Projektgruppe HeISs:
Hierarchische Inferenz evolvierender Spielstrategien

1 Zeitraum

Wintersemester 2013/2014 und Sommersemester 2014

2 Veranstalter

Dipl.-Inf. Hasan Ibne Akram <hasan.akram@tu-dortmund.de>, Tel. 7757
Dipl.-Inf. Malte Isberner <malte.isberner@cs.uni-dortmund.de>, Tel. 5806
Prof. Dr. Bernhard Steffen <steffen@cs.tu-dortmund.de>, Tel. 5801
Informatik Lehrstuhl 5, Otto-Hahn-Straße 14, Raum 135/131

3 Aufgabe

Sowohl bei traditionellen Strategie-Brettspielen wie *Go* oder *Schach* als auch bei modernen Strategie-Computerpielen wie *Civilization* oder *StarCraft* ist das 'Durchschauen' des Gegners wesentlicher Bestandteil erfolgreicher Spielstrategien. Bei ansonsten weitgehend ebenbürtigen Gegnern, seien es Computer oder Menschen, gibt die Kennntis gegnerischer Stärken und Schwächen typischerweise den Ausschlag. Tatsächlich kann dieser Faktor sogar die 'objektiv' größere Stärke eines Gegner kompensieren.

Bei Computerspielen kommt oft erschwerend hinzu, dass die *genauen Spielregeln/Gesetze* im Laufe des Spiels erst noch ermittelt werden müssen. Man denke z.B. an die implementierte Physik, die konkrete Berechnung von Schäden, Fahr- oder Flugbahnen, Bremswegen, usw.. Erst wenn man die verstanden hat, kann man optimal agieren.

3.1 Verfahrenstechnischer Hintergrund

Im Rahmen der PG sollen zwei klassische Verfahren aus dem Bereich des Automatenlernens [3] zur Ermittlung gegnerischer Strategien bzw. der konkreten Spielregeln eingesetzt, evaluiert, verglichen und insbesondere kombiniert und optimiert werden (vgl. Abb. 1).

Passives Automatenlernen: [5] hier wird versucht, aus allen verfügbaren Spielzugprotokollen von Spielen des Gegners nachträglich dessen strategisches Konzept möglichst genau zu bestimmen. Dieses Vorgehen erinnert beispielsweise an die akribischen Stärken/Schwächenanalysen eines Schachprofis vor dem Kandidatenturnier oder gar einer Weltmeisterschaft - eben Fällen, wo der spezifische Gegner bekannt ist.

Aktives Automatenlernen: [2] hier wird versucht, durch geeignete Spielführung, das gegnerische Strategiekonzept gezielt auszuloten. Stärke dieses Vorgehens ist es, dass auf diese Weise Annahmen

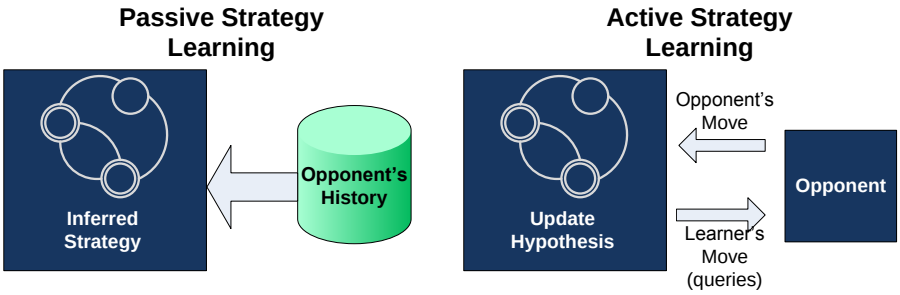

Abbildung 1: Passives und Aktives Lernen von Spielstrategien.

über das gewonnene Strategiekonzept explizit hinterfragt/validiert werden können. Da die validierenden Spielführungen aber typischerweise nicht gewinnbringend sind, ist dieses Vorgehen nur während Tranings- oder Aufwärmphasen sinnvoll.

3.2 Anwendungsszenario

Untersucht werden soll das Zweipersonenspiel ChainReaction[1] (Abb. 2), bei dem es darum geht, alle Spielsteine des Gegners auf dem Spielbrett zu erobern. Dieses Spiel ist deshalb besonders geeignet, da hier sowohl die konkreten Spielregeln, (d.h. die genaue Berechnung des Effekts eines Spielzuges), als auch die gegnerische Spielstrategie Lernthema sein können.

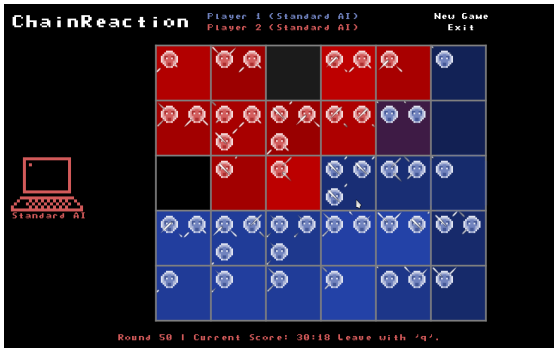

Abbildung 2: Screenshot aus ChainReaction.

Bei ChainReaction können gegnerische Zellen erobert werden, indem eine eigene Zelle durch das Platzieren eines Spielsteins ihre „Kapazität“[2] erreicht und alle auf ihr befindlichen Steine gleichmäßig auf all ihre horizontalen und vertikalen Nachbarzellen verteilt. Abb. 3 illustriert, wie das zu einer Kettenreaktion führen kann, die jedoch je nach nach Auswertungsreihenfolge der Zellen zu sehr unterschiedlichen Ergebnissen führen kann.

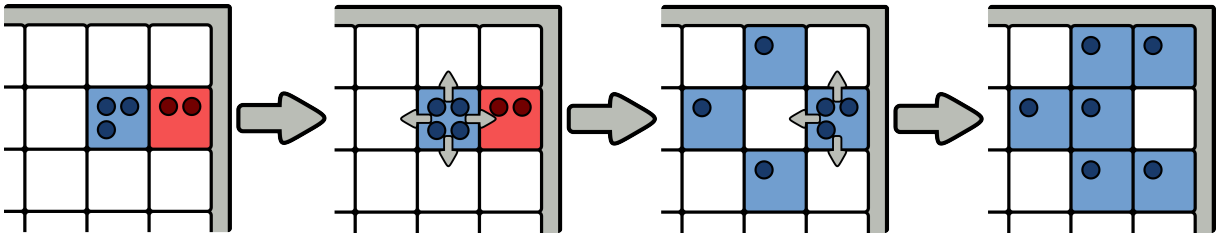

Abbildung 3: Visualisierung der Kettenreaktion durch Hinzufügen eines Steins in die linke Zelle.

[1] `http://cr.freewarepoint.de/`
[2] Die Kapazität einer Zelle entspricht hierbei immer der Anzahl aller horizontalen und verikalen Nachbarzellen.

Das führt zu folgender zweischrittigen Vorgehensweise:

1. Lernen der Auswertungsstrategie zur Berechnung des Effekts eines Spielzuges. Diese Strategie ist typischerweise deterministisch und bleibt unverändert. Damit können die klassichen Lernverfahren unmodifiziert zur Anwendung gebracht werden. Das gleiche gilt auch für beliebige sich nicht ändernde, deterministische Spielstrategien.
2. Lernen sich ändernder gegnerischer Spielstrategien. Hier geht es z.B. um die Behandlung von Nichtdeterminismus oder Strategieevolution durch (ggf. ebenfalls lernbasierte) Strategieveränderungsprozesse.

3.3 Evolution von Strategien

Um den Kern dieser Begriffe zu illustieren, genügt die Betrachtung des Spiels "5 in einer Reihe", wo zwei Spieler abwechselnd Felder eines karierten Bogens markieren, bis es einem gelingt, fünf Felder in einer Reihe für sich markiert zu haben.

Eine sehr naive Strategie könnte zum Kern haben, immer dann wenn der Gegner 4 Felder in einer Reihe hat, das fünfte Feld zu blockieren. Ist eine derartig primitve Defensive erkannt, lässt sie sich leicht schlagen, indem man 'offene' Vierer konstruiert, d.h. vier markierte Felder in einer Reihe, die an beiden Seiten noch ein unmarkiertes Feld haben. Hier scheitert die primitve Defensive unmittelbar. Erkennt der Gegner diese Schwäche, könnte er seine Strategie verbessern, indem er schon alle Situationen zu blockieren versucht, die die Konstruktion eines offenen Vierers im nächsten Schritt erlauben. Aber auch zu dieser Defensivstrategie gibt es, sofern sie erkannt wurde, wieder geeignete Gewinnstrategien, zum Beispiel die Konstruktion zweier offener Dreier.

3.4 Ziel der PG

Ziel der PG ist es, durch Kombination und Erweiterung von passivem und aktivem Lernen ein effektives Verfahren zu entwickeln und zu implementieren, das es ermöglicht, gegnerische Strategien unterschiedlicher Komplexität aufzudecken. Basisfall sind dabei sich nicht ändernde, deterministische Strategien. Aufbauend auf eine Lösung für den Basisfall, soll das Verfahren erweitert werden, um Effekte von Nichtdeterminismus und Evolution mit zu erfassen. Am Beispiel des Zweipersonenspiels ChainReaction sollen die entwickelten Verfahren evaluiert und illustriert werden.

3.5 Geplantes Vorgehen

Das erste PG-Semester wird vorwiegend von der Einarbeitung in die verschiendenen Lernverfahren und Werkzeuge, wie LearnLib [4, 6] für aktives Lernen und GIToolBox [1] für passives Lernen, sowie von Umsetzung des ersten Schrittes geprägt sein. Im zweiten PG-Semester stehen dann die Evaluation sowie die Generalisierungen für Schritt zwei im Vordergrund. Untersucht werden soll, unter welchen Bedingungen welche Verfahren anwendbar bleiben, wo Verfahrenserweiterungen helfen, wo grundsätzliche Probleme/Grenzen liegen, und welche realistischen Anwendungsbereiche abgedeckt werden können. Letzteres muss sich keinesfall auf Spiele beschränken. Vielmehr hat sich gezeigt, dass Automatenlernen ein z.B. adäquates Hilfsmittel zur testbasierten Qualitätssicherung evolvierender Systeme ist [7].

4 Teilnahmevoraussetzungen

Vorausgesetzt werden:

- Fundierte Kenntnisse in mindestens einer objektorientierten Programmiersprache, bevorzugt Java

- Elementare Kenntnisse über Modellierungstechniken, wie sie zum Beispiel in den Vorlesungen „Formale Methoden des Systementwurfs“, „Virtualisierung und Compilation“, „Softwarekonstruktion“ und „Dienstleistungsinformatik“ vermittelt werden.

Wünschenswert sind zudem:

- Kenntnisse im Umgang mit Eclipse-Werkzeugen und -Programmierung
- Kenntnisse über Geschäftsprozesse

5 Minimalziel

Im Rahmen der PG sollen mindestens folgende Punkte umgesetzt werden:

- Inferieren der Zugauswertungsregeln von ChainReaction mittels aktiven und passiven Automatenlernverfahren
- Modellieren von (ausführbaren) Spielstrategien für ChainReaction als Automaten/Prozesse
- Adaption der zuvor verwendeten Automatenlernverfahren auf Inferenz dieser Spielstrategien
- Evaluation der Lernverfahren anhand einer kleinen "Competition“ innerhalb der PG.
- Analyse zur Eignung der entwickelten Verfahren bei evolvierenden und/oder nicht-deterministischen Spielstrategien.

6 Literatur

[1] Hasan Ibne Akram, Colin de la Higuera, Huang Xiao, and Claudia Eckert. Grammatical inference algorithms in matlab. In *Proceedings of ICGI*, volume 6339 of *Lecture Notes in Computer Science*, pages 262–266. Springer-Verlag, 2010.

[2] Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[3] Colin de la Higuera. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press, 2010.

[4] Falk Howar, Malte Isberner, Maik Merten, and Bernhard Steffen. Learnlib tutorial: From finite automata to register interface programs. In *Proceedings of ISoLA (1)*, volume 7609 of *Lecture Notes in Computer Science*, pages 587–590. Springer Verlag, 2012.

[5] José Oncina and Pedro García. Identifying regular languages in polynomial time. In H. Bunke, editor, *Advances in Structural and Syntactic Pattern Recognition*, volume 5 of *Series in Machine Perception and Artificial Intelligence*, pages 99–108. World Scientific, 1992.

[6] Bernhard Steffen, Falk Howar, and Maik Merten. Introduction to active automata learning from a practical perspective. In *Proceeding sof SFM*, volume 6659 of *Lecture Notes in Computer Science*, pages 256–296. Springer Verlag, 2011.

[7] Stephan Windmüller, Johannes Neubauer, Bernhard Steffen, Falk Howar, and Oliver Bauer. Active continuous quality control. In *Proceedings of CBSE*. ACM, 2013.

7 Rechtlicher Hinweis

Die Ergebnisse der Projektarbeit und die dabei erstellte Software sollen der Fakultät für Informatik uneingeschränkt für Lehr- und Forschungszwecke zur freien Verfügung stehen. Darüber hinaus sind keine Einschränkungen der Verwertungsrechte an den Ergebnissen der Projektgruppe und keine Vertraulichkeitsvereinbarungen vorgesehen.