Learn Your Opponent's Strategy
(in Polynomial Time) !

Thema:

Es soll die Erlernbarkeit von Strategien untersucht werden.

Dazu wird ein Ansatz aus der Spieletheorie benutzt:

- Mehrere Agenten befinden sich in einer gemeinsamen Umgebung.
- Koordination der Strategien verbessert den Nutzen der Agenten.

Spieletheoretische Grundlagen

- Bei Spielen soll von den Spielregeln abstrahiert werden.
- Jeder Spieler wählt anfangs eine Strategie, die zu jeder Situation und jedem Wissen den auszuführenden „Zug“ bestimmt.
- Nach Wahl der Strategien verläuft das Spiel deterministisch.
- Das Spielergebnis wird durch Auszahlungsvektoren dargestellt.
- Spiele lassen sich als Abbildung der möglichen Strategienkombinationen auf Auszahlungstupel darstellen.
- 2-Personenspiele mit endlicher Strategienmenge werden meistens in Matrixform notiert.

Beispiel für ein 2-Personen-Spiel:

	L	**R**
T	**2** **1**	**0** **0**
D	**0** **0**	**1** **2**

Abb.1 : „Kampf der Geschlechter“

Im „Kampf der Geschlechter“ hat

- Spielerin A (links) die Wahl zwischen „T“ und „D“,
- Spieler B (oben) die Wahl zwischen „L“ und „R“.

Falls z.B. die Konstellation „D“ / „R“ gewählt wird erhält...
... A die Auszahlung 2,
... B die Auszahlung 1.

Von besonderem Interesse sind sogenannte „Gleichgewichtspunkte“.

Definition 1:

Ein **Nash-Gleichgewichtspunkt** bei einem n-Personen-Spiel ist ein Strategien-n-Tupel, mit der Eigenschaft, daß kein Spieler eine höhere Auszahlung erhalten kann, wenn er seine Strategie ändert, während alle übrigen Spieler ihre Strategie beibehalten.

Anmerkungen:

Es kann mehrere Nash-Gleichgewichtspunkte mit unterschiedlichen Auszahlungsvektoren geben.

In Abb.1 sind „T“ / „L“ und „D“ / „R“ Nash-Gleichgewichtspunkte:

- Beide Spieler erhalten eine Auszahlung von mehr als 0.
- Falls nur eine Partei die Strategie ändert, so beträgt deren Auszahlung anschließend 0.
- Es entsteht ein Interessenskonflikt aufgrund der unterschiedlichen Auszahlungstupel der Nash-Gleichgewichtspunkte.

Wiederholt gespielte Spiele

- Von nun an sollen Spiele wiederholt gespielt werden.
- Die Spieler haben in jeder Runde die Wahl zwischen „Aktionen“.
- Als „Strategie“ wird nun der langfristige Entscheidungsplan, der auch auf Aktionen der Gegner reagiert, bezeichnet.
- Anstelle der Auszahlung einer Runde wird häufig die durchschnittliche Auszahlung über mehrere Runden betrachtet.
- Für wiederholte 2-Personen-Spiele kann man sich eine Spielmatrix mit wiederholten Strategien denken, die aber nicht endlich ist.
- Für wiederholte Strategien bleibt die Definition der Nash-Gleichgewichtspunkte unverändert erhalten.
- Falls ein Spieler auf eine Teilmenge aller möglichen Strategien beschränkt wird, so ändern sich die möglichen Gleichgewichtspunkte.

Wahl eines konkreten 2-Personen-Spiels

	D	**C**
D	**–5** **–5**	**–10** **0**
C	**0** **–10**	**–2** **–2**

Abb.2 : „Gefangenendilemma“

	D	**C**
D	**P** **P**	**S** **T**
C	**T** **S**	**R** **R**

Abb.3 : „Verallgemeinertes Gefangenendilemma“

Es soll gelten: $S < P < R < T$ und $R > \frac{1}{2} * (T + P)$

2-Personen-Nullsummenspiele:

Ein Spieler gewinnt jeweils, was der andere verliert.

Die Interessen sind vollständig gegensätzlich.

$\Rightarrow$ Es besteht kein Verhandlungsinteresse.

„Gefangenendilemma“ ist **kein** Nullsummenspiel.

Im Gefangenendilemma ex. ein Verhandlungsinteresse!

Begründung:

- „D“ führt bei jeder Aktion des Mitspielers zu einer höheren Auszahlung als „C“ ⇒ Rationale Spieler werden „D“ / „D“ spielen.
- „C“/„C“ führt aber zu einer besseren Auszahlung für beide Spieler.
- Problem bei „C“ / “C“:
  Eine einseitige Abweichung zu „D“ / „C“ oder „C“ / „D“ führt zu einer noch höheren Auszahlung für den entsprechenden Spieler.

⇒ Die Möglichkeit bindender Verträge wäre hilfreich.

Beschränkungen:

- Hier soll davon ausgegangen werden, daß die einzig mögliche Art der Kommunikation die Beobachtung gegnerischer Aktionen ist.
- Die Spieler müssen ihre Strategien aufeinander abstimmen, um gute durchschnittliche Auszahlungen zu erhalten.
- Das einzige Mittel, kooperatives Verhalten zu Erzwingen, liegt in der Wiederholung des Spiels.

Die Beschränkungen reichen noch nicht, um Lern-Modell zu erstellen:

Falls beide Spieler die Strategie des anderen erlernen wollen und dies auch wissen, müssen „Täuschungsmanöver“, die die Hypothesen des anderen zum eigenen Vorteil beeinflussen, ebenfalls berücksichtigt werden. Dies führt zu einem zu komplexen Modell.

Weitere Beschränkung:

Komplexität wird eingeschränkt, in dem ein Spieler nur Strategien, die von **endlichen Automaten** repräsentiert werden können, wählen darf:

- Aktionen werden vom aktuellen Zustand des Automaten bestimmt.
- Die gegnerischen Aktionen bestimmen die Zustandsübergänge.
- Um irreversible Fehler zu vermeiden, sollen nur solche Automaten verwendet werden, bei denen jeder Zustand von jedem anderen aus erreichbar ist.

Beispiel für eine DFA–Strategie

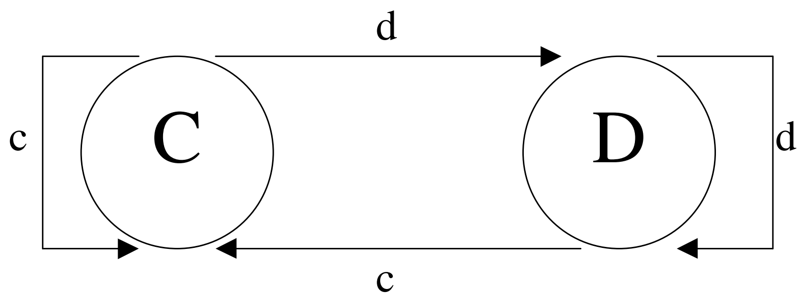


Abb.4 : „TIT-FOR-TAT“

Modell

- Spieler A wählt eine feste DFA-Strategie.
- Spieler B probiert, A’s Strategie zu erlernen. Dies sollte höchstens Polynomialzeit in der Größe des Automaten von A beanspruchen.
- Nachdem B die Strategie von A kennt, entwirft B einen optimalen Antwort - DFA.

Wie gut sind DFA-Strategien ?

Zunächst soll ein Algorithmus hergeleitet werden, der Automaten konstruiert, die vor Ausbeutung durch den Gegner sicher sind.

Feststellung:

- Beim Spiel gegen eine Automatenstrategie stößt man im Spielverlauf auf Zyklen.
- Um seine Auszahlung zu maximieren genügt es, einfache Zyklen zu analysieren.
- Gesucht: Ein Zyklus mit maximaler durchschnittlicher Auszahlung.

Notationen ($i \in \{A, B\}$):

A_i^Q Automatenstrategie des Spielers i mit Q Zuständen

Π Auszahlungsfunktion, bildet die Aktionen der Spieler auf Auszahlungstupel ab

Π_i Auszahlung des Spielers i (Projektion von Π)

Π^*_i durchschnittliche Auszahlung des Spielers i über mehrere Runden (arithmetisch gemittelt)

Definition 2:

Ein Zyklus C **implementiert** $< \alpha, \beta >$,
falls $\Pi^*_A(C) = \alpha$ und $\Pi^*_B(C) = \beta$ ist.

Definition 3:

Ein Zyklus C in A_i^Q **unterstützt** $< \alpha, \beta >$, falls:
1. C implementiert $< \alpha, \beta >$.
2. $\forall C'$: Aus C' implementiert $< \alpha', \beta' >$ folgt ($\alpha' < \alpha \Rightarrow \beta' < \beta$).

Definition 4:

Ein **Zyklus** C in A_i^Q **ε-unterstützt** $< \alpha, \beta >$, falls
er ein $< \alpha', \beta' >$, mit $\| < \alpha', \beta' > - < \alpha, \beta > \| \leq \varepsilon$, unterstützt.

Definition 5:

Ein **Automat** A_i^Q **ε-unterstützt** $< \alpha, \beta >$, falls
ein Zyklus C in A_i^Q existiert, so daß C $< \alpha, \beta >$ ε-unterstützt.

Anschaulich:

Die ε-Unterstützung von $< \alpha, \beta >$ durch einen Automaten (Spieler A) bedeutet, daß Spieler B nur folgende Möglichkeiten hat:

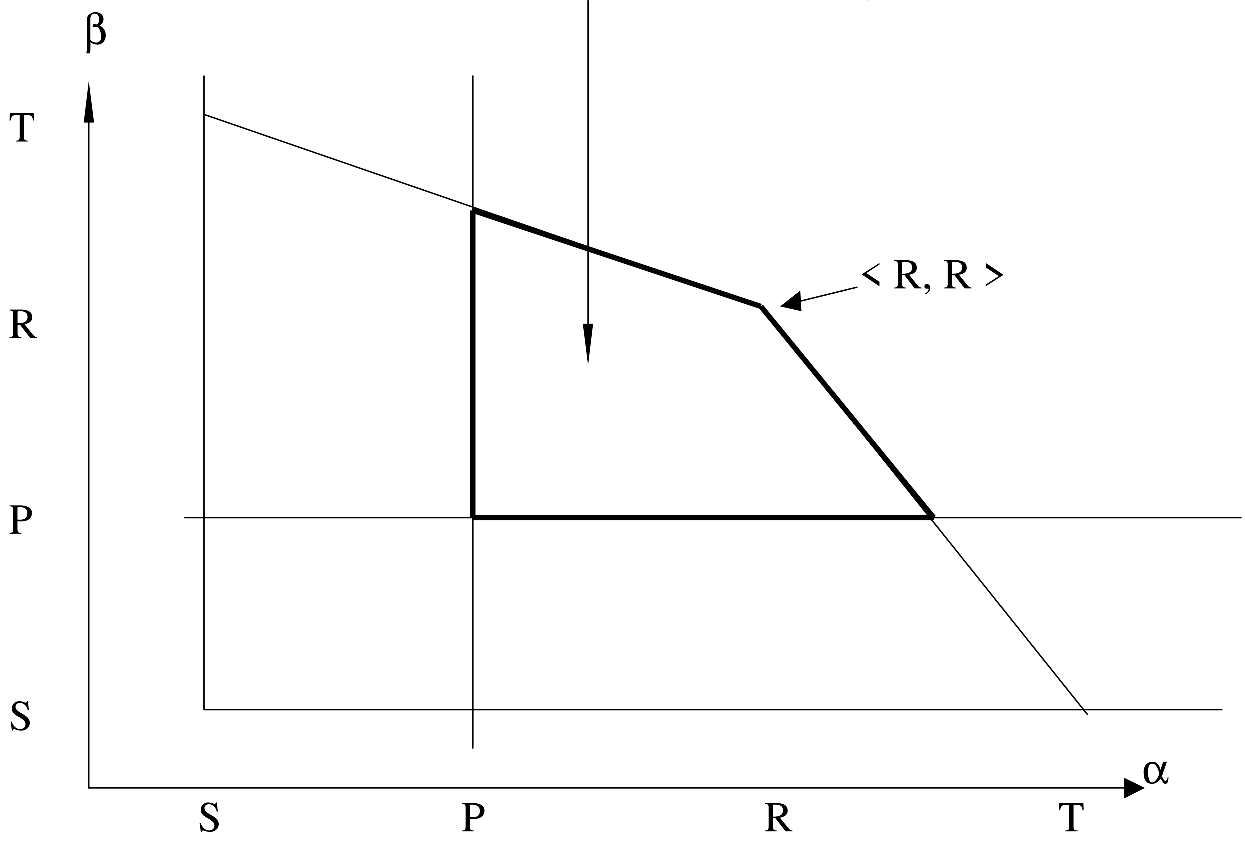
1. Spieler B erhält eine durchschnittliche Auszahlung von (etwa) β
   $\Rightarrow$ Spieler A erhält die Auszahlung ($\approx$) α
2. Spieler B wählt eine Strategie, deren Auszahlung für Spieler A weniger als ($\approx$) α beträgt.
   $\Rightarrow$ Spieler B bekommt weniger als ($\approx$) β ausbezahlt.

Hierbei schwächt „ε" den Vektor $< \alpha, \beta >$ auf eine ε-Umgebung ab.

Fazit:

Automaten, die einen geeigneten Vektor ε-unterstützen, sind vor Ausbeutung durch den Gegner abgesichert. Falls der Gegner die vorgesehene Auszahlung verweigert, schadet er sich selbst.

„The Individual Rational Region“



- Abb. 5 -

Nach Neymann [3] und Rubinstein [4] gilt:

Falls ein Spieler auf DFA-Strategien beschränkt ist, so kann jeder Punkt innerhalb der „Individual Rational Region“ zum Gleichgewichtspunkt gemacht werden.

Begrenzung der „Individual Rational Region“ :

1.) $\alpha, \beta \geq P$, denn mit weniger Auszahlung als P braucht sich kein Spieler zufrieden zu geben (Spiele nur D!).

2.) Die Vektoren müssen sich mittels $< S, T >$, $< R, R >$, $< T, S >$ und $< P, P >$ linear kombinieren lassen um als durchschnittliche Auszahlung in Frage zu kommen.

Konstruktion einer Automatenstrategie

Theorem 1:

Es existiert zu jedem $< \alpha, \beta >$ aus der „Individual Rational Region" ein Automat A_i^Q, der $< \alpha, \beta >$ ε-unterstützt ($\varepsilon = c * Q^{-1}$).

Außerdem existiert ein Algorithmus, der für gegebenes $< \alpha, \beta >$, in polynomieller Zeit in Q, den entsprechenden Automaten konstruiert.

Idee der Konstruktion:

- Der Automat enthält einen Konsens-Zyklus. Dieser implementiert eine bessere Auszahlung als < P, P > für beide Spieler.
- Spieler B bleibt zu jedem Zeitpunkt nur eine mögliche Aktion, um im Konsens-Zyklus zu bleiben. Andernfalls springt der Automat an den Anfang einer Strafkette.
- Die Strafkette sieht wie folgt aus:
  Der Automat spielt „D" bis Spieler B oft genug hintereinander „C" gespielt hat. Spielt B einmal „D", so führt dies wieder zum Anfang der Strafkette.
- Das Ende der Strafkette wird mit dem Konsens-Zyklus verbunden.

Falls Spieler B nicht im Konsens-Zyklus bleibt bleiben nur 2 Möglichkeiten (die Strafkette sei bei 2. als „hinreichend lang" angenommen):

1. für immer in der Strafkette bleiben
   → Die durchschnittliche Auszahlung konvergiert gegen P.
2. bis zum Ende der Strafkette „C" spielen
   → Die Auszahlung beträgt mehrfach S (S < P < R < T).

⇒ Die beste Strategie für Spieler B ist im Konsens-Zyklus zu bleiben.

Konstruktionsansatz

1.) Konstruktion der Strafkette

Es genügt, eine Länge für die Strafkette festzulegen. Wähle ½Q.

2.) Konstruktion des Konsens-Zyklus

Ziel: Approximation der durchschnittlichen Auszahlung $< \alpha, \beta >$ mit ½Q Zuständen.

Die Aktionen von Spieler B sind im Konsens-Zyklus erzwungen.
$\Rightarrow$ Zustände können durch Auszahlungen charakterisiert werden. Es genügt sogar Π_B zur eindeutigen Charakterisierung. Man erhält entsprechend die 4 Zustandstypen T, S, R und P.

Berechne den zur Implementierung von $< \alpha, \beta >$ optimalen Anteil jedes Zustandstypen an der Länge des Konsens-Zyklus.
$\rightarrow$ Lösen eines lineares Gleichungssystem konstanter Größe

Aus den Anteilen der Zustandstypen am Konsens-Zyklus läßt sich leicht ein konkreter Konsens-Zyklus mit ½ Q Zuständen herleiten, der $< \alpha, \beta >$ ε-implementiert.

Wähle c groß genug, um den Rundungsfehler in 2.) zu kompensieren, und setze $\varepsilon := c * Q^{-1}$.

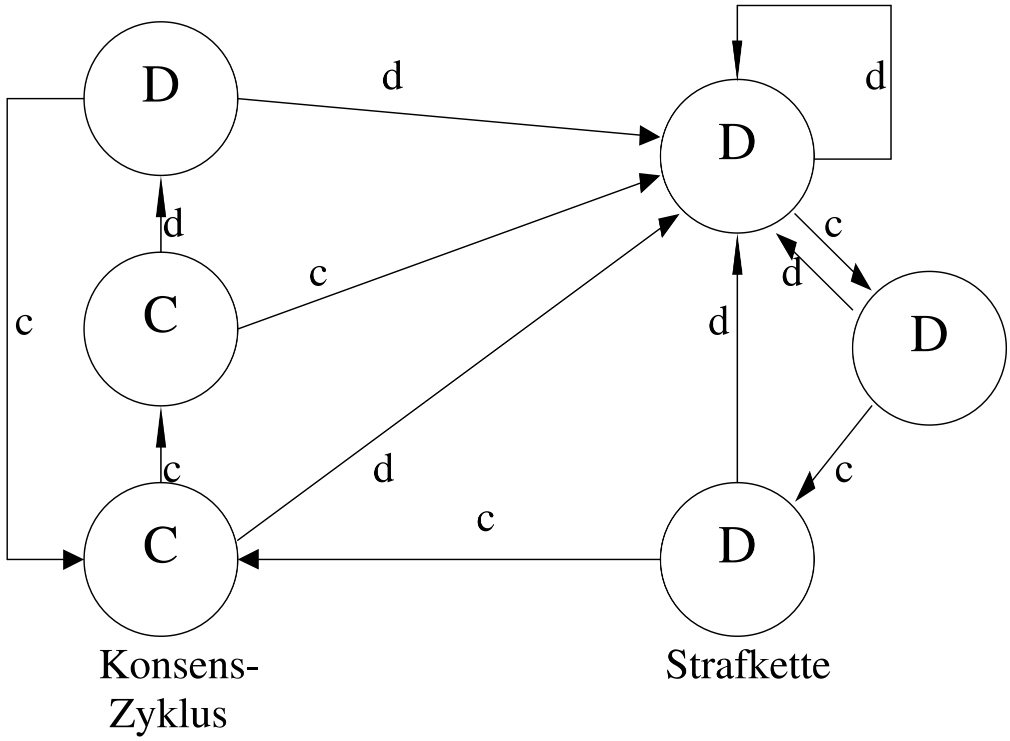
Dann gilt:

Der konstruierte Automat ε-unterstützt $< \alpha, \beta >$.

Rechenzeit :

- Lösen des linearen Gleichungssystems: konstante Zeit
- Konstruktion des Automaten mit berechneter Zustandstypenanzahl: lineare Rechenzeit in der Zustandszahl des Ergebnis-Automaten.

Beispiel – Automat für $\alpha = \beta = (R + S + T) / 3$, $Q = 6$:



Eine nicht in Polynomialzeit lernbare Automaten-Strategie...

Im letzten Abschnitt wurde gezeigt:

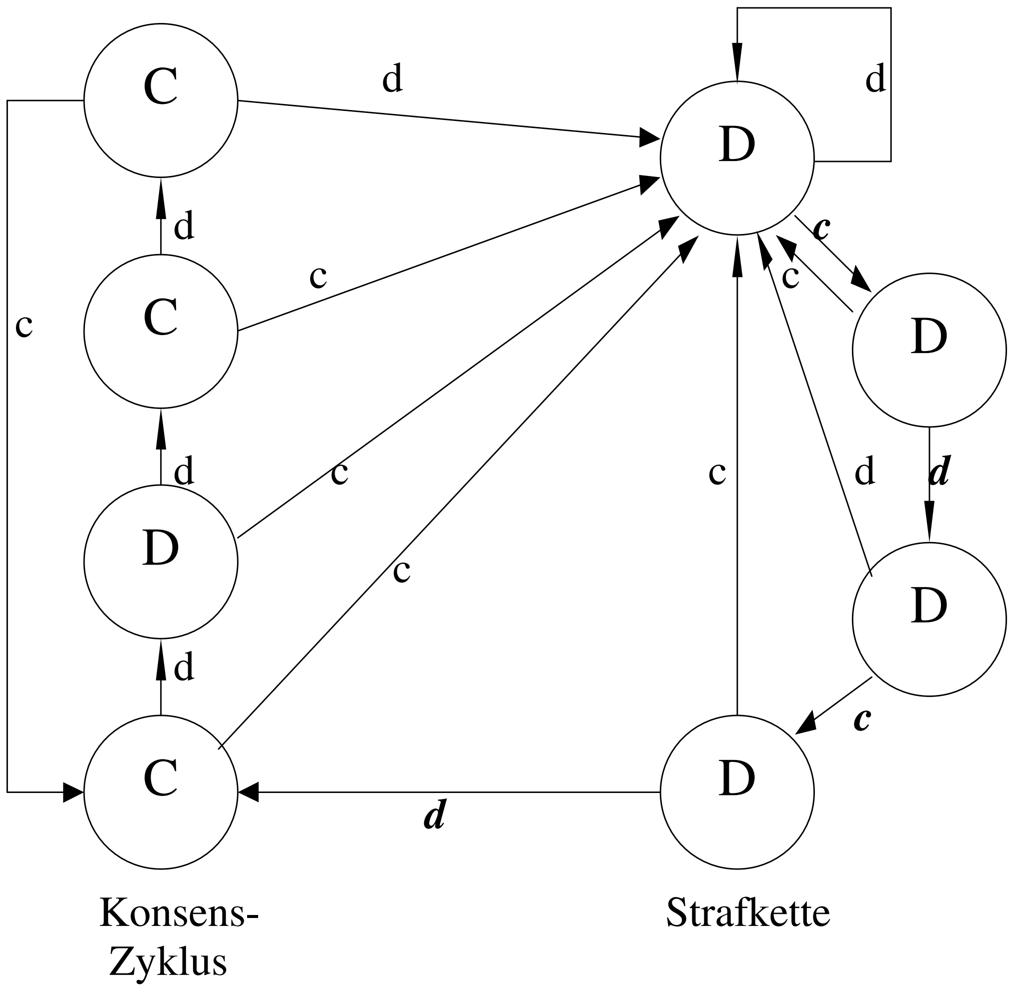
Es lassen sich effizient Automaten konstruieren, die einen bestimmten Auszahlungsvektor in der „Individual Rational Region“ als Gleichgewichtspunkt „anbieten“.

Offene Frage: Sind diese Automaten allgemein effizient lernbar ?

Hierbei soll „effizient lernbar“ genauer bedeuten:

- Das Verhalten des Automaten soll exakt erlernbar sein.
- Die benötigte Zeit darf nur polynomiell in der Zustandszahl sein.

Ein Beispiel für nicht effizient lernbare Automaten:



Eigenschaften des skizzierten Automaten:

- Es existieren wieder ein Konsens-Zyklus und eine Strafkette, die beide ½Q Zustände enthalten.
- Im Konsens-Zyklus führt eine Transition zum Nachfolger im Zyklus, eine Transition führt zum Start der Strafkette.
- In der Strafkette führt eine Transition zum Nachfolger, eine Transition führt zum Start der Strafkette zurück.
- Die Sequenz zum Entkommen von der Strafkette wird zufällig gewählt, ist also i.a. von „½Q mal C spielen“ verschieden.
  Im Beispiel ist diese Sequenz cdcd.

Warum sind solche Automaten nicht effizient lernbar ?

- Entkommenssequenz und Konsens-Zyklus sind anfangs unbekannt.

- Mit einer Wahrscheinlichkeit von ($1 - (½)^{½Q}$) wird die Strafkette betreten, da in jedem Zustand des Konsens-Zyklus die „richtige“ Aktion nur mit Wahrscheinlichkeit ½ erraten werden kann.

- Um aus der Strafkette zu entkommen, muß eine binäre Sequenz der Länge ½Q gespielt werden. Da der Automat innerhalb der Strafkette nur „D“ spielt, also keine Informationen preis gibt, kann die Sequenz nur geraten werden. Dies erfordert eine erwartete Zeit $\Omega(2^Q)$.

- Da mit einer Wahrscheinlichkeit von ($1 - (½)^{½Q}$) die erwartete Zeit zum Lernen des Automaten exponentiell in Q ist, ist die erwartete Zeit auch insgesamt exponentiell in Q.

Beobachtung:

- Automaten-Strategien wurden zur Beschränkung der Komplexität gewählt.

- Durch die Verallgemeinerung der zunächst vorgestellten Automatenklasse wird die Komplexität offensichtlich erhöht.

- Die Größe des Automaten ist kein ausreichendes Komplexitätsmaß.

- Falls Spieler A einen rationalen Auszahlungsvektor anstrebt gibt es keinen Grund die Lernbarkeit seiner Strategie zu erschweren.

- Es könnte Sinn machen, die Strategien von Spieler A noch stärker (als auf allgemeine DFA-Strategien) einzuschränken.

Lernbare Automaten-Strategien

Verfeinerung des Komplexitätsmaßes

Zustandstyp eines Zustandes:

Auszahlung für Spieler B (in der nächsten Runde), wenn Spieler B gemäß seiner besten Strategie spielt.

Chunks sind zusammenhängende Zustände gleichen Typs.

Als weiteres Komplexitätsmaß soll nun die Anzahl Chunks gelten.

Notation:

#(A)	Anzahl Zustände im Automaten A
#C(A)	Anzahl Chunks in A

Definition 6:

Ein Automat A ist genau dann **einfacher** als ein Automat A‘,

notiert durch $A <_C A'$, falls

$\#(A) < \#(A')$,

oder $\#(A) = \#(A') \wedge \#C(A) < \#C(A')$.

Definition 7:

Die Klasse C_{simp} sei wie folgt definiert:

$$C_{simp} \equiv \{A \mid (\exists <\alpha, \beta>, \text{ so daß } A <\alpha, \beta> \text{ unterstützt}) \wedge (\forall A' : A' \text{ unterstützt } <\alpha, \beta> \Rightarrow A <_c A') \}$$

Anschaulich:

C_{simp} besteht aus den (bezüglich $<_c$) einfachsten Automaten, die einen Vektor $< \alpha, \beta >$ unterstützen.

Zu zeigen:

Automaten/Strategien aus der Klasse C_{simp} sind effizient lernbar.

Zwischenschritt:

Es werden zunächst Eigenschaften aller Automaten in C_{simp} gezeigt.

Lemma 1:

Jeder Automat in C_{simp} besteht aus einem Konsens-Zyklus und einer Strafkette.

Beweis:

- Alle $A \in C_{simp}$ unterstützen einen Vektor $< \alpha, \beta >$, also existiert ein Zyklus in A, der $< \alpha, \beta >$ implementiert. Wähle den am wenigsten komplexen dieser Zyklen, und nenne ihn C_{kons}.
- Verläßt eine Transition C_{kons}, so gibt es einen Weg zurück zu C_{kons}, da alle Zustände verbunden sind.
  Nenne die Zusammenhangskomponente außerhalb von C_{kons}, die Spieler B die geringste durchschnittliche Auszahlung liefert, C_{Straf}.
- Falls es noch Zustände außerhalb von C_{kons} und C_{Straf} gibt:
  Ändere alle Transitionen des Automaten A, die C_{kons} verlassen, ab, so daß sie zum Start von C_{Straf} führen.
  Mindestens ein Zustand ist nicht mehr erreichbar und kann entfernt werden. Der neue Automat A‘ unterstützt $< \alpha, \beta >$ und A‘ $<_c$ A.
  Widerspruch zu $A \in C_{simp}$!

Lemma 2:

In der Strafkette aller Automaten $A \in C_{simp}$ kommt nur Zustandstyp S (A spielt „D" und Spieler B spielt „C" um zu entkommen) vor.

Beweis:

Annahme:
$\exists A \in C_{simp} \mid$ A's Strafkette enthält nicht nur Zustandstyp S.

Fall 1: C_{straf} enthalte verschiedene Zustandstypen.

A' entstehe aus A durch Ersetzung indem alle Zustandstypen von C_{straf} durch S ersetzt werden.

Da $S < P < R < T$ sinkt die durchschnittliche Auszahlung von C_{Straf} für Spieler B (bei Wahl der optimalen Strategie) durch die Ersetzung.

$\Rightarrow$ A' unterstützt den gleichen Vektor $< \alpha, \beta >$ wie A.

Es ist aber $A' <_c A$, da $\#C(A') < \#C(A)$.
Dies steht im Widerspruch zu $A \in C_{simp}$.

Fall 2: Die Strafkette enthalte nur einen Zustandstyp ungleich S.

Betrachte wieder Automat A' aus Fall 1.
Da $S < P < R < T$, genügen weniger Zustände vom Typ S, für höchstens die gleiche durchschnittliche Auszahlung des Gegners, beim Durchlaufen der Strafkette.

Damit läßt sich die Länge der Strafkette, also die Komplexität von A verringern, im Widerspruch zu $A \in C_{simp}$.

Lemma 3:

Alle $A \in C_{simp}$ enthalten höchstens einen Chunk jeden Zustandstyps.

Beweis:

Gäbe es zu einem Zustandstyp mehrere Chunks in einem Automaten aus $A \in C_{simp}$, so könnte man einen einfacheren Automaten mit dem gleichen unterstützen Vektor erhalten, indem man die Zustände gleichen Typs zusammen gruppiert.

Lemma 4:

Es gibt in C_{simp} weniger als

$$4! * Q^3$$

verschiedene Automaten mit Q Zuständen.

Beweis:

Es gibt 4 Zustandstypen, also nur 4! Möglichkeiten, diese anzuordnen.

Anzahl Zustände jedes Zustandstyp:

- Es stehen Q Zustände zur Verfügung.
- Anzahl Zustände eines Typs ist durch die übrigen festgelegt.

$\Rightarrow$ Es gibt weniger als Q^3 Aufteilungen der Q Zustände auf Typen.

Daraus ergibt sich direkt die obere Schranke $4!Q^3$.

Theorem 2:

C_{simp} ist effizient lernbar.

Beweis:

Nach Lemma 4 ist der Hypothesenraum nur polynomiell groß in Q.

Hypothesen lassen sich effizient überprüfen:

- Alle Zustandstypen hängen in Automaten aus C_{simp} zusammen.
- Es kann mit linearem Aufwand in Q verifiziert werden, ob der Konsens-Zyklus dem des Hypothese-Automaten entspricht.

Um alle Hypothesen zu einem festen Q zu überprüfen, genügt ein in Q polynomieller Zeitaufwand.

Falls Q nicht bekannt ist, so bleibt der Zeitaufwand polynomiell in Q, falls aufsteigend über alle möglichen Größen iteriert wird.

Folgerung aus Theorem 2:

Angenommen Spieler A ist auf Strategien aus C_{simp} beschränkt, die sich von Automaten mit Q_{max} Zuständen repräsentieren lassen.

Dann kann Spieler B einen Automaten polynomieller Größe in Q_{max} konstruieren, der gegen alle möglichen Automaten von A optimal spielt.

Zusammenfassung:

Rahmen: 2-Personen-Spiel, zum Teil gemeinsame Interessen

→ Koordination der Strategien ist von Vorteil

Annahme : keine Kommunikation außerhalb der Spiel-Aktionen

Untersuchungsgegenstand:
Lernbarkeit von Strategien zum Zwecke der Koordinierung

Der Begriff der Strategie ist zunächst zu mächtig für ein Lernmodell.

→ Einschränkung auf endliche Automaten

Die Klasse der DFA-Strategien ist noch immer hinreichend mächtig.

Es können effizient Automaten zu allen, unter realistischen Annahmen wünschenswerten, Auszahlungsvektoren $< \alpha, \beta >$ konstruiert werden.

Der Mitspieler hat bei diesen Automaten nur die Wahl zwischen:

1.) einer vorgeschriebenen Strategie mit Auszahlungsvektor $< \alpha, \beta >$,
2.) einer schlechteren Auszahlung als β.

Fall 1 setzt aber voraus, daß der Mitspieler die Strategie versteht.

Problem dabei:

Die Klasse der Automaten-Strategien ist nicht effizient lernbar.

⇒ Es ist sinnvoll die Menge der Strategien weiter einzuschränken.

Die Klasse C_{simp} enthält nach einem bestimmten Kriterium einfachste Automaten zu allen unterstützbaren Auszahlungsvektoren.

C_{simp} ist effizient lernbar !