein sog. Bias-Neuron.

[6] In der vorliegenden Arbeit verwenden alle neuronalen Netze nur ein einziges inneres Neuron. Dies ist für die behandelten Probleme völlig ausreichend und macht die Netze einfacher.

[7] Der aktuelle Forschungs- und Entwicklungsstand des Neurosimulators FAUN kann auf der FAUN Projekthomepage des Instituts für Wirtschaftsinformatik (http://www.iwi.uni-hannover.de) des Fachbereichs Wirtschaftswissenschaften der Universität Hannover unter http://www.iwi.uni-hannover.de/faun.html abgerufen werden.

[8] s. auch http://www.iwi.uni-hannover.de/mitarbeiter/mb.html

[9] s. auch http://www.iwi.uni-hannover.de/mitarbeiter/fk.html

[10] Der Autor verwendet einen Pentium IV Rechner mit 2.4 GHz. Dies ist jedoch nur exemplarisch zu betrachten. Selbst mit dem zur Zeit der Fertigstellung dieser Arbeit schnellsten Pentium IV Prozessor mit 3.2 GHz ergäben sich ähnlich hohe, in den meisten Fällen unzumutbare Rechenzeiten.

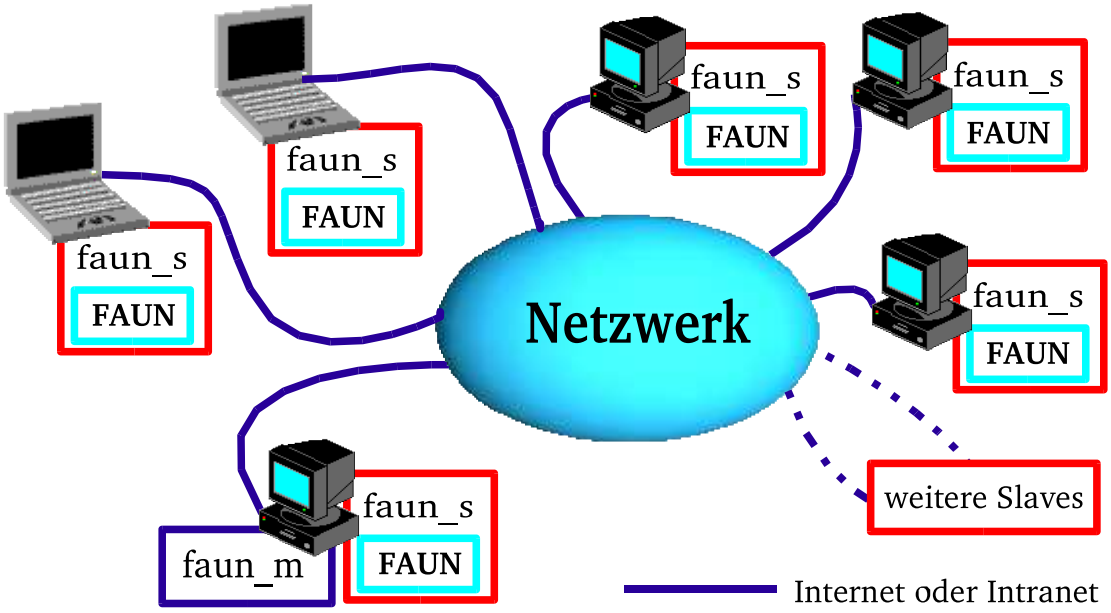

Abbildung 1.1: Schematische Darstellung der Funktionsweise des FAUN 1.0 PVM-HPC Programmpakets. Das Master Programm faun_m kontrolliert mehrere sog. Slave-Programme (Tochter-Programme) faun_s, die den eigentlichen Neurosimulator FAUN 1.0 starten. Auf dem Master-Rechner können ebenfalls Instanzen von faun_s laufen, da die administrativen Aufgaben den Master-Rechner kaum belasten. Es ist auch möglich, pro Rechner mehrere Instanzen von faun_s zu starten. Dies ist insbesondere dann zu empfehlen, wenn auch Mehrprozessor Rechner eingebunden werden sollen. In diesem Falle wird je Prozessor z. B. eine Instanz von faun_s gestartet. Die Rechner müssen im Normalfall mit einem Netzwerk verbunden sein, das direkte IP-Verbindungen erlaubt. Diese Einschränkung ist jedoch durch eine zusätzliche Entwicklung des Autors weitestgehend aufgehoben, s. Abschnitt 2.5. Das verbindende Netzwerk kann ein lokales Intranet sein (z. B. in einer Universität) jedoch auch das langsame Internet.

Durch Parallelisierung läßt sich dieses Problem lindern: Die Aufgabe wird nicht mehr nur von einem Prozessor bearbeitet, sondern mehrere Prozessoren teilen sich die Arbeit. Diese muss passend aufgeteilt werden. Es wäre möglich, die unterschiedlichen Berechnungsschritte der Netze aufzuteilen, eine sog. fein granulare[11] Parallelisierung. Dies würde jedoch schnelle und teure Datenleitungen zwischen den Prozessoren erfordern, mithin einen Mehrprozessorrechner[12]. Dies ist jedoch im Falle des Neurosimulators FAUN 1.0 nicht nötig. Einfacher ist es, wenn die Prozessoren ganze Netze berechnen, das Problem also grob granular aufgeteilt wird. Dies hat den Vorteil, dass keine Änderungen am eigentlichen Neurosimulator FAUN nötig sind. Stattdessen ist ein Mantelprogramm erforderlich, dass die Verteilung der Netze steuert. Ein weiterer Vorteil besteht darin, dass die Prozessoren durch langsamere[13] Datenleitungen verbunden sein können. Dies ermöglicht es, mehrere, günstige Arbeitsplatzrechner[14] zu einem virtuellen Superrechner zusammenzuschalten[15] und dies zu einem Bruchteil der Kosten[16], die für einen herkömmlichen Mehrprozessorrechner gleicher Leistung aufzubringen sind.

Das in dieser Arbeit vorgestellte und in großen Teilen entwickelte Programmpaket FAUN 1.0 PVM-HPC[17] ermöglicht es, verschiedene Rechner für die Berechnung der neuronalen Netze heranzuziehen. Die Kommunikation der Rechner untereinander wird vom Programmpaket PVM[18] geregelt und unterstützt. Dies wird durch eine Zweiteilung der Aufgabe erreicht. Das Programm faun_m ist der sog. Master, der die Tochterprozesse (sog. Slaves) faun_s steuert. Die Slaves übernehmen ihrerseits die Kommunikation mit dem Programmkern, dem Neurosimulator FAUN. Hierbei können die Slaves auf Rech-

[11] Die Granularität beschreibt die “Körnigkeit” eines Problems, also die Möglichkeit, ein Problem in feinere Abschnitte zu untergliedern.

[12] Die Prozessoren eines Mehrprozessorrechners sind mit Datenübermittlungsgeschwindigkeiten von mehreren Gbit/s verbunden.

[13] Das sehr verbreitete Fast Ethernet mit Übertragungsgeschwindigkeit von 100 Mbit/s ist völlig ausreichend.

[14] Damit wird der ebenfalls geläufig Begriff “PC” umschrieben.

[15] Ein solcher Rechnerverbund von meist homogenen Rechnern wird als Cluster bezeichnet.

[16] Je nach Konfiguratio kann die Ersparnis im Bereich von 50 bis 70 Prozent liegen, teilweise sogar darüber.

[17] HPC = High performance computing, etwa “Hochleistungsrechnen”. Diese Bezeichnung soll ausdrücken, dass das entwickelte Programmpaket insbesondere für das Hochleistungsrechnen geeignet ist. Dadurch wird die Anwendung im geringen Leistungsbereich nicht eingeschränkt. FAUN 1.0 PVM-HPC ist ebenso auf wenigen Rechnern mit gleichem relativem Leistungsgewinnen lauffähig. Im Extremfall kann das Programmpaket auch auf nur zwei Rechnern laufen.

[18] PVM = Parallel virtual machine, s. auch [6].

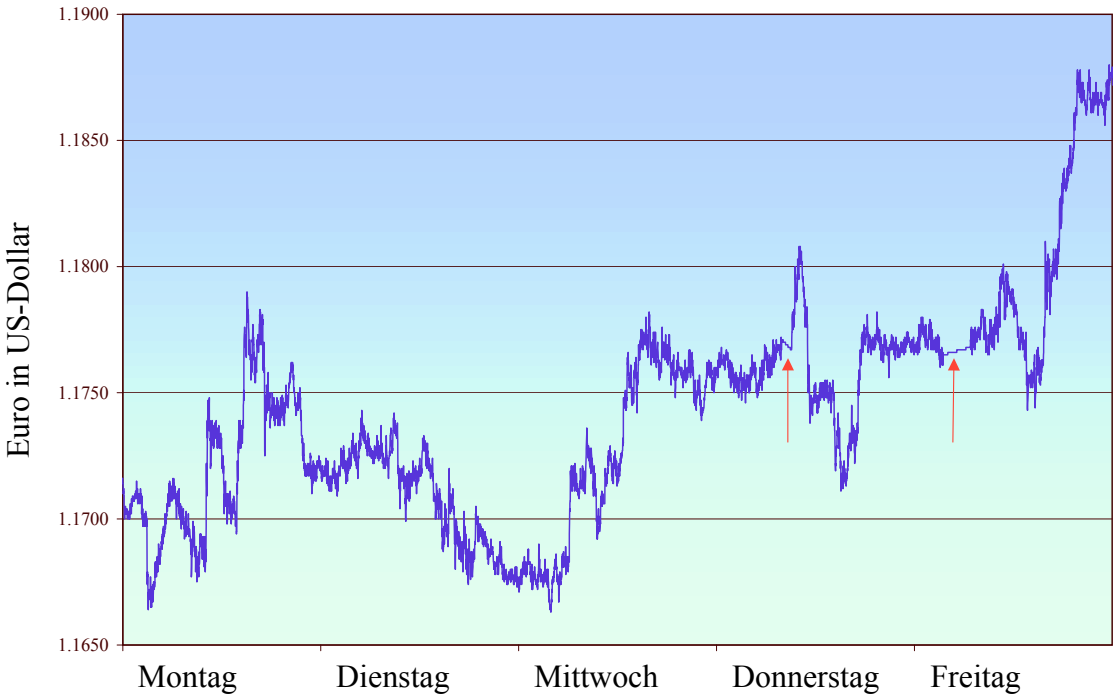

Abbildung 1.2: Wechselkurs des Euro-US-Dollars in der Woche vom 9. bis 13. Juni 2003. Die Kurse werden minütlich aufgenommen. An zwei Stellen gibt es (technisch bedingte) Lücken im Datensatz, die durch rote Pfeile markiert sind. Die Lücken werden linear interpoliert. Der vorliegende Datensatz dient dem Neurosimulator als Trainingsdatensatz.

nern unterschiedlicher Art und Leistung laufen. Das Master Programm übernimmt die Verteilung und sorgt dafür, dass jeder Rechner stets beschäftigt ist. Auch wird dafür gesorgt, dass schnellere Rechner mehr Netze berechnen als langsame. Eine schematische Darstellung der Funktionsweise des Programmpakets und das Zusammenspiel von Master, Slave und FAUN-Kernprogramm ist in Abbildung 1.1 dargestellt. Das Programmpaket befinde sich samt Anwendungsbeispielen auf der dieser Arbeit im vorderen Buchdeckel beiliegenden CD-ROM[19].

Als erstes Anwendungsbeispiel ist die kurzfristige Prognose des Euro-US-Dollar Wechselkurses anzuführen. Eine kurzfristige Wechselkursprognose ist relevant für Geldinstitu-

[19] Der Neurosimulator FAUN 1.0 befinde sich jedoch nicht auf der CD-ROM. Der Neurosimulator ist keine Share- oder Freeware. Bei Interesse am Neurosimulator ist bei Herrn Prof. Dr. M. H. Breitner nachzufragen, s. Fußnote 8.

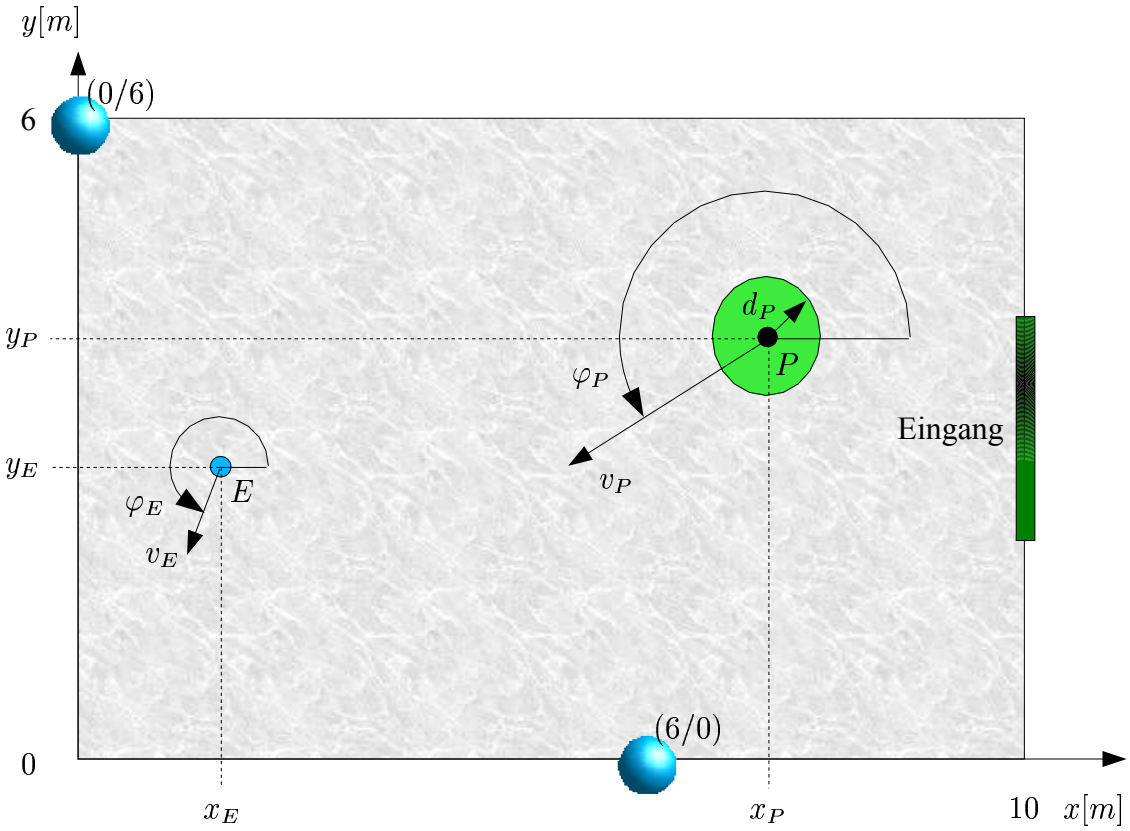

Abbildung 1.3: Der geometrische Aufbau des Enhanced Cornered Rat Game: Die Katze P verfolg die Ratte E. Katze und Ratte können ihren Winkel beliebig ändern. Durch zwei Löcher im Raum an den Positionen $(0[m]/6[m])$ und $(6[m]/0[m])$ kann die Ratte der Katze endgültig entfliehen Die Graphik wird im Kapitel 4 genauer erläutert.

te aber auch für große Unternehmen, die täglich Beträge in Milliardenhöhe am internationalen Devisenmarkt, Forex[20] genannt, tauschen. Auch wenn die Schwankungen im Wechselkurs nur gering sind, macht selbst die kleinste Schwankung bei hohen Beträgen eine beträchtliche Summe aus. Täglich werden Euros und US-Dollars in der Größenordnung von 1 Billion(!) Euro getauscht. Damit ist dieser Markt der liquideste[21] der Welt und

[20] Forex = Foreign exchange, etwa "ausländischer Tausch"

[21] Das bedeutet, dass Käufe und Verkäufe zum aktuellen Kurs praktisch immer realisierbar sind. Das Gegenteil ist ein illiquider Markt, in dem Angebot und Nachfrage nicht ausgeglichen sind. In diesem Falle können Kauf-/Verkaufaufträge nicht immer erfüllt werden. Es gibt einen Angebots- oder Nachfrageüberschuss.

kann nicht von einem einzelnen Marktteilnehmer dauerhaft beeinfluss werden. Die Forex wird zentral von einem Rechner gesteuert und ist unabhängig von den Öffnungszeiten der Börsen. Ein 24-Stunden Handel von Montag bis Freitag möglich. Dies erlaubt es, über 5 Tage kontinuierlich Kurse aufzunehmen und verschiedene Indikatoren mit den gelieferten Daten auszuprobieren. In der vorliegenden Arbeit werden die minütlichen Wechselkurse der Woche vom 9. bis 13. Juni 2003 verwendet, s. Abbildung 1.2. Um eine Prognose zu erstellen, werden gleitende Durchschnitte (s. Abschnitt 3.2) und Candlesticks (s. Abschnitt 3.3) verwendet.

Das zweite Anwendungsbeispiel ist das sog. Enhanced Cornered Rat Game (ECRG)[22]. Das ECRG ist ein typisches dynamisches Verfolgungsspiel: Eine Katze, der Verfolger P[23], verfolgt eine Ratte, den Verfolgten E[24], s. Abbildung 1.3. Die Ratte E ist in einem rechteckigen Raum eingeschlossen, hat aber zwei Möglichkeiten, der Katze endgültig zu entkommen: der Raum hat zwei "Schlupflöcher in einer Seitenwand und in einer Ecke, durch die die Ratte E entfliehe kann. Die Katze P ist doppelt so schnell wie die verfolgte Ratte E. Es ergeben sich Situationen in denen die Ratte E bei optimaler Strategie sicher entfliehe kann und andere Situationen in denen die Katze P die Ratte E sicher fangen kann. Diese optimalen Strategien lassen sich in akzeptabler Zeit mathematisch exakt berechnen. In der vorliegenden Arbeit werden die optimalen Strategien mit den Ergebnissen verglichen, die ein neuronales Netz liefert. Ein solcher Vergleich ist wichtig, um die Güte der gefundenen Netze und des Neurosimulators in Anwendungen zu beurteilen. Das ECRG bietet sich für einen solchen Vergleich an. Es ist ein hochkomplexes Spiel, das viele Variationsmöglichkeiten bietet. Die Lösungen lassen sich jedoch exakt berechnen. Informationen über die Entwicklung des Spiels vermitteln [11, 13, 5].

Insgesamt soll in dieser Arbeit die Funktionsweise des entwickelten Programmpakets FAUN 1.0 PVM-HPC dargestellt und an Anwendungsbeispielen erläutert werden. Dass Parallelisierung ein aktuelles und wichtiges Thema in der Wissenschaft ist, zeigen z. B. [4, 14, 15].

[22] Etwa "Erweitertes Spiel der in einer Ecke eingeschlossenen Ratte"
[23] von engl. pursuer = Verfolger
[24] von engl. evader = Flüchtiger

5 Fazit und Ausblick

Die vorliegende Arbeit zeigt zunächst, wie sich durch Parallelisierung Probleme, die sonst extrem lange Rechenzeiten erforderten, in akzeptabler Zeit lösen lassen. Einen Überblick über die Rechenzeiten liefert Tabelle 5.1. Dort wird die vom CLIC benötigte Rechenzeit auf einen einzelnen aktuellen Rechner umgerechnet. Inklusive aller Testrechnungen hätten die in diese Arbeit eingebrachten Rechnungen auf nur einem Rechner eine Rechenzeit von ca. 40 Tagen erfordert. Evtl. wäre dieser Wert auch höher gewesen, da es nötig ist, mit den Parametern des verwendeten Neurosimulators FAUN 1.0 zu experimentieren. Nicht erfolgreiche Rechnungen können abgebrochen werden, jedoch wird erst eine gewisse Rechenzeit benötigt, um feststellen zu können, dass die Berechnung nicht erfolgreich verläuft.

Insofern stellt das in dieser Arbeit weiter entwickelte Mantelprogramm FAUN 1.0 PVM-HPC für den Neurosimulator FAUN 1.0 einen wesentlichen Aspekt dar. Es erlaubt, die Vorteile der neuen Version 1.0 des Neurosimulators auch parallel zu nutzen. Die Verwendung grob granularer Parallelisierung erfordert keinerlei Änderungen am Kernprogramm.

Mittelfristige Ziele bei der Weiterentwicklung der Parallelisierung sind insbesondere die Portierung von FAUN 1.0 PVM-HPC auf andere Betriebssysteme, z. B. Windows und Macintosh. Außerdem sollen Programmversionen für Mehrprozessorrechner und Vektorrechner entworfen werden. Darüberhinaus kann die Benutzerfreundlichkeit von FAUN 1.0 PVM-HPC noch erhöht werden. Hierunter fällt die Integrierung in die graphische Oberfläch für FAUN.

Ein weiterer Punkt dieser Arbeit ist die Prognose des Euro-US-Dollar Wechselkurses mit Hilfe des Neurosimulators unter Verwendung der parallelisierten Version. Als Indikatoren werden sowohl klassische gleitende Durchschnitte als auch die neuere Methode

Problem	Zeit [s]	Rechner	entspricht [Std.]
Euro-Dollar, Skalierung [-0.8;0.8]			
Prognosehorizont 5 Min.			
kurzer GD	1146	100	11.4
kurzer GD, Ausreißer 100 mal	967	100	9.6
Euro-Dollar, Skalierung [-0.95;0.95]			
Prognosehorizont 5 Min.			
kurzer GD	1821	100	18.1
kurzer GD, Ausreißer 100 mal	1265	100	12.6
mittlerer GD	1270	100	12.6
langer GD	1466	100	14.6
alle GD	1272	100	12.6
3 Kerzen à 5 Min	1158	200	23.0
3 Kerzen à 5 Min, 15 Min. GD	2108	200	42.0
5 Kerzen à 5 Min, 15 Min. GD	2078	200	41.3
Prognosehorizont 15 Min.			
kurzer GD	1008	100	10.0
mittlerer GD	1575	100	15.7
3 Kerzen à 15 Min.	829	200	16.5
3 Kerzen à 15 Min., 45 Min. GD	1420	200	28.2
5 Kerzen à 15 Min.	1767	200	35.1
5 Kerzen à 15 Min., 45 Min GD	1255	200	25.0
Prognosehorizont 45 Min.			
kurzer GD	1051	100	10.5
mittlerer GD	1614	100	16.1
langer GD	1373	100	13.7
alle GD	1296	100	12.9
alle GD, Ausreißer 30 mal	2125	100	21.1
Enhanced Cornered Rat Game			
1. Entkommenszone, Ratte	1154	200	23.0
1. Entkommenszone, Katze	356	200	7.1
2. Entkommenszone, Ratte	600	175	10.4
2. Entkommenszone, Katze	149	175	2.6
		ges.	445.7

Tabelle 5.1: Überblick über die Berechnungszeiten, umgerechnet auf die Rechenzeit, die ein heute üblicher Pentium IV mit 2.4 GHz benötigt. Insgesamt ergeben sich 445.7 Std. Berechnungszeit auf einem einzelnen Rechner. Dies entspricht knapp 19 Tagen ununterbrochener Rechnung! Da der Autor zahlreiche Testrechnungen durchgeführt hat, die in der Übersicht nicht auftauchen, kann der Wert verdoppelt werden.

der japanischen Kerzencharts genutzt. Es ist festzustellen, dass die Prognosen keine hohe Güte aufweisen. Die Prognose extremer Schwankungen gelingt nicht. Dies liegt nach Ansicht des Autors daran, dass auf dem Devisenmarkt viele unvorhergesehene Ereignisse auftreten können. Spekulative Geschäfte können z. B. zu jeder beliebigen Uhrzeit auftreten, ohne dass die *genaue* Uhrzeit eine Rolle spielt. Verkauft z. B. ein Marktteilnehmer spekulativ 1 Mrd. Euro, kann er dies um 10.00 aber auch um 10.05 tun, ohne dass dies vorhersehbar ist. Dennoch beeinfluss dieser Verkauf den Kurs kurzzeitig. Solche Ereignisse können nicht prognostiziert werden.

Es sollte aber möglich sein, bessere Prognoseergebnisse zu erzielen. Der Autor wird weiterhin das Problem der Wechselkursprognose bearbeiten. Es bietet sich an, mit weiteren Indikatoren zu experimentieren. Auch verspricht möglicherweise eine längerfristige Prognose mehr Erfolg. Hiermit ist ein Prognosehorizont im Bereich von Tagen gemeint. Der Autor wird sich weiterhin mit Derivaten auf Devisenkurse und den damit verbundenen Problemen beschäftigen.

Schließlich wird der Neurosimulator dazu verwendet, Strategien für ein dynamisches Spiel zu finden Das vorliegende Spiel, das Enhanced Cornered Rat Game, ist ein typisches Verfolgungsspiel. Diese theoretische Anwendung ist von Vorteil, da die Lösungen des Spiels analytisch genau berechnet werden können. Es ist somit möglich, die gelieferten Ergebnisse direkt zu überprüfen. Es wird gezeigt, dass bei einer sinnvollen Ergänzung der neuronalen Strategie neuronale Netze ein dynamisches Spiel mit beachtlichem Erfolg kontrollieren können.

Möglichkeiten zur Optimierung sind auch hier mit Sicherheit noch gegeben. Es ist denkbar, mit anderen Parametern für den Neurosimulator zu experimentieren.

Insgesamt hofft der Autor, dem Leser eine gute Einführung in die Problematik der Parallelisierung und der damit bearbeiteten Anwendungen zu geben. Die vorgestellten Aspekte können mit Sicherheit noch vertieft werden. Dies wird auch geschehen, denn, wie Erich Kästner sagt:

Es gibt nichts Gutes,
außer: man tut es.