Sommerakademie Leysin, Studienstiftung des deutschen Volkes
12. August 2018 – 25. August 2018

AG4: Können Computer denken? Automatisches Beweisen und künstliche Intelligenz

Themen für die Projektphase

S. Koch, L. S. Krapp, J. Roos

Im folgenden werden mögliche Themen für die Gruppenarbeitsphase in der zweiten Woche umrissen. Die Themen werden kurz beschrieben und es werden Literaturhinweise gegeben.

1 Entscheidungsalgorithmus für Aussagenlogik in Prolog

Es gibt zahlreiche Algorithmen, die in einer bestimmten Sprache der Aussagenlogik (d.h. für eine bestimmte Auswahl an Verknüpfungssymbolen) für einen gegebenen Satz entscheiden, ob dieser wahr oder falsch ist. Wir wollen (z.B. basierend auf [2]) einen solchen Entscheidungsalgorithmus erarbeiten, in der Programmiersprache Prolog implementieren und ihn auf einige Beispielsätze anwenden. Um zu verdeutlichen, welche Art von mathematischen Problemen auf aussagenlogischer Ebene gelöst werden können, behandeln wir Beispiele, für die unser Entscheidungsalgorithmus in der Lage ist, eine Lösung auszugeben, wie das Flussüberquerungsrätsel (Wolf, Ziege, Kohlkopf) und das Vier-Farben-Problem für gegebene Landkarten.

2 Entscheidungsalgorithmus für reelle Zahlen

Im mathematischen Kontext begegnen einem oft Sätze in FOL. So sind zum Beispiel die Axiomensysteme für Gruppen, Ringe und Körper in dieser formalisierbar. Umgekehrt kann man auch eine mathematische Struktur wie den angeordneten Körper $(\mathbb{R}, +, -, \cdot, 0, 1, <)$ und die Menge aller FOL-Sätze betrachten, die dieser erfüllt. Beispielsweise gilt in $\mathbb{R}$ der Satz $\exists x\ x \cdot x = 1+1$ („Die Zahl 2 hat eine Quadratwurzel.“), wohingegen $\exists x\ x \cdot x + 1 = 0$ („Die Zahl -1 hat eine Quadratwurzel.“) nicht gilt. Nun stellt sich die Frage, ob es einen Algorithmus gibt, der für beliebige solche Sätze entscheidet, ob sie über die reellen Zahlen wahr oder falsch sind.

Ziel dieses Themas ist es, einen theoretischen Entscheidungsalgorithmus für die reellen Zahlen zu schreiben und per Hand auf einige Sätze anzuwenden. Der Entscheidungsalgorithmus unterscheidet sich vom Thema *Tarskis Entscheidungsalgorithmus*

für elementare Algebra und Geometrie insoweit, dass er allgemeiner und damit auf andere mathematische Strukturen anwendbar ist, da er darauf basiert, Zeichenketten zu manipulieren und nicht den mathematischen Inhalt eines Satzes zu analysieren.

Falls die Zeit ausreicht, kann der Algorithmus zu Teilen (z.B. in Prolog) implementiert werden.

3 Tarskis Entscheidungsalgorithmus für elementare Algebra und Geometrie

Wie schon in der Beschreibung vom Thema *Entscheidungsalgorithmus für reelle Zahlen* erklärt, ist eine typische Fragestellung der mathematischen Logik, ob eine bestimmte mathematische Struktur entscheidbar ist, also es einen Algorithmus gibt, der für FOL-Sätze ausgibt, ob sie in der Struktur wahr oder falsch sind. Für eine der wichtigsten mathematischen Strukturen, den angeordneten Körper der reellen Zahlen, wurde diese Frage in den 1940er Jahren durch Alfred Tarski positiv beantwortet. In [11] präsentierte Tarski einen Algorithmus, der durch mathematische Analyse eines gegebenen Satzes mittels analytischer Argumente feststellt, ob dieser gilt oder nicht. Eine Besonderheit an Tarskis Algorithmus ist, dass sowohl die logischen als auch die analytischen Argumente auf sehr grundlegender Ebene nachvollzogen werden können.

Ziel dieses Themas ist es, Tarskis Entscheidungsalgorithmus zu verstehen und daraufhin per Hand auf einige Sätze anzuwenden. Dabei werden wir die Beschreibung des Algorithmus in moderner mathematischer Sprache aus [3, Kapitel 3] verwenden. Weiterhin soll untersucht werden, welche Art von mathematischen Aussagen von diesem Algorithmus entschieden werden können, wie die Komplexität des Satzes die Laufzeit beeinflusst und wo die Grenzen des Verfahrens liegen. Letztlich möchten wir den Algorithmus erweitern, indem wir die sogenannte Modellvollständigkeit des angeordneten Körpers der reellen Zahlen nutzen, und den neuen mit dem klassischen Algorithmus vergleichen.

Auch wenn keine Kenntnis in grundlegender Analysis nötig ist, um den Algorithmus anzuwenden, ist diese hilfreich, um seine Funktionsweise zu verstehen.

4 Gödelscher Vollständigkeitssatz und Gödelscher Unvollständigkeitssatz

Der Vollständigkeitssatz und der Unvollständigkeitssatz des österreichischen Mathematikers Kurt Gödel hatten eine revolutionäre Bedeutung für das Gebiet der mathematischen Logik und lösten mitunter in den 1930er Jahren Kontroversen in der Mathematik, Philosophie und frühen Informatik aus. Die grundlegende Idee der beiden Sätze ist relativ simpel: Der Vollständigkeitssatz besagt in etwa „Jeder Satz, der in jedem Modell bestimmter Axiome gilt, ist auch aus den Axiomen herleitbar.“, wohingegen der Unvollständigkeitssatz die Existenz eines Satzes formuliert, der zwar wahr, aber nicht herleitbar ist. Selbstverständlich sind diese beiden Formulierungen noch sehr unpräzise und weit vom eigentlichen Kern der Aussagen entfernt.

In diesem Thema möchten wir den Vollständigkeitssatz untersuchen und einen Beweis erarbeiten. Für den deutlich komplizierteren Unvollständigkeitssatz versuchen wir, die Grundlagen von Gödelisierungen zu verstehen und die Idee der Konstruktion eines selbstreferenziellen Satzes zu beleuchten. Anschließend soll diskutiert werden, welche Auswirkungen die beiden Sätze auf automatische Theorembeweiser und die theoretischen Existenz von Entscheidungsalgorithmen haben.

Auch eine philosophische bzw. historische Betrachtung des Gödelschen Vollständigkeits- und Unvollständigkeitssatzes kann stattfinden. Diese kann auf [5] basieren.

5 NIP, O-Minimalität und neuronale Netze

In der Modelltheorie wird eine angeordnete Struktur, in der jede definierbare Teilmenge des Trägers eine endliche Vereinigung offener Intervalle und Punkte ist, als *o-minimal* bezeichnet. O-Minimalität impliziert viele nützliche geometrische sowie analytische Eigenschaften der gegebenen Struktur. Insbesondere besitzen o-minimale Strukturen eine gewisse kombinatorische Eigenschaft namens *NIP* (*not the independence property*). Überraschenderweise erfolgte die Beschreibung von NIP zeitgleich in einem modelltheoretischen Kontext sowie im Kontext von neuronalen Netzen Anfang der 1970er Jahre. Die Verbindung zwischen diesen beiden Gebieten wurde jedoch erst 20 Jahre später gefunden (siehe [13]).

In diesem Thema soll erarbeitet werden, wie NIP O-Minimalität impliziert und welche Auswirkung die O-Minimalität des reellen Exponentialkörpers mit eingeschränkten analytischen Funktionen auf Neuronalnetzwerktraining hat.

6 Vertiefung automatische FOL Beweiser

In Woche 1 haben wir ein paar grundlegende Ideen zu automatischen FOL Theorembeweisern kennengelernt. In diesem Projekt soll es darum gehen, dieses Thema soweit es geht theoretisch und möglichst auch praktisch zu vertiefen. Ein sinnvoller Einstiegspunkt wäre Harrison [4] Kapitel 3, ab Abschnitt 3.9. Relevant für dieses Thema sind auch effiziente SAT-Löser aus Kapitel 2 (e.g. DPLL, eine Verfeinerung des Davis-Putnam-Algorithmus). Weitere wichtige grundlegende Aspekte sind unification und die grundsätzliche Unterscheidung zwischen top-down und bottom-up Methoden.

7 Automatische FOL Beweiser mit Gleichheit

Gleichheit kann in FOL durch ein zweistelliges Prädikat und die für die Gleichheit gültigen Axiome kodiert werden. Damit können mathematische Aussagen, die Gleichheit beinhalten, prinzipiell durch FOL Beweiser bewiesen werden. Praktisch ist dieser Ansatz jedoch ineffizient, da unnötig viele Rechenschritte auf die Behandlung von "trivialen" Anwendungen der Axiome des Gleichheitszeichens verwendet werden. Ein anderer Ansatz ist es, die Definition von FOL durch die Gleichheit zu

erweitern. Dadurch ergeben sich effizientere Möglichkeiten in automatischen Theorembeweisern mit Gleichheit umzugehen. Ein mögliches (erstes) Ziel kann es sein Harrison [4], Kapitel 4 durchzuarbeiten. Ein wichtiger Meilenstein ist der Algorithmus von Knuth und Bendix. Die aktuell mächtigsten Systeme verwenden *superposition calculus* [1] (siehe auch [10]). Ein solches System ist der "E prover" von S. Schulz [9]. Eine besondere Stärke von E ist ausserdem die Möglichkeit flexible Suchheuristiken basierend auf machine learning zu verwenden [10]. Als Motivation für dieses Projekt sei erwähnt, dass der Einsatz automatischer Beweiser, die effizient mit Gleichheit umgehen können bereits zur Lösung einer offenen mathematischen Vermutung geführt hat (Robbins-Vermutung [7]).

8 Interaktive Theorembeweiser

Die Idee eines interaktiven Theorembeweisers ist, dass Mensch und Maschine zusammen an einem Beweis arbeiten. Ein Beispiel für ein solches System ist der Beweisassistent Isabelle. In diesem Projekt soll es darum gehen, die Grundlagen der Funktionsweise eines interaktiven Theorembeweisers zu verstehen. Eine geeignete Quelle ist Harrison [4], Kapitel 6.

9 Implementierung von Thiele-Logik

In dem Preprint [12] von C. Thiele geht es um die Grundlagen der Mathematik. Dazu wird eine alternative Formelsprache mit einer gewissen Syntax und Semantik und gewissen Schlussregeln definiert. Ziel dieses Projektes ist es zunächst den Artikel zu verstehen und dann die dort eingeführte Logik zu implementieren (zum Beispiel in OCaml). Im nächsten Schritt kann der Versuch unternommen werden, einen einfachen automatischen Theorembeweiser zu implementieren.

10 Automatisches Entdecken und Beweisen hypergeometrischer Identitäten

Eine hypergeometrische Summe ist eine Summe über Terme t_k, sodass t_{k+1}/t_k eine rationale Funktion von k ist. Eine hypergeometrische Identität ist eine Gleichung, die für eine solche Summe eine geschlossene Form angibt. Zum Beispiel:

$$\sum_{k=0}^{n} k\binom{n}{k} = n2^{n-1}$$

$$\sum_{k=0}^{n}(-1)^k\binom{2n}{k}\binom{2k}{k}\binom{4n-2k}{2n-k} = \binom{2n}{n}^2$$

In dem Buch [8] von M. Petkovšek, H. Wilf und D. Zeilberger werden verschiedene Algorithmen eingeführt, um automatisiert hypergeometrische Identitäten zu entdecken und zu beweisen. Diese sollen verstanden und praktisch ausgetestet (je nach Neigung auch implementiert) werden.

11 Deep Network Guided Proof Search

Das Paper [6] entwickelt ein neuronales Netzwerk, welches den Theorembeweiser E [9] durch Heuristik-Selektion unterstützt. Dafür werden mehrere Netzwerke mit Hilfe existierender FOL Beweise von FOL Theoremen der Mizar Mathematical Library trainiert und verglichen. Es wird gezeigt, dass dieser Ansatz sowohl die Anzahl nötiger Schritte in der Beweissuche verkürzt, als auch Beweise für Theoreme findet, für welche bisher kein FOL Beweis existierte. Ein wichtiger Beitrag des Papers besteht darüber hinaus darin, Phasen der Netzwerk-gestützten Heuristik-Auswahl mit der deterministischen Anwendung schneller Heuristiken abzuwechseln. Dieses Paper soll erarbeitet und verstanden werden, sowie unter Umständen der Versuch unternommen werden die vorgestellten Netzwerke zu implementieren.

Literatur

[1] L. Bachmair, H. Ganzinger, *Rewrite-Based Equational Theorem Proving with Selection and Simplification*, Journal of Logic and Computation, 3(4):217–247, 1994.

[2] M. Fitting, *First-Order Logic and Automated Theorem Proving*, 2. Auflage (Springer, 1996).

[3] L. S. Gütlein, 'The Ordered Field of Real Numbers with Exponentiation: Model Completeness, Decidability and Schanuel's Conjecture', Master's Thesis, Universität Konstanz, 2017.

[4] J. Harrison, *Handbook of Practical Logic and Automated Reasoning*, Cambridge University Press, 2009.

[5] D. W. Hoffmann, *Die Gödel'schen Unvollständigkeitssätze*, 2. Auflage (Springer, 2017).

[6] S. Loos, G. Irving, C. Szegedy, C. Kaliszyk, *Deep Network Guided Proof Search*, arXiv:1701.06972, EPiC Series in Computing, vol. 46, pages 85-105, EasyChair, 2017.

[7] W. McCune, *Solution of the Robbins Problem*, J. Automat. Reason. 19, 263–276, 1997.

[8] M. Petkovšek, H. Wilf, D. Zeilberger, *A=B*, 1997.

[9] S. Schulz, *The E Theorem Prover*, https://wwwlehre.dhbw-stuttgart.de/~sschulz/E/E.html

[10] S. Schulz, *Learning Search Control Knowledge for Equational Deduction*, Ph.D. Thesis, TU München, 2000.

[11] A. Tarski, *A decision method for elementary algebra and geometry* (RAND Corporation, Santa Monica, CA, 1948).

[12] C. Thiele, *Definitions in mathematics*, arXiv:1706.08905, 2017.

[13] M. Tressl, 'Introduction to o-minimal structures and an application to neural network learning', Preprint, 2010.