Rheinisch-Westfälische Technische
Hochschule Aachen
Lehrstuhl für Informatik IV
Prof. Dr. rer. nat. Otto Spaniol

QoS in Ad-Hoc-Netzwerken

Seminar Datenkommunikation und
Verteilte Systeme
WS 2001/02

Marc Wolter
marc.wolter@post.rwth-aachen.de
Matrikelnummer: 219921

Betreuung: Dipl.-Inform. Mesut Günes
Lehrstuhl für Informatik IV, RWTH Aachen

Inhaltsverzeichnis

Zusammenfassung

Diese Arbeit beschäftigt sich mit der Qualität in Ad-Hoc-Netzwerken und wie sie sichergestellt werden kann. Die Verwaltung der Ressourcen eines dezentralen Netzwerks (wie z.B. des Internets, aber auch in mobilen Netzwerken), dessen Kanäle zeitvariierend sind, ist eine schwierige Aufgabe, die aber in vielen Aspekten der heutigen Netzwerktechnik eine grosse Rolle spielt. Das grosse Problem ist die Unwissenheit der einzelnen Teilnehmer über die Anzahl der Mitbenutzer, deren Verhalten sowie die Art und Struktur des Netzes und sonstige Umwelteinflüsse. Trotz all dieser Ungewissheiten ist es aber möglich in bestimmten Situationen nahezu optimale Ergebnisse zu erlangen.

1 Einleitung

In dieser Arbeit wird ein Lösungsversuch für die optimale Ressourcenverwaltung anhand von lernenden Automaten vorgeschlagen, der sich auf die Theorie von nicht-kooperativen Spielen stützt. Da diese Theorie schon seit langem untersucht wird, können bekannte Sätze verwendet werden, um dieses neue Problem zu lösen. Wie bei (fast) jedem Spiel haben dabei die Benutzer Strategien, nach denen sie spielen, um in diesem Spiel besonders gut abzuschneiden. Es zeigt sich, das die optimale Strategie in einem asynchronen Netzwerk mit Umwelteffekten in fester Zeit erreicht werden kann. Die Schnelligkeit der Konvergenz ist leider immer noch ein offenes Problem.

Das verwendete Modell soll kurz vorgestellt werden:
Es sei angenommen, das m Benutzer an n Empfänger senden. Dazu existiert eine zeitvariante Verbindungsmatrix, die für jede Verbindung einen Wert (entweder $0 =$ Verbindung existiert, $1 =$ Verbindung getrennt oder einen Wert $\in [0, 1]$ für die Güte der Verbindung) beinhaltet, deren Werte aber ungewiss sind, da sie ja in der Zeit variieren. Man betrachtet nun die Benutzer als Spieler, die versuchen mit den ungewissen Daten (Verbindungen und Umwelt) eine für sich möglichst maximale Übertragung aufzubauen (der sogenannte *payoff*). Jeder Spieler benutzt dabei mehrere Strategien, deren Wirksamkeit er sich merkt und so im Laufe der Zeit schwächere Strtageien eliminiert, bis nach einer gewissen Zeit nur noch die stärkste, soll heissen beste Strategie vorhanden ist. In diesem Zeitpunkt spricht man von einem Gleichgewicht (das sog. $Nash - Gleichgewicht$), wenn also alle Benutzer die für sie günstigste Strategie gefunden haben.

Diese Übertragung von Ad-Hoc-Netzwerken zu (nicht-) kooperativen Spielen ist recht intuitiv und erlaubt es, vorhandene Hilfsmittel der Spieltheorie einzusetzen.

Kapitel 2 liefert eine Einführung in die Spieltheorie und erläutert die verwendeten Begriffe, die zum Beschreiben von nicht-kooperativen Spielen notwendig sind, daruf aufbauend wird in Kapitel 3 das verwendete Modell für Ad-Hoc-Netzwerke genauer vorgestellt. Kapitel 4 beschreibt einen Lösungsversuch für das Spiel anhand von lernenden Automaten, deren Defintion und Verhalten aufgezeigt werden.

2 Nicht-kooperative Spiele

Gegenstand der Spieltheorie sind interaktive Strategien von Parteien, die entgegengesetzte oder konfligierende Interessen verfolgen. Diese Situation liegt auch in einem Ad-Hoc–Netzwerk vor. Bevor einige wichtige Begriffe zu nicht-kooperativen Spielen folgen, kurz erst einmal ein einfaches Beispiel, was nicht-kooperative Spiele überhaupt sind:

2.1 Das Gefangendilemma

Zwei Gefangene, deren Namen A und B lauten, sind verdächtig, gemeinsam eine Straftat begangen zu haben und hatten das Pech, von der Polizei in Gewahrsam genommen worden zu sein. Die Höchststrafe für das Verbrechen beträgt 5 Jahre. Der Richter macht jedem der beiden folgendes Angebot: „Wenn Du auspackst und somit deinen Partner belastest, kommst Du ohne Strafe davon und er muss die vollen 5 Jahre absitzen. Wenn Ihr beide schweigt, haben wir genügend Indizienbeweise, um euch für 2 Jahre einzusperren, wenn Ihr beide gesteht, müsst ihr 4 Jahre eures Lebens hier verbringen.“

Jeder der Gefangenen hat zwei Möglichkeiten (Strategien): schweigen oder gestehen oder (aus Sicht des anderen Gefangenen) kooperieren oder sabotieren. Jede Strategie liefert, abhängig von der gewählten Strategie des Mitgefangenen, ein unterschiedliches Ergebnis. Die beiden befinden sich in getrennten Zellen, sie können also nicht kooperieren und sich nicht absprechen, sind also völlig auf sich alleine gestellt.
Die Werte für jede der vier möglichen Zugkombinationen sind in einer *payoff*-Matrix angegeben, und zwar in der Form (Gefängnisstrafe für A, Gefängnisstrafe für B). Punkte in diesem Spiel werden in ersparten Jahren Gefängnis gemessen. Jeder der beiden Gefangenen will seinen Gewinn maximieren, d.h. ihn interessiert nicht, wie viele Punkte der Gegner bekommt, sondern nur sein eigenes Abschneiden. Die *payoff* Matrix:

B A	schweigen	gestehen
schweigen	(-2,-2)	(-5,0)
gestehen	(0,-5)	(-4,-4)

Zum Beispiel erhält jeder Spieler drei Punkte, wenn beide schweigen, da beide nur zwei Jahre ins Gefangnis kommen anstatt der üblichen fünf für das Verbrechen. Andererseits bekommt der Spieler, der gesteht, fünf Punkte falls sein Mitgefangener die unglückliche Entscheidung macht, dicht zu halten und somit die volle Strafe absitzt (= 0 Punkte).

Wenn man nun die möglichen Strategien betrachtet, so stellt sich heraus, das Sabotieren eine gute Strategie ist, da man, egal was der Gegner macht, nie Verlust macht. Verhalten sich also beide Parteien nach dieser Strategie, so bekommt zwar jeder immer einen Punkt, das Ergebnis selbst ist aber suboptimal, da man höhere Freiheitsgewinne erzielen könnte, wenn man seinen Partner ab und zu reinlegt.

Dieses Beispiel zeigt die Vielseitigkeit und die Schwierigkeit von nicht-kooperativen Spielen. Die Spieltheorie selber kam durch das Buch "Games and Economic Behaviour"von John v. Neumann und Oscar Morgenstern 1944 zu wissenschaftlicher Relevanz und ist seitdem stark weiterentwickelt und erforscht worden, wodurch sie ein geeignetes Mittel zur Modellierung darstellt. Nicht-kooperative Spiele sind in vielen wissenschaftlichen Bereichen wie z.B. der Betriebswirtschaft oder Biologie von Interesse, um hier z.B. das Verhalten an Märkten zu simulieren, bei denen ja auch der Mitbewerber (im Normalfall) keinen Einblick auf andere Strategien hat oder das Ergebnis seiner Bemühungen im Voraus detailliert berechnen kann.

2.2 Grundlegende Begriffe

Definition 2.1 Ein Spiel $G = (I, \Sigma, r_1, \ldots, r_n)$ besteht aus einer Menge I von n Spielern, einer Strategiemenge Σ sowie einer *payoff* - Funktion r_i für jeden Spieler, wobei $\Sigma_i \in \Sigma$ die Menge der Strategien σ_i ist, die der Spieler i spielen darf.

Wie eine Strategie explizit aussieht, sei hier offen gelassen, denn es hängt stark von dem verwendeten Modell ab. Im folgenden werden noch die Begriffe *reine* und *gemischte* Strategie verwendet, wobei *reine* Strategien einfache Strategien sind, die in keiner Form bewertet sind und *gemischte* Strategien mit einer Wahrscheinlichkeit verbunden sind, dass sie gespielt werden (wie es in Kapitel 4 benötigt wird).
Zu unterscheiden sind noch synchrone (one-shot) oder asynchrone Spiele. Bei ersteren wählen alle Spieler gleichzeitig ihre Strategie (am Anfang jeder Runde), während bei asynchronen Spielen jeder Spieler seine Strategie zu jedem Zeitpunkt beliebig ändern kann.

Definition 2.2 Das *Strategieprofil* oder die Strategiekonfiguration $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ ist das Resultat aller Strategiewahlen und induziert ein Spielergebnis (*outcome*), das jeder Spieler anhand seiner Auszahlungsfunktion r_i bewertet. $\sigma(i)$ ist hiermit gleichbedeutend mit σ_i, also der Strategie des Spielers i. In einem zeitdiskreten System wird statt σ auch σ^t geschrieben, was das Strategieprofil zum Zeitpunkt t beschreibt.

Definition 2.3 Ein *nicht-atomares* Spiel ist ein Spiel, in dem kein einzelner Spieler die anderen Spieler nur durch Änderung seiner Aktionen beeinflussen kann. Ein nicht-atomares externes Spiel (NAEG) ist ein nicht-atomares Spiel mit m Klassen von Spielern und submodularen *payoff* Funktionen.

Die Restriktion, die hier an den *payoff* gestellt wird, ist, das der *payoff* nur von der eigenen gewählten Strategie $\sigma(i)$ und einer Umgebung $f(\sigma)$ abhängt. Die Umgebung f entsteht durch die Aktionen der anderen Spieler und sonstigen Umwelteinflüssen. Ausserhalb des Modells würde man die Umgebung einfach Interferenz nennen.
Submodular bedeutet hier, das die Differenzen abnehmen, also $\forall \sigma, \sigma' \in \Sigma_i$ und $\sigma > \sigma'$ gilt:

$$r_i(\sigma, f) - r_i(\sigma', f) \tag{1}$$

ist nicht-steigend in f.

Es stellt sich die Frage: Welche Strategie soll nun gewählt werden, um einen möglich hohen *payoff* zu erzielen?

Wenn das System nach endlicher Zeit stationär wird, so bedeutet dies, dass das Spiel zu einem bestimmten Strategieprofil konvergiert und dort ein Gleichgewicht bildet (das sogenannte Nash-Gleichgewicht).

Definition 2.4 Ein Strategieprofil σ heisst *Nash-Gleichgewicht* (nash equilibria), wenn für (fast) alle Spieler i und alle $s \in \Sigma$ gilt:

$$r_i(\sigma(i), f(\sigma)) \geq r_i(s, f(\sigma)) \tag{2}$$

Wenn die Strategieprofile auf ein Profil konvergieren so ist dies gleichbedeutend mit einem Ausschluss von bestimmten Strategien, die in dem Gleichgewicht aufgrund ihrer Schwäche nicht mehr vorkommen. „Schwäche“ ist hier aber nicht genau definiert, also benötigt man noch ein Konzept, um die Strategien untereinander zu bewerten. Eine fortlaufende Bewertung jeder Strategie ist nicht nötig und meisst auch nicht sinnvoll, und aus diesem Grund werden hier drei Konzepte, die „stärkere“ Strategien von den anderen abheben, vorgestellt:

Um eine gute Strategie gegen das Strategieprofil σ zu berechnen, muss der Spieler nur $f(\sigma)$ kennen, also die Auswirkungen des Strategieprofils. Diese besten Strategien sind wie folgt definiert:

Definition 2.5 Sei $BR_i(f(\sigma))$ die Menge der besten Antworten (best response) für Spieler i auf Strategieprofil σ.
Sei weiterhin $BR^0(\Sigma) = \Sigma$ und $BR^{n+1}(\Sigma) = BR(BR^n(\Sigma))$, dann ist $\Sigma^* = \lim_{n\to\infty} BR^n(\Sigma)$. Ein externes Spiel heisst *BR-stabil* wenn $\forall i \in I$ gilt: $|\Sigma^*| = 1$, d.h. die beste Antwort ist eindeutig.

Da aber das Bekanntsein von $f(\sigma)$ eine sehr starke Bedingung ist, die im verwendeten Modell nicht gegeben ist, ist die BR-Strategie bei nicht-atomaren Spielen nicht zwingend berechenbar. Deshalb werden zwei weitere Begriffe eingeführt, die ohne Kenntnisse der Umgebung $f(\sigma)$ verwendet werden können.

Definition 2.6 Für Spieler i *dominiert* Strategie σ_i die Strategie σ_i' genau dann wenn $\forall f: \quad r_i(\sigma_i, f) > r_i(\sigma_i', f)$.

D^∞ bezeichnet die Menge der Strategien nach unendlich ofter Elimination von dominierten Strategien, also die Strategien, die selbst nicht dominiert werden. Ein Spiel heisst D-lösbar wenn $|D^\infty| = 1$ für fast alle i.

Es ist leicht zu sehen, das BR-Stabilität D-Lösbarkeit impliziert. Bei NAEG ist sogar die Umkehrung korrekt (was sie im allgemeinen nicht ist).

Satz 2.1 Ein externes Spiel ist BR-stabil genau dann wenn es D-lösbar ist.

Eine stärkere Bedingung als Dominanz ist das Übertreffen:

Definition 2.7 Für Spieler i *übertrifft* (overwhelms) Strategie σ_i die Strategie σ_i' in Bezug auf Stragiemenge Σ gdw $\forall f, f': \quad r_i(\sigma_i, f) > r_i(\sigma_i', f')$.

O^∞ bezeichnet die Menge der Strategien nach unendlich ofter Elimination von übertroffenen Strategien, also die Strategien, die selbst nicht übertroffen werden. Ein Spiel heisst O-lösbar wenn $|O^\infty| = 1$ für fast alle i.

Anders als bei der Dominanz impliziert O-Lösbarkeit zwar auch BR-Stabilität, die Umkehrung ist jedoch nur bei bestimmten Klassen von NAEG korrekt. Eine Klasse von NAEG sind *einfache NAEG*.

Definition 2.8 Ein NAEG heisst *einfach*, wenn für alle Spieler $i \in I$ $\Sigma_i = \sigma_-, \sigma_+$, $r_i(\sigma_-, f)$ ist nicht-steigend in f und $\sigma_- < \sigma_+$.
Das typische einfache NAEG ist der, in dem als Strategien nur „teilnehmen"/„nicht teilnehmen" existieren, also $r_i(\sigma_-, f) = 0$ und $r_i(\sigma_+, f) = 1$. In dieser Klasse gilt nun:

Satz 2.2 Wenn ein einfaches NAEG D-lösbar ist, dann ist es auch O-lösbar.

Beweis: Nach Definition gilt $D(i) \subseteq O(i)$. Angenommen σ dominiert σ' und es gilt $\sigma > \sigma'$ (der andere Fall ist analog). Dann gilt für alle f $r_i(\sigma, f) - r_i(\sigma', f) > 0$. Betrachtet man nun das Paar f > f', dann gilt $r_i(\sigma, f') - r_i(\sigma', f') > 0$ und somit auch $r_i(\sigma, f) - r_i(\sigma', f') > 0$. Falls f < f', so gilt $r_i(\sigma, f') - r_i(\sigma', f) > 0$, da r_i nichtsteigend in f ist. Also gilt $D(i) = O(i)$.

Satz 2.3 Wenn ein einfaches NAEG BR-stabil ist, dann ist es auch O-lösbar.
Beweis: Folgt unmittelbar aus Satz 2.1.1 und Satz 2.1.2.

2.3 Beispiele

- *Einwählen in den Uni-Rechner*
  Betrachtet man einen Service Provider, der k Modems zur Verfügung hat. Es gibt nur eine Klasse von Usern, m=1. Der User muss sich entscheiden, ob er sich zu Hause einwählt oder den weiten Weg zur Uni macht, um den Rechner dort zu verwenden, also $\Sigma = 0, 1$. Wird von zu Hause eingewählt und kein Modem ist frei, dann gerät der User in eine Warteschlange, deren Wartezeit $W(f)$ für die Umgebung f beträgt. Wenn die Online-Zeiten der User z.B. exponentialverteilt sind, ist W(f) bekannt, nämlich die M/M/k-Warteschlange im FIFO-System. Seien weiterhin $c_i()$ die nichtsinkende Kostenfunktion für das Warten und b_i die Kosten, um zur Uni zu fahren, so erhält man einen einfachen NAEG mit $r_i(0, f) = -b_i$ und $r_i(1, f) = -c_i(D(f))$.

- *Benutzen einer Schnittstelle*
  Mehrere Benutzer senden Daten über eine gemeinsame Netzwerkverbindung. Hierbei ist $\sigma_i \in \Sigma_i$ die Rate, mit der der Benutzer i Daten sendet. Dabei gilt $\Sigma_i \in [0, \sigma_{max}]$, die Umgebung $f(\sigma)$ ist die gesamte Datenrate auf der Verbindung und durch die gemeinsame Nutzung treten Wartezeiten auf. Die Wartezeit $\sigma_i W^{\kappa}(f)$ hängt von der Zeit der Übertragung und der Wartezeit auf das Freiwerden der Verbindung ab, wobei κ die Kapazität der Verbindung angibt, $W^{\kappa}(f)$ die erwartete Wartezeit pro Packet. Auch hier entspricht der *payoff* den Wartekosten.

3 Ad-Hoc-Netzwerke als Spiele

Im Fall das sich n Benutzer in einem zeitvarianten Netzwerk Bandbreite teilen liegen einige Ungewissheiten vor. Einigen oder sogar allen Benutzern fehlen Informationen über das Spiel, wie z.B. die *payoffs* der anderen Spieler oder sogar die eigene *payoff* Funktion, die Umgebung f und die gewählten Strategien der Mitspieler (da es ja ein nicht-kooperatives Spiel ist).
Diese Situation wird zukünftig in Ad-Hoc-Netzwerken auftreten, in denen ein verteiltes System mit minimalen Informationen operieren muss. Ein Modell, das versucht diese Situation darzustellen, wird in [6] dargestellt. Dieses Modell ist auf viele dezentrale Systeme wie Ad-Hoc-Netzwerke, aber auch das Internet anwendbar, das die gleichen Schwierigkeiten hat, und zwar unvollständige Informationen und asynchrone Aktionen der Teilnehmer.

Wichtig ist, dass trotz der Ungewissheit aber eine Verfügbarkeit der Ressourcen gewährleistet bleiben muss, wobei der Begriff Quality of Service (kurz QoS) ins Spiel kommt. Einige mögliche Definitionen für *Quality of Service*:

- Zusammenfassung verschiedener Kriterien, die die Güte eines Dienstes im Netz hinsichtlich Geschwindigkeit und Zuverlässigkeit beschreiben.
- Die Bereitstellung einer garantierten Dienstgüte nach obiger Definition in einem Datennetz.
- Die Fähigkeit eines Datennetzes, dass bestimmte Anwendungen eine Dienstgüte nach obiger Definition anfordern können und dann diese dann auch garantiert eingehalten wird.

Wie QoS beschrieben wird, hängt von dem verwendeten Modell ab. Z.B. wird der QoS in [7] wie folgt definiert:

der Rauschabstand α liegt über einer Schwelle $\overline{\alpha}$,

wobei die Einschränkung gilt: die Übertragungsstärke x_i für Benutzer i liegt in $0 < x_i < X_i$, d.h. jeder Benutzer hat nur ein gewisses Potential an Übertragungsstärke.

Der *Rauschabstand* α (auch SNR, signal to noise ratio) ist der Quotient aus Signalstärke und Rauschen, wobei Rauschen im weiteren Sinne Störungen, die den Betrieb eines Gerätes beeinflussen, aller Art sind. Daraus folgt, um so höher der Rauschabstand (der normalerweise in Dezibel gemessen wird) um so „besser“ ist das Signal, da entweder das Signal sehr stark oder das Rauschen sehr gering ist.
In der Kommunikationstechnik bezeichnet Rauschen „unerwünschte elektrische Signale, die entweder auf natürlichem Wege oder durch die Schaltung selbst hervorgerufen werden und die Qualität oder Leistung eines Kommunikationskanals herabsetzen“ [8].

Der QoS $\alpha > \overline{\alpha}$ verlangt also eine gewisse Güte von dem Übertragungssignal, welches durch die maximale Übertragungsstärke X_i beschränkt wird. Dies ist natürlich nur ein Beispiel für die Modellierung von QoS und die benötigte Dienstgüte ist stark von der Verwendung des Netzes abhängig, aber es liefert einfache Konzepte, die in der Spieltheorie übernommen werden können.

Um das Modell vorzustellen erst einmal alle verwendeten Termini:

1. Der *Quality of Service* wird durch den Rauschabstand (SNR) definiert.
2. x_i ist die Übertragungsstärke des Signals von Benutzer i.
3. g_i ist ein Faktor, der den Übertragungskanal des Benutzers i beschreibt. Im erweiterten Fall ist $G = (g_{ij})$ eine $m \times m$-Matrix, die sogenannte „link gain matrix“, deren Elemente g_{ij} den Zugewinn/Verlust von übertragenen Signalen, die von Benutzer i an Benutzer j gesendet worden sind, beschreiben.
4. $f(x)$ beschreibt die Störungen, die die Übertragung des Signals der Stärke x begleitet. Diese Interferenzen werden auch in der Interferenz-Matrix F angegeben und entstehen in diesem Modell nur durch die Übertragung der anderen Benutzer, weitere Umwelteinflüsse werden ausgeschlossen.

5. $\overline{\alpha}$ ist die untere Grenze für den Rauschabstand. Es ist natürlich auch möglich, das jeder Benutzer einen anderen Schwellenwert verwendet, der Einfacherheit halber soll von einem gemeinsamen Schwellenwert ausgegangen werden.

Mit den gegebenen Werten lässt sich der Rauschabstand wie folgt angeben (Signalstärke / Interferenzen durch andere Benutzer):

$$\alpha_i = \frac{g_i x_i}{\sum_{j \neq i} g_j x_j} \tag{3}$$

Diese Werte werden nun spieltheoretisch beschrieben:

- Jeder Benutzer des Systems ist ein Spieler.
- Der *payoff* r_i eines Spielers ist der Rauschabstand α_i.
- Die Strategie σ_i eines Spielers ist seine Übertragungsstärke x_i, wobei gilt $\Sigma_i = [0, X_i]$.

Das Ziel des Spieles ist es, den *payoff* zu maximieren, also ein möglichst deutliches Signal zu erhalten. Da das verwendete Spiel ein NAEG ist, ist es nötig zu überprüfen, ob der *payoff* nur von den eigenen Aktionen abhängt und ob die *payoff* Funktion submodular ist (siehe Gl. 1):
Die Auszahlungsfunktion lässt sich wie folgt schreiben: $\alpha(x, f) = \alpha_i(x_i, f(x)) = \frac{g_i x_i}{f(x)}$ und entspricht damit Gl. (3). Für zwei Übertragungsstärken x, x' aus dem Bereich gültiger Strategien mit $x > x'$ gilt somit:

$$\alpha_i(x, f) - \alpha_i(x', f) = \frac{g_i(x - x')}{\sum_{j \neq i} g_j x_j} \tag{4}$$

was in der Tat nichtsteigend in f ist.

Da aber die Situation in dem Netzwerk ständig variiert bedeutet dies, dass nicht mehr direkt mit dem *payoff* gerechnet werden kann sondern jetzt alle ungewissen Funktionen durch deren Verteilung bzw. deren Erwartungswert ersetzen werden müssen. Besonders die „link gain matrix“ G sowie damit verbunden die Umgebung F ändern sich ständig.

Der Hauptunterscheid zwischen dem Model für das Internet und dem Model eines verteilten dezentralen System ist, dass im ersten Fall für die Verbindungskoeffizienten g_{ij} binäre Werte ausreichen, während im zweiten Fall $0 \leq g_{ij} \leq 1$ gilt. In einem Netzwerk mit festen Knoten stellt sich nur die Frage, ob der Knoten angeschlossen ist oder nicht, während bei einem mobilen System die Qualität der Verbindung stark schwanken kann.

4 Lernende Automaten

Da das aufgestellte Modell voller Ungewissheiten und Variationen ist, sind herkömmliche Lösungsansätze wenig effizient. Es besteht ein grosser Unterschied zu herkömmlichen Spielen, in denen die Spieler die „best response"-Strategie anhand der vorherigen Spiels, der Historie aller vorhergehenden Spiele oder komplexen bayesischen Berechnungen errechnen können, da in allen diesen Fällen das Lernverhalten von der Beobachtung der Mitspieler und Wissen über die *payoff* - Matrix abhängt, was hier jedoch nicht gegeben ist. Dagegen erweist sich der Einsatz von Algorithmen, die stochastisch lernende Automaten (learning automata, LA) verwenden, als guter Ansatz. Gerade weil die eigene *payoff* Funktion, die Anzahl und das Verhalten der Mitspieler und die gesamte Umgebung unbekannt ist, bleibt nur ein „trial-and-error"-System, das durch geschicktes Experimentieren versucht, möglichst optimales Verhalten zu zeigen.

Welche sinnvollen Eigenschaften sollten solche Automaten haben?

Die notwendigste Eigenschaft ist, das, wenn man in einer stationären zufälligen Umgebung spielt (z.B. wird der *payoff* durch eine gegebene Zufallsverteilung errechnet), der Algorithmus irgendwann die Strategie(n) mit dem höchsten durchschnittlichen *payoff* spielt.
Es muss aber noch eine zweite Eigenschaft gelten: in diesem Kontext kann ein Spieler den erwarteten *payoff* nicht direkt betrachten, eine Änderung des Spieles kann nur durch Herumexperimentieren festgestellt werden. Solche Änderungen treten in der Realität recht häufig auf, es reicht ja schon das Einloggen eines neuen Benutzers in ein Netzwerk, um die gesamte Situation völlig zu verändern. Ein sinnvoller lernender Automat sollte sich also Veränderungen des Spiels anpassen können.
Im Folgenden werden zwei Arten von solchen Algorithmen vorgestellt, erst ein einfacher Ansatz und danach dessen Verbesserung.

4.1 Learning Automata

Die verwendeten Automaten sind stochastisch lernende Automaten, deren Prinzip sehr einfach ist: Betrachten wir zunächst nur einen Benutzer in einem zeitdiskreten System mit einem *payoff* von r_i^t für Strategie i zum Zeitpunkt t, so kann man den zugehörigen LA wie folgt beschreiben.
Der *payoff* kann zufällig sein und seine Berechnung auf die gesamte Historie vor Zeitpunkt t zurückgreifen, der Einfachheit halber gelte $\forall i, t : 0 \leq r_i^t \leq 1$.

Jeder Zustand des Automaten ist ein Vektor $p^t = (p_1^t, p_2^t, \ldots, p_m^t)$, der zur Zeit t die Wahrscheinlichkeiten p_i^t beinhalten, Strategie i zu spielen. Dabei gilt $\sum_{i=1}^m p_i^t = 1$ und es stehen m mögliche Strategien zur Verfügung. Welche Strategie jetzt gespielt wird, entscheidet der Automat zufällig unter Berücksichtigung der Wahrscheinlichkeiten.
Das Lernende an diesem Automaten liegt an einer Aktualisierung des Wahrscheinlichkeitsvektors p^{t+1} aufgrund des erzielten *payoffs* r^t. Eine einfache Update - Funktion aus [2] mit Parameter $\alpha > 0$:

$$p_i^{t+1} = p_i^t + \alpha r_i^t (1 - p_i^t) \tag{5}$$

$$\forall j \neq i : p_j^{t+1} = p_j^t (1 - \alpha r_i^t) \tag{6}$$

Wird also zur Zeit t die Strategie i gespielt, so erhält sie zum nächsten Zeitpunkt eine veränderte Wahrscheinlichkeit, gespielt zu werden (bemessen am Parameter α sowie dem erzielten Gewinn r_i^t), während alle anderen Strategien entgegengesetzt bewertet werden. War der erzielte *payoff* hoch, so wird die Strategie das nächste Mal wahrscheinlicher gespielt, bei niedrigem *payoff* jedoch seltener. Der Parameter α gibt an, wie stark der *payoff* bewertet wird, ob also Erfolge bzw. Misserfolge sofort gravierende Auswirkungen haben oder ob eine Strategie erst nach vielen guten Ergebnissen wirklich öfter gespielt wird.

Diese LA sind ϵ-optimal, d.h. für alle $\epsilon > 0$ existiert ein α, so dass:

$$\lim_{t \to \infty} E(\sum_{i=1}^m p_i^t r_i^t) > \max\ (E[r_i^s]) - \epsilon \tag{7}$$

Dies bedeutet, dass LA's einen asymptotisch erwarteten *payoff* beliebig nahe an dem optimalen *payoff* erzielen können. Es heisst hingegen nicht, dass der durchschnittliche *payoff* für ein gegebenes Spiel in die Nähe des Optimums konvergiert.
In der Tat verhält es sich bei den LA's so, dass sie mit Wahrscheinlichkeit eins zu einer immer wiederholten Strategie konvergieren. Für die Folge von Spielen eines LA gilt also falls $lim_{0 \to \infty} p_i^t = \delta_{i,k}$ für eine Strategie k, dass für $i \neq k \quad \delta_{i,k} = 0$ und $\delta_{k,k} = 1$.
Diese LA haben jedoch einen Nachteil: wenn sich der Automat erstmal auf wenige Strategien eingespielt hat und einige Strategien wegen ihren Misserfolgen verworfen wurden (d.h. ihre Spielwahrscheinlichkeit $p_i^t = 0$), so

ist der Automat nicht mehr anpassungsfähig an Veränderungen der Umgebung. Es kann nämlich vorkommen, das eine verworfene Strategie ab einem bestimmten Zeitpunkt die besten Erfolge erzielt und somit erreicht der Automat höchstens ein suboptimales Ergebnis.

4.2 Responsive Learning Automata

Um diese Unflexibilität zu beseitigen muss der einfache LA leicht verändert werden, wodurch man einen reagierenden lernenden Automaten (RLA, responsive learning automaton) erhält. Man erlaubt einfach nicht, das die Spielwahrscheinlichkeit p_i^t unter $\alpha/2$ sinkt. Da niemals der Fall $p_i^t = 0$ auftritt wird in einem unendlichen Spiel jede Strategie unendlich oft gespielt, auch diejenigen mit schlechtem *payoff*.

Die Update-Funktion muss leicht variiert werden (RLA_α):

$$p_i^{t+1} = p_i^t + \alpha r_i^t \sum_{j \neq i} a_j^t p_j^t \tag{8}$$

$$\forall j \neq i : p_j^{t+1} = p_j^t - \alpha r_i^t a_j^t p_j^t \tag{9}$$

wobei $a_j^t = \min\left(1, \frac{p_j^t - \alpha/2}{\alpha p_j^t r_i^t}\right)$, $\alpha < 1$, $r_i < 1$.

Bemerkung 1 Ein Vektor p^t heisst *gültig* wenn $\sum_{i=1}^m p_i^t = 1$ und $\forall i : p_i^t \geq \frac{\alpha}{2}$. Die Updateregeln verwandeln gültige Vektoren wieder in gültige Vektoren.

Bemerkung 2 Wenn für alle $p_j \geq \alpha$ gilt, so entsprechen die Regeln RLA_α den Regeln LA_α.

4.3 Konvergenz zum Optimum

Eine sinnvolle Anforderung an lernende Algorithmen ist, dass das System zu einem optimalen Zustand konvergiert. Wenn man jedoch von einer ständig verändernden Umgebung ausgeht ist es zu schwer zu einem Optimum zu gelangen. Da man also Optimalität nicht in allen Umgebungen erreichen kann geben wir uns damit zufrieden Optimalität in „einfachen“ Umgebungen zu zeigen. Nützlich ist es z.B. zu zeigen, das in einem ab einem bestimmten Zeitpunkt statischen System der RLA zum Optimum konvergiert.
Während LA in statischen Umgebungen ϵ-Optimalität erreichen, verhalten sich RLA unterschiedlich. Die meiste Zeit spielen RLA optimale Strategien,

wandern aber ab und zu herum und probieren andere Strategien aus.
Wie dieses Wandern aussieht, zeigt die folgende Defintion, die benötigt wird, um die benötigte Konvergenz zu beschreiben.

Definition 4.1 Ein durch α parametrisierter Prozess x^t in diskreter Zeit α-konvergiert gegen 0 wenn positive Konstanten $\alpha_0, \gamma, b_1, b_2, b_3$ und q existieren, so dass für jedes $0 < \alpha < \alpha_0$ gilt:

1. $\lim_{T\to\infty}(\frac{1}{T}\int_0^T dtP[x^t > \sqrt{\alpha}]) < \alpha$
2. Wenn zum Zeitpunkt τ_f zum ersten Mal $x^t \leq \gamma\alpha$ gilt, dann $E[\tau_f] \leq b_1/\alpha^q$
3. Wenn zum Zeitpunkt τ_r zum ersten Mal $x^t \geq \sqrt{\alpha}$ mit $x^0_\alpha \leq \gamma\alpha$ gilt, dann $E[\tau_r] \geq b_2 e^{b_3/\sqrt{\alpha}}/\alpha$

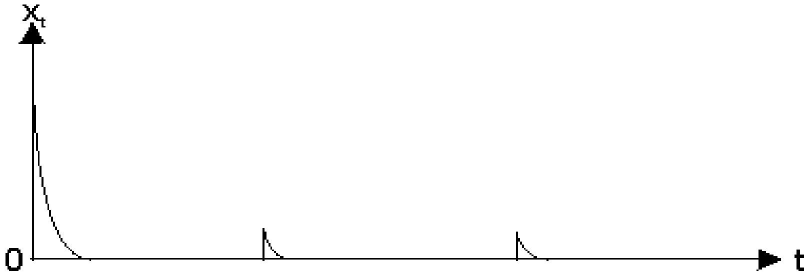

Abbildung 1: α-Konvergenz

Anschaulich betrachtet wird α-Konvergenz durch folgende Beschreibung charakterisiert: eine schnelle (polynomielle) Konvergenz gegen Null gefolgt von einem sehr langen (exponentiellen) Zeitraum in der Nähe von Null, bis zufällige Variationen x^t wieder von Null wegbewegen.
Die exponentiell lange Phase, in der x^t nahe Null bleibt dominiert im Durchschnitt immer die polynomiell lange Phase die x^t benötigt, um Null von einer Anfangsposition aus zu erreichen. Diese Definition kann nun weiter für das folgende Ergebnis verwendet werden:

Theorem 1
Sei A eine Menge von Strategien und $p^t_A = \sum_{i\in A} p^t_i$. Angenommen es existieren $\gamma > 0$ und $T > 0$, so dass

$$E[r^t_i|h^t] < (1 - \frac{m}{\gamma})E[r^t_j|h^t] \tag{10}$$

für alle $t \geq T$, für alle $i \in A$ und für alle $j \notin A$, wobei $h^t = (r^1, r^2, \ldots, r^{t-1})$. Dann α-konvergiert p_A^t gegen 0 von jeder gültigen Ausgangsposition.

Wenn ein RLA in einer statischen Umgebung gegen ein Optimum α-konvergiert, so bedeutet dies, das die Wahrscheinlichkeit nichtoptimale Strategien zu spielen gegen null α-konvergiert. Mit der höchsten Wahrscheinlichkeit werden optimale Strategien gespielt, jedoch manchmal auch nichtoptimale. Hieraus folgt trivialerweise, dass RLA auch ϵ - optimal sind. Genauer: in stationären Umgebungen, in denen r_i^t durch die Verteilungsfunktionen $F_i(x)$ beschrieben sind, konvergiert das Ergebnis eines Spieles innerhalb ϵ gegen den optimalen *payoff*, d.h. für alle $\epsilon > 0$ existiert ein α so dass

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \sum_{i=1}^{m} p_i^t r_i^t > \max\ (E[r_i^s]) - \epsilon \tag{11}$$

fast sicher.

Also erreicht in jedem einzelnen Spiel eines RLA der *payoff* in einer Umgebung, die ab einem Zeitpunkt an stationär ist, schliesslich den optimalen Wert. Dies ist eine Eigenschaft die die einfachen LA nicht haben, die zwar fast optimal sind, wenn man das Mittel über alle Spiele bildet, bei denen einzelne Spiele aber deutlich suboptimal sein können. Der Beweis des Theorems soll an dieser Stelle wegen seines Umfanges nicht gezeigt werden, kann aber unter [2] nachgelesen werden.

4.4 Monotonie oder das Gesetz des Effektes

Eine weitere wichtige Eigenschaft, die das Verhalten betrifft ist das Gesetz des Effektes (law of the effect) von E. Thorndike (Psychologe, 1898), welches für diesen Fall interpretiert besagt, dass Strategien, die in der Vergangenheit zu guten Ergebnissen geführt haben, in der Zukunft öfter gespielt werden. Mathematisch ausgedrückt ist dies eine Forderung der Monotonie an den lernenden Algorithmus, in denen durch Erhöhung des *payoffs* von bestimmten Strategien gleichzeitig ihre Spielwahrscheinlichkeit in der Zukunft erhöht wird. Diese formalere Aussage kann bewiesen werden:

Theorem 2
Sei RLA_α ein reagierender lernender Automat, der gegen eine Umgebung mit den Auszahlungsfunktionen r_i^t spielt. Der gleiche Automat spiele auch gegen eine Umgebung mit den *payoffs* $\tilde{r}_i^t$, wobei p_i^t und $\tilde{p}_i^t$ die Spielwahrscheinlichkeiten in den beiden Fällen angeben. Sei A eine Menge von Strategien. Dann gilt:

wenn $\tilde{r}_i^t$ stochastisch grösser oder gleich r_i^t für alle t und alle $i \in A$ ist und $\tilde{r}_i^t$ ist stochastisch kleiner oder gleich r_i^t für alle t und für alle $i \notin A$, dann gilt $\tilde{p}_A^t = \sum_{i \in A} \tilde{p}_i^t$ dominiert $p_A^t = \sum_{i \in A} p_i^t$ stochastisch für alle t.

Beweis:
Sei $p_A^t = \sum_{i \in A} p_i^t$. Wenn Strategie i zur Zeit t gewählt werden, sind die Update-Funktionen für p_A^t wie folgt:

$$i \in A: \quad p_A^{t+1} = p_A^t + \alpha r_i^t \sum_{j \notin A} a_j^t p_j^t \tag{12}$$

$$i \notin A: \quad p_A^{t+1} = p_A^t - \alpha r_i^t \sum_{j \in A} a_j^t p_j^t \tag{13}$$

wobei

$$a_j^t = \min\,[1, \frac{p_j^t - \alpha/2}{\alpha p_j^t r_i^t}] \tag{14}$$

Also ist p_A^{t+1} monoton steigend in p_A^t, monoton steigend in r_i^t mit $i \in A$ und monoton fallend in r_i^t mit $i \notin A$.

4.5 Konvergenz in RLA - Spielen

Zwei wichtige Eigenschaften von lernenden Algorithmen sind in den vorhergehenden Kapiteln beschrieben, was bis jetzt aber noch nicht betrachtet worden ist, ist das allgemeine Verhalten von dezentralen nicht-kooperativen Spielen. Dabei stellt sich heraus, dass dieses Verhalten stark davon abhängt, ob das Spiel synchron oder asynchron ist, ob also die Strategien in jeder Runde des Spiels oder ständig während des Spiels aktualisiert werden. Das Ergebnis von [2] sagt aus, dass bei synchronen Spielen die Elimination von dominierten Strategien als Konvergenz ausreicht, während bei asynchronen Spielen irgendwann nur noch unübertroffene Strategien gespielt werden. Diese Aussage ist in den beiden folgenden Theoremen formuliert:

Theorem 3
Für jede Gruppe von n (n > 1) synchronen reagierenden lernenden Automaten RLA_α, die ein normalisiertes Spiel spielen, und für jedes $p \geq 1$ gilt für jeden Automaten a

$$p_D^t \equiv \sum_{i \notin D^\infty} p_i^t \tag{15}$$

α-konvergiert gegen 0.

Theorem 4
Für jede Gruppe von n (n > 1) asynchronen reagierenden lernenden Automaten RLA_α, die ein normalisiertes Spiel spielen, und für jedes $p \geq 1$ gilt für jeden Automaten a

$$p_D^t \equiv \sum_{i \notin O^\infty} p_i^t \tag{16}$$

α-konvergiert gegen 0.

Bemerkung
Ein normalisiertes Spiel bedeutet, dass der Wert des *payoffs* in [0,1] liegt, wie es in Kapitel 2 angenommen wird.

Die genaue Anwendung der α-Konvergenz sowie die Beweise dieser beiden Theoreme kann [2] entnommen werden, ist aber an dieser Stelle nicht ausschlaggebend.

Das übertroffene Strategien im Gegensatz zu dominierenden Strategien das passende Konzept in asynchronen Umgebungen sind, ist leicht zu verdeutlichen:
Angenommen, Strategie i dominiert Strategie j, aber ein anderer Spieler reagiert auf i völlig anders als auf j, so kann sich herausstellen, das es vorteilhafter ist j zu spielen. Dieser Fall tritt bei übertroffenen Strategien nicht auf, da Übertreffen unabhängig von den Reaktionen der anderen Spieler ist.

Bemerkung Es gilt $U^\infty \subseteq O^\infty$, da das Eliminationskriterium stärker ist, d.h. eine übertroffene ist gleichzeitig eine dominierte Strategie, aber nicht andersrum.

Man kann also zusammenfassen, das der RLA konvergiert. Im besten Fall ist O^∞ eindeutig, da es eine einelementige Menge ist, es gibt jedoch auch Spiele in denen dies nicht der Fall ist, in denen obiges Theorem also kein eindeutiges Ergebnis liefert. Solche Fällen sind aber bei lernenden Algorithmen eine unausweichliche Schwierigkeit und es konnte bisher noch nicht gezeigt werden, dass diese umgangen werden können.

Angewendet auf die Situation eines Ad-Hoc-Netzwerkes benötigt man ein asynchrones Spiel, um den Verlauf im Netzwerk korrekt wiederzugeben. Jeder Benutzer mag zwar in einer gewissen weise synchron arbeiten, doch alle Benutzer zusammen arbeiten asynchron, wodurch die Verwendung von RLA's nach dem obigen Ergebnis nötig wird.

5 Zusammenfassung

In der vorhergehenden Kapiteln wurde eine Analyse der Ressourcenverteilung in dezentralen verteilten Netzwerken, seien es nun Ad-Hoc-Netzwerke oder Systeme wie das Internet, vorgestellt, die die größte Schwierigkeit solcher Systeme beinhaltet, nämlich Ungewissheit. Durch die Verwendung eines Modells aus der Spieltheorie lassen sich so verschiedene Lösungsansätze für die Probleme von verteilten Systemen gewinnen, indem einfach die gesamte Theorie der Spiele ausgeschöpft wird, die sich im Laufe der Jahre entwickelt hat, und für die Zwecke eines jüngeren, moderneren Problems genutzt wird.
Der hier gebrachte Lösungsvorschlag mit Hilfe von reagierenden (stochastisch) lernenden Algorithmen wurde genauer betrachtet und es stellt sich heraus, dass unter Verwendung von RLA's in asynchronen Systemen der Algorithmus irgendwann konvergiert, auch im schlechtesten Fall. Wie schnell diese Konvergenz aber eintritt, ist noch unbekannt und bedarf noch weiterer Forschung, bevor diese Idee in der Realität umgesetzt werden kann. Ein weiterer Vorteil der RLA's ist die einfache Implementierung, da diese Automaten nicht sehr komplex sind und ihr „Lernen“ durch die Wahrscheinlichkeiten relativ einfach umgesetzt wird.
Sicher bedarf es noch einiger Simulationen und Erfahrungen auf diesem Gebiet, es ist aber deutlich geworden, dass die Verwendung von Ad-Hoc-Netzwerken immer näher rückt, da man mit heutigen Mitteln und auch mit der Theorie von „älteren“ Themen immer mehr Probleme dieser schwierigen Netzwerke voller Ungewissheit lösen kann.

Literatur

[1] Learnability in a Class of Non-Atomic Games arising on the Internet. E. Friedman 1998

[2] Synchronous and Asynchronous Learning by Responsive Learning Automata. E. Friedman und S. Shenker 1996

[3] A Minimax Game of Power Control Under Incomplete Information. T. Heikkinen

[4] Stochastic Power Control for Cellular Radio Systems. S. Ulukus und R. Yates 1998

[5] Performance of Optimum Transmitter Power Control in Cellular Radio Systems. J. Zander 1992

[6] On Learning and the Quality of Service in a Wireless Network. T. Heikkinen 2000, LNCS 1815

[7] Power Control and Ressource Management for a Multimedia CDMA Wireless System. A. Sampath, P. Kumar, J. Holtzman 1995

[8] www.lexios.de