Köchel, P. (Hrsg.): KI-Methoden in der simulationsbasierten Optimierung - 13. Workshop der ASIM-Arbeitsgruppe „Simulation und Künstliche Intelligenz". Chemnitzer Informatik-Berichte CSR-99-03, Technische Universität Chemnitz, 1999, S. 79-88

Ein Multiagentenansatz zur Realisierung selbstadaptierender hybrider Optimierungsverfahren

Sven Hader
Fakultät für Informatik, TU Chemnitz
D-09107 Chemnitz, Deutschland
E-Mail: sha@informatik.tu-chemnitz.de

Zusammenfassung

Entwurfsaufgaben, deren Lösungsraum sich durch einen parametrierbaren Prototyp beschrieben läßt, können mit den Mitteln der Parameteroptimierung rechnergestützt gelöst werden. In dieser Arbeit wird eine Klasse selbstadaptierender hybrider Optimierungsverfahren auf Multiagentenbasis vorgestellt, die zur Lösung derartiger Aufgaben in einer den eingesetzten Mitteln angemessenen Qualität geeignet ist. Die Implementierung eines solchen Verfahrens wird anhand des hybriden Optimierungssystems MASCOT beschrieben.

1 Einführung

Der Entwurf technischer Systeme ist geprägt durch ständig steigende Anforderungen an deren Leistungsfähigkeit, Flexibilität und Qualität. Aus Gründen der Wirtschaftlichkeit existieren außerdem starke Restriktionen bezüglich der notwendigen finanziellen Aufwendungen für Entwurf und Betrieb der Systeme. Verschärfend kommt hinzu, daß wegen der immer kürzer werdenden Innovationszyklen für den eigentlichen Entwurfsprozeß immer weniger Zeit zur Verfügung steht. Um die Entwurfsaufgabe innerhalb der vorgegebenen Zeit in der geforderten Qualität lösen zu können, ist deshalb der möglichst durchgängige Einsatz rechentechnischer Konzepte notwendig.

Nach Gero (1990) können Entwurfsaufgaben in die drei Bereiche *Routine-Entwurf*, *innovativer Entwurf* und *kreativer Entwurf* unterteilt werden. Beim Routine-Entwurf entsteht die Lösung durch Verfeinerung oder Parametrierung existierender Prototypen, beim innovativen Entwurf durch Kombination von Prototypen und beim kreativen Entwurf durch Erzeugung neuer Prototypen. Die meisten der in der Praxis anfallenden Entwurfsaufgaben können dem Routine-Entwurf zugeordnet werden. Das Teilgebiet des Routine-Entwurfs, bei dem der Lösungsraum durch einen parametrierbaren Prototyp beschrieben werden kann, wird als *parametrischer Entwurf* bezeichnet. Bei Aufgaben des parametrischen Entwurfs reduziert sich der Entwurfsprozeß auf die Suche nach geeigneten Wertbelegungen für die Parameter des Prototyps. Derartige Aufgaben können als Optimierungsprobleme formuliert und mit den Mitteln der *Parameteroptimierung* rechnergestützt gelöst werden.

Parametrische Entwurfsaufgaben repräsentierende Optimierungsprobleme weisen gegenüber „rein mathematischen“ Problemen meist eine Reihe wesentlicher Besonderheiten auf:

- Zielfunktion und/oder Nebenbedingungen liegen nicht in analytischer sondern in algorithmischer Form vor (i. allg. als ausführbare Programme)
- große Anzahl von Entwurfsvariablen
- große Anzahl von linearen/nichtlinearen Nebenbedingungen
- Vielzahl lokaler Optima

Diese Besonderheiten in Verbindung mit der im Vergleich zur Größe des Suchraums nur sehr geringen verfügbaren Zeit erschweren das Finden optimaler und oft sogar „annehmbarer“ Entwürfe. Deshalb stellt die Auswahl eines geeigneten Optimierungsverfahrens und dessen problembezogene Parametrierung[1] eine wesentliche Voraussetzung für die termingerechte und qualitativ hochwertige Lösung praxisrelevanter Entwurfsaufgaben mittels Optimierung dar. Oftmals ist dem Ingenieur (hier als Synonym für den am Entwurfsprozeß Beteiligten verwendet) dieses Problem nicht bewußt oder kann wegen des begrenzten Wissens zur Optimierung nicht in der notwendigen Qualität gelöst werden. Folge davon ist die Verwendung von Optimierungsverfahren bzw. Parametrierungen, die für die konkrete Entwurfsaufgabe nicht geeignet sind. Dieser Zustand kann insbesondere bei komplexeren Aufgaben durchaus unbemerkt bleiben, solange der gefundene Entwurf nicht „offensichtlich“ inoptimal ist.

Eine Möglichkeit, um diese unbefriedigende Situation zu verbessern, besteht in der Bereitstellung eines leistungsfähigen Optimierungsverfahrens, das in der Lage ist, bei minimalem Parametrierungsaufwand für möglichst alle Aufgaben des parametrischen Entwurfs eine den eingesetzten Mitteln (Rechnerkapazität, Zeit, u. ä.) angemessene Lösungsqualität zu erreichen. Das Interesse der Forschung richtet sich dabei zunehmend auf zusammengesetzte, sogenannte *hybride* Verfahren.

2 Hybride Optimierungsverfahren

Trotz der in den letzten Jahren zunehmenden Forschungstätigkeit auf dem Gebiet der hybriden Optimierung existiert bis heute weder eine abgeschlossene formale Theorie noch eine allgemein anerkannte Terminologie. Für das Verständnis dieser Arbeit erscheint es deshalb sinnvoll, einige Begriffe und Aspekte der hybriden Optimierung zu erläutern. Der Schwerpunkt liegt dabei eher auf einer allgemeinen als auf einer formal vollständigen Darstellung.

Die *hybride Optimierung* beschäftigt sich mit der Lösung von Optimierungsproblemen durch Anwendung einer Kombination aus mindestens zwei verschiedenen Optimierungsverfahren.

Ein Optimierungsverfahren wird als *hybrid* bezeichnet, wenn es sich aus mindestens zwei verschiedenen, auch eigenständig anwendbaren Optimierungsverfahren zusammensetzt.

[1] Der ein Optimierungsverfahren beschreibende Algorithmus besitzt meist gewisse Freiheitsgrade bezüglich seiner Ausführung, sogenannte *Verfahrensparameter* (z. B. Schrittweiten). Die Festlegung konkreter Werte für diese Parameter wird als *Parametrierung* des Verfahrens bezeichnet.

Letztere werden in dieser Arbeit auch als „Teilverfahren“ bezeichnet. Ein Beispiel für ein hybrides Optimierungsverfahren zeigt Abbildung 1. Alle nicht-hybriden Optimierungsverfahren werden als *atomar* bezeichnet, siehe SYRJAKOW und SZCZERBICKA (1995). Sie machen den weit überwiegenden Teil der heute existierenden Optimierungsverfahren aus.

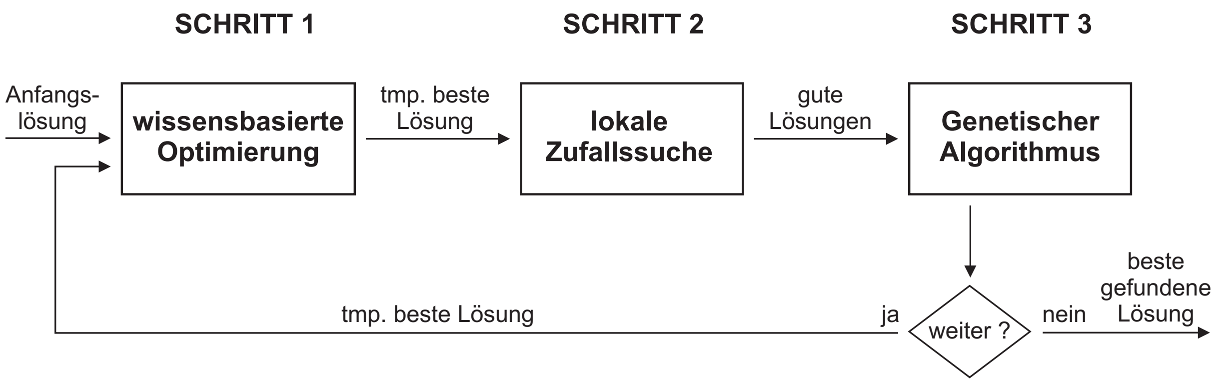

Abbildung 1: Beispiel eines hybriden Optimierungsverfahrens (in Anlehnung an Powell, Skolnick und Tong (1990))

Bei der Entwicklung hybrider Optimierungsverfahren wird meist von einem bereits existierenden atomaren Verfahren ausgegangen. Mit der Erweiterung zum hybriden Verfahren wird mindestens eines der folgenden Ziele verfolgt:

1. Steigerung der Effizienz des Verfahrens

   Optimierungsverfahren sind für die verschiedenen Phasen des Optimierungsprozesses unterschiedlich gut geeignet. So zählt z. B. bei Evolutionären Algorithmen die globale Erforschung des Suchraums zu den besonderen Stärken während die abschließende lokale Konvergenz zum Optimum eher langsam erfolgt. Deshalb liegt es nahe, in den „schwachen“ Phasen andere Optimierungsverfahren einzusetzen, die dafür besser geeignet sind. Nach diesem Prinzip werden gewöhnlich globale Optimierungsverfahren mit effizienten lokalen Verfahren für die Schlußphase der Optimierung kombiniert (HART 1994). Dadurch wird eine Verbesserung der Effizienz des globalen Verfahrens erreicht, ohne die Lösungsqualität zu verschlechtern.

2. Erweiterung des Anwendungsbereichs des Verfahrens

   Optimierungsverfahren sind für verschiedene Aufgabenklassen unterschiedlich gut geeignet. So sind für Aufgaben mit komplexen Suchräumen Evolutionäre Algorithmen besser geeignet als Gradientenverfahren; für Aufgaben mit konvexer Zielfunktion gilt das Gegenteil. Deshalb kann durch Kombination von Optimierungsverfahren für verschiedene Aufgabenklassen das entstehende hybride Verfahren einen größeren Anwendungsbereich abdecken als jedes der Teilverfahren für sich. Bei der Lösung einer konkreten Aufgabe wird innerhalb des hybriden Verfahrens entweder das (wahrscheinlich) am besten geeignete Verfahren ausgewählt (PINTO 1989) oder alle geeignet erscheinenden Verfahren werden nacheinander angewendet (KRUG 1997).

3. Verbesserung der Lösungsqualität des Verfahrens

   Die meisten effizienten Optimierungsverfahren zeigen einen Hang zur Konvergenz in (dem Startpunkt am nächsten liegenden) lokalen Optima, sind also lokale Verfahren. Durch den Einsatz von den Suchraum „erforschenden“ Optimierungsverfahren können zu Beginn der Optimierung erfolgversprechende Teilräume identifiziert werden, auf die sich die lokalen Verfahren anschließend konzentrieren (Siochi 1995). Dadurch erhöht sich die Wahrscheinlichkeit, bessere lokale Optima oder sogar das globale Optimum zu finden.

Wissensbasierte Optimierungsverfahren verwenden explizites Wissen über die zu lösende Aufgabe, um durch Konstruktion oder Modifikation optimale Lösungen zu erzeugen. Dabei kann es je nach Qualität und Umfang des verwendeten Wissens früher oder später zum sogenannten *Sudden-Death*-Phänomen kommen, bei dem die Optimierung abrupt abbricht, da für eine aufgetretene Situation kein anwendbares Wissen vorliegt. In solchen Fällen kann die weitere Optimierung durch andere, numerische Optimierungsverfahren fortgesetzt werden, wobei die Möglichkeit besteht, daß das wissensbasierte Verfahren in einer späteren Phase wieder einsetzbar wird. Ebenso ist es möglich, durch den Einsatz numerischer Optimierungsverfahren die Benutzung qualitativen Wissens zu erleichtern (Hader 1998a).

Bei der Entwicklung hybrider Optimierungsverfahren muß eine Reihe von Entwurfsentscheidungen getroffen werden. Die wichtigsten von ihnen sind:

- Welche Optimierungsverfahren sollen kombiniert werden ?
- Wie können die Optimierungsverfahren geeignet parametriert werden ?
- Wie sollen die Optimierungsverfahren interagieren (Reihenfolge der Anwendung, „Umschalt“-Bedingungen, usw.) ?

3 Realisierung hybrider Optimierungsverfahren durch Multiagentensysteme

Die im vorigen Kapitel erläuterten Vorteile der Hybridisierung von Optimierungsverfahren erwecken den Eindruck, als ob hybride Verfahren wegen der Verbesserung von Effizienz, Lösungsqualität und/oder Anwendungsbreite den klassischen, atomaren Verfahren in jedem Fall vorzuziehen wären. Dabei wird jedoch außer acht gelassen, daß diese Vorteile neben einem erhöhten Entwicklungsaufwand vor allem durch einen wesentlichen Nachteil erkauft werden: einer nicht-trivialen Parametrierung. Neben den Verfahrensparametern, die jedes der Teilverfahren besitzt (und die natürlich erhalten bleiben), kann ein hybrides Verfahren als Ganzes eine Vielzahl weiterer Verfahrensparameter aufweisen. Zu letzteren gehören vor allem Parameter, die die Anwendungsreihenfolge und -dauer der Teilverfahren festlegen.

Damit aus der Kombination verschiedener Optimierungsverfahren wirklich die gewünschten Vorteile erwachsen können, müssen für alle Verfahrensparameter eines hybriden Verfahrens möglichst gut geeignete Werte festgelegt werden. In traditionellen hybriden Optimierungsverfahren sind diese Werte entweder fest vorgegeben oder müssen vom Nutzer angegeben werden (vgl. das Konzept des *Optimierungsplans* von Powell, Skolnick und Tong (1990)). Bei der ersten Variante kann das hybride Verfahren nicht ausreichend an die betrachtete Aufgabenklasse bzw. die konkret zu lösende Aufgabe angepaßt werden, was zu einer Verschlechterung von Effizienz und Lösungsqualität führt. Bei der zweiten Variante ist der Nutzer, falls es sich nicht um einen Optimierungsexperten handelt, mit der Wahl der Verfahrensparameterwerte überfordert. Beide Varianten erschweren also den Einsatz von hybriden Optimierungsverfahren in der Praxis.

Der im folgenden vorgestellte Ansatz für eine Klasse *selbstadaptierender hybrider Optimierungsverfahren* löst das beschriebene Problem durch automatische Festlegung der Verfahrensparameterwerte während der Optimierung. In den verwendeten Algorithmus gehen dabei folgende Informationen ein:

1. eine formale Beschreibung der zu lösenden Aufgabe
   (Parameter, Nebenbedingungen, Zielfunktion u. ä.)
2. die für die aktuelle Optimierung verbleibende Zeit
3. Erfahrungen aus früheren Optimierungen ähnlicher Aufgaben
4. Erfahrungen aus dem bisherigen Verlauf der aktuellen Optimierung
5. Heuristiken über die Bestimmung geeigneter Verfahrensparameterwerte aus den obengenannten Informationen

Während die Informationsarten 1 und 2 (Beschreibung, Zeit) vom Nutzer initial und die Informationsart 5 (Heuristiken) vom Entwickler bereitzustellen sind, werden die Informationsarten 3 und 4 (Erfahrungen) vom hybriden Verfahren automatisch gesammelt. Besonderes Augenmerk legt der Parametrierungsalgorithmus auf die Einbeziehung der Größe der noch verbleibenden Zeit, die Rückschlüsse auf die Maximalzahl während der Optimierung noch untersuchbarer Entwürfe erlaubt. Auf Basis dieser Kenngröße können u. a. Entscheidungen über die als nächstes anzuwendenden Optimierungsverfahren getroffen werden[2].

Abbildung 2 zeigt den allgemeinen Aufbau der Steuerung des entwickelten Hybridansatzes. Wie bereits erwähnt, stellt der Nutzer eines hybriden Optimierungsverfahrens dieser Klasse eine formale Beschreibung der zu lösenden Aufgabe als Optimierungsproblem sowie die Größe der für die Optimierung maximal aufzuwendenden Zeit bereit. Aus der Problembeschreibung werden durch Analyse die wesentlichen explizit bestimmbaren Problemmerkmale abgeleitet. Dazu gehören u. a. Anzahl und Definitionsbereiche der Parameter sowie Aufbau von Zielfunktion und Nebenbedingungen (linear, quadratisch, usw.). Diese Merkmale werden zusammen mit einer Schätzung der Maximalzahl untersuchbarer Entwürfe bei der *strategischen Planung* der Optimierung verwendet, um einen groben Plan des Optimierungsprozesses aufzustellen. Der Plan unterteilt den Optimierungsprozeß in einzelne Phasen (z. B. Erforschungsphase, Konvergenzphase) und legt die dafür maximal aufzuwendenden Zeiten fest. Anschließend werden die einzelnen Phasen abgearbeitet. Die in der jeweils aktuellen Phase zu lösenden Teilprobleme und die dafür maximal aufzuwendenden Zeiten werden durch die *taktische Planung* sukzessiv festgelegt[3]. Für jedes zu lösende Teilproblem erfolgt die *Auswahl* eines geeigneten Verfahrens der Verfahrensbibliothek (meist aus mehreren potentiellen Kandidaten) sowie dessen *Parametrierung*. Beide Vorgänge werden wissensbasiert gesteuert, wobei insbesondere vom Entwickler vorgegebene Heuristiken sowie Erfahrungswissen aus früheren Optimierungen bzw. dem bisherigen Verlauf der aktuellen Optimierung genutzt werden

[2] Bei einer verbleibenden Maximalzahl von 100 untersuchbaren Entwürfen ist es z. B. nicht sinnvoll, einen Genetischen Algorithmus mit einer Populationsgröße von 200 anzuwenden. In dieser Situation eignet sich eher ein effizientes lokales Suchverfahren.

[3] Teilprobleme können z. B. von der Art „verbessere Entwurf X“ oder „löse das reduzierte Optimierungsproblem mit Entwurfsvariable $x_1 = 3.7$“ sein.

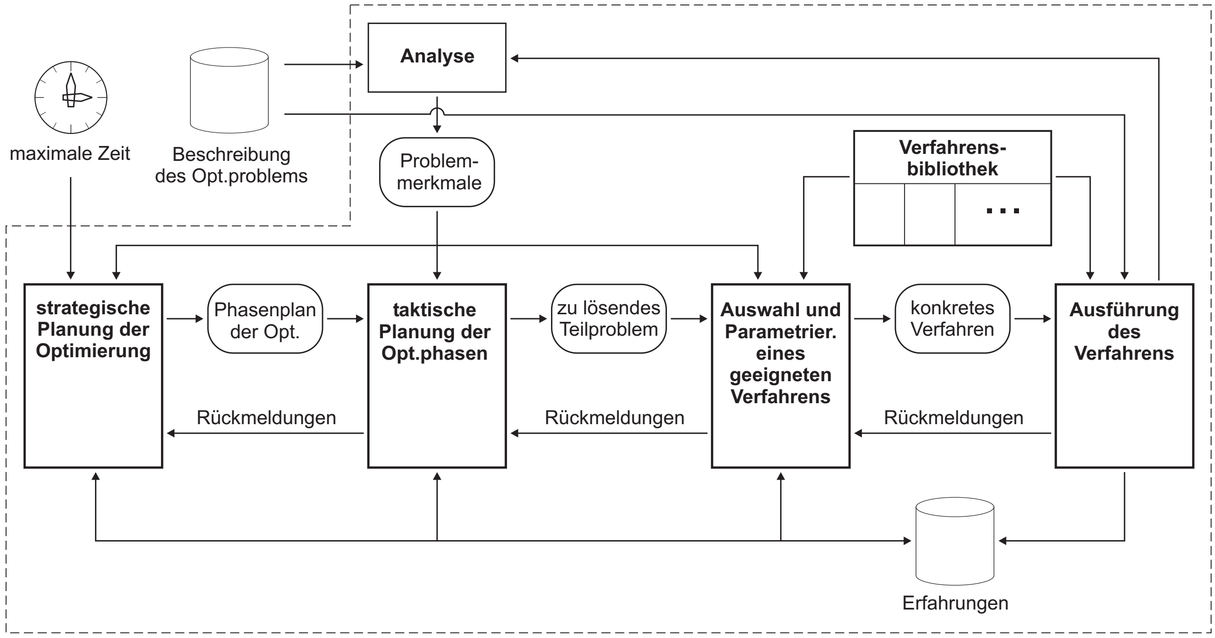

Abbildung 2: Aufbau der Steuerung des selbstadaptierenden Hybridansatzes

(Selbstadaptivität). Anschließend erfolgt die *Ausführung* des gewählten Verfahrens, die entweder zur Lösung des Teilproblems führt oder mit einem Mißerfolg endet. In letzterem Fall wird zur Auswahl zurückgekehrt, die versucht, ein alternatives Verfahren zu finden. Gelingt dies nicht, muß zur taktischen oder sogar zur strategischen Planung zurückgekehrt werden, um auf den Mißerfolg durch Umplanung zu reagieren.

Exkurs Multiagentensysteme:

> In den letzten beiden Jahrzehnten haben sich sogenannte *Multiagentensysteme*, die dem Fachgebiet der Verteilten Künstlichen Intelligenz entstammen, von einem theoretischen Forschungsgegenstand zu einer praktisch einsetzbaren Technologie entwickelt (siehe z. B. Müller (1993)). Ein Multiagentensystem besteht aus einer Ansammlung autonomer Systeme, die *Agenten* genannt werden. Jeder Agent besitzt spezielle Fähigkeiten sowie eigenes Wissen, Pläne und Ziele. Die Lösung von Problemen in Multiagentensystemen wird nicht zentral gesteuert, sondern ergibt sich dezentral durch die Prinzipien von Kooperation und Konkurrenz.

Der Gedanke liegt nahe, jedes der in der Verfahrensbibliothek enthaltenen Verfahren als einen Agenten zu modellieren. Dieser Agent enthält neben dem eigentlichen Verfahrensalgorithmus („Fähigkeit") auch Informationen über seine Eignung für die Lösung bestimmter Probleme und die dann jeweils günstigste Parametrierung („Wissen"). Die Informationen sind initial vom Entwickler vorzugeben, können danach jedoch entsprechend der gesammelten Erfahrung automatisch modifiziert werden. Die Gesamtheit der Verfahrensagenten bildet ein Multiagentensystem, das in der Lage ist, für ein von der taktischen Planung festgelegtes Teilproblem selbständig das am besten geeignete Verfahren zu ermitteln und auszuführen. Dazu bedient es sich eines verhandlungsbasierten Ansatzes, der als *Contract-Net-Protokoll* bezeichnet wird und auf Ausschreibung-Bewerbung-Zuschlag-Zyklen beruht. Näheres dazu bei Smith (1980) und Hader (1998c).

Die wesentlichen Vorteile der Realisierung der Komponenten „Auswahl und Parametrierung eines Verfahrens“ und „Ausführung eines Verfahrens“ durch ein Multiagentensystem sind:

- jedes Verfahren bildet zusammen mit dem Wissen (Heuristiken, Erfahrungen, usw.) über das Verfahren eine abgeschlossene Einheit und kann somit problemlos ausgetauscht werden
  $\longrightarrow$ leichte Erweiterbarkeit des hybriden Verfahrens
- da Multiagentensysteme ihrer Natur nach *verteilte Systeme* sind, können die einzelnen Verfahren verteilt und ggf. parallel auf verschiedenen Rechnern eines Netzes abgearbeitet werden
  $\longrightarrow$ erhöhte Effizienz des hybriden Verfahrens

Um Erkenntnisse über die beschriebene Klasse hybrider Optimierungsverfahren zu sammeln und insbesondere ihre Leistungsfähigkeit nachzuweisen, wurde das hybride Optimierungssystem MASCOT konzipiert, das im folgenden Kapitel vorgestellt wird.

4 MASCOT: Ein hybrides Optimierungssystem

MASCOT (**M**ulti-**A**gent **S**ystem for the **C**ombination of **O**ptimization **T**echniques) ist ein automatisches Optimierungssystem, das auf dem im vorigen Kapitel beschriebenen Ansatz zur Realisierung selbstadaptierender hybrider Optimierungsverfahren durch Multiagentensysteme basiert (Hader 1997, 1998b, 1998c). Das Anwendungsgebiet des Systems sind Aufgaben der statischen Parameteroptimierung mit einer Zielfunktion, wobei insbesondere Aufgaben gelöst werden können, deren Zielfunktion und/oder Nebenbedingungen durch zeitintensive externe Programme, speziell Simulationsprogramme, gegeben sind.

In MASCOT werden die hauptsächlichen Softwarekomponenten in Form von eigenständigen Programmen bereitgestellt, die zur Laufzeit als parallele Betriebssystem-Prozesse ausgeführt werden. Zur Verwaltung der einzelnen Prozesse wird das PVM-System (*Parallel Virtual Machine*) eingesetzt, das es ermöglicht, die Prozesse verteilt auf verschiedenen Rechnern eines Netzes auszuführen. Die Kommunikation zwischen den Prozessen erfolgt durch Austausch von Nachrichten via TCP/IP. Dabei wird ein auf der *Sprechakttheorie* basierendes Dialogkonzept verwendet, das u. a. festlegt, wie ein Prozeß auf den Empfang einer Nachricht A zu reagieren hat (i. allg. durch interne Aktionen sowie das Senden einer Nachricht B). Die beschriebene Systemarchitektur besitzt zwei wesentliche Vorteile:

1. jede Softwarekomponente kann auf der am besten geeigneten Hardwareplattform ausgeführt werden
2. Softwarekomponenten können problemlos ausgetauscht werden (in eingeschränktem Umfang sogar zur Laufzeit), wodurch das System offen für Erweiterungen und Anpassungen ist

Jede durch einen eigenen Prozeß repräsentierte Softwarekomponente von MASCOT wird als *Agent* angesehen und kann mit anderen Agenten kommunizieren und bestimmte Aufgaben, sogenannte *Tasks*, ausführen (wie „führe Programm X aus“ oder „löse Optimierungsproblem Y“). Alle Agenten zusammen bilden eine *Agentengesellschaft*, deren gemeinsames Ziel es ist, vom Nutzer vorgegebene Optimierungsprobleme in einer der aufwendbaren Zeit angemessenen Qualität zu lösen. Diesem Ziel ordnen sich alle Agenten unter, verhalten sich also altruistisch. Jeder mit der Lösung eines (z. B. von der taktischen Planung formulierten) Problems betraute Agent besitzt somit die Möglichkeit, das Problem zu zerlegen und alle oder einen Teil der entstehenden Teilprobleme mittels einer Ausschreibungsnachricht anderen Agenten zur Lösung anzubieten. Die angeschriebenen Agenten schätzen ihre Eignung zur Problemlösung selbst ein, wobei sie sich auf ihnen verfügbare lokale Informationen (Heuristiken, Erfahrungen, aktuelle Auslastung, u. ä.) stützen. Wenn sie sich für geeignet halten, schicken sie entsprechende Bewerbungsnachrichten zurück. Nach Eingang der Bewerbungen wird der am besten geeignete Agent (und bei dessen Scheitern der jeweils nächstbeste Agent) mit der Lösung des Teilproblems betraut.

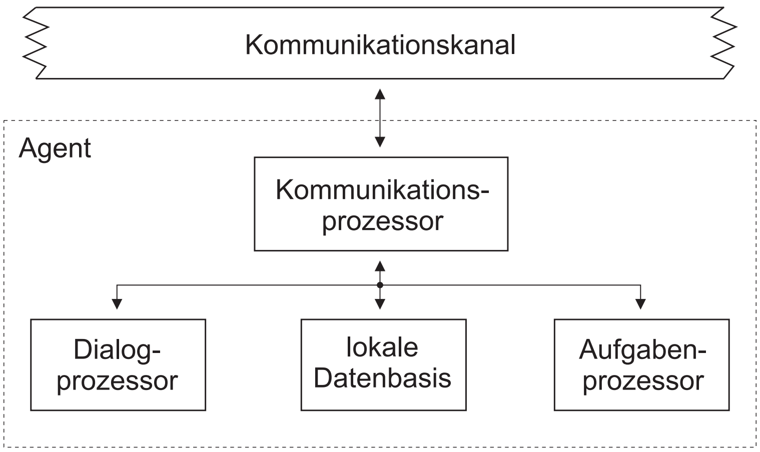

Abbildung 3: Aufbau eines MASCOT-Agenten

Um die beschriebene Funktionalität zu realisieren, besteht jeder Agent aus vier Komponenten: einem Kommunikationsprozessor, einem Dialogprozessor, einem Taskprozessor und einer lokalen Datenbasis (Abbildung 3). Der Kommunikationsprozessor ist für Empfang und Zustellung von Nachrichten via TCP/IP zuständig und unterstützt insbesondere die für fehlertolerante verteilte Systeme notwendigen Acknowledge- und Timeout-Mechanismen. Der Dialogprozessor steuert die mit anderen Agenten geführten Dialoge, indem er empfangene Nachrichten auswertet, entsprechende interne Tasks startet und zu sendende Nachrichten generiert. Der Taskprozessor steuert alle aktuell laufenden Tasks des Agenten über einen prioritätsbasierten Scheduler. Die lokale Datenbasis speichert alle lokal verfügbaren Informationen, z. B. über die Eignung für bestimmte Probleme, frühere Erfolge/Mißerfolge und die aktuelle Auslastung.

MASCOT-Agenten lassen sich einer der folgenden Kategorien zuordnen (Abbildung 4):

- Nutzerschnittstellen-Agenten
  Diese Agenten (meist genau einer) stellen eine Schnittstelle zwischen dem Nutzer und MASCOT zur Verfügung.
- Programmschnittstellen-Agenten
  Diese Agenten können externe Programme ausführen (z. B. Simulationsprogramme für Zielfunktionsberechnungen) und Daten mit diesen austauschen.
- Datenbank-Agenten
  Diese Agenten dienen der Verwaltung globaler Daten, wie z. B. aller bisher untersuchten Entwürfe.

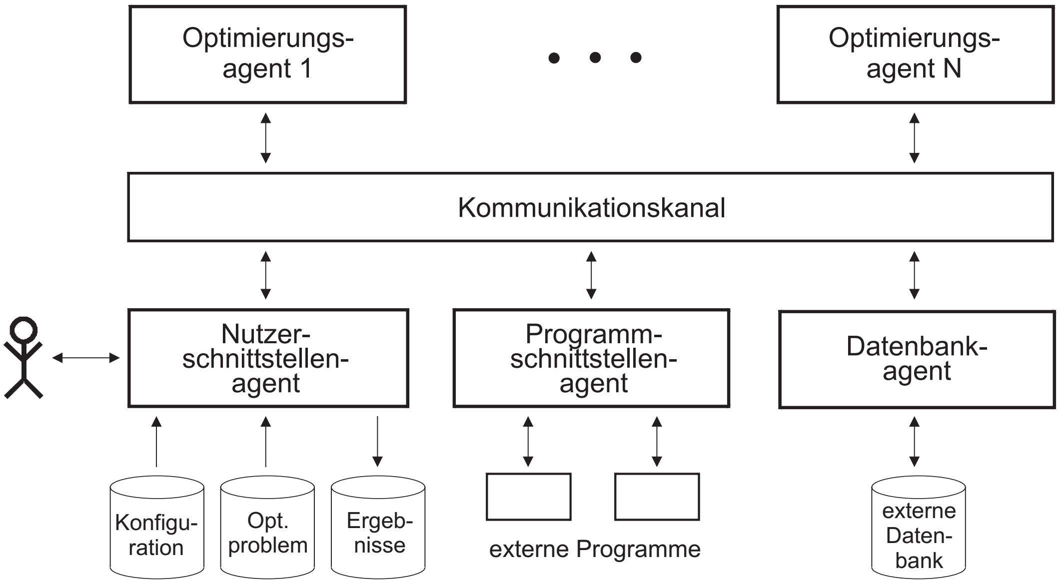

Abbildung 4: Architektur des Optimierungssystems MASCOT

- Optimierungs-Agenten
  Diese Agenten können Optimierungsprobleme durch Anwendung jeweils genau eines Verfahrens lösen. Neben numerischen Verfahren zur lokalen und globalen Optimierung sind auch wissensbasierte Verfahren vertreten, um explizites problemspezifisches Wissen in den Optimierungsprozeß einbeziehen zu können.

MASCOT befindet sich zur Zeit noch in der Entwicklungs- und Testphase. Es wurde bisher vorrangig zur Lösung von Problemen aus den Bereichen Logistik und Produktion eingesetzt. Um eine vollständige Praxistauglichkeit zu erreichen, müssen sowohl Effizienz und Robustheit als auch die Bedienung des Systems weiter verbessert werden.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde eine Klasse selbstadaptierender hybrider Optimierungsverfahren vorgestellt. Der Vorteil hybrider Verfahren besteht im Potential, eine bessere Effizienz und Lösungsqualität sowie einen größeren Anwendungsbereich als atomare Verfahren zu erreichen. Dieses Potential kann allerdings nur genutzt werden, wenn es gelingt, die gegenüber atomaren Verfahren stark vergrößerte Anzahl Verfahrensparameter möglichst gut einzustellen. Da diese Einstellung dem Nutzer, d. h. dem Ingenieur im Entwurfsprozeß, nicht zugemutet werden kann, wurde ein Ansatz beschrieben, der die Verfahrensparameter automatisch während der Optimierung einstellt bzw. anpaßt. Dieser Ansatz betrachtet alle im hybriden Optimierungsverfahren enthaltenen Teilverfahren als Agenten eines Multiagentensystems, die lokales Wissen über ihre Eignung für bestimmte Probleme sowie über die jeweils günstigen Verfahrensparameterwerte besitzen. Das Wissen wird initial vom Entwickler vorgegeben (Heuristiken o. ä.), kann aber durch Einbeziehung von Erfahrungen automatisch erweitert werden. Das für ein Problem jeweils am besten geeignete Teilverfahren kann dadurch während der Optimierung mittels eines Ausschreibung-Bewerbung-Zuschlag-Mechanismus dynamisch ermittelt und parametriert werden.

Das hybride Optimierungssystem MASCOT demonstriert einen Weg, den vorgestellten Ansatz in ein praxisorientiertes Softwaresystem zu überführen. Einige Aspekte des Ansatzes bedürfen allerdings weiterer Forschung. Dazu gehören u. a. theoretische und empirische Effizienz- und Konvergenzuntersuchungen hybrider Verfahren sowie die Entwicklung leistungsfähiger Methoden zur Nutzung automatisch gesammelten Erfahrungswissens bei der Steuerung zukünftiger Optimierungen.

Literatur

Gero, J.S. (1990), Design Prototypes: A Knowledge Representation Schema for Design, *AI Magazine*, 11(1990)2, pp. 26–36.

Hader, S. (1997), MASCOT: A Multiagent System for Hybrid Optimization, in: *Proceedings of the Third ISIR Summer School on Inventory Modelling in Production and Supply Chains*.

Hader, S. (1998a), *Das wissensbasierte Optimierungssystem DIM_EXPERTE*, Forschungsbericht, Technische Universität Chemnitz, Innovationskolleg INK 17/A1-1 „Bildung eines vernetzten Logistik- und Simulationszentrums“.

Hader, S. (1998b), Ein Multiagentensystem zur simulationsbasierten Optimierung, in: *Proceedings des Workshop SiWis '98 – Simulation in Wissensbasierten Systemen*.

Hader, S. (1998c), A Multiagent Simulation Optimization System, in: *Proceedings of the 10th European Simulation Symposium – ESS '98*, pp. 124–128.

Hart, W.E. (1994), *Adaptive Global Optimization with Local Search*, Ph. D. thesis, University of California, San Diego.

Krug, W. (1997), *Intelligentes Simulations- und Optimierungssystem für Prozesse der Fertigung, Organisation und Logistik – ARENA/ISSOP*, Society for Computer Simulation International, San Diego.

Müller, J. (Hrsg.) (1993), *Verteilte Künstliche Intelligenz – Methoden und Anwendungen*, BI Wissenschaftsverlag, Mannheim.

Pinto, I.V. (1989), *Wissensbasierte Unterstützung bei der Lösung von Optimierungsaufgaben*, Dissertation, Universität Dortmund, Fachbereich Informatik.

Powell, D.J., M.M. Skolnick, S.S. Tong (1990), EnGENous: A Unified Approach to Design Optimization, in: Gero, J.S., *Applications of Artificial Intelligence in Engineering V – Vol. 1: Design*, pp. 137–157.

Siochi, F.C. (1995), *Building a Knowledge-based Simulation Optimization System with Discovery Learning*, Ph. D. thesis, Virginia Polytechnic Institute and State University, Blacksburg.

Smith, R.G. (1980), The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Transactions on Computers*, 29(1980)12, pp. 1104–1113.

Syrjakow, M., H. Szczerbicka (1995), Simulation and Optimization of Complex Technical Systems, in: *Proceedings of the SCSC'95*, pp. 86–95.