Echtzeitaspekte bei der Aufgabenverteilung in selbst-organisierenden autonomen Systemen

Gerhard Fuchs, Falko Dressler

Universität Erlangen-Nürnberg
Lehrstuhl für Rechnernetze und Kommunikationssysteme (Informatik 7)
Martensstr. 3, 91058 Erlangen, Germany
{gerhard.fuchs,dressler}@informatik.uni-erlangen.de

Zusammenfassung. Aufgabenverteilung ist eine wichtige Kernaufgabe in selbst-organisierenden autonomen Systemen. Aus den unterschiedlichen Einsatzgebieten derartiger Systeme resultieren verschiedene Anforderungen und Problemstellungen. In diesem Paper werden Echtzeitaspekte betrachtet. Es wird ein Verfahren beschrieben, mit dem die für die Verteilung von Aufgaben benötigte Zeit und die Verweildauer der Aufgaben im System bestimmt werden kann. Der Ansatz beruht darauf, dass ein Koordinator die Verwaltung einer Aufgabe übernimmt. Ihm werden sowohl die Ankunft, als auch der Beginn und das Ende der Ausführung angezeigt. Auf diese Weise können die gesuchten Zeiten durch einfache Subtraktion ermittelt werden. Das Prinzip wurde experimentell und simulativ mit einem auf MURDOCH (lokale Statusverwaltung), bzw. OAA (zentrale Statusverwaltung) basierenden Aufgabenverteilungsmechanismus getestet. Dadurch konnten erste Einblicke in das Echtzeitverhalten bei Koordinierungsfragen in autonomen Systemen gewonnen werden.

1 Einleitung

Selbst-organisierende autonome Systeme wie z.B. Sensornetze und Roboterschwärme setzten sich aus mehreren einzelnen Knoten zusammen, die von außen betrachtet als eine Einheit agieren. Wird eine Aufgabe an ein solches System gestellt, sollen sich die einzelnen Knoten ohne weiteres Zutun koordinieren und diese lösen. Hierbei ist die Aufgabenverteilung, also welcher Knoten was zum Gesamtziel beiträgt, eine wichtige Kernaufgabe. Der hierfür verwendete Mechanismus hat maßgeblichen Einfluss auf die Leistungsfähigkeit des gesamten Systems. Je nach Einsatzgebiet und Anwendung werden unterschiedliche Anforderungen an das System gestellt und es treten verschiedene Problemstellungen (u.a. Energie-, Kommunikations-, Echtzeitaspekte …) in den Vordergrund.

Ziel des Papers ist die Beschreibung eines Verfahrens, mit dem die für die Verteilung von Aufgaben benötigte Zeit und die Verweildauer der Aufgaben im autonomen System, bestimmt werden kann.

Das nachfolgende Paper gliedert sich wie folgt: In Abschnitt 2 wird die Einbettung der Untersuchungen in unser Forschungsvorhaben und einige Beispiele des Einsatzes

von Aufgabenverteilung im Umfeld von mobilen Sensornetzen aufgezeigt. Abschnitt 3 beschreibt zwei exemplarische Aufgabenverteilungsmechanismen. Diese werden mit dem in Abschnitt 4 vorgestellten Verfahren sowohl simulativ, als auch im Experiment untersucht. Die Ergebnisse sind in Abschnitt 5 dargestellt. In Abschnitt 6 werden die gewonnenen Erkenntnisse diskutiert und zusammengefasst.

2 Einbettung in Forschungsvorhaben und related Work

2.1 ROSES

Die primären Forschungsziele der Autonomic Networking Gruppe des Lehrstuhls für Rechnernetze und Kommunikationssysteme sind das Entwickeln und Beherrschen von selbst-konfigurierenden, selbst-organisierenden und adaptiven Methoden für Kommunikationssysteme und interagierende autonome Systeme. Unter diesen Gesichtspunkten wird im Rahmen des ROSES[1]-Projekts (Robot assisted Sensor Networks) eine Kombination aus mobilen Robotern und stationären Sensornetzen untersucht. Wir unterscheiden zwischen Sensornetzwerk unterstützten Roboterschwärmen, d.h., dass die Roboter das Sensornetzwerk z.B. zur präziseren Lokalisierung verwenden [3] und Roboter unterstützten Sensornetzen, d.h., dass die mobilen Roboter das Sensornetzwerk z.B. rekonfigurieren [4].

Unsere Gruppe forscht u.a. an der Optimierung der Aufgabenverteilung in derartigen Systemen. Neben Energiebetrachtungen [2] ist das Echtzeitverhalten, im Speziellen das Einhalten von Zeitschranken, ein wichtiger Aspekt.

2.2 Related Work

[5] gibt einen Überblick zu existierenden Aufgabenverteilungsmechanismen in Multi-Roboter-Systemen, klassifiziert und bewertet sie bezüglich ihrer Komplexität. Echtzeitfähige Verteilungsmechanismen in diesen Systemen werden z.B. in [1, 8] beschrieben. Ein Beispiel für Aufgabenverteilung in Sensornetzen ist [9], für mobile Sensornetze [6].

3 Eigenschaften der Aufgabenverteilungsmechanismen

3.1 MURDOCH

Der in Abb. 1 gezeigte Mechanismus ist an MURDOCH [5] angelehnt. Der Koordinator verwaltet keine Statusinformationen und weiß demnach nicht, welche Roboter sich in seiner Umgebung befinden. Die einzelnen Roboter können anhand von loka-

[1] Projekt-Homepage: http://www7.informatik.uni-erlangen.de/~dressler/projects/roses/

len Profilen (P) beurteilen, wie gut sie eine Aufgabe ausführen können. Die Aufgaben werden nach dem FIFO-Prinzip verteilt.

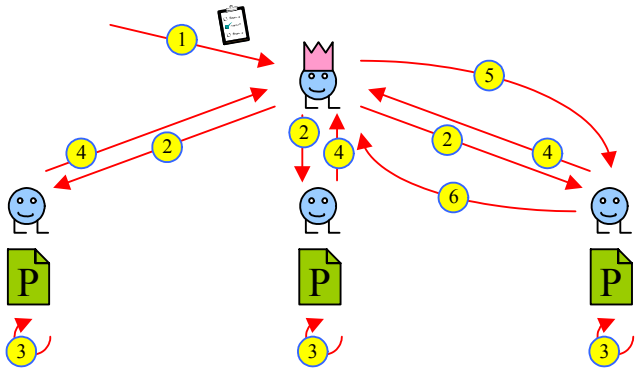

Abb. 1. MURDOCH, verteilte Profilverwaltung

1. Ankunft der zu lösenden Aufgabe beim Koordinator; die Aufgabe wird in eine Warteschlange abgelegt.
2. Ist die Aufgabe an der Reihe, wird sie per Broadcast an alle erreichbaren Roboter weitergeleitet (**task announcement**).
3. Die Roboter berechnen anhand ihrer Profile (P) lokal ihre Fitness (**metric evaluation**).
4. Können die Roboter die Aufgabe ausführen, geben sie ein ihrer Fitness entsprechendes Gebot ab (**bid submission**).
5. Der Koordinator ermittelt nach Ablauf einer festen Auktionszeit den Gewinner, also den Roboter, der die höchste Fitness geboten hat (**close of auction**) und benachrichtigt diesen. Hat die Auktion keinen Gewinner hervorgebracht, wird sie wiederholt (2).
6. Der Gewinner startet die Aufgabe und benachrichtigt den Koordinator über den Start und das Ende der Ausführung.

3.2 Open Agent Architecture (OAA)

Der in Abb. 2 beschriebene Mechanismus ist an [7] angelehnt. Der Koordinator verwaltet Statusinformationen, d.h. er erfragt und verwaltet die Profile der einzelnen Roboter zentral. Weiterhin weiß er Bescheid, welche Roboter gerade Aufgaben ausführen können. Um diese Informationen aktuell zu halten, ist ein zusätzlicher Koordinations- und somit Kommunikationsaufwand nötig. Die Aufgaben werden nach dem FIFO-Prinzip verteilt.

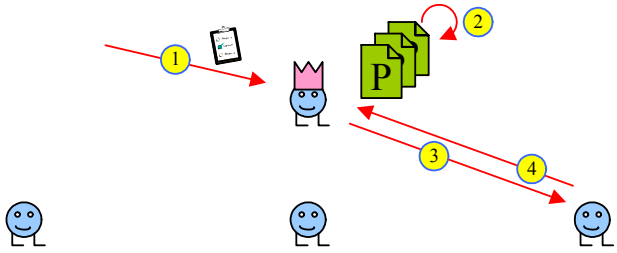

Abb. 2. Open Agent Architecture (OAA), zentrale Profilverwaltung

1) Ankunft der zu lösenden Aufgabe beim Koordinator.
2) Der Koordinator ermittelt anhand der ihm vorliegenden Profile den für die Lösung der Aufgabe am besten geeigneten Roboter.
3) Dem ausgewählten Roboter wird die Aufgabe übertragen.
4) Er startet die Ausführung und zeigt dies dem Koordinator an. Außerdem meldet er das Ende, so dass nur freie Roboter vom Koordinator berücksichtigt werden.

4 Quantitative Untersuchungsverfahren

4.1 Prinzip der Zeitmessung

Das Prinzip der Zeitmessung wird in Abb. 3 gezeigt: Ein Aufgabengenerator wählt aus einer endlichen Menge verschiedene Aufgaben aus und schickt eine Anfrage an den Koordinator (A). Mit Hilfe des Aufgabenverteilungsmechanismus wird aus den vorhandenen Robotern der am besten geeignetste ausgewählt und die Anfrage an diesen weitergeleitet (C). Der Roboter startet die Aufgabe und meldet dies dem Koordinator (F). Das Ende der Ausführung wird ebenfalls dem Koordinator mitgeteilt (G). Da alle Zeitmessungen an einem einzigen Knoten erfolgen, kann auf eine Synchronisation der Uhren im System verzichtet werden.

Definitionen:

- Aufgabenverteilzeit T_{AV} Zeit zwischen (B) und (D)
- Verweildauer T_{V} Zeit zwischen (B) und (E)
- Anfrageabstand T_{ANF} Zeit zwischen (A) und (H)
- Ausführungszeit T_{EX} Zeit zwischen (F) und (G)

Eine Anfrage des Aufgabengenerators setzt sich aus einem Aufgabennamen `X` und einer maximalen Ausführzeit T_{MAX} zusammen. Auf den Robotern befinden sich jeweils ein Profil `P(X)`→`[0,1]`, das jedem `X` einen Wert zwischen 0 und 1 zuordnet, der aussagt, wie gut die Aufgabe gelöst werden kann (0 ist schlecht, 1 ist gut). Bekommt ein Roboter eine Aufgabe zugeteilt, ist er für die Zeit $T_{EX} = T_{MAX} * (1-P(X))$ blockiert und kann keine weiteren Aufgaben ausführen.

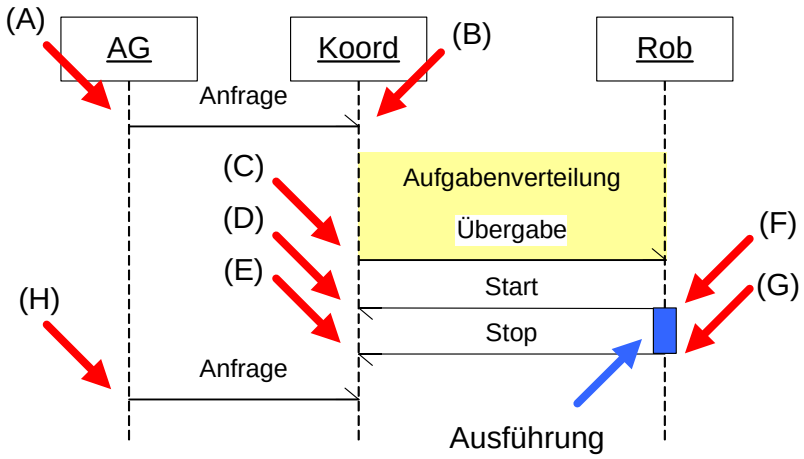

Abb. 3. Prinzip der Zeitmessung (AG: Aufgabengenerator; Koord: Koordinator; Rob: Roboter)

4.2 Experimentelle Umsetzung

Bei der experimentellen Umsetzung laufen der Aufgabengenerator und der Koordinator auf einem handelsüblichen PC. Als Roboter kommen 3 Robertino[2]-Plattformen zum Einsatz. Es existieren 10 verschiedene Aufgaben aus denen bei den Anfragen der Reihe nach ausgewählt wird. Die Profile sind zufällig gewählt.

4.3 Simulationsmodell

Das Simulationsmodell ist mit AnyLogic[3] erstellt. Es werden 100 Roboter simuliert. Um zu verhindern, dass die Ausführungszeiten bei den Robotern zu 0 wird (bei 100 Robotern sehr wahrscheinlich), gilt für die Profile `P(X)→[0,0.9]`. Es existieren 10 verschiedene Aufgaben, aus denen bei den Anfragen zufällig ausgewählt wird. Die Profile sind zufällig gewählt.

5 Messergebnisse

5.1 Experimentelle Untersuchung

Bei der experimentellen Messung wurden die Parameter so gewählt, dass das mit OAA arbeitende System gut ausgelastet, das mit MURDOCH arbeitende leicht über-

[2] Projekt-Homepage: http://www.openrobertino.org
[3] Homepage: http://www.xjtek.com

lastet ist. Dementsprechend steigt bei MURDOCH die gegen die Sequenznummer (Reihenfolge beim Aufgabeneingang) aufgetragene Aufgabenverteilzeit tendenziell leicht an, während sie bei OAA auf nahezu gleichem Niveau bleibt (Abb. 4). Abb. 5 zeigt die nach Aufgabentyp sortierte durchschnittliche Verweildauer der Aufgaben im System. Sie ist bei MURDOCH deutlich höher als bei OAA. Das Ergebnis kann mit der Simulation verifiziert werden.

5.2 Simulation

Bei der Simulation wurden die beiden Systeme einmal unter mittlerer und einmal unter extremer Belastung getestet. Beim 1. Simulationsexperiment entspricht die Aufgabenverteilzeit der Zeit, die die Mechanismen für die Auswahl brauchen. OAA bestimmt den Sieger sofort, MURDOCH erst nach einer Sekunde Auktion (Abb. 6). Die durchschnittliche Verweildauer der einzelnen Aufgaben im System unterscheidet sich ebenfalls genau um diese Zeitspanne (Abb. 7), d.h., bei OAA ist sie um eine Sekunde kürzer als bei MURDOCH.

Beim 2. Simulationsexperiment wurde die Auktionszeit von MURDOCH verlängert. Hierdurch haben mehrere Knoten die Möglichkeit, sich an der Auktion zu beteiligen. Das System ist nach ca. 1000s überlastet, d.h. alle Knoten sind mit Aufgaben belegt und die Anfragen werden in eine Warteschlange eingefügt. Die Aufgabenverteilzeit steigt bei beiden Systemen ab diesem Punkt linear an, jedoch bei OAA steiler. Vor der Überlastung ist das gleiche Verhalten wie beim Simulationsexperiment 1 zu beobachten (Abb. 8). Die durchschnittliche Verweildauer der einzelnen Aufgaben im System ist bei OAA höher als bei MURDOCH (Abb. 9).

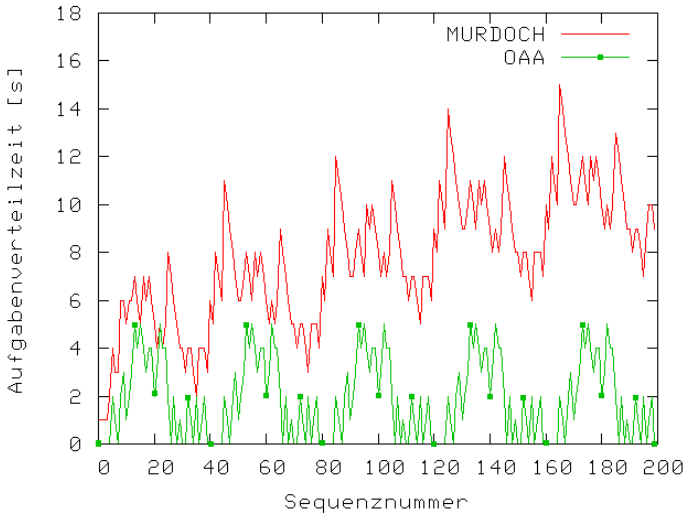

Abb. 4. Experimentelle Untersuchung - Aufgabenverteilzeit (T_{ANF} = 2s; T_{MAX} = 11s; Anzahl der Anfragen = 200; Auktionszeit von Murdoch = 1s)

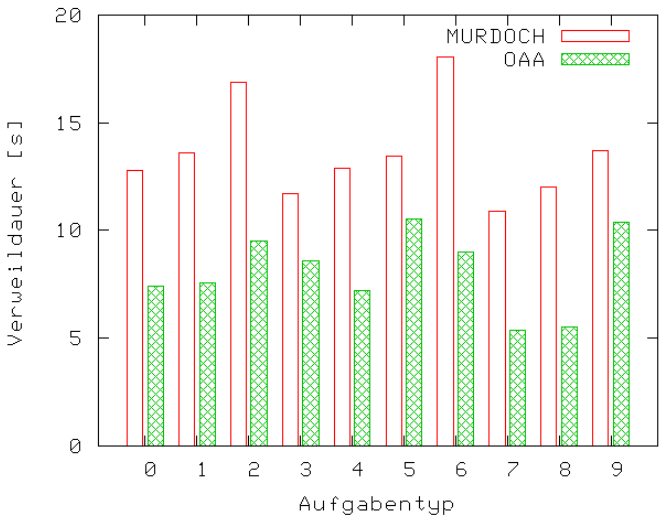

Abb. 5. Experimentelle Untersuchung - Verweildauer der Aufgaben im System (T_{ANF} = 2s; T_{MAX} = 11s; Anzahl der Anfragen = 200; Auktionszeit von Murdoch = 1s)

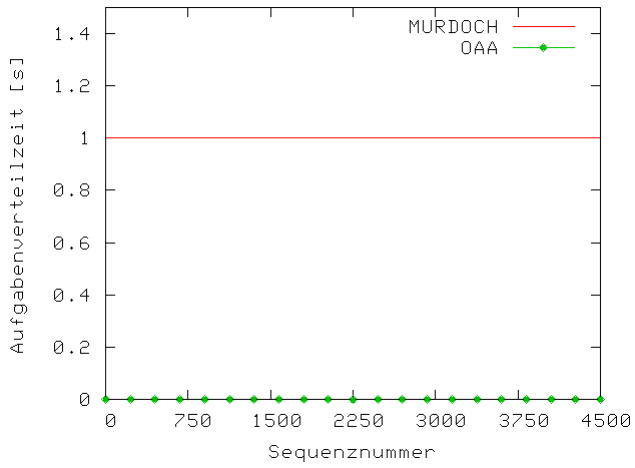

Abb. 6. Simulation - Aufgabenverteilzeit (T_{ANF} = 2s; T_{MAX} ϵ [100, 900] – für jede Anfrage zufällig ausgewählt; Anzahl der Anfragen = 4500; Auktionszeit von Murdoch = 1s)

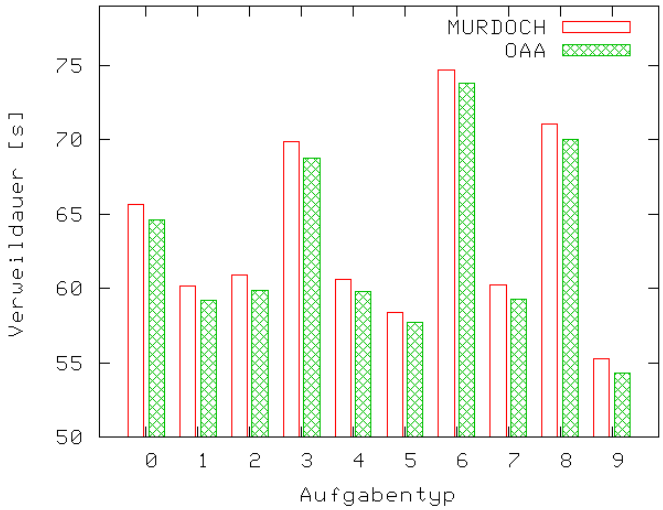

Abb. 7. Simulation - Verweildauer der Aufgaben im System (T_{ANF} = 2s; T_{MAX} ϵ [100, 900] – für jede Anfrage zufällig ausgewählt; Anzahl der Anfragen = 4500; Auktionszeit von Murdoch = 1s)

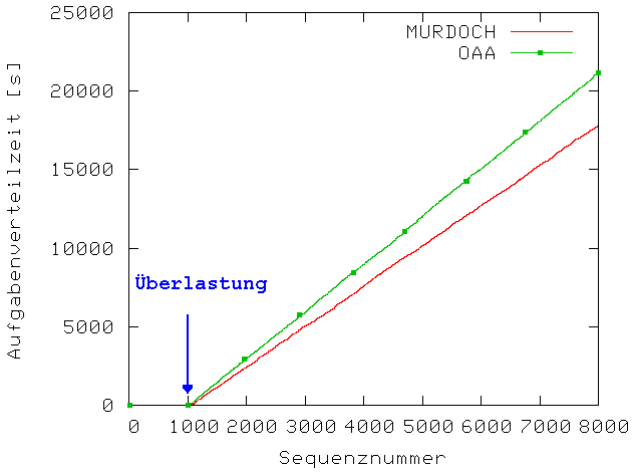

Abb. 8. Simulation - Aufgabenverteilzeit (T_{ANF} = 2,2s; T_{MAX} = 600s; Anzahl der Anfragen = 8000; Auktionszeit von Murdoch = 2s)

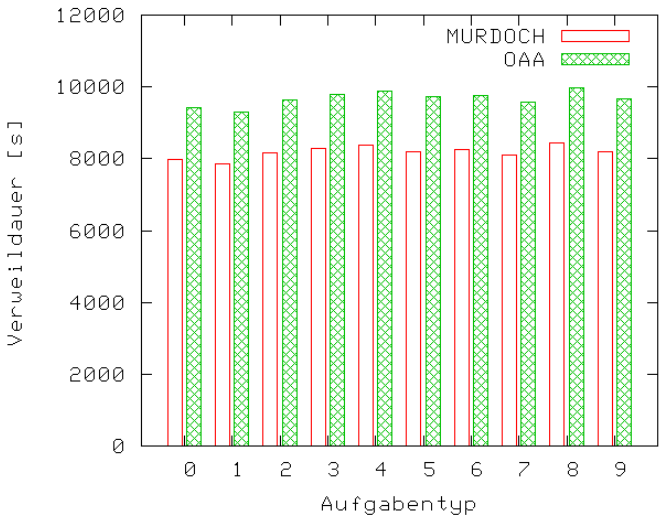

Abb. 9. Simulation - Verweildauer der Aufgaben im System (T_{ANF} = 2,2s; T_{MAX} = 600s; Anzahl der Anfragen = 8000; Auktionszeit von Murdoch = 2s)

6 Diskussion und Zusammenfassung

Mit dem hier vorgestellten quantitativen Untersuchungsverfahren ist es möglich, die zur Aufgabenverteilung benötigte Zeit und die Verweilzeit von Aufgaben in autonomen Systemen zu bestimmen. Dadurch dass ein Koordinator eine Aufgabe verwaltet, können an einem einzigen Knoten alle wichtigen Zeiten gemessen werden. Für diese Methode müssen die Uhren der einzelnen Knoten im System also nicht unbedingt synchron laufen.

Dieses Untersuchungsverfahren wurde experimentell und simulativ angewendet, um einen auf OAA (zentrale Statusverwaltung) und einen auf MURDOCH (lokale Statusverwaltung) basierenden Mechanismus im Zeitverhalten zu verglichen. Vom Prinzip her sind beide Mechanismen gleich. Ein Koordinator wählt aus einer Menge freier Knoten den für eine Aufgabe am besten geeignetsten aus und übergibt diesem die Aufgabe. Der Unterschied liegt darin, dass bei OAA ein zentraler Status des Gesamtsystems geführt wird. Somit kann ohne langwierige Auktion ein freier Knoten ausgewählt werden. Der Vorteil von MURDOCH liegt darin, dass nicht der erst beste Knoten zur Lösung der Aufgabe verwendet, sondern eine sorgfältigere Auswahl getroffen wird. Hierdurch können vor allem in Überlastsituationen bessere Ergebnisse erzielt werden.

Zusammenfassend gesagt, konnten wir erste Einblicke in das Echtzeitverhalten bei Koordinierungsfragen in autonomen Systemen gewinnen.

Literatur

1. M. Berhault, M. G. Lagoudakis, P. Keskinocak, A. J. Kleywegt, and S. Koenig, "Auctions with Performance Guarantees for Multi-Robot Task Allocation," Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, September 28 - October 2 2004.
2. F. Dressler and G. Fuchs, "Energy-aware Operation and Task Allocation of Autonomous Robots," Proceedings of 5th IEEE International Workshop on Robot Motion and Control (IEEE RoMoCo'05), Dymaczewo, Poland, June 2005, pp. 163-168.
3. F. Dressler, "Sensor-Based Localization-Assistance for Mobile Nodes," Proceedings of 4. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze, Zurich, Switzerland, March 2005, pp. 102-106.
4. G. Fuchs, S. Truchat, and F. Dressler, "Profilbasierte Rekonfiguration kleiner Endgeräte - Roboter unterstützte Sensornetzwerke," Proceedings of 2. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste, Stuttgart, Germany, June 2005, pp. 50-53.
5. B. P. Gerkey, "On Multi-Robot Task Allocation," Faculty of the Graduate School, University of southern California, 2003.
6. K. H. Low, W. K. Leow, and M. H. Ang, "Autonomic Mobile Sensor Network with Self-Coordinated Task Allocation and Execution," *IEEE Transactions on Systems, Man, and Cybernetics--Part C: Applications and Reviews*, 2005 *(accepted for pubilcation)*.
7. D. Martin, A. Cheyer, and D. Moran, "The Open Agent Architecture: a framework for building distributed software systems," *Applied Artificial Intelligence*, vol. 13, pp. 91-128, 1999.
8. S. Sariel and T. Balch, "Real Time Auction Based Allocation of Tasks for Multi-Robot Exploration Problem in Dynamic Environments," Proceedings of The Twentieth National Conference on Artificial Intelligence (AAAI), Integrating Planning into Scheduling Workshop, Pittsburgh, Pennsylvania, July 2005.
9. M. Younis, K. Akkaya, and A. Kunjithapatham, "Optimization of Task Allocation in a Cluster-Based Sensor Network," Proceedings of 8th IEEE International Symposium on Computers and Communications, Kemer-Antalya, Turkey, June 2003, pp. 329-340.