

März 2025

Computeralgebra

Rundbrief

> Ausgabe 76

- ▶ Tagung der Fachgruppe in Leipzig
- ▶ SAT-Solver für Formeln in XNF
- ▶ polymake updated



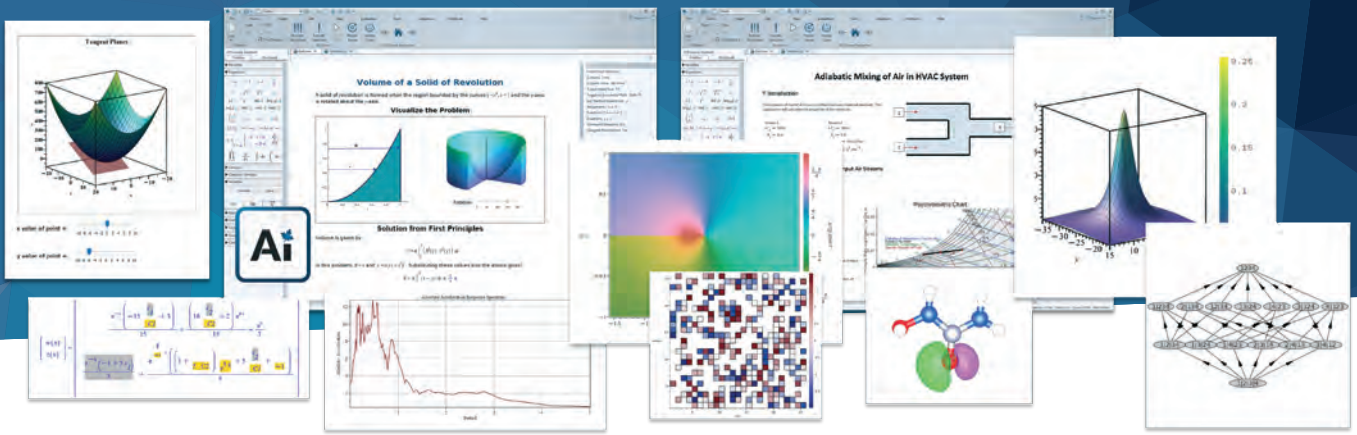
DMV





Mehr leisten mit Maple 2025!

Lösen Sie mehr Probleme noch leichter mit Maple 2025



Neues Maple 2025 jetzt verfügbar!

Mit mehr Mathematik, einer neuen Benutzeroberfläche, verbesserten Tools für Bildung und Vernetzung und vielem mehr ist die leistungsstärkste und umfassendste Umgebung zum Erforschen, Visualisieren und Lösen selbst der schwierigsten mathematischen Probleme besser denn je.

Probieren Sie Maple kostenlos
für 15 Tage ohne Verpflichtungen

www.maplesoft.com/CAR2025

Wenden Sie sich direkt an Ihr
lokales Maplesoft-Team.



Tel.: +49 241 980977-30
Email: vertrieb@maplesoft.com



Inhaltsverzeichnis

Inhalt	3
Impressum	4
Mitteilungen der Sprecher	5
Tagungen der Fachgruppe	6
Themen und Anwendungen	8
<i>SAT-Solver für Formeln in XNF</i> (B. Andraschko, J. Danner, M. Kreuzer)	8
Neues über Systeme	12
<i>polymake updated</i> (M. Joswig)	12
Berichte über Arbeitsgruppen	16
<i>DFG Priority Program SPP 2458: Combinatorial Synergies</i> (T. Kahle)	16
Publikationen über Computeralgebra	20
Besprechungen zu Büchern der Computeralgebra	21
<i>V. Diekert, M. Kreuzer: Finitely Presented Groups</i> (B. Eick)	21
Promotionen in der Computeralgebra	23
Berichte von Konferenzen	24
Hinweise auf Konferenzen	25
Fachgruppenleitung Computeralgebra 2023–2026	27

Impressum

Der Computeralgebra-Rundbrief wird herausgegeben von der Fachgruppe Computeralgebra der GI in Kooperation mit der DMV und der GAMM (verantwortlicher Redakteur: Dr. Fabian Reimers car@mathematik.de)

Der Computeralgebra-Rundbrief erscheint halbjährlich, Redaktionsschluss 15.02. und 15.09. ISSN 0933-5994. Mitglieder der Fachgruppe Computeralgebra erhalten je ein Exemplar dieses Rundbriefs im Rahmen ihrer Mitgliedschaft. Fachgruppe Computeralgebra im Internet: <https://www.fachgruppe-computeralgebra.de>.

Konferenzankündigungen, Mitteilungen, einzurichtende Links, Manuskripte und Anzeigenwünsche bitte an den verantwortlichen Redakteur.

GI (Gesellschaft für Informatik e.V.)
Wissenschaftszentrum
Ahrstr. 45
53175 Bonn
Telefon 0228-302-145
Telefax 0228-302-167
bonn@gi.de
<https://gi.de>

DMV (Deutsche Mathematiker-Vereinigung e.V.)
Mohrenstraße 39
10117 Berlin
Telefon 030-20377-306
Telefax 030-20377-307
dmv@wias-berlin.de
<https://www.mathematik.de>

GAMM (Gesellschaft für Angewandte Mathematik und Mechanik e.V.)
Technische Universität Dresden
Institut für Statik und Dynamik der Tragwerke
01062 Dresden
Telefon 0351-463-33448
Telefax 0351-463-37086
GAMM@mailbox.tu-dresden.de
<https://www.gamm-ev.de>



Mitteilungen der Sprecher

Liebe Mitglieder der Fachgruppe Computeralgebra,

so schnell vergeht eine Wahlperiode. Ende des Jahres steht die Neuwahl der Fachgruppenleitung für die nächste Periode von 3 Jahren an. Es wäre schön, wenn wir diesmal wieder das eine oder andere neue Gesicht unter den Kandidierenden begrüßen dürften. Interessierte dürfen sich gerne an uns beide oder jedes andere Fachgruppenleitungsmitglied wenden, idealerweise noch vor dem Sommer. Besonders am Herzen liegt es uns, wieder eine Kandidatur aus dem Interaktionsbereich Computeralgebra in der Schule zu finden, nachdem das Resort in dieser Wahlperiode unbesetzt war. Die Liste der Kandidierenden wird in der Fachgruppenleitungssitzung im Herbst aufgestellt. Im Herbstrundbrief wird sich dann die traditionelle Vorstellung all derer finden, die für einen Sitz in der Fachgruppenleitung kandidieren.

Ehe es aber an die Neuwahlen geht, steht uns noch ein sehr schönes Highlight bevor: Die Fachgruppenpentagung 2025 in Leipzig. Mit 5 Hauptvortragenden, einem Sondervortrag zu Forschungsdatenmanagement und natürlich den Nachwuchsvorträgen, für die die Einreichungsfrist derzeit noch läuft, werden es wieder intensive zweieinhalb Tage mit viel Computeralgebra und viel Gelegenheit zum informellen Austausch. Wir freuen uns darauf, Sie dort begrüßen zu dürfen. Sollten Sie Doktoranden oder Post-Docs in Computeralgebra in Ihrer Gruppe haben, die bereits schöne Ergebnisse vorzuweisen haben, weisen Sie sie gerne auf diese Möglichkeit zur Einreichung eines Vortrags und auf unseren Nachwuchspreis hin.

Ein anderes Thema beschäftigte uns schon im letzten Rundbrief: die Zukunft von ISSAC und SIGSAM. Hier sind in der Zwischenzeit noch keine Entscheidungen gefallen, aber zumindest gab es seither einige intensive und konstruktive Gespräche zwischen ACM und SIGSAM sowie zwischen ACM und ISSAC. In der heutigen Zeit mit ihren sich schnell wandelnden Randbedingungen ist so ein intensives Ringen um die Sache schon ein positives Signal. Wir drücken weiter die Daumen, dass bald befriedigende Lösungen verkündet werden können.

Doch nun möchten wir Sie nicht länger von der Computeralgebra abhalten. Thematische Schwerpunkte dieses Rundbriefs sind SAT-Solver, Neuigkeiten zu dem System POLYMAKE sowie eine Vorstellung des DFG Schwerpunktprogramms SPP 2458 Combinatorial Synergies.

Wir wünschen Ihnen eine kurzweilige und anregende Lektüre.

Anne Frühbis-Krüger

Michael Cuntz

Tagungen der Fachgruppe

Tagung der Fachgruppe Computeralgebra

Leipzig

02.06.–04.06.2025

<https://fachgruppe-computeralgebra.de/leipzig-2025>

Für die 11. Computeralgebra-Tagung der Fachgruppe ist diesmal das MPI-MIS in Leipzig unser Gastgeber. Nach den Unregelmäßigkeiten der letzten Jahre aufgrund der Pandemie sind wir damit wieder im bewährten 2-jährigen Rhythmus angekommen. Die Tagung wird am Montagmittag beginnen und am Mittwochmittag enden.

Ganz in der Tradition der früheren Tagungen werden auch diesmal mehrere Hauptvortragende Übersichtsvorträge über wichtige Themen aus Computeralgebra und über Computeralgebrasysteme halten, während in den anderen Vorträgen dem wissenschaftliche Nachwuchs Gelegenheit gegeben wird, seine Ergebnisse vorzustellen. Deutsch und Englisch sind dabei wie immer gleichberechtigte Konferenzsprachen. Für den besten Nachwuchsvortrag vergibt die Fachgruppe auch dieses Mal wieder einen mit 500,- Euro dotierten Preis.

Hierzu sind Nachwuchswissenschaftler und -wissenschaftlerinnen (**Promovendi, Post-Docs**) aufgefordert, sich bis zum **15.04.2025** mit einem **Vortrag anzumelden**.

Als Hauptvortragende konnten wir folgende Wissenschaftlerinnen und Wissenschaftler gewinnen:

- **Clemens Hofstadler** (Linz)
- **Thomas Kahle** (Magdeburg)
- **Marta Panizzut** (Tromsø)
- **Lorenz Panny** (München)
- **Milena Wrobel** (Oldenburg)

In einem weiteren eingeladenen Vortrag wird uns Christiane Görgen einen Vortrag zu dem Thema 'Forschungsdatenmanagement' halten – ein aktuelles Thema für alle aktiven Forscher und Forscherinnen – und insbesondere für jüngere Teilnehmende sicher eine gute Gelegenheit sich zu informieren und ihre Fragen im Plenum oder individuell zu stellen.

Die Eckdaten zur Konferenz im Überblick:

Website: <https://fachgruppe-computeralgebra.de/leipzig-2025>

Termin und Ort: Die Tagung findet in der Zeit vom 02. – 04. Juni 2025 am MPI-MIS in Leipzig statt. Sie wird am 02. Juni 2025 um circa 13:00 Uhr eröffnet (Anreisetag) und endet am 04. Juni 2025 um circa 12:30 Uhr (Abreisetag).

Anmeldung: Die Anmeldung eines Vortrags ist bis 15. April 2025 möglich. Die Anmeldung ohne Vortrag ist bis 10. Mai 2025 möglich. Details zur Anmeldung finden Sie auf der Website der Tagung.

Nachwuchspreis: Die Fachgruppe Computeralgebra vergibt für den besten Vortrag von Promovendi/PostDocs wieder einen mit 500€ dotierten Nachwuchspreis. Verbunden mit dem Geldpreis ist die Einladung, auf der nächsten Tagung der Fachgruppe einen Hauptvortrag zu halten.



LEIBNIZ
UNIVERSITÄT
HANNOVER

THE 11TH CONFERENCE OF THE FACHGRUPPE COMPUTERALGEBRA

SPEAKERS

Clemens Hofstadler

Johannes Kepler University Linz

Thomas Kahle

Otto von Guericke University Magdeburg

Marta Panizzut

The Arctic University of Norway

Lorenz Panny

Technical University of Munich

Milena Wrobel

Carl von Ossietzky University Oldenburg

THEME TALK

Christiane Görgen

Leipzig University

SCIENTIFIC ORGANIZERS

Anne Frühbis-Krüger

Carl von Ossietzky
University Oldenburg

Alheydis Geiger

Max Planck Institute
for Mathematics in the Sciences

Max Horn

RPTU University
Kaiserslautern-Landau

YOUNG TALENT AWARD

On the final day, a prize of €500 will be awarded for the best contributed talk. The cash prize is accompanied by an invitation to give a main talk at the next conference.

If you would like to give a talk at the conference, please register by April 15, 2025.

JUNE 2 – 4, 2025

**MAX PLANCK INSTITUTE
FOR MATHEMATICS IN THE SCIENCES**

Leibniz lecture hall (E1 05)

Inselstraße 22, 04103 Leipzig

Administrative contact:

Saskia Gutzschebauch & Mirke Olschewski

Mail: bs_sec@mis.mpg.de



FURTHER INFORMATION &
ONLINE REGISTRATION VIA

www.mis.mpg.de/computeralgebra

SAT-Solver für Formeln in XNF

Bernhard Andraschko, Universität Passau

Julian Danner, Universität Passau

Martin Kreuzer, Universität Passau

bernhard.andraschko@uni-passau.de

julian.danner@uni-passau.de

martin.kreuzer@uni-passau.de



SAT-Solver sind weit verbreitete Programme zur Lösung des Booleschen Erfüllbarkeitsproblems (SAT). Insbesondere in der Kryptoanalyse können sie hilfreiche Werkzeuge sein. Viele Probleme aus der Kryptoanalyse stellen jedoch eine Herausforderung für herkömmliche SAT-Solver dar, da die entsprechenden logischen Darstellungen oft sehr viele *exklusive Disjunktionen (XORs)* enthalten. Die meisten SAT-Solver nehmen als Eingabe Formeln in konjunktiver Normalform (KNF), welche XORs nur sehr ineffizient darstellen können. Bisherige Ansätze zur Verarbeitung von Formeln mit XORs nehmen als Eingabe eine Formel in KNF in Kombination mit so genannten *XOR-Bedingungen*, also exklusiven Disjunktionen von Literalen [5, 8, 10]. Ein fortschrittlicherer Ansatz ist das Beweissystem *s-Res*, welches logische und algebraische Schlussfolgerungen kombiniert, indem es sowohl die klassische Resolution als auch Buchbergers S-Polynome verallgemeinert [7]. Auf *s-Res* basierende SAT-Solver existieren jedoch bisher noch nicht.

Das Ziel unserer Arbeit [1] ist die Untersuchung neuer Methoden zur Lösung XOR-reicher Boolescher Polynome. Im ersten Teil führen wir dazu die *XOR-OR-AND-Normalform (XNF)* aussagenlogischer Formeln ein, die die KNF verallgemeinert, indem Literale durch XORs von Literalen (sogenannten *Linalen*) ersetzt werden. Sie wird auch für den Input von *s-Res* verwendet. Wir beschreiben Methoden zur effizienten Konvertierung von Systemen Boolescher Polynome zu 2-XNF, also Formeln in XNF, bei denen jede Klausel maximal zwei Linalen enthält.

Im zweiten Teil geben wir graphbasierte Pre- und In-Processing-Techniken an und erklären *2-Xornado*, unseren ersten SAT-Solver für Formeln in 2-XNF. Abschließend vergleichen wir *2-Xornado* anhand von Beispielen aus der Kryptoanalyse mit etablierten SAT-Solvern mit XOR-Unterstützung und algebraischen Solvern, welche Boolesche Polynome als Eingabe nehmen.

Die XOR-OR-AND-Normalform

Zuerst wiederholen wir die Definition der KNF.

Definition 1 Sei F ein aussagenlogische Formel.

- Eine **Literal** ist eine logische Variable X oder die Negation $\neg X$ einer logischen Variablen.
- Ist $F = L_1 \vee \dots \vee L_s$ eine Disjunktion von Literalen L_1, \dots, L_s , so heißt F eine **Klausel**.
- Ist $F = C_1 \wedge \dots \wedge C_r$ eine Konjunktion von Klauseln C_1, \dots, C_r , so ist F in **konjunktiver Normalform (KNF)**.
- Sei $k \in \mathbb{N}$. Ist F in KNF und enthält jede Klausel von F höchstens k Literale, so sagen wir, dass F in **k -KNF** ist.

Eine Besonderheit der KNF ist, dass aussagenlogische Probleme einfach zu einer äquivalenten KNF umgewandelt werden können. Beispielsweise ist die Formel $(X_1 \wedge X_2) \rightarrow Y$ für aussagenlogische Variablen X_1, X_2 und Y äquivalent zur Klausel $(\neg X_1 \vee \neg X_2 \vee Y)$ in dem Sinne, dass beide Formeln die gleichen Wahrheitswerte liefern.

Als nächsten Schritt führen wir eine neue Normalform ein, welche die KNF verallgemeinert, indem sie XORs von Literalen anstelle von Literalen zulässt.

Definition 2 Sei F eine aussagenlogische Formel.

- Ein **Linal** ist eine exklusive Disjunktion von Literalen.
- Eine Disjunktion von Linalen nennen wir eine **XNF-Klausel**.
- Die Formel F ist in **XOR-OR-AND-Normalform (XNF)**, wenn F eine Konjunktion von XNF-Klauseln ist.
- Sei $k \in \mathbb{N}$. Ist F in XNF und enthält jede XNF-Klausel von F höchstens k Linalen, so sagen wir, dass F in **k -XNF** ist.

Ein klassisches Resultat der Aussagenlogik ist, dass jede Formel in KNF erfüllbarkeitsäquivalent zu einer Formel in 3-KNF ist. Beispielsweise kann die Formel $(X_1 \vee X_2 \vee X_3 \vee X_4)$ für Variablen X_1, X_2, X_3, X_4 durch Einführung der Zusatzvariablen Y in die Formel

$$(X_1 \vee X_2 \vee Y) \wedge (\neg Y \vee X_3 \vee X_4)$$

umgewandelt werden. Im allgemeineren Fall der XNF gilt sogar ein stärkeres Resultat. Dieses folgt direkt daraus, dass für alle logischen Variablen Y, X_1, X_2 gilt

$$\begin{aligned} Y &\leftrightarrow (X_1 \vee X_2) \\ &\equiv (Y \vee \neg X_2) \wedge (\neg(Y \oplus X_1) \vee X_2). \end{aligned} \quad (1)$$

Mit dieser Äquivalenz ist es möglich, Disjunktionen zweier Literale durch Zusatzvariablen zu ersetzen und die Information der Substitution mithilfe von (1) zu kodieren.

Satz 3 Sei F eine aussagenlogische Formel. Dann kann man in polynomieller Zeit eine erfüllbarkeitsäquivalente Formel G in 2-XNF berechnen.

Zur Vereinfachung der Notation wechseln wir nun zu einer algebraischen Formulierung von Formeln in XNF. Im Folgenden bezeichnen wir mit

$$\mathbb{B}_n = \mathbb{F}_2[x_1, \dots, x_n] / \langle x_1^2 - x_1, \dots, x_n^2 - x_n \rangle$$

den Ring der Booleschen Polynome in den Unbestimmten x_1, \dots, x_n . Weiterhin schreiben wir $\mathbb{L}_n = \langle 1, x_1, \dots, x_n \rangle_{\mathbb{F}_2}$ für den \mathbb{F}_2 -Vektorraum der Booleschen Polynome vom Grad ≤ 1 . Für jede logische Formel F in n Variablen existiert nun ein eindeutiges Ideal I_F in \mathbb{B}_n , sodass die erfüllenden Belegungen für F durch die Identifikation $\text{true} \equiv 1$ und $\text{false} \equiv 0$ genau den Nullstellen von I_F entsprechen. Dieses Ideal I_F heißt die *algebraische Repräsentation* von F .

Aus algebraischer Sicht entspricht dann ein Literal einem linearen Polynom in \mathbb{L}_n , eine XNF-Klausel einem Produkt von linearen Polynomen aus \mathbb{L}_n , und eine Formel in XNF einem Ideal erzeugt von Produkten von linearen Polynomen. Diese Perspektive zeigt, dass Formeln in XNF auf natürliche Weise im Beweissystem s -Res vorkommen. Eine algebraische Formulierung von Satz 3 ist nun, dass jedes System Boolescher Polynome transformiert werden kann in eines der Form

$$\{f_1 g_1, \dots, f_k g_k, \ell_1, \dots, \ell_s\} \subseteq \mathbb{B}_n$$

wobei $f_i, g_i, \ell_j \in \mathbb{L}_n$ sind. Eine algebraische Formulierung von (1) liefert jetzt einen Algorithmus, der ein beliebiges System Boolescher Polynome in polynomieller Zeit in 2-XNF umwandelt.

Beispiel 4 `Ascon-128` ist ein 128-Bit-Kryptosystem, das auf einer 5-Bit S-Box basiert und 2023 von NIST als Standard der *Lightweight Cryptography* ausgewählt wurde [6].

Mit unseren Algorithmen konnten wir eine Darstellung in 2-XNF des kompletten Kryptosystems

`Ascon-128` berechnen, die aus lediglich 6080 Variablen und 17 664 XNF-Klauseln besteht. Herkömmliche Methoden zur Konvertierung zu KNF wie `PolyBoRi` [3], die Methoden aus [8] oder `Bosphorus` [5] benötigen zur Darstellung mindestens 12 224 Variablen und 137 739 Klauseln.

Die Kodierung XOR-reicher Probleme in 2-XNF ermöglicht also weitaus kompaktere Darstellungen als die üblichen Verfahren zur Umwandlung in KNF-Klauseln. Im nächsten Abschnitt zeigen wir, wie dies gewinnbringend genutzt werden kann.

Graphbasiertes 2-XNF SAT-Solving

Es ist bekannt, dass zu einer erfüllbaren Formel in 2-KNF in linearer Zeit eine erfüllende Belegung gefunden werden kann [2]. Obwohl die Methoden zum Finden einer solchen Lösung nicht direkt auf Formeln in 2-XNF übertragen werden können, lassen sich einige der zugrunde liegenden Ideen nutzen.

Definition 5 Sei F eine Formel in 2-XNF. Ein Tupel $G = (L, V, E)$, wobei $L, V \subseteq \mathbb{L}_n$ und $E \subseteq V^2$ sind, heißt **Implikationsgraphenstruktur (IGS)** für F , wenn (V, E) ein nichtreflexiver gerichteter Graph ist und die folgenden Eigenschaften erfüllt:

- (1) $I_F = \langle L \rangle + \langle fg \mid (f + 1, g) \in E \rangle$ und
- (2) $(f + 1, g) \in E$ impliziert $(g + 1, f) \in E$.

Eine IGS für eine Formel F in 2-XNF kann direkt aus den XNF-Klauseln abgeleitet werden. Weiterhin entspricht jede Kante $(f, g) \in E$ der Implikation $f \in I_F \Rightarrow g \in I_F$. Die Menge L enthält alle bekannten linearen Informationen über I_F . Um nun eine 2-XNF-Instanz zu lösen, schlagen wir folgenden DPLL-Ansatz vor: Starte mit einer trivialen IGS und verbessere diese schrittweise durch *Propagation*, *In-Processing* und *Entscheidungen* so lange, bis wir bei einem IGS mit einem leeren Graph ankommen, also bis I_F von den linearen Polynomen in L erzeugt wird. Sofern die Entscheidungen korrekt waren, kann daraus eine erfüllende Belegung für F bestimmt werden. Wenn die Polynome in L keine gemeinsame Nullstelle haben, springen wir zur letzten Entscheidung zurück.

Propagation. Lineare Informationen werden folgendermaßen propagiert. Für alle Knoten $f \in V$ berechnen wir den *normalen Rest* $\text{NR}_\sigma(f, L)$ bezüglich der Polynome in L und einer Termordnung σ . Reduziert sich eine Kante $(f, g) \in E$ dadurch zu $(0, g')$ mit $g' \neq 0$, so erhalten wir $f \in I_F$ und können somit g' zu L hinzufügen. Reduziert sich (f, g) zu $(f', 1)$ mit $f' \neq 1$, so erhalten wir $g + 1 \in I_F$ und können $f' + 1$ zu L hinzufügen. Dies wird durch Algorithmus 1 umgesetzt.

In-Processing. Basierend auf den Techniken für 2-KNF-Solver untersuchen wir die *starken Zusammenhangskomponenten (SZKs)* des Implikationsgraphen, um neue lineare Polynome zu finden. Für verschiedene $f_i, f_j \in V$, die zur gleichen SZK gehören, gilt dann $f_i + f_j \in I_F$. Ein Resultat aus [2] zeigt, dass SZKs

mit linearem Zeit- und Platzaufwand berechnet werden können, was dies zu einer effizienten In-Processing-Methode macht.

Algorithm 1: GGCP (Graph Gaussian Constraint Propagation)

Input : Eine IGS (L, V, E) für eine Formel F in 2-XNF, eine Termordnung σ .

Output: Eine ergänzte IGS (L', V', E') für F .

- 1 Setze $(L', V', E') = (L, \emptyset, \emptyset)$.
 - 2 **for** $(f, g) \in E$ **do**
 - 3 Let $f' = \text{NR}_\sigma(f, L)$ und $g' = \text{NR}_\sigma(g, L)$.
 - 4 **if** $f' = 0$ **and** $g' \neq 0$ **then** füge g' zu L' hinzu.
 - 5 **if** $g' = 1$ **and** $f' \neq 1$ **then** füge $f' + 1$ zu L' hinzu.
 - 6 **if** $f' \notin \mathbb{F}_2$ **and** $g' \notin \mathbb{F}_2$ **and** $f' \neq g'$ **then**
 - 7 füge (f', g') zu E' hinzu
 - 8 füge f' und g' zu V' hinzu.
 - 9 **if** $L \neq L'$ **then**
 - 10 Setze $(L, V, E) = (L', V', E')$ und gehe zu Zeile 1.
 - 11 **else return** (L', V', E') .
-

Entscheidungen. Im Entscheidungsschritt des Algorithmus müssen wir eine Entscheidung treffen, also ein Tupel (L_0, L_1) von Teilmengen von \mathbb{L}_n finden, sodass entweder die Annahme von L_0 oder die von L_1 zu einer Lösung von F führt – falls überhaupt eine existiert. Während KNF-basierte SAT-Solver lediglich Entscheidungen der Form $(\{x_i\}, \{x_i + 1\})$ oder $(\{x_i + 1\}, \{x_i\})$ erlauben, können wir auch mehrere lineare Polynome auf einmal raten. Zum Beispiel bildet jeder Pfad $f_1 \rightarrow \dots \rightarrow f_r$ in (V, E) eine Entscheidung $(\{f_1 + 1, f_r\}, \{f_1 + f_i \mid i \in \{2, \dots, r\}\})$.

Insgesamt verwenden wir den folgenden Algorithmus zur Lösung von Formeln in 2-XNF.

Algorithm 2: G_2XNF_DPLL (Graph 2-XNF DPLL-Solver)

Input : Eine IGS (L, V, E) für eine Formel F in 2-XNF, eine Termordnung σ .

Output: UNSAT oder eine Nullstelle $a \in \mathcal{Z}(I_F)$.

- 1 Setze $(L, V, E) = \text{GGCP}((L, V, E), \sigma)$.
 - 2 Nutze die SZKs in (L, V, E) , um lineare Polynome L_{FL} in I_F zu berechnen und füge diese zu L hinzu.
 - 3 // In-Processing
 - 4 **if** $L_{\text{FL}} \neq \emptyset$ **then** gehe zu Zeile 1.
 - 5 **if** $1 \in \langle L \rangle_{\mathbb{F}_2}$ **then return** UNSAT
 - 6 **if** $E = \emptyset$ **then return** $a \in \mathcal{Z}(L) \subseteq \mathbb{F}_2^n$.
 - 7 Berechne eine Entscheidung (L_0, L_1) für (L, V, E) .
 - 8 // Entscheidung
 - 9 **if** $\text{G_2XNF_DPLL}_\sigma(L \cup L_0, V, E)$ **returns** $a \in \mathbb{F}_2^n$ **then**
 - 10 **return** a .
 - 11 **else return** $\text{G_2XNF_DPLL}_\sigma(L \cup L_1, V, E)$.
-

Bevor der Hauptalgorithmus ausgeführt wird, führen wir noch ein kurzes **Pre-Processing** aus, welches auf folgendem Resultat basiert: Für ein Boolesches Polynom $f \in V$ definieren wir den *Raum der Nachfolger*

$$\Delta_f = \langle f \rangle_{\mathbb{F}_2} + \langle g \in V \mid \exists \text{ ein Pfad } f \rightarrow g \text{ in } (V, E) \rangle_{\mathbb{F}_2}.$$

Satz 6 Sei (L, V, E) eine IGS für eine Formel F in 2-XNF. Dann ist für alle $f \in V$ auch $\Delta_f \cap \Delta_{f+1} \subseteq I_F$.

Durch die Berechnung der Schnittmenge dieser beiden Unterräume von \mathbb{L}_n für alle $f \in V$ können wir neue lineare Polynome in I_F finden.

Experimente

Sowohl der 2-XNF-Solver 2-Xornado als auch der Brute-Force-Solver `xfn_bf` wurden von uns in C++ implementiert. Bei 2-Xornado betrachteten wir dabei Versionen mit drei verschiedenen Entscheidungsstrategien `MaxPath`, `MaxReach` und `MaxBottleneck`. Zum Vergleich dienten KNF-basierte SAT-Solver mit Unterstützung für XOR-Bedingungen: der CDCL-Solver `CryptoMiniSat` [10] sowie der SLS-Solver `xfnSAT` [9]. Weiterhin bezogen wir die algebraischen Solver `PolyBoRi` [3] und `Bosphorus` [5] sowie den KNF-Solver `SBVA-CaDiCaL` [4] mit ein.

Für einen ersten Leistungstest wurde eine Benchmark-Suite von zufälligen 2-XNF-Instanzen erzeugt. Hierbei zeigte sich, dass 2-Xornado die herkömmlichen SAT-Solver auf diesen Instanzen weit übertrifft (siehe Abb. 1). Dies kann größtenteils auf die um einen Faktor von 60-80 kleinere benötigte Anzahl an Entscheidungen zurückgeführt werden.

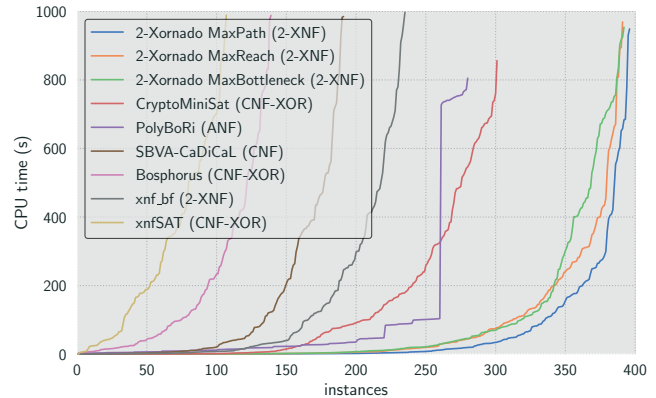


Abbildung 1: Benchmark-Suite bestehend aus 400 zufälligen, lösbaren 2-XNF-Instanzen in n Variablen und $3n$ Klauseln für $n \in \{21, \dots, 40\}$.

Ein zweiter Benchmark lief auf Instanzen, die aus algebraischen Angriffen auf rundenreduzierte Versionen von `Ascon-128` (siehe Beispiel 4) entstehen. Auch hier übertrifft 2-Xornado die anderen Programme und stellt sich als bester Solver für diese Instanzen heraus. Besonders hervorzuheben ist, dass einige der Instanzen bereits in der Pre-Processing-Phase gelöst wurden. In allen anderen Fällen benötigte 2-Xornado wieder um einen Faktor von 60-80 weniger Entscheidungen als z.B. `CryptoMiniSat`.

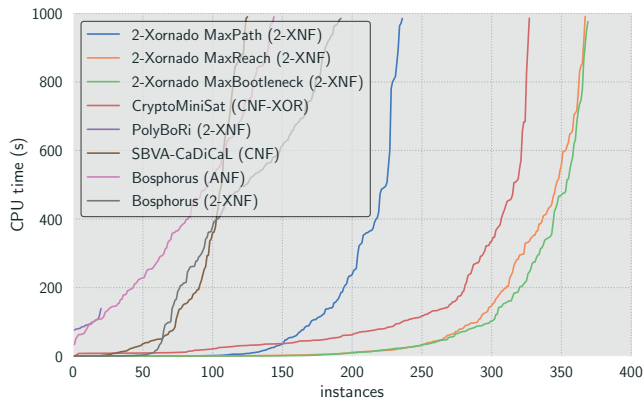


Abbildung 2: Benchmark-Suite bestehend aus 400 lösbaren Instanzen im Zusammenhang mit algebraischen Angriffen auf rundenreduziertes *Ascon-128*.

Zusammenfassung. Mit der XNF haben wir eine Verallgemeinerung der KNF eingeführt, welche kompakte Darstellungen XOR-reicher Probleme ermöglicht. Darauf aufbauend wurde ein graphbasierter 2-XNF-Solver mit fortschrittlichen Pre-Processing Methoden entwickelt, welcher sowohl auf zufällig generierten Instanzen als auch auf Instanzen aus der Kryptoanalyse moderne SAT-Solver übertrifft.

Literatur

- [1] B. Andraschko, J. Danner, and M. Kreuzer, SAT Solving Using XOR-OR-AND Normal Forms, *Math. Comput. Sci.* **18** (2024), article 20.
- [2] B. Aspvall, M. Plass, and R. Tarjan, A linear-time algorithm for testing the truth of certain quantified boolean formulas, *Inf. Process. Lett.* **8** (1979), 121–123.
- [3] M. Brickenstein and A. Dreyer, PolyBoRi: A framework for Gröbner-basis computations with Boolean polynomials, *J. Symb. Comput.* **44** (2009), 1326–1345.
- [4] A. Biere et al., CaDiCaL 2.0, in: *Proc. Computer Aided Verification (CAV 2024)*, Montreal 2024, LNCS 14681, Springer Nature, Cham 2024, pp. 133-152.
- [5] D. Choo, M. Soos, K. M. A. Chai, and K. S. Meel, Bosphorus: Bridging ANF and CNF solvers, in: *Proc. Design, Automation, and Test in Europe (DATE)*, Florence 2019, IEEE Xplore, pp. 468-473.
- [6] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, *Ascon v1.2*: Technical report, National Institute of Standards and Technology, 2019.
- [7] J. Horáček and M. Kreuzer, Refutation of products of linear polynomials, in: *Proc. Third Int. Workshop on Satisfiability Checking and Symbolic Computation (SC²)*, Oxford 2018, available at <http://ceur-ws.org/Vol-2189/>.
- [8] P. Jovanovic and M. Kreuzer, Algebraic attacks using SAT-solvers, *Groups Complexity Cryptology* **2** (2010), 247–259.
- [9] W. Nawrocki, Z. Liu, A. Fröhlich, M. J. H. Heule, and A. Biere, XOR local search for Boolean Brent equations, in: *Theory and Applications of Satisfiability Testing (SAT 2021)*, LNCS 12831, Springer Nature, Cham 2021, pp. 417–435.
- [10] M. Soos, K. Nohl, and C. Castelluccia, Extending SAT solvers to cryptographic problems, in: *Theory and Applications of Satisfiability Testing (SAT 2009)*, LNCS 5584, Springer-Verlag, Berlin 2009, pp. 244–257.

polymake updated

Michael Joswig, TU Berlin & MPI-MiS, Leipzig

joswig@math.tu-berlin.de



Introduction

polymake is free open source software for computations in polyhedral geometry, with applications to algebraic and tropical geometry, as well as combinatorial optimization and combinatorial topology. Since its beginnings in 1997 [14] polymake became a standard reference; it includes interfaces to numerous other software libraries including, e.g., cddlib [13], lrslib [2], PPL [6] and Normaliz [9]; conversely, polymake is an optional package of SageMath [20].

The purpose of this note is to describe features of the most recent polymake version 4.13. For several key algorithmic problems polymake has more than one algorithm or implementation. We give an example to show why this is useful. Via the Julia interface layer Polymake.jl [18] polymake is one of four cornerstones of the new computer algebra system OSCAR [10]. In two showcases we highlight some capabilities of polymake and OSCAR combined, in particular with respect to nonrational polytopes. A brief outlook sketches polymake's future.

Convex hull computations

The bread-and-butter algorithmic problem in polyhedral geometry is the computation of convex hulls. The input is a finite set of points in \mathbb{R}^d , and the output is a list of (nonredundant) affine linear inequalities describing the convex hull of the input. There is also the dual problem, to convert inequalities into points. However, via cone polarity the primal and the dual problem are equivalent. So in terms of implementations it suffices to solve one of the two problems. It is known that the time required by the known algorithms and their implementations to solve an instance of the convex problem may vary considerably [3, 1].

In our first experiment we construct a polytope as the convex hull of 300 points chosen uniformly at random on the unit sphere in \mathbb{R}^6 . In fact, the function produces rational approximations of such points. We fix the

seed value of the random number generator in order to be able to reproduce the same data. It turns out that our polytope has precisely 30401 facets; printing that number triggers the convex hull computation.

```
polytope > use Benchmark qw(:all);
polytope > $P = rand_sphere(6, 300,
  ↪ seed=>11);
polytope > $t0=Benchmark->new; print
  ↪ $P->N_FACETS; $t1=Benchmark->new;
polymake: used package ppl
The Parma Polyhedra Library
  ↪ ([[wiki:external_software#PPL]]): A C++
  ↪ library for convex polyhedra
and other numerical abstractions.
http://www.cs.unipr.it/ppl/
```

30401

```
polytope > print timestr(timediff($t1,$t0));
35 wallclock secs (35.02 usr + 0.04 sys =
  ↪ 35.06 CPU)
```

The user is notified that PPL [6] was used, which is polymake's current default. The algorithm is standard Fourier–Motzkin elimination. All timings are taken on a 16-core AMD Ryzen 9 7950X with Manjaro Linux, with kernel 6.12.11-1.

Instead of working with the default we can request a specific code like, e.g., lrslib [2] for the convex hull computations. The algorithm is reverse search [4]. Of course, we get the same number of facets as the result.

```
polytope > prefer_now "lrs";
  ↪ $t0=Benchmark->new; print $P->N_FACETS;
  ↪ $t1=Benchmark->new;
30401
```

```
polytope > print timestr(timediff($t1,$t0));
19 wallclock secs (18.56 usr + 0.00 sys =
  ↪ 18.56 CPU)
```

For this specific input lrslib is considerably faster than PPL. It is also more memory efficient. We can do the same computation a third time but with Normaliz [9] instead of lrslib [2]. Normaliz employs a method called *pyramid decomposition* [8].

```

polytope > $P = rand_sphere(6, 300,
↳ seed=>11);

polytope > prefer_now "libnormaliz";
↳ $t0=Benchmark->new; print $P->N_FACETS;
↳ $t1=Benchmark->new;
30401

polytope > print timestr(timediff($t1,$t0));
1 wallclock secs ( 7.59 usr + 0.40 sys =
↳ 7.99 CPU)

```

Among the codes presented here `Normaliz` is the only one which runs in parallel (via OpenMP). By default, `Normaliz` limits the number of threads to eight. The parallelization explains the discrepancy between the wallclock and CPU times.

Several comments are in order. First, picking the right kind of implementation for a specific input may make a difference in terms of running time. There are even more drastic examples known than the one shown here. Second, we see that `PPL` is inferior to `lrslib` and `Normaliz` for this specific input. Other input may show an entirely different behavior, and this actually happens in practice. Third, parallelization in polyhedral geometry remains a challenge. The single-threaded version of `Normaliz` takes about two seconds on this input. So we observe a speed-up factor of only two with eight cores; yet also the benefit from parallelization varies with the input. Further, there is also `mplrs` [5], which is a parallel implementation of reverse search. For more details on convex hull codes and their differences the reader is referred to [3] and [1].

The language used in the command line interface is `polymake`'s own dialect of `Perl`. The relevant algorithms are implemented in `C++` or `C`.

Fields of rational functions

Polyhedral geometry makes sense over any ordered field. For instance, we can consider the ring $R = \mathbb{Q}[t]$ of univariate polynomials with rational coefficients. We can sort the terms of any such polynomial by, say, ascending degree; then the leading term is the one of lowest order. The ring R does not have zero divisors; and the field of rational functions $\mathbb{Q}(t)$ is the field of fractions. Multiplying the sign of the leading coefficient of the numerator $p \in \mathbb{Q}[t]$ and the sign of the leading coefficient of the denominator $q \in \mathbb{Q}[t]$ defines a sign for the fraction $p/q \in \mathbb{Q}(t)$. In this way we obtain an ordered field. Allowing for rational exponents in the construction above does not change much. The resulting fields, called *Puiseux fractions* in [17, §2.6], admit a valuation by mapping to the lowest order degree. Because of the latter property, the Puiseux fractions occur naturally, e.g., in tropical geometry.

Here we look at a standard example from linear programming through the lens of Puiseux fractions. The *Klee-Minty* cube KM_d is a d -dimensional unit cube whose facet description is perturbed by a sufficiently small parameter. Note that, in the ordered field of Puiseux fractions, where the leading term is defined by

lowest degree (that is chosen by the `Min` template parameter below), the rational function t is smaller than any rational constant.

```

polytope > $m =
↳ monomials<Rational,Rational>(1);

polytope > $t = new PuiseuxFraction<Min>($m);
↳ set_var_names<UniPolynomial<Rational,Rational>>('t');

polytope > $KM = klee_minty_cube(3, $t);

```

We can look at the facet description in human readable form. The coefficients are rational functions (with rational exponents) in t .

```

polytope > print_constraints($KM);
Facets:
0: x1 >= (0)
1: -x1 >= (- 1)
2: -(t) x1 + x2 >= (0)
3: -(t) x1 - x2 >= (- 1)
4: -(t) x2 + x3 >= (0)
5: -(t) x2 - x3 >= (- 1)

```

Applying standard algorithms from polyhedral geometry gives the volume as a function of t , provided that t is small enough.

```

polytope > print $KM->VOLUME;
(1 -2*t + t^2)

```

The algorithm that is applied in this particular computation is basic: the polytope is first triangulated, and then the volumes of the maximal simplices are added up. In `polymake` this is implemented in `C++`, where the ordered field of coefficients occurs as a template type. So the code works over any ordered field, for which an implementation of the arithmetic is provided.

Using `polymake` via `OSCAR`

In addition to polytopes and derived objects such as linear programs and polyhedral fans, `polymake` also features matroids, finite simplicial complexes, tropical hypersurfaces and more. That considerable range of computational methods is vastly expanded within the `OSCAR` project, where `polymake` joins forces with `GAP` [19] (for group and representation theory), `Singular` [11] (for commutative algebra and algebraic geometry) and `ANTIC/Nemo/Hecke` [15, 12] (for number theory).

`OSCAR` is written in `Julia`. One feature which makes `Julia` attractive as a programming language, is the ease of installation. With `Julia` running the following two lines suffice to install and start up `OSCAR`.

```

julia> using Pkg; Pkg.add("Oscar")
julia> using Oscar

```

Under the hood there is a fully featured `polymake` running, next to its friends `GAP` and `Singular`. So we can redo the convex hull benchmarks from the beginning, with `Julia` 1.10.8 and `OSCAR` 1.2.2.

```

julia> P = rand_spherical_polytope(6, 300;
  ↪ seed=11)
Polytope in ambient dimension 6

julia> Polymake.prefer("lrs") do
  @time n_facets(P)
end
19.258279 seconds (10.63 M allocations:
  ↪ 815.511 MiB, 0.40% gc time, 0.48%
  ↪ compilation time)
30401

julia> Polymake.prefer("libnormaliz") do
  @time n_facets(P)
end
1.307261 seconds (7.18 M allocations:
  ↪ 288.132 MiB)
30401

```

Observe that the same seed parameter is used, and thus the polytopes constructed by `polymake` and OSCAR are actually the same. The Julia interface layer `Polymake.jl` [18] can also be used independently from OSCAR.

Real algebraic number fields

One obvious benefit of using `polymake` from within OSCAR is the access to a wide range of additional functionality. For instance, OSCAR is good at computing with algebraic number fields, some of which arise as subfields of the reals via a fixed embedding. The latter fields are suitable for polyhedral geometry.

Below we construct a regular 17-gon and compute its exact area. Internally, the same C++ code from `polymake` is used as in the Klee–minty example above.

```

julia> n = 17; rho =
  ↪ root_of_unity(algebraic_closure(QQ), n)
Root 0.932472 + 0.361242*im of x^16 + x^15 +
  ↪ x^14 + x^13 + x^12 + x^11 + x^10 + x^9 +
  ↪ x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 +
  ↪ x + 1

julia> P = convex_hull(stack( [[ real(rho^k)
  ↪ for k=0:n-1 ], [ imag(rho^k) for k=0:n-1
  ↪ ] ] ))
Polyhedron in ambient dimension 2 with
  ↪ QQBarFieldElem type coefficients

julia> volume(P)
Root 3.07055 of 4294967296x^16 -
  ↪ 1318823395328x^14 + 166748733046784x^12 -
  ↪ 11187053393870848x^10 +
  ↪ 427447629075906560x^8 -
  ↪ 9264927360220274688x^6 +
  ↪ 106494023009804634624x^4 -
  ↪ 549585225889884632256x^2 +
  ↪ 827240261886336764177

```

The construction is pretty direct; a more fancy version is implemented in OSCAR’s function `n_gon`. The vertices of a regular polygon correspond to the powers of a suitable root of unity. Via mapping to the real and imaginary parts, we may view a complex number as a point in \mathbb{R}^2 . Of course, the challenging part of this computation is the arithmetic in the number field. For this,

ANTIC/Nemo/Hecke [15, 12] rely on Arb [16], which provides exact interval arithmetic. The floating numbers in the output serve as approximate values. More importantly, they specify unique roots of the given minimal polynomials. Recently, OSCAR’s polyhedral geometry functions over embedded number fields were exploited in an application to the representation theory of finite groups [7].

Easter eggs

OSCAR’s polyhedral geometry function rely on `polymake` and its dependencies. It is worth noting, however, that all `polymake` functions can also be called directly. This is the purpose of `Polymake.jl` [18]. The translation from the `polymake` syntax to the Julia syntax is straightforward. For instance, the following computes the reduced integral homology of the real projective plane.

```

julia> rp2 =
  ↪ Polymake.topaz.real_projective_plane();

julia> rp2.HOMOLOGY
pm::Array{<topaz::HomologyGroup{pm::Integer}>
  ({} 0)
  ({} (2 1) 0)
  ({} 0)
}

```

The output above says that the reduced integral homology modules H_0 and H_2 are trivial, and H_1 is isomorphic to $\mathbb{Z}/2$.

Pressing `$` within OSCAR in the Julia REPL even gives access to `polymake`’s original Perl command line. And backspace brings you back.

Outlook

The overall design goals of `polymake` and OSCAR are rather different. `polymake` addresses the needs of experts in polyhedral geometry, who need all the bells and whistles to allow for really large computations. OSCAR, on the other hand, wants to be more accessible to a wider audience; this is natural because the total range of methods is much larger. To allow for both usage patterns to continue, `polymake` will remain the polyhedral geometry engine of OSCAR and, at the same time, an independent product.

A final word on programming languages. From the start `polymake` was designed as a hybrid between Perl (as an interpreted language) and C++ (as a compiled language). The distinction between interpreted and compiled languages is still useful, even if modern languages such as Julia blur the line between these two concepts. Perl, however, was first released in 1987, when the Internet was still in its infancy. So it shows signs of age (similar to Python, by the way, which is from 1991). For this reason the `polymake` team currently works on a near-complete rewrite, with more, and more modern, C++ and less Perl. As an additional benefit the upcoming `polymake` version 5.0

will be fully thread-safe. In the long run we will keep C++ and replace Perl by Julia.

Acknowledgments

I would like to thank all current and previous developers of `polymake` and `OSCAR`. In particular, Alexej Jordan and Benjamin Lorenz contributed greatly to the new support of polyhedral geometry over arbitrary number fields in `OSCAR`. Since 2017 `polymake` and `OSCAR` has been funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), „Symbolic Tools in Mathematics and their Application“ (TRR 195, project ID 286237555).

References

- [1] Benjamin Assarf et al., *Computing convex hulls and counting integer points with `polymake`*, *Math. Program. Comput.* **9** (2017), 1–38.
- [2] David Avis, *`lrslib` (Version 7.3)*, <https://cgm.cs.mcgill.ca/~avis/C/lrs.html> (2024).
- [3] David Avis, David Bremner and Raimund Seidel, *How good are convex hull algorithms?*, *Comput. Geom.* **7** (1997), 265–301.
- [4] David Avis and Komei Fukuda, *Reverse search for enumeration*, *Discrete Appl. Math.* **65** (1996), no. 1-3, 21–46.
- [5] David Avis and Charles Jordan, *`mplrs`: A scalable parallel vertex/facet enumeration code*, *Math. Program. Comput.* **10** (2018), 267–302.
- [6] Roberto Bagnara, Patricia M. Hill and Enea Zaffanella, *The Parma Polyhedra Library: toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems*, *Sci. Comput. Programming* **2** (2008), 3–21.
- [7] Thomas Breuer, Michael Joswig and Gunter Malle, *Zeros of S -characters*, preprint [arXiv:2408.16785](https://arxiv.org/abs/2408.16785) (2025).
- [8] Winfried Bruns, Bogdan Ichim and Christof Söger, *The power of pyramid decomposition in `Normaliz`*, *J. Symbolic Comput.* **74** (2016), 513–536.
- [9] Winfried Bruns, Bogdan Ichim, Christof Söger and Ulrich von der Ohe, *`Normaliz`. Algorithms for rational cones and affine monoids. (Version 3.10.4)* <https://www.normaliz.uni-osnabrueck.de> (2024).
- [10] Wolfram Decker, Christian Eder, Claus Fieker, Max Horn and Michael Joswig, *The Computer Algebra System `OSCAR`: Algorithms and Examples*, volume 32 of *Algorithms and Computation in Mathematics*, Springer, 2025.
- [11] Wolfram Decker, Gert-Martin Greuel, Gerhard Pfister, and Hans Schönemann, *`Singular 4-4-0` — A computer algebra system for polynomial computations*, <http://www.singular.uni-kl.de> (2024).
- [12] Claus Fieker, William Hart, Tommy Hofmann, and Fredrik Johansson, *`Nemo/Hecke`: Computer Algebra and Number Theory Packages for the Julia Programming Language*. In *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC ’17*, pages 157–164, New York, NY, USA, 2017. ACM.
- [13] Komei Fukuda et al., *`cddplus` (Version 0.92m)*, <https://github.com/cddlib/cddlib> (2020).
- [14] Ewgenij Gawrilow und Michael Joswig, *`polymake`: a framework for analyzing convex polytopes*, *Polytopes—combinatorics and computation* (Oberwolfach, 1997), DMV Sem. **29**, pp.43–73, Birkhäuser, Basel, 2000.
- [15] William Hart, *`ANTIC`: Algebraic Number Theory in \mathbb{C}* , *Computeralgebra Rundbrief* **56** (2015), 10–11.
- [16] Fredrik Johansson, *`Arb`: efficient arbitrary-precision midpoint-radius interval arithmetic*, *IEEE Trans. Comput.*, **66** (2017), 1281–1292.
- [17] Michael Joswig, *Essentials of tropical combinatorics*, *Graduate Studies in Mathematics* **219**, American Mathematical Society, Providence, RI, 2021.
- [18] Marek Kaluba, Benjamin Lorenz and Sascha Timme, *`Polymake.jl`: a new interface to `polymake`*, In: *Mathematical software — ICMS 2020*, 377–385. *Lecture Notes in Comput. Sci.*, 12097.
- [19] The GAP Group, *`GAP`—Groups, Algorithms, and Programming* (Version 4.14.0), <https://www.gap-system.org> (2024).
- [20] The Sage Developers, *`SageMath`, the Sage Mathematics Software System (Version 10.5)*, <https://www.sagemath.org> (2024).

DFG Priority Program SPP 2458: Combinatorial Synergies

Overview

The DFG Priority Program SPP 2458 *Combinatorial Synergies* is a collaborative research initiative funded by the German Research Foundation (DFG). With this program, the DFG and the program committee aim to advance combinatorial research as a unifying and interdisciplinary field, fostering synergies across diverse mathematical and applied domains. It launched in 2024 and will run for 6 years. Since its launch, it has brought together leading experts and emerging scholars in combinatorics, creating a vibrant network of scientific exchange and innovation. An overview of the manifold activities and opportunities can be found under

www.combinatorial-synergies.de

Combinatorics is the study of finite and discrete structures. Starting from fundamental questions of ordering, decomposition and structuring of finitely many objects or states, combinatorics represents the nanotechnology of mathematics and its applications. Due to its interdisciplinarity, it is a central mathematical research area with influence across disciplinary boundaries. Questions are unified and sound theories with intrinsic questions and methods are developed from structurally related approaches. The two 2022 Fields Medals to June Huh and Maryna Viazovska for solutions to central combinatorial problems emphasize both the importance and timeliness as well as the potential of this field.

Discrete data has always been a source for the development of mathematical theories. Their analysis is comparable to the derivation of physical laws from observations of phenomena in nature. Due to the complexity of mathematical observations, the field is seeing a revolution for the development cycles in the interplay of data and structure. This change in research methodology is being met worldwide with versatile programs focused on combinatorics. The SPP 2458 aims to unlock the potential of extensive combinatorial databases which are seen as mathematical research data. Following successful examples like the Online Encyclopedia of Integer Sequences (OEIS), the program will develop new ways to make these mathematical treasures more accessible and valuable for the research community.

This priority program links the huge potential of excellent and dynamic combinatorics groups. It enables breakthrough advances within and across the thematic areas. In the process, the accessibility and usability of discrete data acts as a multiplier. A globally visible combinatorics network in modern basic mathematical research is being created.



Researchers across various institutions contribute to the overarching theme, exploring synergies within and between the 9 themes of the program: Enumeration, Dynkin Classification, Commutative Algebra, Matroids, Convexity, Lattice Points, Statistics, Non-linear Optimization and Mathematical Physics.

Program Structure and Funding Opportunities

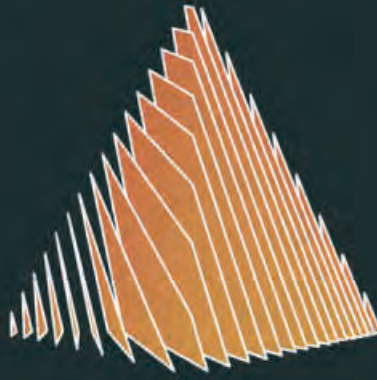
A DFG priority program is decentralized, meaning that it funds individual projects, which are led by principal investigators all over Germany. To this end there are 2 funding rounds, to which any person with a PhD and based at a higher education institution in Germany can apply. The first funding round was concluded in the beginning of 2024, funding about 30 positions in 20 projects. A second call for proposals will be announced in 2026 with a submission deadline in the fall of 2026. An international review panel formed by the DFG reviews all proposals and makes the funding decisions. The funded 3-year projects will commence in the spring of 2027.

Next to funding positions, a main purpose of the priority program is to provide opportunity for scientific exchange and the dissemination of new ideas. To this end the program committee has published ongoing open calls for various activities. There are no deadlines and such funding is granted on a rolling basis by the program committee.

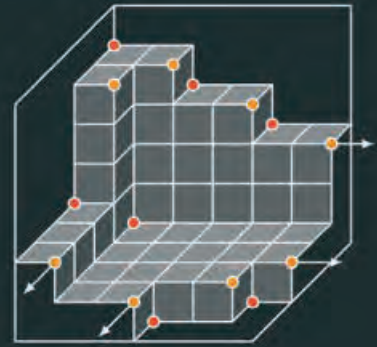
The most common scientific activity is the **Workshop**. The SPP supports program related conferences and thematic workshops, e.g. by travel grants, or centrally covering accommodation costs. Anybody eligible for DFG funding is invited to submit proposals to run a workshop or conference, regardless of having a funded project within the SPP. Such an application is a short document of at most 4 pages, which should include a description of the event, including dates, speakers, etc. and a budget, including a list of items that should be funded by the SPP. Any SPP-funding should normally be matched by other funds. Funding is possible up to 10.000 EUR for a week-long event.



ENUMERATION



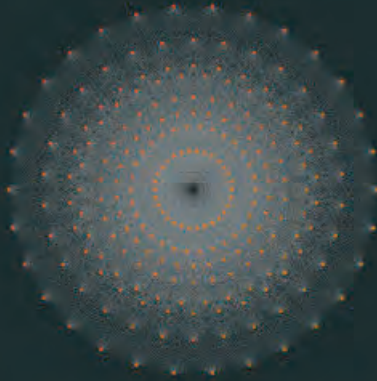
STATISTICS



COMMUTATIVE ALGEBRA



MATHEMATICAL PHYSICS



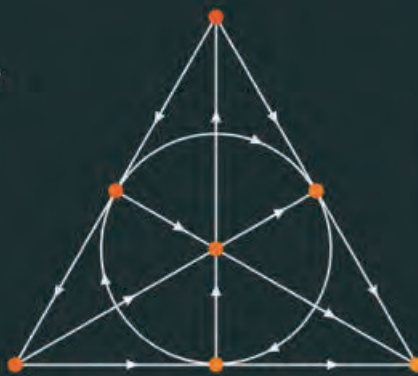
DYNKIN CLASSIFICATION



CONVEXITY



LATTICE POINTS



MATROIDS



NONLINEAR OPTIMIZATION



The **Dive into Research (DiR)** is our program for engaging next generation mathematicians in combinatorial research. Based on successful models like REU and DAAD-RISE, the DiR provides hands-on research experience to advanced undergraduate students. Each DiR combines an introduction to a specific combinatorial topic with team-based research projects. Programs run for 3-4 weeks, with participants working in-person at host institutions. The introductory component includes lectures and reading courses, which may be conducted online. Projects span the program's research areas and emphasize innovative approaches to research data. Throughout, participants receive mentoring from PIs, postdocs, and PhD students.

Mercator Fellows are renowned researchers from abroad who engage in extended research stays, typically totaling around 3 months, at one or preferably multiple German institutions. During these visits, they work on collaborative projects that advance the SPP's research goals. The program centrally administers funding support for Mercator Fellows who can meaningfully contribute to the SPP's work and success.

The entire program is dedicated to fostering **excellence through diversity** through several concrete initiatives. It interconnects with decentralized equal opportunity offices and partners with the Gender Consulting Office at Ruhr-University Bochum, which provides implicit bias training and evaluates our diversity measures. The program actively promotes family-friendly best practices across the network. A Diversity Symposium will assess these measures.

Outlook

The program committee invites and encourages participation with proposals in the upcoming funding round and of course through the many activities already planned. The regularly updated calendar on the homepage shows all events and activities. Once per year, the entire SPP comes together in a central meeting. The next such annual conference will take place in September 3-5 2025 in Hannover. Anybody interested in combinatorial research and this SPP is welcome to attend.

The next funding round will be announced in 2026, inviting applications for innovative research projects. Such proposals should follow the rules of the DFG Sachbeihilfe funding scheme. The program committee can answer questions about the application process, for example at the annual conference. In the second funding phase, specific emphasis will be put on creating synergies between the themes and projects. Joint projects with several principal investigators are possible and encouraged. An individual project will typically consist of funds for one PhD student or postdocs (per PI) for 3 years and a small amount of funds for travel.

SPP 2458 *Combinatorial Synergies* is an active and growing research initiative, fostering interdisciplinary collaboration and advancing the field of combinatorics. With its distributed project structure, extensive scientific activities, as well as the present and upcoming funding opportunities, it continues to shape the future of combinatorial research.

Thomas Kahle (Magdeburg)



Workshop-Förderung der Fachgruppe:

Sie veranstalten einen Workshop zu einem Thema aus dem Bereich der Computeralgebra und könnten mit einer kleinen finanziellen Unterstützung den Workshop deutlich interessanter oder effektiver gestalten? Die Fachgruppe Computeralgebra unterstützt Workshops mit bis zu 1000,- Euro.

Anträge können mit einer kurzen Beschreibung des Workshops (ca. 1 DIN A4 Seite; kurze Beschreibung des Gebiets, Thema des Workshops, Zielgruppe, Budget-Planung) und einer Darstellung, inwiefern diese Förderung einen deutlich erkennbaren Beitrag zum Gelingen des Workshops und zur Nachwuchsförderung liefert, an die Sprecherin der Fachgruppe gerichtet werden:

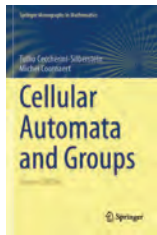
anne.fruehbis-krueger@uni-oldenburg.de,

bitte „**Workshop-Förderung**“ im Betreff angeben.

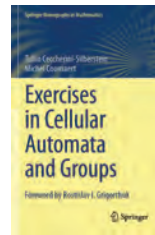


Publikationen über Computeralgebra

Neuerscheinungen:



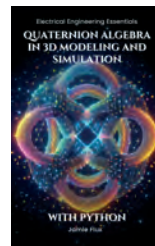
T. Ceccherini-Silberstein,
M. Coornaert,
Cellular Automata and Groups,
Springer-Verlag,
2. Auflage, Jan. 2024,
577 Seiten,
ISBN 978-3031433276



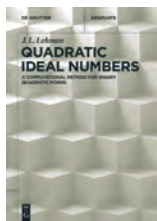
T. Ceccherini-Silberstein,
M. Coornaert,
*Exercises in
Cellular Automata and Groups*,
Springer-Verlag,
Nov. 2024, 648 Seiten,
ISBN 978-3031103933



P. Ferragina, F. Luccio,
*Computational Thinking:
First Algorithms, Then Code*,
Springer-Verlag,
2. Auflage, Okt. 2024,
209 Seiten,
ISBN 978-3031599217



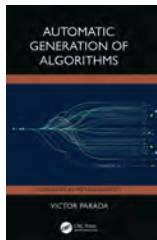
J. Flux,
*Quaternion Algebra in 3D Modeling
and Simulation: With Python*,
Golden Dawn Engineering,
Nov. 2024, 381 Seiten,
ISBN 979-8345942864



J. L. Lehman,
*Quadratic Ideal Numbers:
A Computational Method
for Binary Quadratic Forms*,
De Gruyter Verlag,
Jan. 2025, 254 Seiten,
ISBN 978-3111319353



Naoya und Hiroshi Nakazawa,
*Random Number Generators
on Computers*,
Jenny Stanford Publishers,
Nov. 2024, 120 Seiten,
ISBN 978-9814968492



Victor Parada,
*Automatic Generation
of Algorithms*,
CRC Press,
Feb. 2025, 214 Seiten,
ISBN 978-1032894454

Die Rubrik Publikationen ist nicht allein auf eine Liste von Neuerscheinungen und Neuauflagen beschränkt. Sie lebt vor allem von fundierten Rezensionen von Fachgruppenmitgliedern für Fachgruppenmitglieder, die wir an dieser Stelle gerne abdrucken. Sollte eines der oben genannten Bücher, insbesondere eine der Neuerscheinungen, Ihr Interesse geweckt haben, und Sie möchten dieses für den Computeralgebra-Rundbrief besprechen, nehmen Sie bitte Kontakt zu Martin Kreuzer (martin.kreuzer@uni-passau.de) auf.

Volker Diekert und Martin Kreuzer, Finitely Presented Groups: With Applications in Post-Quantum Cryptography and Artificial Intelligence

De Gruyter, Berlin 2024, 252 Seiten, ISBN 978-3111473376

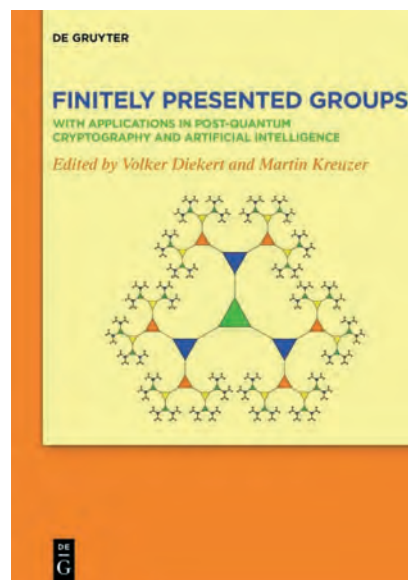
Das Buch „Finitely Presented Groups: With Applications in Post-Quantum Cryptography and Artificial Intelligence“ bietet eine faszinierende und tiefgehende Untersuchung eines hochaktuellen Themas an der Schnittstelle von Mathematik, Informatik und angewandter Kryptographie. Geschrieben von einer Gruppe von Experten auf ihrem Gebiet, präsentiert das Werk sowohl theoretische Grundlagen als auch praktische Anwendungen.

Das Buch besteht aus vier Teilen zu den Themen: Part I: Theory, Part II: Algorithms, Part III: Applications und Part IV: The Life and the Work of Gerhard Rosenberger. Der vierte und letzte Teil unterstreicht, dass das Buch als Geburtstagsband zum 80. Geburtstag von Gerhard Rosenberger zusammengestellt worden ist von den beiden Editoren Volker Diekert und Martin Kreuzer.

Im ersten Teil des Buches (Theory) sind fünf Artikel zur Theorie der endlich präsentierten Gruppen zu finden. Diese beleuchten unterschiedliche theoretische Aspekte in diesem Gebiet. Insbesondere gibt es einen Survey zum Thema „The Universal Theories of Various Classes of Groups“ von A. Gaglione und D. Spellman.

Im zweiten Teil des Buches (Algorithms) sind spezielle Themen zur algorithmischen Theorie der endlich präsentierten Gruppen zu finden und, insbesondere, ein Survey „Complexity of Some Algorithmic Problems in Groups“ von V. Shpilrain.

Im dritten Teil des Buches (Applications) werden Anwendungen der Gruppentheorie auf Themen wie Kryptographie und künstliche Intelligenz vorgestellt. Zum Beispiel gibt es einen Artikel „Secret Sharing Schemes Using Representation Theory of Finite Groups“ von D. Kahrobaei und K. Mallahi-Karai.



Im vierten und letzten Teil des Buches ist eine Biographie von Gerhard Rosenberger von Martin Kreuzer zu finden.

Insgesamt finde ich es ein sehr interessantes Buch mit vielen lesenswerten Artikeln aus ganz unterschiedlichen Bereichen der Theorie der endlich präsentierten Gruppen und ihren Anwendungen. Es regt zu eigenem Nachdenken und Nachforschen an und ist sehr empfehlenswert.

Bettina Eick (Braunschweig)



Dissertationspreis der Fachgruppe Computeralgebra:

Die Fachgruppe Computeralgebra möchte herausragende Dissertationen im Themenbereich der Computeralgebra durch die Vergabe eines Dissertationspreises würdigen. Die Ausschreibung erfolgt zum ersten Mal im Jahr 2024 und danach jährlich, jeweils mit der Einreichungsfrist 1. April.

Eingereicht werden können deutsch- oder englischsprachige Dissertationen, die innerhalb von 12 Monaten vor der Einreichungsfrist verteidigt und veröffentlicht wurden. Zugelassen sind Dissertationen aus dem deutschsprachigen Raum, mit einem betreuenden Institut aus Deutschland, Österreich oder der Schweiz. Das Thema der Dissertation soll einen klaren Bezug zur Computeralgebra (Theorie, Algorithmen oder Implementierung) aufweisen.

Einreichungen können entweder als Eigenbewerbung oder als Nominierung durch die wissenschaftlichen Betreuerinnen und Betreuer per E-Mail an die Fachgruppe Computeralgebra

`ca-promotionspreis@mathematik.de`

erfolgen.

Einzureichen sind in elektronischer Form die Dissertation, eine Kurzfassung (max. 1/2 Seite), akademischer Werdegang mit Publikationsliste und optional ein Empfehlungsschreiben.

Der Dissertationspreis ist mit 500 Euro dotiert. Die Kurzfassungen aller eingereichten Dissertationen werden im Rundbrief der Fachgruppe Computeralgebra veröffentlicht.



Florian Walsh: Computing the Binomial Part of Polynomial Ideals

Betreuer: Martin Kreuzer (Passau)

Zweitgutachter: Manuel Kauers (JKU Linz)

November 2024

Abstract: The aim of this doctoral thesis is to provide algorithms for solving the following three related problems in computational commutative algebra. Emphasis is placed on making these algorithms implementable in existing computer algebra systems.

- *Computing the Binomial Part:* Let K be a field. Given generators of an ideal I in $K[x_1, \dots, x_n]$, determine a finite set of generators of the ideal generated by all binomials contained in I .

- *Computing the Unit Group of a Finite \mathbb{Z} -Algebra:* Given a finite commutative \mathbb{Z} -algebra R , i.e., a commutative ring which is finitely generated as a \mathbb{Z} -module, determine a presentation of the unit group R^\times .

Both tasks can be reduced to the third problem.

- *Computing Exponent Lattices:* Let R be a commutative ring. Given a tuple of units $(f_1, \dots, f_k) \in R^k$, determine a basis of the \mathbb{Z} -module

$$\{(a_1, \dots, a_k) \in \mathbb{Z}^k \mid f_1^{a_1} \cdots f_k^{a_k} = 1\}$$

of all multiplicative relations between these units.

In this thesis we examine the problems mentioned above in more detail and give our motivation for studying them. We also provide an overview of the current state of the art and related work, as well as a detailed description of our approaches for solving these problems.

Combinatorial Synergies: Priority Program Kick-Off Meeting

Osnabrück, 11.09. – 13.09.2024

https://combinatorial-synergies.de/activities/2024_09_KickOff

Die erste Jahrestagung des Schwerpunktprogramms „Combinatorial Synergies“ fand im September 2024 in Osnabrück statt. Lokale Organisatoren waren Paul Breiding, Martina Juhnke und Tim Römer. Zum Programm gehörten 9 einstündige Vorträge mit folgenden Speakern: Gennadiy Averkov, Karin Baur, Mareike Dressler, Alheydis Geiger, Giulio Salvatori, Lisa Seccia, Christian Stump, Martin Wahl, Martin Winter. Außerdem stellten sich die großen Bereiche des Schwerpunkts mit anschließender Diskussion vor. Es gab ferner eine Sitzung zum Thema „Mentoring for PhD students“ von Bernd Sturmfels und einen Workshop über „Implicit Biases“ von Sabine Müller.

Dieses bunte Programm (mit vielen fruchtbaren Kaffeepausen) fand im schön gelegenen Haus des botanischen Gartens der Universität Osnabrück statt. Wie bei den meisten Aktivitäten dieses Schwerpunktprogramms war auch hier wieder auffällig, dass die Community aus jungen, sehr motivierten und engagierten Personen besteht, die mutig mit innovativen Konzepten experimentieren. Das Kick-Off Meeting in Osnabrück war ein sehr gelungener Auftakt für ein vielversprechendes Schwerpunktprogramm.

Michael Cuntz (Hannover)

Women in Algebra and Symbolic Computation III

Bad Dürkheim, 09.12.–11.12.2024

<https://www.computeralgebra.de/women-in-algebra-and-symbolic-computation-iii/>

The workshop, organized by Gunter Malle, Hannah Markwig, Claus Fieker, Gabriela Weitze-Schmithüsen, and Michael Kunze, provided a platform for female researchers in algebra and symbolic computation, with a focus on symbolic methods and computations that connect fields such as group theory, algebraic geometry, commutative and non-commutative algebra, tropical geometry, number theory, and random matrix theory.

Participants took part in insightful talks and a poster session, with key discussions exploring conical zeta values, b -divisors, and convex bodies in tropical geometry; the challenges of solving Diophantine equations in number theory; and the structure of quotients of abelian varieties in algebraic geometry. These themes were explored in keynote lectures by Ana Botero, Rachel Newton, and Cécile Gachet, enriching the program with dynamic and thought-provoking content.

Beyond its academic focus, the workshop fostered an open and engaging atmosphere, encouraging participation from researchers at all career stages. It provided an opportunity to connect with established mathematicians, exchange ideas, and present individual research. While the event primarily aimed to support female mathematicians, male researchers were also welcomed, contributing to a diverse and stimulating exchange of ideas. The workshop facilitated collaboration and mentorship, helping participants build professional networks, gain insights from experts, and enrich academic collaboration and interdisciplinary research.

Firoozeh Dastur (Kaiserslautern)

Hinweise auf Konferenzen

GAMM - 95th Annual Meeting

Posen, Polen, 07.04. – 11.04.2025

jahrestagung.gamm-ev.de

The GAMM Annual Meeting 2025 will be hosted by Poznan University of Technology.

It will take place from April 7th to 11th, 2025, in Poznan, a city where the energies and inventiveness of Eastern and Western European people intertwine.

Submission of Abstracts will be open by 1st of October 2024.

SYMCOMP 2025

Lissabon, Portugal, 10.04. – 11.04.2025

symcomp2025.isel.pt

The ECCOMAS Thematic Conference on Numerical and Symbolic Computation: Developments and Applications, SYMCOMP2025 is the seventh conference in a series that started in 2013, and it aims bringing together academic and scientific communities that are involved with Numerical and Symbolic Computation in the most various scientific areas.

Exchanging experiences and knowledge about current and emerging research and development areas, is a major goal. The multidisciplinary character of this Conference promotes a privileged forum to establish and cross-fertilize new multidisciplinary and cross-sector collaborations.

CoCoA School 2025

Genua, Italien, 14.07. – 18.07.2025

sites.google.com/view/cocoaschool2025

Vom 14.7. bis 18.7.2025 findet an der Universität Genua (Italien) die nächste Ausgabe der internationalen Doktorandenschule *CoCoA School* statt. Wie immer gibt es zwei Kurse, nämlich *Multivariate Cryptography and Polynomial Systems*, gehalten von Alessio Caminata (Genua), und *Liaison Theory*, gehalten von Elisa Gorla (Neuchatel). Die zugehörigen Tutorien, bei denen das Computeralgebrasystem CoCoA zum Einsatz kommt, werden von Giulia Gaggero und Lisa Seccia (beide Neuchatel) durchgeführt.

Im Anschluss an die Schule folgt am 18.7. eine Minikonferenz zu Ehren des 80ten Geburtstags von Lorenzo Robbiano (Genua). Weitere Details, insbesondere die Anmeldemodalitäten, folgen demnächst auf der angegebenen Webseite der CoCoA School 2025.

ACA 2025

Heraklion (Kreta), Griechenland, 14.07. – 18.07.2025

aca2025.github.io

The Applications of Computer Algebra is scheduled on 14-18 July, 2025 and will be held at Heraklion (Crete), Greece. The ACA conference series is devoted to promoting all kinds of computer algebra applications, and encouraging the interaction of developers of computer algebra systems and packages with researchers and users (including scientists, engineers, educators, and mathematicians).

ISSAC 2025

Guanajuato, Mexiko, 28.07. – 01.08.2025

www.issac-conference.org/2025

The International Symposium on Symbolic and Algebraic Computation (ISSAC) is the premier conference for research in symbolic computation and computer algebra. ISSAC 2025 will be the 50th meeting in the series, which started in 1966 and has been held annually since 1981. The conference presents a range of invited speakers, tutorials, short communications, software demonstrations and vendor exhibits with a center-piece of contributed research papers.

ISSAC 2025 will be held from July 28th to August 1st, 2025, at the Center for Research in Mathematics (CIMAT) in Guanajuato, Mexico.

ÖMG-DMV 2025

Linz, Österreich, 01.09. – 05.09.2025

www.jku.at/en/faculty-of-engineering-natural-sciences/organization/subject-areas/mathematics/oemg-dmv-2025

Organized by the Department of Mathematics at the Johannes Kepler University Linz (JKU), the annual joint meeting of the Österreichische Mathematische Gesellschaft (ÖMG), opens an external URL in a new window and the Deutsche Mathematiker-Vereinigung (DMV), opens an external URL in a new window will take place in Linz in 2025. The conference will consist of sections, mini-symposia, and several satellite events.

There will be a section on computer algebra organized by Christoph Koutschan (Linz) and Daniel Robertz (Aachen).

Sommer school “Computer Algebra with OSCAR” of the SFB-TRR 195

Lambrecht, Deutschland, 15.09. – 19.09.2025

oscar-system.org/school/

This summer school aims to teach participants knowledge about computational algebra, both theoretical and practical.

The schedule will encompass:

- introductory courses by experts on selected topics of computational algebra
- introductory sessions on working with git, Julia, OSCAR
- hand-on training sessions implementing selected algorithms in Julia and OSCAR

The introductory courses require only few prerequisites and will be accompanied by hands-on programming sessions. While the courses deal with particular mathematical problems, the techniques introduced have a wide range of applications, in particular from a practical point of view.



Antrag auf Mitgliedschaft in der Fachgruppe Computeralgebra

Die Fachgruppe Computeralgebra sieht es als ihre Aufgabe an, Lehre, Forschung, Entwicklung, Anwendungen, Informationsaustausch und Zusammenarbeit auf dem Gebiet der Computeralgebra in Deutschland zu fördern.

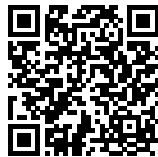
Eine Mitgliedschaft in der Fachgruppe Computeralgebra gibt es bereits ab 7,50 € pro Jahr (für Mitglieder von DMV, GI oder GAMM; ansonsten 9 €).

Vorteile einer Mitgliedschaft:

- Sie fördern durch Ihren Beitrag die Workshops, Seminare, Tagungen und andere Aktivitäten auf dem Gebiet der Computeralgebra, die die Fachgruppe organisiert und unterstützt.
- Sie erhalten zweimal im Jahr den Computeralgebra-Rundbrief mit vielen interessanten Informationen rund um die Computeralgebra frei Haus.
- Sie verleihen unserer Stimme an Gewicht, die wir aktiv in Diskussionen um die Stellung der Computeralgebra in der Ausbildung in Schule und Hochschule einbringen.

Wir würden uns sehr über Ihre Unterstützung freuen. Die Mitgliedschaft in der Fachgruppe steht allen offen. Weiter Informationen zur Mitgliedschaft und einen Aufnahmeantrag finden Sie auf unserer Webseite unter folgender Adresse, oder scannen Sie einfach den QR-Code.

<https://fachgruppe-computeralgebra.de/aufnahmeantrag>



Fachgruppenleitung Computeralgebra 2023–2026

**Sprecherin:**

Prof. Dr. Anne Frühbis-Krüger
Carl-von-Ossietzky Universität Oldenburg
Carl-von-Ossietzky-Straße 11, 26129 Oldenburg
0441 798-3233
anne.fruehbis-krueger@uni-oldenburg.de
<https://uol.de/anne-fruehbis-krueger>

**Stellvertretender Sprecher:**

Prof. Dr. Michael Cuntz
Leibniz Universität Hannover
Welfengarten 1, 30167 Hannover
0511 762-4252
cuntz@math.uni-hannover.de
<https://www.iazd.uni-hannover.de/de/cuntz>

**Vertreterin der GI:**

Prof. Dr. Erika Abraham
RWTH Aachen
Ahornstr. 55, 52056 Aachen
0241 80-21242, -22243 (Fax)
abraham@cs.rwth-aachen.de
<https://ths.rwth-aachen.de/people/erika-abraham/>

**Fachreferentin Industrie:**

Xenia Bogomolec
Quant-X Security & Coding
Engelbosteler Damm 15, 30167 Hannover
0173 3031816
xb@quant-x-sec.com
<https://quant-x-sec.com>

**Fachreferent CA-Systeme und -Bibliotheken:**

Prof. Dr. Claus Fieker
RPTU Kaiserslautern-Landau
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
0631 205-2392, -4427 (Fax)
fieker@mathematik.uni-kl.de
<https://www.mathematik.uni-kl.de/~fieker>

**Fachreferent Physik:**

Dr. Thomas Hahn
Max-Planck-Institut für Physik
Föhringer Ring 6, 80805 München
089 32354-300, -304 (Fax)
hahn@feynarts.de
<https://wwwth.mpp.mpg.de/members/hahn>

**Vertreter der DMV:**

Prof. Dr. Florian Heß
Carl-von-Ossietzky Universität Oldenburg
Institut für Mathematik, 26111 Oldenburg
0441 798-2906, -3004 (Fax)
florian.hess@uni-oldenburg.de
<https://uol.de/florian-hess>

**Fachreferent CA-Systeme und -Bibliotheken:**

Jun.-Prof. Dr. Tommy Hofmann
Universität Siegen
Walter-Flex-Straße 3, 57072 Siegen
0271-740-2868
tommy.hofmann@uni-siegen.de
<https://www.thofma.com/>

**Fachexperte SFB-TRR 195:**

Prof. Dr. Max Horn
RPTU Kaiserslautern-Landau
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
0631 205-2730, -4427 (Fax)
mhorn@rptu.de
<https://www.quendi.de/de/mathe>

**Fachreferent Themen und Anwendungen:**

Prof. Dr. Gregor Kemper
Technische Universität München
Boltzmannstr. 3, 85748 Garching
089 289-17454, -17457 (Fax)
kemper@ma.tum.de
<https://www.math.cit.tum.de/algebra/kemper>

**Fachreferent Publikationen:**

Prof. Dr. Martin Kreuzer
Universität Passau
Innstr. 33, 94030 Passau
0851 509-3120, -3122 (Fax)
martin.kreuzer@uni-passau.de
<https://staff.fim.uni-passau.de/kreuzer/>

**Fachreferent Redaktion Rundbrief:**

Dr. Fabian Reimers
Technische Universität München
Boltzmannstr. 3, 85748 Garching
089 289-17474
reimers@ma.tum.de
<https://www.math.cit.tum.de/algebra/reimers>

**Vertreterin der GAMM:**

Prof. Dr. Eva Zerz
RWTH Aachen
Pontdriesch 14/16, 52062 Aachen
0241 80-94544, -92108 (Fax)
eva.zerz@math.rwth-aachen.de
<https://www.math.rwth-aachen.de/~Eva.Zerz/>

