http://dx.doi.org/10.25673/123350

KI-gestützte Erweiterung der statischen Analyse von GRAFCET-Spezifikationen durch LLM-basierte Extraktion und Integration von Zusatzinformationen

Darko Divkovic[1], Franz C. Kunze[1], Gianluca Manca[1] und Alexander Fay[1]

Abstract: Die Überprüfung kritischer Eigenschaften in GRAFCET-Spezifikationen – wie Sicherheit, Erreichbarkeit und Deadlock-Freiheit – ist für zuverlässige Automatisierung wichtig. Model Checking ist hierfür eine etablierte Methode, leidet jedoch unter Zustandsraumexplosion. Statische Analyse kann diesen Aufwand reduzieren, erfasst jedoch ohne Domänenwissen keine Fehler, die aus semantischen Eigenschaften der Anlagenvariablen resultieren. Dieser Beitrag stellt einen hybriden Ansatz vor, der statische Analyse mit Large Language Models (LLMs) kombiniert, um diese Einschränkung zu überwinden. Die hybride Methode identifiziert erfolgreich synthetisch eingebrachte Modellfehler in einer industriellen GRAFCET-Fallstudie, die klassische statische Analyse übersieht.

Keywords: GRAFCET, Statische Analyse, Formale Verifikation, LLMs

1 Einleitung

1.1 Hintergrund zur GRAFCET

GRAFCET ist gemäß [IEC 60848] eine internationale Norm zur graphischen Spezifikation von Steuerungsabläufen. Steuerungen werden technologieunabhängig über Schritt-Transition-Ketten beschrieben, wobei Transitionen erst bei erfüllter Bedingung und aktivem Schritt im Vorbereich schalten. Basis-Elemente sind Schritte, Transitionen und Aktionen, die abhängig vom Aktivierungszustand der Schritte bzw. von booleschen Bedingungen kontinuierlich oder gespeichert wirken können (siehe Abb. 1). Schritte repräsentieren Zustände, während den Transitionen die Bedingungen zugeordnet sind, die den Übergang von einem Schritt zum nächsten auslösen. Abläufe können sequenziell, parallel oder alternativ definiert werden. Im Vergleich zur Ablaufsprache [IEC 61131-3], die auf GRAFCET basiert [DA10, LA25], unterstützt GRAFCET hierarchische Strukturen wie einschließende Schritte und Makroschritte für komplexe Steuerungen. Zwangssteuerungen ermöglichen das gezielte Einfrieren oder Schalten von Abläufen und sind insbesondere für Störungsreaktionen nützlich, da sie das Modell in definierte Situationen überführen, wobei sie die anderen Regeln von GRAFCET aussetzen. Für eine detaillierte Beschreibung von GRAFCET sei der Leser auf [IEC 60848] verwiesen. IEC 60848 ist ein semiformaler Standard. Um die formale Verifikation von GRAFCET-

[1] Lehrstuhl für Automatisierungstechnik, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, {vorname.name}@rub.de

Modellen zu ermöglichen, ist eine Formalisierung erforderlich. In [Mr22] stellen Mroß et al. eine Formalisierung von GRAFCET vor, und in [Mr23a] präsentieren sie eine eindeutige Interpretation von IEC 60848, die beide hier übernommen werden.

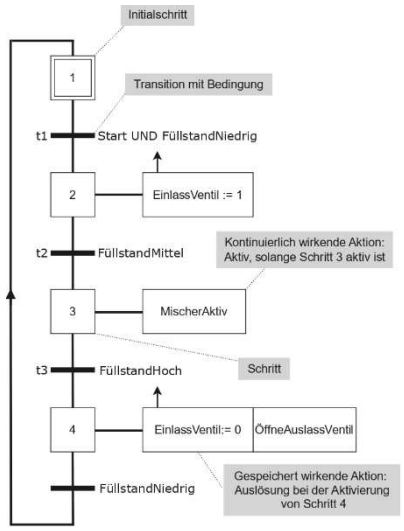

Abb. 1: Beispiel einer GRAFCET-Spezifikation für ein Tanksystem, das mit Fluid befüllt, im Tank gemischt und anschließend entleert wird

1.2 Möglichkeiten und Grenzen der klassischen statischen Analyse von GRAFCET-Spezifikationen

Die Sicherstellung der Korrektheit von Grafcet-Spezifikationen, wobei Grafcet eine Instanz von GRAFCET bezeichnet, ist entscheidend für die Qualität der zugrunde liegenden Steuerungsdokumentation. Zudem beeinflusst bei der automatischen Generierung von Steuerungscode aus Grafcet-Spezifikationen die Validität der Spezifikation maßgeblich die Fehlerfreiheit des resultierenden Steuerungsverhaltens [Ju17, SF14].

Unter statischer Analyse (SA) versteht man Verfahren zur Überprüfung der Struktur und Syntax eines Programms oder einer Spezifikation ohne deren Ausführung. Im Gegensatz zu Model Checking, das alle erreichbaren Zustände für einen festgelegten Anfangszustand erkundet [BK08], ist SA oft skalierbarer, da sie keine vollständige Exploration des erreichbaren Zustandsraums erfordert. Model Checking ist hingegen für das sogenannte Zustandsraumexplosionsproblem bekannt [BK08].

1.3 Verwandte Arbeiten und Einschränkungen

Statische Analyse Methoden liefern wertvolle Informationen zur Unterstützung des Model Checkings, indem sie Variablenausschlüsse und -einschränkungen identifizieren. So wenden beispielsweise Schnakenbeck et al. eine abstrakte Interpretation auf den GRAFCET-Kontrollfluss an, um nicht erreichbare Kombinationen zu identifizieren und die Verifikationskomplexität zu reduzieren [Sc23a]. Die abstrakte Interpretation als ein Framework der statischen Analyse ist dadurch charakterisiert, dass statt einer konkreten Ablaufstruktur abstrakte Zustände analysiert werden, die viele konkrete Zustände zusammenfassen. In der Standardform berücksichtigt dieser Ansatz kein explizites domänenspezifisches Wissen über die Anlage. In [Sc23b] verwenden dieselben Autoren statische Analyse, um parallele Schritte in einer Grafcet-Spezifikation zu identifizieren und die möglichen Werte interner Variablen und Ausgabevariablen sowie Bedingungen zu überapproximieren, um eine Verifikation auf Spezifikationsebene zu ermöglichen. In [Mr23b] stellen Mroß et al. Reduktionstechniken für das Model Checking von Grafcets vor, die mittels statischer Analyse implementiert werden können.

Trotz etablierter Ansätze der SA bleiben bestimmte Fehler in Grafcet-Spezifikationen unentdeckt, da klassische SA nicht über Anlagen-Domänenwissen verfügt. Klassische Verfahren der SA sind auf die Struktur der Grafcet-Spezifikation beschränkt und ignorieren Wissen über die Struktur bzw. das dynamische Verhalten der Anlage, wie z.B. Zustands- oder Konsistenzbedingungen von Variablen, die sich aus deren Semantik ableiten lassen. Beispielsweise können sich Positionssensoren je nach Aktorstellung gegenseitig ausschließen. Dieses Wissen ist in realen Anlagen zwar vorhanden, liegt jedoch meist nur in technischer, nicht maschinenlesbarer Dokumentation vor. In [Sc24] wird das Streckenverhalten mittels Invarianten und prozesstechnisch interpretierter Grafcets modelliert, was jedoch aufwändig ist, da die zusätzlichen Informationen manuell erfasst werden müssen. Die Forschungslücke bestehender SA-Verfahren hinsichtlich GRAFCET-Spezifikationen liegt in der unzureichenden systematischen Berücksichtigung domänenspezifischen Anlagenwissens. Insbesondere ist offen, welche semantischen Informationen hierfür erforderlich sind und wie diese mit geringem oder ohne manuellen Modellierungsaufwand effizient berücksichtigt werden können. Die systematische Einbeziehung domänenspezifischer Informationen aus technischer Dokumentation bietet Potenzial, die Grenzen der SA zu überwinden. Ein vielversprechender Ansatz zur Extraktion solcher Informationen ist der Einsatz von LLMs.

1.4 Motivation für KI-gestützte statische Analyse

Jüngste Fortschritte bei LLMs haben ein wachsendes Interesse an deren Einsatz für ingenieurtechnische Aufgaben geweckt, insbesondere dort, wo unstrukturierte Dokumentationen oder Spezifikationen in strukturierte, maschineninterpretierbare Formate überführt werden müssen. Im Bereich der Automatisierung wurden LLMs untersucht für die Zuordnung von IO-Listen zu Funktionsbausteinen [Ko25], die Extraktion kausaler Abhängigkeiten aus Prozessbeschreibungen [KMA24], die

Generierung von SPS- und DCS-Steuerungscode aus Prompts [KGA23], die iterative Verfeinerung von endlichen Zustandsmaschinen mit Sicherheitsbeschränkungen [AV25], Fehlerbehandlungssysteme, die LLM-Agenten mit Digital-Twin-Umgebungen kombinieren [Gi25], sowie die Entwicklung von Skill-Implementierungen aus Fähigkeitsmodellen mittels Retrieval-Augmented Generation (RAG) [Si25]. Diese Anwendungen verdeutlichen das Potenzial von LLMs, als Assistenten bei der Überbrückung der Lücke zwischen tabellarisch technischer Dokumentation und formalen ingenieurtechnischen Artefakten zu fungieren.

1.5 Vorgeschlagener Ansatz und Beiträge

Auf Basis der identifizierten Lücke in der SA und Verifikation von Grafcet-Spezifikationen stellt diese Arbeit einen hybriden Ansatz vor, der klassische SA mit domänenspezifischem Wissen kombiniert, das durch LLMs gewonnen wird. Konkret werden LLMs eingesetzt, um technische Dokumentationen wie Variablentabellen zu verarbeiten und automatisch gegenseitig ausschließende Signale zu identifizieren, die von der klassischen SA nicht erfasst werden können; weitergehende, erst durch die Verknüpfung mit der GRAFCET-Struktur erkennbare Ausschlüsse liegen außerhalb des Rahmens dieses Beitrags. Diese Zusatzinformationen werden anschließend in den Verifikationsprozess integriert, um Modellierungsfehler wie unerfüllbare oder redundant formulierte Transitionsbedingungen zu erkennen. Die Beiträge dieser Arbeit sind: (i) dass LLMs zur Extraktion semantischer Informationen über mögliche Variablenausschlüsse aus Variablentabellen eines Automatisierungsprojekts eingesetzt werden können, (ii) die Anreicherung der statischen Analyse mit diesen Informationen ermöglicht die Erkennung potenziell unerfüllbarer oder redundant formulierter Transitionsbedingungen in GRAFCET-Spezifikationen, die sonst unentdeckt blieben würden.

2 Vorgeschlagene Methode

Um Ausschlüsse von Eingangs-, internen und Ausgangsvariablen in einer Grafcet-Spezifikation zu analysieren, wird ein hybrider Ansatz vorgestellt, der KI-gestützte Extraktion mit statischer Analyse kombiniert. Zwei boolesche Variablen x und y schließen sich aus, wenn $\neg (x \wedge y)$ als Anlageninvariante gilt. Die Methode besteht aus zwei Teilen: Der erste Teil (*Identifikation von Ausschlüssen*) nutzt LLMs zur Extraktion von Ausschlüssen aus der Dokumentation und überführt diese in ein strukturiertes Format. Der zweite Teil wendet statische Analyse an, angereichert mit dem extrahierten Domänenwissen, um potenzielle Modellierungsfehler zu identifizieren (siehe Abb. 2).

Als Eingabe dient eine – in jedem Automatisierungsprojekt vorliegende – Variablentabelle, in der jede Variable einer Anlagenkomponente zugeordnet ist und Name, Signaltyp, Datentyp sowie einen beschreibenden Kommentar enthält (vgl. Tab. 1).

In Schritt 1 werden die aus PDF-Dateien extrahierten Tabellen mittels eines Python-Skripts an das LLM übergeben und von diesem verarbeitet, um Ausschlüsse zwischen Variablen abzuleiten – etwa, dass *LinearHinten1* und *LinearVorne1* niemals gleichzeitig wahr sein können. Bei größeren Dokumenten könnte ggf. eine RAG-Pipeline [Ga24, Le20] eingesetzt werden; dies ist Gegenstand zukünftiger Arbeiten. Die identifizierten Ausschlüsse werden strukturiert exportiert und in ein bereits aus dem DFG-Projekt AGRAFE vorhandenes Verifikations-Framework importiert.

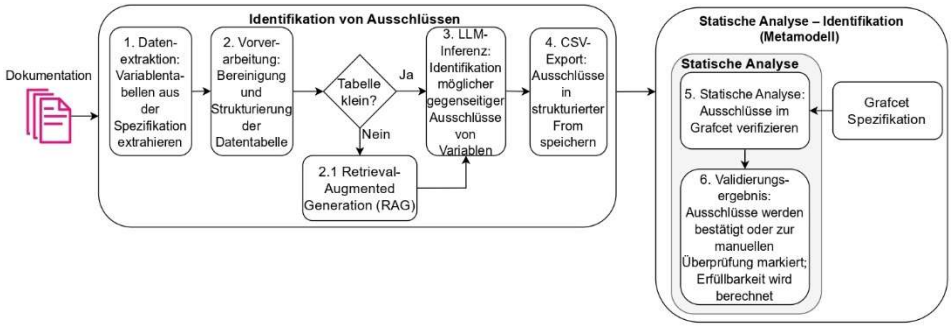

Abb. 2: Vorgehensweise bei der KI-gestützten statischen Analyse

Komponente	Signalname	Signaltyp	Datentyp	Kommentar
Station 1	LinearHinten1	Input	Bool	Lineareinheit ist in hinterer Position
	LinearVorne1	Input	Bool	Lineareinheit ist in vorderer Position

Tab. 1: Ausschnitt aus der Variablentabelle

Die Methode umfasst sechs Schritte: (1) Extraktion von Tabellen aus PDF-Dokumentation und Speicherung als CSV; (2) Vorverarbeitung und Bereinigung der Daten gemäß einer vordefinierten Kopfzeilenstruktur (Komponente, Signalname, Signaltyp, Datentyp, Kommentar); (3) Konfiguration der LLM-API und Konstruktion von Prompts zur Ausschlussextraktion[2]; (4) Validierung der LLM-Ausgabe auf JSON-Konformität und Generierung einer CSV-Datei als Eingabe für den zweiten Methodenteil; (5) Anwendung eines statischen Analysealgorithmus mit einem SMT-Solver, der alle Transitionsbedingungen der GRAFCET-Spezifikation unter Einbindung der gefundenen Ausschlüsse hinsichtlich ihrer Erfüllbarkeit prüft – unerfüllbare oder redundante Bedingungen werden zur menschlichen Überprüfung markiert, da LLMs Halluzinationen

[2] Vollständige Daten dieses Beitrags sind im Paper Repository verfügbar: https://github.com/divkovic-AUT-research/TEFVGRAFCETLLMs

unterliegen können [Ka25]; (6) Präsentation der Ergebnisse in textueller Form mit Angabe der Transitionen, Bedingungen und ihrer Erfüllbarkeit.

3 Evaluierung und Diskussion

Dieser Abschnitt gliedert sich in drei Unterabschnitte: Systembeschreibung, experimenteller Aufbau sowie Evaluierungsergebnisse und Diskussion.

3.1 Systembeschreibung

1) Industrielle Grafcet-Spezifikation: Das betrachtete System ist eine industrielle automatische Prüfmaschine zur Qualitätskontrolle elektromechanischer Bauteile [SF14], deren Grafcet-Spezifikation unter https://github.com/Project-AGRAFE verfügbar ist. Sie umfasst acht Teil-Grafcets mit 60 Schritten, 62 Transitionen, 61 Aktionen und insgesamt 80 booleschen sowie ganzzahligen Variablen (45 Eingangs-, 20 Ausgangs- und 15 interne Variablen) und wurde bereits zur Evaluierung klassischer statischer Analyseansätze verwendet [Sc23a, Sc23b]. Diese Angaben gelten für die korrekte und die fehlerhafte GRAFCET-Spezifikation.

2) Fehlerhaftes Modell eines Teil-Grafcets: Abb. 3 zeigt ein speziell für dieses Paper erstelltes fehlerhaftes Teil-Grafcet für Station 3. Transition t305 enthält die Bedingung *(StoesselOben3* AND *StoesselUnten3)* – zwei sich gegenseitig ausschließende Positionssensoren, die einen Modellierungsfehler darstellen. Die Bedingung von Transition t303 *(¬AusloeserHinten3* AND *AusloeserVorne3)* ist logisch korrekt, jedoch aufgrund der gegenseitigen Ausschließlichkeit der Variablen redundant formuliert und könnte vereinfacht werden zu *AusloeserVorne3*.

3.2 Experimenteller Aufbau

Die Experimente werden mit Python 3.13 und Eclipse IDE 2025-09 durchgeführt. Der Schritt *Identifikation von Ausschlüssen* (Abb. 2) wird in Python implementiert. Aus PDF-Spezifikationen extrahierte Tabellen werden von den in Tab. 3 aufgeführten LLM-Modellen mit Temperatur 0 und maximal 4000 Token analysiert, um Ausschlüsse zwischen Variablen je Systemkomponente abzuleiten. Eine Temperatur von 0 reduziert die Zufälligkeit beim Sampling und führt in der Regel zu weitgehend deterministischen Ergebnissen, ohne jedoch vollständigen Determinismus zu garantieren [Re24], wobei eine Übertragbarkeit dieser Wahl auf die vorliegende Aufgabe nicht zwingend gegeben ist und eine systematische Evaluierung zukünftigen Arbeiten vorbehalten bleibt. Die identifizierten Ausschlüsse werden strukturiert als CSV exportiert.

Der Teil *Statische Analyse – Identifikation* (Abb. 2) wird in Java (JavaSE-17) implementiert, konstruiert die Grafcet-Spezifikation und führt die statische Analyse durch. Das Metamodell ist in [Ju19, Mr22] beschrieben; zur Überprüfung der Erfüllbarkeit boolescher Ausdrücke wird der SMT-Solver Z3 [MB08] verwendet. Zur Evaluierung wird eine Ground Truth Table (GTT) auf Basis der Variablentabelle erstellt, die 23 Variablenausschlüsse dokumentiert (Ausschnitt in Tab. 2).

Komponente	**Variable 1**	**Variable 2**	**Begründung**
Station 1	LinearVorne1	LinearHinten1	Die Lineareinheit kann nicht gleichzeitig in hinterer und vorderer Position sein

Tab. 2: Fragment der Ground-Truth Tabelle

3.3 Evaluierungsergebnisse und Diskussion

Die in Tab. 3 angegebenen Werte sind als Recall bezogen auf 23 Ground-Truth-Ausschlüsse zu verstehen. Die LLM-Analyse (Ergebnisse in Tab. 3) erzeugte keine falsch positiven Ergebnisse, was zeigt, dass LLMs die relevanten Ausschlüsse effektiv erfassen und damit die Vollständigkeit der Systemdokumentation verbessern.

Model	**claude-sonnet-20250514**	**claude-3-7-sonnet-20250219**	**gpt-4o**	**gpt-5 (no temp./token setting)**	**grok-4**	**mistral-medium-2508**	**mistral-large-2411**
Ausschlüs.	82 % (19/23)	82 % (19/23)	78 % (18/23)	87 % (20/23)	82 % (19/23)	82 % (19/23)	82 % (19/23)

Tab. 3: Evaluierungsergebnisse der eingesetzten LLM-Modelle; mehrere Läufe mit identischem Prompt zeigten geringe Varianz

Die anschließende KI-gestützte statische Analyse wurde auf eine korrekte und eine fehlerhafte Grafcet-Spezifikation angewendet. Für die korrekte Spezifikation wurden erwartungsgemäß keine Modellierungsfehler erkannt; zwei möglicherweise redundante Bedingungen im globalen Grafcet wurden zur menschlichen Überprüfung markiert. Für die fehlerhafte Spezifikation (Abb. 3) zeigt sich der Vorteil gegenüber SA ohne Domänenwissen. Während diese alle Variablen als unabhängig behandelt, nutzt die KI-gestützte Methode die identifizierten Ausschlüsse, um Transition t305 als potenziell unerfüllbar und t303 als redundant formuliert zur identifizieren und zur menschlichen Überprüfung vorzuschlagen.

Die Evaluierung zeigt, dass die Methode eine gute Balance zwischen Präzision und Nützlichkeit erreicht: LLMs decken den Großteil der GTT-Ausschlüsse ohne falsch positive Ergebnisse ab, während die Integration mit statischer Analyse potenziell fehlerhafte Transitionen identifiziert, die bisherige Ansätze übersehen würden.

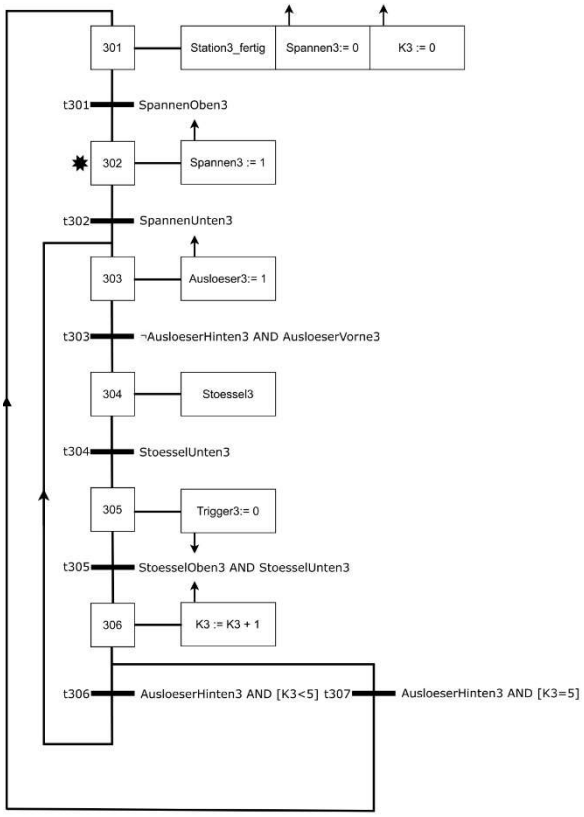

Abb. 3: Fehlerhaftes Teil-Grafcet von Station 3

4 Zusammenfassung

Die KI-gestützte statische Analyse hat gezeigt, dass Modellierungsfehler aus der Systembeschreibung erkannt werden können, die bisherige Ansätze ohne Domänenwissen übersehen würden. Durch die automatische Identifikation sich gegenseitig ausschließender Variablen besitzt diese Methode das Potenzial, den manuellen Aufwand

für Ingenieure zu reduzieren und eine stärkere Konzentration auf übergeordnete Designverbesserungen zu ermöglichen.

Für sicherheitskritische Systeme bleibt jedoch menschliche Überprüfung notwendig, da ein einziger übersehener Ausschluss schwerwiegende Folgen haben kann. Model Checking stellt eine formale Alternative dar, lag jedoch außerhalb des Rahmens dieser Studie. Die quantitative Bewertung des Mehrwerts LLM-abgeleiteter Ausschlüsse gegenüber klassischer SA bleibt zukünftigen Arbeiten vorbehalten.

Der LLM-Einsatz beschränkte sich hier auf die Erkennung von Variablenausschlüssen aus tabellarischen Spezifikationen – ein erster Schritt einer umfassenderen Forschungsagenda. Zukünftige Arbeiten könnten untersuchen, wie LLMs die statische Analyse erweitern durch: (i) Erkennung semantischer Widersprüche in Transitionsbedingungen, (ii) Identifikation semantischer Inkonsistenzen in Grafcet-Spezifikationen, (iii) Vorschläge von Korrekturmaßnahmen, (iv) Minderung kombinatorischer Explosion bei parallelen Abläufen sowie (v) Generierung von Hilfsbedingungen zur Verbesserung von Skalierbarkeit und Präzision des Model Checkings.

Literaturverzeichnis

[IEC 60848] IEC, *GRAFCET specification language for sequential function charts*, IEC 60848:2013.

[IEC 61131-3] IEC, *Programmable controllers – Part 3: Programming languages*, IEC 61131-3:2025.

[DA10] R. David und H. Alla, *Discrete, continuous, and hybrid Petri nets*. Springer, 2010.

[LA25] L. Gniewek und M. Markiewicz, "Extended sequential function chart," *IEEE Access*, vol. 13, pp. 85585–85596, 2025.

[Mr22] R. Mroß et al., "Transformation of GRAFCET into GAL for verification purposes based on a detailed meta-model," *IEEE Access*, vol. 10, pp. 125652–125665, 2022.

[Mr23a] R. Mroß et al., "Unambiguous interpretation of IEC 60848 GRAFCET based on a literature review," in: *Proceedings of the IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2023, pp. 1–8.

[Ju17] R. Julius et al., "Transformation of GRAFCET to PLC code including hierarchical structures," *Control Engineering Practice*, vol. 64, pp. 173–194, 2017.

[SF14] F. Schumacher und A. Fay, "Formal representation of GRAFCET to automatically generate control code," *Control Engineering Practice*, vol. 33, pp. 84–93, 2014.

[BK08] C. Baier und J.-P. Katoen, *Principles of model checking*. MIT Press, 2008.

[Sc23a] A. Schnakenbeck et al., “A control flow based static analysis of GRAFCET using abstract interpretation,” in: *Proceedings of the IEEE 21st International Conference on Industrial Informatics (INDIN)*, 2023, pp. 1–7.

[Sc23b] ´ A. Schnakenbeck et al., “Structural analysis of GRAFCET control specifications,” in: *Proceedings of the IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2023, pp. 1–8.

[Sc24] A. Schnakenbeck et al., *Ansätze für die Modellierung von Streckeninformationen zur Verifikation im Steuerungsentwurf*, EKA 2024.

[Mr23b] R. Mroß et al., “GRAFCET reduction techniques for model checking,” in: *Proceedings of the IEEE 21st International Conference on Industrial Informatics (INDIN)*, 2023, pp. 1–6.

[BR02] T. Ball und S. K. Rajamani, “The SLAM project: debugging system software via static analysis,” *SIGPLAN Notices*, vol. 37, no. 1, pp. 1–3, 2002.

[Ko25] H. Koziolek et al., “Io-automapper: leveraging LLMs to bind I/O signal list entries,” in: *Proceedings of the IEEE 30th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2025.

[KMA24] F. C. Kunze et al., “Automated extraction of conditional causal rules using logic programming and LLMs,” in: *Proceedings of IEEE ETFA 2024*, pp. 1–4.

[KGA23] H. Koziolek et al., “ChatGPT for PLC/DCS control logic generation,” in: *Proceedings of IEEE ETFA 2023*, pp. 1–8.

[AV25] A. King und V. Vyatkin, “LLM-based iterative refinement of finite-state machines,” in: *Proceedings of IEEE ETFA 2025*.

[Gi25] M. S. Gill et al., “Leveraging LLM agents and digital twins for fault handling,” in: *Proceedings of IEEE ETFA 2025*.

[Si25] L. M. V. da Silva et al., “Capability-driven skill generation with LLMs,” in: *Proceedings of IEEE ETFA 2025*.

[Ga24] Y. Gao et al., “Retrieval-augmented generation for large language models: A survey,” arXiv:2312.10997, 2024.

[Le20] P. Lewis et al., “Retrieval-augmented generation for knowledge-intensive NLP tasks,” arXiv:2005.11401, 2020.

[Ka25] A. T. Kalai et al., “Why language models hallucinate,” arXiv:2509.04664, 2025.

[Re24] M. Renze, “The effect of sampling temperature on problem solving in LLMs,” in: *Findings of EMNLP*, 2024, pp. 7346–7356.

[Ju19] R. Julius et al., “A metamodel based environment for GRAFCET specifications,” in: *Proceedings of IEEE SysCon*, 2019, pp. 1–7.

[MB08] L. de Moura und N. Bjørner, “Z3: an efficient SMT solver,” in: *TACAS*, Springer, 2008, pp. 337–340.