NEFCON-I: Eine Simulationsumgebung für Neuronale Fuzzy-Regler

Detlef Nauck
Institut für Betriebssysteme und Rechnerverbund
Technische Universität Braunschweig
Bültenweg 74 / 75, 38106 Braunschweig
Tel. (05 31) 391-3155, Fax: (05 31) 391-5936, email: nauck@ibr.cs.tu-bs.de

1 Einführung

Bisher aus der Literatur bekannte Kombinationen von Neuronalen Netzen und Fuzzy-Reglern (für einen Überblick s. [6, 7]) sind meist kooperativer Natur, wobei ein Neuronales Netz zum Erlernen von Fuzzy-Mengen oder Fuzzy-Regeln genutzt wird und dessen Ergebnisse zur Konstruktion eines konventionellen Fuzzy-Reglers herangezogen werden. Hybride Ansätze hingegen versuchen eine neue Architektur zu finden, die Neuronale Netze und den Fuzzy-Regler vereinen [2, 3]. Nachteile dieser Ansätze sind vor allem im Bereich der Interpretierbarkeit der Lernresultate zu sehen, insbesondere bei der Verwendung einer rein neuronalen Architektur (für eine Diskussion s. [8]).

Das in diesem Aufsatz vorgestellte Modell eines Neuronalen Fuzzy-Reglers hat den Vorteil, daß es sowohl als Neuronales Netz mit Gewichten in Form von Fuzzy-Mengen als auch als Fuzzy-Regler interpretiert werden kann. Auf diese Weise läßt sich ein Lernalgorithmus angeben, der keine strukturellen Veränderungen nach sich zieht, wodurch die dem Fuzzy-Regler zugrundeliegenden Semantik erhalten bleibt. Das Lernverfahren zur Veränderung der Zugehörigkeitsfunktionen des Modells basiert auf einem *Fuzzy-Fehlermaß,* daß den Zustand des zu regelnden Systems beschreibt [9], und entspricht dem reinforcement learning für Neuronale Netze [1].

Der Neuronale Fuzzy-Regler wurde im Rahmen der Simulationsumgebung NEFCON-I implementiert und getestet.

2 Das NEFCON-Modell

Die folgenden Betrachtungen legen ein technisches System S mit n Meßgrößen $\xi_1 \in X_1, \ldots, \xi_n \in X_n$ und einer Stellgröße $\eta \in Y$ zugrunde. Jede Menge $X_i, i = 1, \ldots, n$ ist durch p_i Fuzzy-Mengen $\mu_1^{(i)}, \ldots, \mu_{p_i}^{(i)}$ und die Menge Y durch die q Fuzzy-Mengen $\nu_1, \ldots, \nu_q$ partitioniert, die mit den entsprechenden linguistischen Termen $A_1^{(i)}, \ldots, A_{p_i}^{(i)}$ und $B_1, \ldots, B_q$ assoziiert sind. Das Wissen über die Stellgröße η wird durch k linguistische Kontrollregeln $R_1, \ldots, R_k$ beschrieben.

Das Bild 1 stellt die Architektur eines NEFCON-Systems mit zwei Eingangsgrößen, fünf Regeln und einer Stellgröße dar. Das Modell entspricht einem vorwärtsbetriebenen dreischichtigen Neuronalen Netz. Die Eingabeschicht besteht aus Einheiten, die lediglich die externe Eingabe entgegennehmen und zunächst keine weitere Verarbeitung leisten. Müssen die Meßgrößen jedoch transformiert werden, dann erfüllen die Eingabeeinheiten diese Aufgabe. Die innere Schicht enthält Einheiten, die linguistische Regeln repräsentieren. Ihre Aktivierungen entsprechen dem Erfüllungsgrad ihrer Prämissen. Die Ausgabeschicht enthält genau eine Ausgabeeinheit, die einen scharfen Stellwert liefert. Die Verarbeitungseinheiten des Netzes werden gemäß ihrer Funktion genau so wie die Eingangsgrößen, die linguistischen Regeln und die Stellgröße des betrachteten Systems S bezeichnet.

Eine Besonderheit gegenüber gewöhnlichen Neuronalen Netzen stellt die Art der Verbindung zwischen den

Wird auf dem ersten GI-Workshop "Fuzzy-Systeme '93" vom 21.-22.10.93 in Braunschweig vorgetragen.

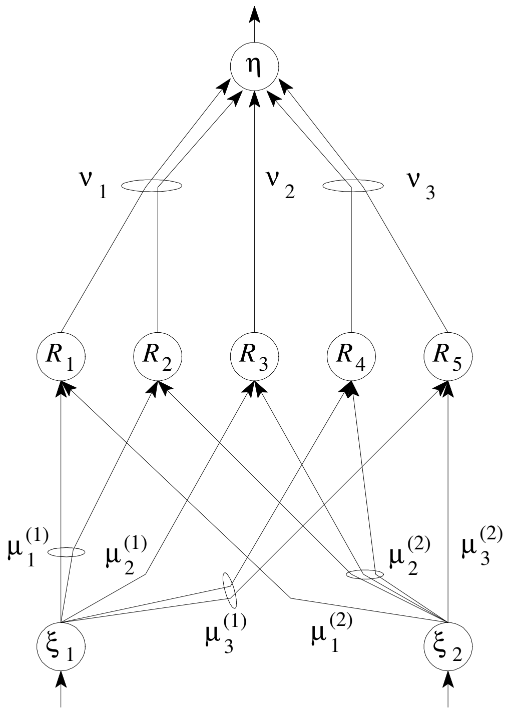

Bild 1: Ein NEFCON-System mit 2 Eingangsgrößen und 5 Regeln

einzelnen Schichten dar. Es gibt Verbindungen, denen ein Gewicht gemeinsam ist (shared weights, coupled links). In Bild 1 tragen z.B. die Verbindungen von der Eingabeeinheit ξ_1 zu den inneren Einheiten R_1 und R_2 dasselbe Gewicht $\mu_1^{(1)}$. Ebenso ist den Verbindungen von den Einheiten R_4 und R_5 zur Ausgabeeinheit η das Gewicht ν_3 gemeinsam. Ein Lernalgorithmus muß gemeinsame Gewichte berücksichtigen und seine Änderungen an einer Verbindung auf allen gekoppelten Verbindungen in gleicher Weise durchführen.

Die Verbindungsgewichte sind keine Zahlenwerte, sondern Fuzzy-Mengen. Die Gewichte zwischen der Eingabeschicht und der inneren Schicht repräsentieren die Fuzzy-Mengen der Regelprämissen, und die Gewichte zwischen innerer Schicht und Ausgabeschicht entsprechen den Fuzzy-Mengen der Regelkonklusionen. Die Gewichte sind genau so wie ihre korrespondierenden Fuzzy-Mengen bezeichnet. Die Regelbasis wird durch die Verbindungsstruktur kodiert.

NEFCON ist als Spezialfall eines *dreischichtigen Fuzzy-Perceptrons* zu interpretieren, das die Architektur eines üblichen Multilayer-Perceptrons besitzt. Die Gewichte werden jedoch nicht als reelle Zahlen, sondern in Form von Fuzzy-Mengen realisiert und die Funktionen des Netzes entsprechend angepaßt. Ein Fuzzy-Perceptron kann als eingeschränkte „Fuzzifizierung" eines normalen Perceptrons interpretiert werden. Es dient wie ein gewöhnliches Perceptron zur Funktionsapproximation, und sein Vorteil besteht darin, seine Struktur in Form linguistischer Regeln interpretieren zu können.

Definition 2.1 *Ein dreischichtiges Fuzzy-Perceptron ist ein dreischichtiges, vorwärtsbetriebenes Neuronales Netz* $(U, W, \mathrm{NET}, A, O, \mathrm{ex})$, *mit den folgenden Spezifikationen:*

(i) $U = \bigcup_{i \in M} U_i$ *ist eine nicht-leere Menge von Verarbeitungseinheiten, und* $M = \{1, 2, 3\}$ *heißt Indexmenge von* U. *Es gilt für alle* $i, j \in M, U_i \neq \emptyset$ *und* $U_i \cap U_j = \emptyset$ *für* $i \neq j$. U_1 *heißt Eingabeschicht,* U_2 *Regelschicht und* U_3 *Ausgabeschicht.*

(ii) Die Netzwerkstruktur ist $W : U \times U \to \mathcal{F}(\mathbb{R})$, *wobei ausschließlich Verbindungen* $W(u, v)$ *mit* $u \in U_i,\ v \in U_{i+1}\ (i \in \{1, 2\})$ *existieren.*

(iii) A *ordnet jeder Einheit* $u \in U$ *eine Aktivierungsfunktion* A_u *zur Berechnung der Aktivierung* a_u *zu. Es gilt*

(a) *für Eingabeeinheiten und Regeleinheiten* $u \in U_1 \cup U_2$:

$$A_u : \mathbb{R} \to \mathbb{R}, \quad a_u = A_u(\mathrm{net}_u) = \mathrm{net}_u,$$

(b) *für Ausgabeeinheiten* $u \in U_3$:

$$A_u : \mathcal{F}(\mathbb{R}) \to \mathcal{F}(\mathbb{R}), \quad a_u = A_u(\mathrm{net}_u) = \mathrm{net}_u.$$

(iv) *O ordnet jeder Einheit* $u \in U$ *eine Ausgabefunktion* O_u *zur Berechnung der Ausgabe* o_u *zu. Es gilt*

(a) *für Eingabeeinheiten und Regeleinheiten* $u \in U_1 \cup U_2$:

$$O_u : \mathbb{R} \to \mathbb{R}, \quad o_u = O_u(a_u) = a_u,$$

(b) *für Ausgabeeinheiten* $u \in U_3$:

$$O_u : \mathcal{F}(\mathbb{R}) \to \mathbb{R}, \quad o_u = O_u(\mathrm{net}_u) = \mathrm{DEFUZZ}_u(\mathrm{net}_u),$$

dabei ist DEFUZZ_u *ein geeignetes Defuzzifizierungsverfahren.*

(v) NET *ordnet jeder Einheit* $u \in U$ *eine Netzeingabefunktion* NET_u *zur Berechnung der Netzeingabe* net_u *zu. Es gilt*

(a) *für Eingabeeinheiten* $u \in U_1$:

$$\mathrm{NET}_u : \mathbb{R} \to \mathbb{R}, \quad \mathrm{net}_u = ex_u,$$

(b) *für innere Einheiten* $u \in U_2$:

$$\mathrm{NET}_u : (\mathbb{R} \times \mathcal{F}(\mathbb{R}))^{U_1} \to [0,1], \quad \mathrm{net}_u = \top_{u' \in U_1} (W(u',u)(o_{u'})),$$

dabei ist $\top$ *eine t-Norm,*

(c) *für Ausgabeeinheiten* $u \in U_3$:

$$\begin{aligned} \mathrm{NET}_u &: ([0,1] \times \mathcal{F}(\mathbb{R}))^{U_2} \to \mathcal{F}(\mathbb{R}), \\ \mathrm{net}_u &: \mathbb{R} \to [0,1], \quad \mathrm{net}_u(x) = \perp_{u' \in U_2} (\top(o_{u'}, W(u',u)(x))), \end{aligned}$$

dabei ist $\perp$ *eine t-Conorm.*
Falls die Fuzzy-Mengen $W(u',u)$, $u' \in U_2$, $u \in U_3$, *über ihren Trägern monoton sind, und* $W^{-1}(u',u)(\tau)$ *das* $x \in \mathbb{R}$ *bestimmt, für das* $W(u',u)(x) = \tau$ *gilt, so kann alternativ für eine Ausgabeeinheit* $u \in U_3$ *gelten:*

$$\mathrm{net}_u(x) = \begin{cases} 1 & \textit{falls } x = \dfrac{\sum_{u' \in U_2} o_{u'} \cdot W^{-1}(u',u)(o_{u'})}{\sum_{u' \in U_2} o_{u'}} \\ 0 & \textit{sonst.} \end{cases}$$

Für die Ausgabe o_u *gilt dann abweichend von (iv.b):*

$$o_u = x, \ \textit{mit } net_u(x) = 1.$$

(vi) $\mathrm{ex} : U_1 \to \mathbb{R}$, *ordnet jeder Eingabeeinheit* $u \in U_1$ *ihre externe Eingabe* $\mathrm{ex}(u) = \mathrm{ex}_u$ *zu. Für andere Einheiten ist* ex *nicht definiert.*

Die Idee des Fuzzy-Perceptrons bildet die Grundlage des NEFCON-Modells. Um einen Neuronalen Fuzzy-Regler zu bilden, muß entweder ein Fuzzy-Regler als Fuzzy-Perceptron dargestellt oder umgekehrt, ein Fuzzy-Perceptron als Fuzzy-Regler interpretiert werden. Dazu sind jedoch Einschränkungen in der Art der Verbindungswahl und der Bestimmung der Fuzzy-Gewichte zu vereinbaren. Gleichzeitig müssen die Gewichte, bzw. die Verbindungen, mit linguistischen Termen assoziiert werden.

Definition 2.2 *Gegeben sei ein dynamisches System S mit n Meßgrößen und einer Stellgröße (s.o.). Im Zusammenhang mit der Regelung von S seien k linguistische Regeln bekannt. Ein NEFCON-System ist ein Fuzzy-Perceptron mit benannten Verbindungen, daß den folgenden Einschränkungen unterliegt:*

(i) $U_1 = \{\xi_1, \ldots, \xi_n\}$, $U_2 = \{R_1, \ldots, R_k\}$, $U_3 = \{\eta\}$.

(ii) Jeder Verbindung zwischen Einheiten $\xi_i \in U_1$ und $R_r \in U_2$ wird mit einem linguistischem Term $A_{j_r}^{(i)}$ $(j_r \in \{1, \ldots, p_i\})$ benannt.

(iii) Jede Verbindung zwischen Einheiten $R_r \in U_2$ und der Ausgabeeinheit η wird mit einem linguistischen Term B_{j_r} $(j_r \in \{1, \ldots, q\})$ benannt.

(iv) Verbindungen, die von derselben Eingabeeinheit ξ_i $(i \in \{1, \ldots, n\})$ wegführen und gleiche Benennungen besitzen, tragen zu jeder Zeit gleiche Gewichte. Die Verbindungen heißen gekoppelt. Analoges gilt für die zu der Ausgabeeinheit η führenden Verbindungen.

(v) $L_{u,v}$ bezeichne die Benennung der Verbindung zwischen den Einheiten u und v. Für alle $v, v' \in U_2$ gilt

$$(\forall\ (u \in U_1)\ \ L_{u,v} = L_{u,v'})\ \Rightarrow\ v = v'.$$

Die Definition 2.2 ermöglicht es, ein NEFCON-System als Fuzzy-Regler zu interpretieren. Die Bedingung (iv) stellt sicher, daß gleiche linguistische Werte einer Variablen durch nur eine Fuzzy-Menge repräsentiert werden. Bedingung (v) legt fest, daß es keine zwei Regeln mit identischen Prämissen gibt. Ein Netz, daß diese Bedingung nicht erfüllt, heißt *überbestimmtes NEFCON-System.*

Die Wissensbasis des durch NEFCON repräsentierten Fuzzy-Reglers ist implizit in der Struktur des Netzwerkes enthalten. Die Eingabeschicht übernimmt die Aufgaben des Fuzzifizierungs-Interface, die Entscheidungslogik ist über die Propagierungsfunktionen verteilt, und die Ausgabeeinheit entspricht dem Defuzzifizierungs-Interface.

Die Abläufe innerhalb des Neuronalen Fuzzy-Reglers entsprechen denen eines vorwärtsbetriebenen Neuronalen Netzes. Die Eingabeeinheiten repräsentieren in ihrer Aktivierung die scharfen Werte der Meßgrößen und reichen sie an die verbundenen Regeleinheiten weiter. Die Propagierungsfunktion der Regeleinheiten ermittelt die Zugehörigkeitswerte zu den Fuzzy-Mengen der Verbindungen zwischen Eingabe- und Regelschicht und bestimmt mittels einer t-Norm, üblicherweise durch den Minimumoperator, den Erfüllungsgrad der jeweiligen Regelprämisse. Dieser Wert wird in der Aktivierung der Regeleinheit repräsentiert.

Die Regeleinheiten geben ihre Aktivierung weiter an die Ausgabeeinheit. Deren Propagierungsfunktion verknüpft den Erfüllungsgrad einer Regeleinheit mit der Fuzzy-Menge der jeweiligen Verbindung zwischen Regel- und Ausgabeeinheit. Dies entspricht der Ermittlung der Ausgabe-Fuzzy-Menge einer Regel. Schließlich werden die Einzelausgaben der Regeln zur Gesamtausgabe zusammengefaßt. Die Vorgehensweise muß dabei in Übereinstimmung mit dem zu realisierenden Fuzzy-Regler und unter Berücksichtigung des Lernalgorithmus gewählt werden.

Das Lernverfahren verlangt, daß jede Regel einen scharfen Ausgabewert liefert. Um das zu gewährleisten, muß entweder eine Defuzzifizierung vor der Zusammenfassung der Einzelergebnisse erfolgen, oder die Zugehörigkeitsfunktionen in den Konklusionen sind monoton über ihrem Träger und können daher invertiert werden. NEFCON verwendet die zweite Möglichkeit, die auch in der Definition des Fuzzy-Perceptrons vorgesehen ist (vgl. Def. 2.1).

3 Fuzzy-Fehler-Propagation – Der Lernalgorithmus

Das Ziel des Lernalgorithmus ist die Anpassung der Zugehörigkeitsfunktionen des Reglers. Dabei wird vorausgesetzt, daß die linguistischen Kontrollregeln adäquat formuliert sind und lediglich die Modellierung der Fuzzy-Mengen aufgrund mangelnden Wissens dem System nicht gerecht wird.

Der Regler soll einen optimalen Systemzustand herstellen. Gewöhnlich wird der Zustand eines Systems jedoch auch dann als *gut* bezeichnet, wenn dieses Optimum nur ungefähr erreicht wird. Es ist daher angemessen, die Güte des Systemzustandes mittels linguistischer Terme zu beschreiben, die durch Fuzzy-Mengen repräsentiert werden. Daraus läßt sich ein Fuzzy-Fehler ableiten, der die Leistung des Neuronalen Fuzzy-Reglers charakterisiert.

Bei der Beurteilung des Systemzustandes lassen sich zwei Fälle unterscheiden. Im ersten Fall befindet sich das System in einem Zustand, in dem alle Meßgrößen in etwa optimale Werte aufweisen. Im zweiten Fall weisen sie dagegen unzulässige Werte auf, die sich jedoch in dem Sinne kompensieren, daß das System in naher Zukunft optimale Werte in den Meßgrößen annehmen wird (kompensatorische Situation).

Definition 3.1 *Gegeben sei ein dynamisches System S mit n Meßgrößen $\xi_1 \in X_1, \ldots, \xi_n \in X_n$ für das s kompensatorische Situationen bekannt sind. Weiterhin seien n Fuzzy-Mengen $\mu_{\text{opt}}^{(i)} : X_i \to [0,1]$, $(i \in \{1, \ldots, n\})$ und s n-stellige Fuzzy-Relationen $\mu_{\text{komp}}^{(j)} : X_1 \times \ldots \times X_n \to [0,1]$, $(j \in \{1, \ldots, s\})$ gegeben, die die optimalen Werte der einzelnen Meßgrößen bzw. kompensatorische Wertekombinationen repräsentieren. Die aktuellen Meßwerte seien durch $(x_1, \ldots, x_n)$ gegeben.*

Die Fuzzy-Güte G des Systems S ist definiert als

$$\begin{aligned} G &: X_1 \times \ldots \times X_n \to [0,1], \\ G(x_1, \ldots, x_n) &\stackrel{def}{=} g\left(G_{\text{opt}}(x_1, \ldots, x_n), G_{\text{komp}}(x_1, \ldots, x_n)\right). \end{aligned}$$

Dabei ist eine geeignete Funktion g zur Kombination der Gütemaße in Abhängigkeit von S zu wählen. G setzt sich zusammen aus der kumulativen Fuzzy-Güte G_{opt}:

$$\begin{aligned} G_{\text{opt}} &: X_1 \times \ldots \times X_n \to [0,1], \\ G_{\text{opt}}(x_1, \ldots, x_n) &\stackrel{def}{=} \top\left\{\mu_{\text{opt}}^{(1)}(x_1), \ldots, \mu_{\text{opt}}^{(n)}(x_n)\right\}, \end{aligned}$$

und der kompensatorischen Fuzzy-Güte G_{komp}:

$$\begin{aligned} G_{\text{komp}} &: X_1 \times \ldots \times X_n \to [0,1], \\ G_{\text{komp}}(x_1, \ldots, x_n) &\stackrel{def}{=} \top\left\{\mu_{\text{komp}}^{(1)}(x_1, \ldots, x_n), \ldots, \mu_{\text{komp}}^{(s)}(x_1, \ldots, x_n)\right\}. \end{aligned}$$

Der Fuzzy-Fehler E eines zur Regelung von S bestimmten NEFCON-Systems ist definiert als

$$E : X_1 \times \ldots \times X_n \to [0,1], \quad E(x_1, \ldots, x_n) \stackrel{def}{=} 1 - G(x_1, \ldots, x_n).$$

Die kumulative Fuzzy-Güte G_{opt} basiert auf den Beschreibungen für die angestrebten optimalen Werte der Meßgrößen, zu deren Modellierung Fuzzy-Zahlen oder Fuzzy-Intervalle geeignet sind. Die Zugehörigkeitsfunktionen $\mu_{\text{komp}}^{(j)}$, die zur Bestimmung der kompensatorischen Fuzzy-Güte G_{komp} verwendet werden, müssen nicht notwendigerweise über allen Meßgrößen definiert sein. Denkbar ist auch, daß sie nur von zwei oder mehreren von ihnen abhängen. Die Funktion g, die die Fuzzy-Güte aus den beiden Gütemaßen G_{opt} und G_{komp} bestimmt, muß entweder eines der beiden Maße für den aktuellen Zustand auswählen oder eine geeignete Kombination aus beiden bilden. Als t-Norm kann z.B. der Minimumoperator gewählt werden.

Neben der in Definition 3.1 angegebenen Vorgehensweise, kann der Fuzzy-Fehler auch mit Hilfe linguistischer Regeln angegeben werden. Diese Möglichkeit wird in NEFCON-I genutzt.

Der Lernalgorithmus für Neuronale Fuzzy-Regler, der auf dem oben definierten Fuzzy-Fehler basiert, wird *Fuzzy-Fehler-Propagierung* genannt. Der Ablauf des Verfahrens ist einerseits dem Backpropagation-Algorithmus für das mehrschichtige Perceptron vergleichbar, andererseits wird jedoch kein von der Ausgabe abhängiger Fehler verwendet [4], was dem reinforcement learning entspricht.

Der individuelle Beitrag einer Regel zum Wert der Stellgröße soll einen Einfluß auf das Ausmaß der Veränderung durch den Lernalgorithmus ausüben. Er ist jedoch nur schwer zu ermitteln, wenn die Konklusionen zu einer einzigen Fuzzy-Menge aggregiert werden, die schließlich defuzzifiziert wird. Liefert jedoch jede Regel einen scharfen Wert und keine Fuzzy-Menge, so fällt dieses Problem weg. Die meisten Ansätze Neuronaler Fuzzy-Regler, die keine stochastischen Lernverfahren wie z.B. in [11] verwenden, beruhen auf dieser Tatsache und verwenden entweder spezielle Zugehörigkeitsfunktionen[2] in den Konklusionen oder spezielle Defuzzifizierungsmethoden, die vor der Aggregation eingesetzt werden [3].

In dem hier diskutierten Ansatz werden als Fuzzy-Mengen für die Regelkonklusionen nur solche mit monotoner Zugehörigkeitsfunktion zugelassen. Aufgrund der Existenz der Umkehrfunktion liefert jede Regel in Abhängigkeit ihres Erfüllungsgrades direkt einen scharfen Wert. Monotone Zugehörigkeitsfunktionen werden von Tsukamotos Variante des Mamdani-Reglers verwendet [5, 12].

Für jede einzelne Regeleinheit R_r deren Aktivierung o_r größer Null ist, muß entschieden werden, ob ihr Beitrag t_r zum Wert der Stellgröße für den aktuellen Zustand einen positiven oder negativen Einfluß auf die Regelung hat. In Abhängigkeit davon wird die Regel durch den Lernalgorithmus entweder „belohnt“ oder „bestraft“. Dazu wird neben der Bewertung des Systemzustandes anhand des Fuzzy-Fehlers das Wissen benötigt, ob in dem gegebenen Zustand der optimale Wert η_{opt} der Stellgröße positiv oder negativ sein sollte. Mit dieser Kenntnis, kann der Lernalgorithmus eine Regel R_r deren Beitrag t_r das korrekte Vorzeichen aufweist „belohnen“ und andernfalls „bestrafen“.

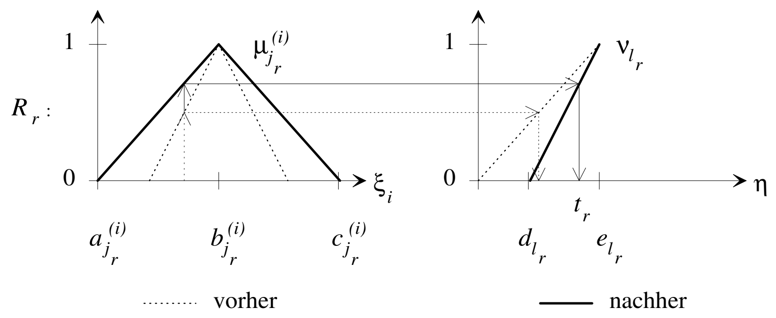

Bild 2: Anpassung der Zugehörigkeitsfunktionen (Belohnungssituation)

Die „Belohnung“ einer Regel (Bild 2) soll sich in einer Erhöhung ihres Beitrages zur Regelaktion auswirken. Dies kann erreicht werden, wenn einerseits der Erfüllungsgrad der Regel bei gleichem Systemzustand erhöht wird und andererseits der Betrag von t_r größer wird. Der Erfüllungsgrad der Regel steigt, wenn die Zugehörigkeitswerte der Meßwerte zu den Fuzzy-Mengen der Prämisse größer werden. Bei der Verwendung der üblichen Dreiecksfunktionen, wird dies durch eine Vergrößerung der Träger erreicht. Der Betrag von t_r wird dagegen größer, wenn der Träger der Fuzzy-Menge ν_{j_r} verringert wird [9]. Bei einer „Bestrafung“ der Regel ist die entsprechend entgegengesetzte Aktion durchzuführen. Die Größe der Veränderung der Zugehörigkeitsfunktionen jeder Regel R_r hängt von deren *Fuzzy-Regel-Fehler*

$$E_{R_r} \stackrel{def}{=} \begin{cases} -o_{R_r} \cdot E & \text{falls } \operatorname{sgn}(t_r) = \operatorname{sgn}(\eta_{\text{opt}}) \\ o_{R_r} \cdot E & \text{sonst} \end{cases}$$

ab.

Die Veränderungen der Zugehörigkeitsfunktionen ν_{j_r} der Konklusionen werden wie folgt bestimmt (vgl. auch Bild 2):

$$\Delta d_{j_r} = \sigma \cdot E_{R_r} \cdot |d_{j_r} - e_{j_r}|$$

wobei σ eine Lernrate darstellt und der Regelknoten R_r durch ν_{j_r} mit η verbunden ist. Stellt ν_{j_r} ein gemeinsames Gewicht dar, so wird es entsprechend oft verändert.

Die Regelfehler werden anschließend zur mittleren Schicht propagiert, in der die Regelknoten eine Veränderung der Zugehörigkeitsfunktionen der Prämissen veranlassen:

$$\Delta\, a_{j_r}^{(i)} \stackrel{def}{=} -\sigma \cdot E_{R_r} \cdot (b_{j_r}^{(i)} - a_{j_r}^{(i)}),$$

$$\Delta\, c_{j_r}^{(i)} \stackrel{def}{=} \sigma \cdot E_{R_r} \cdot (c_{j_r}^{(i)} - b_{j_r}^{(i)}).$$

wobei der Eingabeknoten ξ_i mit R_r durch $\mu_{j_r}^{(i)}$ verbunden ist. Falls ein Gewicht $\mu_{j_r}^{(i)}$ mehreren Verbindungen gemeinsam ist, wird es entsprechend oft verändert.

In [10] wird eine Erweiterung des Lernalgorithmus vorgestellt, wodurch das Modell auch Fuzzy-Regeln erlernen kann. Dabei beginnt der Neuronale Fuzzy-Regler mit einer Architektur, in der zunächst sämtliche linguistischen Kontrollregeln, die sich aus der Kombination von Ein- und Ausgabevariablen bilden lassen, repräsentiert sind (überbestimmtes NEFCON-System). Die Regelbasis muß von dem Lernverfahren durch die Entfernung falscher bzw. überflüssiger Regeln konsistent gemacht werden.

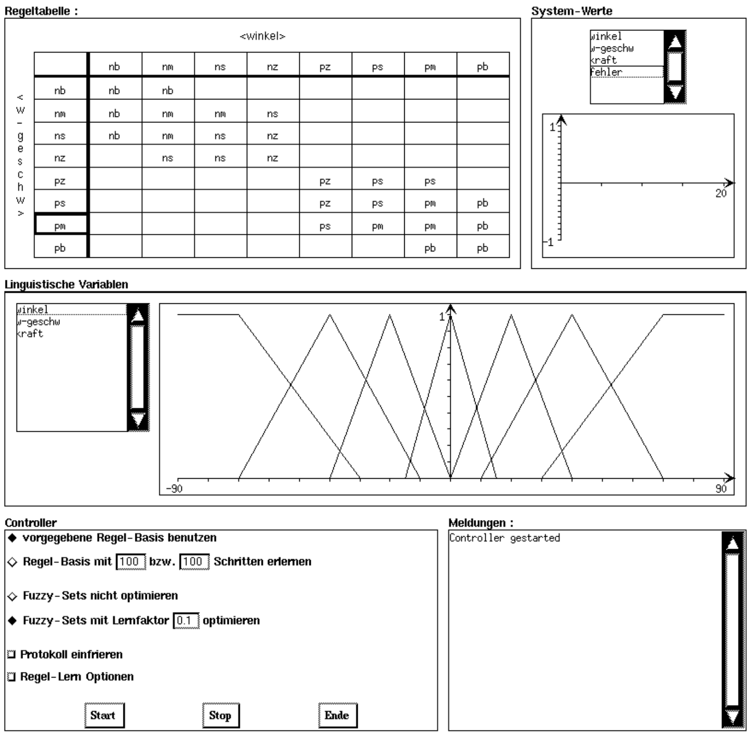

Bild 3: Die Oberfläche von NEFCON-I

In einer ersten Lernphase werden zunächst alle Regeln entfernt, deren Ausgabe ein von η_{opt} verschiedenes Vorzeichen aufweisen. Nach Abschluß dieser Phase enthält die Regelbasis zwar noch Regeln mit unterschiedlichen Konklusionen für dieselbe Prämisse, jedoch produzieren diese Ausgabewerte mit identischem Vorzeichen. Die zweite Phase des Lernalgorithmus muß demnach versuchen, jeweils die Regel herauszufinden, die für eine gegebene Prämisse die korrekte bzw. beste Konklusion enthält.

NEFCON-I implementiert dieses Erweiterung des Lernalgorithmus und enthält einen gegenüber dem ersten Ansatz [10] verbesserten Regelauswahlmechanismus. Während der zweiten Lernphase beteiligt sich eine Regel mit der Wahrscheinlichkeit $\frac{2}{n}$ an dem aktuellen Zyklus, wobei n die Anzahl aller möglichen Konklusionen darstellt. Nach der Durchführung der Regelaktion, wird der Fuzzy-Fehler des Systems ermittelt. Jede Regel R_r akkumuliert während der zweiten Phase einen Fehlerwert, dem jeweils der aktuelle Fuzzy-Regel-Fehler hinzugefügt wird. Am Ende der zweiten Lernphase wird diejenige Regel jeder Regelteilmenge mit identischen Prämissen ausgewählt, die den geringsten Fehlerwert aufweist. Die anderen Regeln werden entfernt. Die verbleibende Architektur ist somit wie ein gewöhnlicher Fuzzy-Regler zu interpretieren.

Während der beiden Lernphasen durchlaufen die Regeleinheiten des Systems zusätzlich einen Alterungsprozeß. Regeln die während eines vorgegebenen Zeitraums nicht oder nur sehr selten gefeuert haben, werden von dem Lernalgorithmus entfernt. In der dritten Lernphase werden, wie bereits beschrieben, die Zugehörigkeitsfunktionen optimiert.

4 NEFCON-I – Eine Implementation

Das am Institut für Betriebssysteme und Rechnerverbund der Technischen Universtität Braunschweig entwickelte Programm NEFCON-I (Neural Fuzzy Controller) stellt eine graphische Oberfläche für Sun-Workstations unter X-Windows zur Verfügung, die es erlaubt, einen Neuronalen Fuzzy Regler zu defi-

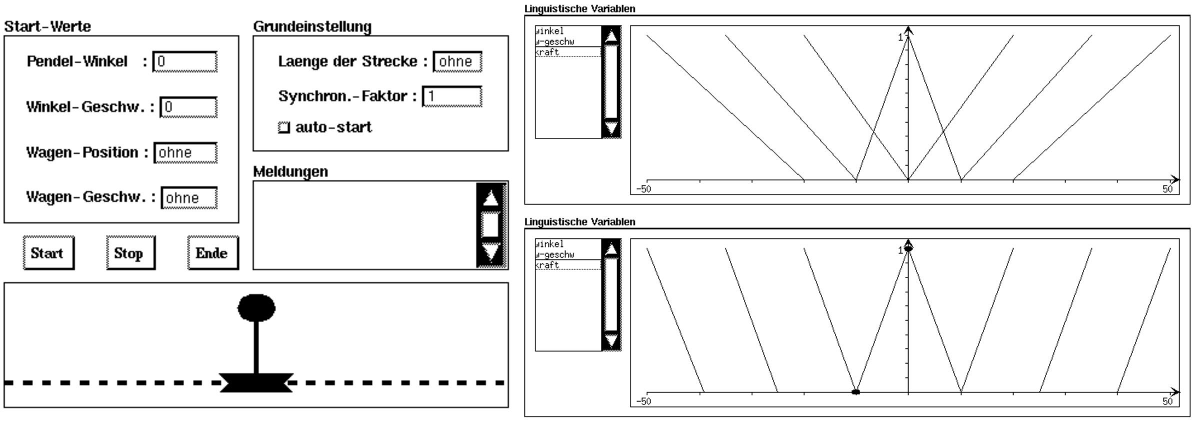

Bild 4: Pendelsimulation und Fuzzy-Mengen der Variablen Kraft vor und nach dem Lernvorgang

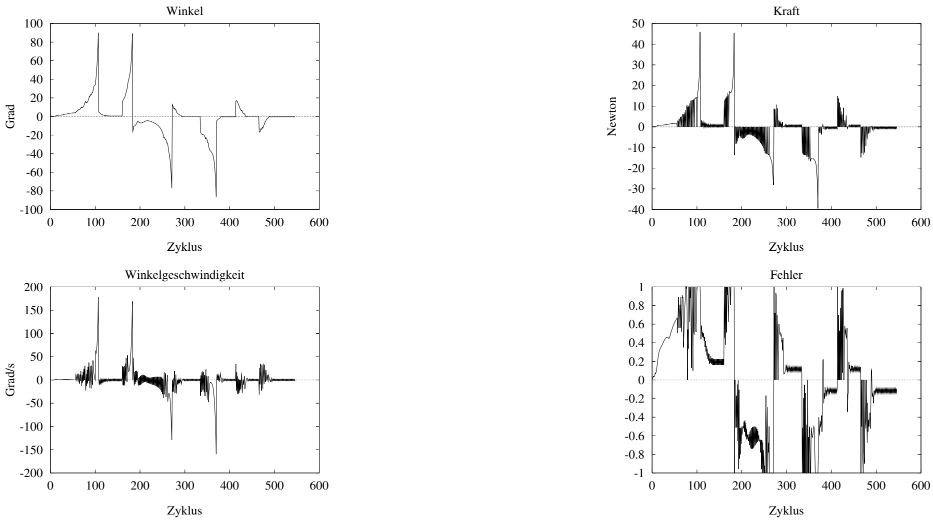

Bild 5: Zustand der Systemvariablen über 574 Zyklen. Das Pendel ist viermal umgefallen; zunächst zweimal ohne äußere Einwirkung und dann zweimal aufgrund einer externen Störkraft. Das Pendel wurde insgesamt viermal mit einer abwechselnd positiven und negativen Kraft gleichen Betrags aus der Balance gestoßen. Die letzen beiden Stöße konnten bereits ausbalanciert werden

nieren und online mit Hilfe einer gesondert implementierten Simulation eines dynamischen Systems zu trainieren (s. Bild 3).

Der Benutzer definiert die linguistischen Variablen und deren anfängliche Partitionierung mit Hilfe von Zugehörigkeitsfunktionen. Er kann entscheiden, ob anschließend eine Regelbasis eingegeben werden soll oder diese vom System erlernt werden soll. Zur Durchführung des Lernverfahrens muß der Fuzzy-Fehler für das zu regelnde dynamische System in Form linguistischer Regeln definiert werden. Im Gegensatz zum ersten in [10] vorgestellten Ansatz ist die Angabe einer Berechnungsvorschrift nicht mehr notwendig. Bevor der Lernvorgang gestartet werden muß ist vom Benutzer ein externes Programm anzugeben, das das physikalische System simuliert. Dieses Programm wird von NEFCON-I gestartet und tauscht mit ihm Steuer- und Meßdaten aus.

Für den Test des Modells wurde eine vereinfachte Version des inversen Pendels herangezogen, bei der lediglich Winkel und Winkeländerung betrachtet wurden. Die Testläufe ergaben, daß der Lernalgorithmus in der Lage ist, die Fuzzy-Mengen derart zu optimieren, daß das Regelverhalten stabiler wird. Die Oberfläche der Pendelsimulation sowie die Änderungen der Zugehörigkeitsfunktionen der Variablen „Kraft“ sind in Bild 4 dargestellt. Das Bild 5 zeigt einen Simulationsverlauf. Vergleiche mit einer Simulation der ARIC-Architektur [2] zeigen, daß NEFCON-I schneller und effektiver lernt. Das Pendel wird über weitere Winkelbereiche beherrscht als dies bei ARIC der Fall ist.

Literatur

[1] ANDREW G. BARTO, RICHARD S. SUTTON und CHARLES W. ANDERSON: *Neuronlike Adaptive Elements that Can Solve Difficult Learning Control Problems.* IEEE Trans. Systems, Man & Cybernetics, 13:834–846, 1983.

[2] HAMID R. BERENJI: *A Reinforcement Learning-Based Architecture for Fuzzy Logic Control.* Int. J. Approximate Reasoning, 6:267 –292, Februar 1992.

[3] HAMID R. BERENJI und PRATAP KHEDKAR: *Learning and Tuning Fuzzy Logic Controllers Through Reinforcements.* IEEE Trans. Neural Networks, 3:724–740, September 1992.

[4] PATRIK EKLUND, FRANK KLAWONN und DETLEF NAUCK: *Distributing Errors in Neural Fuzzy Control.* In *Proc. 2nd Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA'92)*, Seiten 1139–1142, Juli 1992.

[5] CHUEN CHIEN LEE: *Fuzzy Logic in Control Systems: Fuzzy Logic Controller, Part II.* IEEE Trans. Systems, Man & Cybernetics, 20:419–435, 1990.

[6] DETLEF NAUCK, FRANK KLAWONN und RUDOLF KRUSE: *Fuzzy Sets, Fuzzy Controllers, and Neural Networks.* Wissenschaftliche Zeitschrift der Humboldt-Universität zu Berlin, R. Medizin, 41(4):99–120, 1992.

[7] DETLEF NAUCK, FRANK KLAWONN und RUDOLF KRUSE: *Combining Neural Networks and Fuzzy Controllers.* In ERICH PETER KLEMENT und WOLFGANG SLANY (Herausgeber): *Fuzzy Logic in Artificial Intelligence (FLAI93)*, Nummer 695 in *Lecture Notes in Artificial Intelligence*, Seiten 35–46, Berlin, Juni 1993. Springer.

[8] DETLEF NAUCK und RUDOLF KRUSE: *Interpreting Changes in the Fuzzy Sets of a Self-Adaptive Neural Fuzzy Controller.* In *Proc. Second Int. Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems (IFIS'92)*, Seiten 146–152, College Station, Texas, Dezember 1992.

[9] DETLEF NAUCK und RUDOLF KRUSE: *A Neural Fuzzy Controller Learning by Fuzzy Error Propagation.* In *Proc. Workshop of North American Fuzzy Information Processing Society (NAFIPS92)*, Seiten 388–397, Puerto Vallarta, Dezember 1992.

[10] DETLEF NAUCK und RUDOLF KRUSE: *A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation.* In *Proc. IEEE Int. Conf. on Neural Networks 1993*, Seiten 1022–1027, San Francisco, März 1993.

[11] SANDRA M. SULZBERGER, NADINE N. TSCHICHOLD-GÜRMAN und SJUR J. VESTLI: *FUN: Optimization of Fuzzy Rule Based Systems using Neural Networks.* In *Proc. IEEE Int. Conf. on Neural Networks 1993*, Seiten 312–316, San Francisco, März 1993.

[12] Y. TSUKAMOTO: *An Approach to Fuzzy Reasoning Method.* In M.M. GUPTA, R.K. RAGADE und R.R. YAGER (Herausgeber): *Advances in Fuzzy Set Theory.* North-Holland, Amsterdam, 1979.