Erweiterung des Methodenspektrums zur automatisierten Modifikation von PSA-Modellen

Extending the Spectrum of Methods for an Automated Modification of PSA Plant Models

GRS - 783

Gesellschaft für Anlagen-
und Reaktorsicherheit
(GRS) gGmbH

Erweiterung des Methodenspektrums zur automatisierten Modifikation von PSA-Modellen

Extending the Spectrum of Methods for an Automated Modification of PSA Plant Models

Technischer Bericht
Technical Report

Tanja Eraerds

August 2024

Anmerkung:

Das diesem Bericht zugrunde liegende Eigenforschungsvorhaben wurde mit Mitteln des Bundesministeriums für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz (BMUV) unter dem Förderkennzeichen RS1596 durchgeführt.

Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei der GRS.

Der Bericht gibt die Auffassung und Meinung der GRS wieder und muss nicht mit der Meinung des BMUV-übereinstimmen.

GRS - 783
ISBN 978-3-910548-76-3

Deskriptoren

Netzwerkanalyse, Probabilistische Sicherheitsanalyse, pyRiskRobot, übergreifende Einwirkungen

Kurzfassung

Das Softwarewerkzeug pyRiskRobot der GRS wurde für die automatisierte Erzeugung, Duplikation und Modifikation von Fehlerbäumen für probabilistische Sicherheitsanalysen (PSA) entwickelt. In mehreren Anwendungen hat sich bereits der Nutzen von pyRiskRobot gezeigt, da dieses es erlaubt PSA-Anlagenmodelle konsistent und effizient zu erweitern und anzupassen, was insbesondere bei der Modellierung übergreifender Einzeleinwirkungen und Einwirkungskombinationen von Bedeutung ist. Ursprünglich wurde pyRiskRobot ausschließlich für die Modifikation von PSA-Modellen verwandt, die mit der kommerziellen PSA-Software RiskSpectrum® erstellt wurden.

Im Forschungs- und Entwicklungsvorhaben RS1596 wurde pyRiskRobot überarbeitet und erweitert. So wurde pyRiskRobot, das verschiedene open-source Python-Bibliotheken nutzt, zum einen so angepasst, dass es mit den Weiterentwicklungen dieser Bibliotheken Schritt hält. Dadurch profitiert auch pyRiskRobot von den Verbesserungen dieser Bibliotheken. Zum anderen wurde pyRiskRobot erweitert, um neben PSA-Modellen in RiskSpectrum® auch Fehlerbäume in Anlagenmodellen mit dem PSA-Code SAPHIRE bearbeiten zu können. Dies lockert zum einen die Beschränkung in der Wahl der genutzten PSA-Modellierungswerkzeuge, zum anderen erlaubt es, SAPHIRE-basierte Fehlerbäume zu untersuchen und zu adaptieren, was von erheblicher Bedeutung ist, da SAPHIRE insbesondere in den Vereinigten Staaten ein bevorzugtes Werkzeug zur PSA-Modellierung ist. Des Weiteren wurde pyRiskRobot um Funktionsgruppen zur Untersuchung von gemeinsam verursachten Ausfällen erweitert.

Bereits in der Vergangenheit wurde pyRiskRobot um ein Modul für die Netzwerkanalyse erweitert. Dieses Modul ist vor allem für die Vorbereitung einer PSA für übergreifende Einwirkungen von Bedeutung, da sich dabei oftmals komplizierte Raumabhängigkeiten bei der Modellierung der Ausbreitung solcher Einwirkungen ergeben, die sich über die Methoden der Netzwerkanalyse visualisieren und analysieren lassen.

Des Weiteren wurde die Anbindung von pyRiskRobot an das Netzwerkanalysemodul verbessert, um die Ergebnisse einer Netzwerkanalyse bei der automatisierten Erweiterung eines PSA-Modells um übergreifende Einwirkungen besser nutzen zu können. Außerdem erfolgte eine exemplarische Anwendung dieser Verbindung von pyRiskRobot und der Netzwerkanalyse. Das Netzwerkanalysemodul wurde zudem um Analysemethoden für verschiedene, gleichzeitig auftretende übergreifende Einwirkungen ergänzt, dabei wurde auf die Python-Bibliothek Py3plex zurückgegriffen. Diese erlaubt es, multi-

plexe Netzwerke zu visualisieren und zu analysieren. Dabei entspricht jede Dimension einer bestimmten übergreifenden Einwirkung.

Abstract

The software tool pyRiskRobot developed by GRS was designed for an automated generation, duplication and modification of fault trees for probabilistic safety assessment (PSA). The added value of pyRiskRobot has already been demonstrated in several applications as it allows to extend and adapt PSA plant models consistently and efficiently: That is particularly important for modelling individual single hazards or hazard combinations. Originally, pyRiskRobot was exclusively used for the modification of PSA models generated with the commercial PSA software RiskSpectrum®.

In the frame of the research and development project RS1596, pyRiskRobot has been further enhanced and extended. In particular, pyRiskRobot using various open-source python libraries has e.g. been adapted in line with the advancements of these libraries. As a result, pyRiskRobot also benefits from the improvements to these libraries. Moreover, pyRiskRobot has been extended to be able to process fault trees in PSA plant models with the PSA code SAPHIRE in addition to RiskSpectrum® PSA models. This eases the restrictions in the choice of PSA modelling tools used and allows to analyse and adapt SAPHIRE-based fault trees, which is highly important as SAPHIRE is one of the preferred tools for PSA modelling, particularly in the United States. Furthermore, pyRiskRobot has been extended to include function groups for analysing common cause failures.

A module for network analysis has already been added to pyRiskRobot in the past. This module is particularly important for preparing a PSA for hazards as these often result in complicated spatial dependencies in the modelling of the spreading of such hazards, which can be analysed and visualised using network analysis methods.

In addition, the connection of pyRiskRobot to the network analysis module has been further improved to better utilise the results of a network analysis in the automated expansion of a PSA model to consider hazards. The network analysis module has also been expanded to include analysis methods for different, simultaneously occurring hazards using the python library Py3plex. This allows multiplex networks to be visualised and analysed, with each dimension corresponding to a specific hazard.

Inhaltsverzeichnis

Kurzfassung I

Abstract III

1 Einführung 1

2 Weiterentwicklung von Werkzeugen zur generischen automatisierten Modifikation von PSA-Modellen 5

2.1 Umstrukturierung des Werkzeugs pyRiskRobot 5

2.2 Erweiterung der Kernfunktionalitäten des Werkzeugs pyRiskRobot 6

2.3 Anbindung von pyRiskRobot an SAPHIRE 10

2.4 Neue Funktionsgruppen zur Modellierung gemeinsam verursachter Ausfälle 15

2.5 Schnittstelle zur Netzwerkanalyse 17

2.6 Beispielhafte Nutzung von Netzwerkmaßen zur Vorbereitung einer Brand-PSA-Modellierung 19

2.7 Neue Werkzeuge zur Visualisierung und Analyse multipler Hazards als mehrdimensionale Netzwerke und geeignete Netzwerkmaße 32

2.8 Entwicklungen zur Modellierung und Analyse von Unfallabläufen als dynamische Netzwerke 36

3 Zusammenfassung und Ausblick 39

Literaturverzeichnis 43

Abkürzungsverzeichnis 51

Abbildungsverzeichnis 53

Tabellenverzeichnis 55

A Anhang A: Datenbanktabellen zum Auslesen von Ereignissen, Parametern und Attributen 57

1 Einführung

Probabilistische Sicherheitsanalysen (PSA) erlauben eine umfassende Modellierung des Zusammenspiels einer Vielzahl von Komponenten und die Ermittlung des Beitrags dieser Komponenten zu möglichen Unfallabläufen.

Eine PSA basiert auf Ereignis- und Fehlerbäumen, wobei die Ereignisbäume es erlauben, sowohl den zeitlichen Ablauf als auch die Zusammenhänge verschiedener Unfallszenarien zu modellieren. Ein Ereignisbaum beginnt mit einem auslösenden Ereignis (Englisch: IE für initiating event) und stellt dar, wie sich mögliche neue Szenarien ergeben, indem an Verzweigungspunkten (Ereignissen) jeweils zwei mögliche Verlaufsalternativen betrachtet werden. Die verschiedenen binären Verlaufsalternativen stehen dabei oft in Zusammenhang mit der Unverfügbarkeit von im Ereignisablauf beteiligten sicherheitsrelevanten Strukturen, Systemen und Komponenten (Englisch: SSC für structures, systems and components). Die Wahrscheinlichkeit für den Ausfall eines beteiligten Systems kann in einer PSA entweder über einen Fehlerbaum modelliert oder aber direkt angegeben werden. In einem Fehlerbaum ist die Information enthalten, wie verschiedene einzelne Komponenten zur Systemunverfügbarkeit beitragen. So wird in einem sogenannten „top-down“-approach die Wahrscheinlichkeit eines Systemausfalls basierend auf den Ausfallwahrscheinlichkeiten der nächst-weniger komplexen Systemkomponenten modelliert. Die Ausfallwahrscheinlichkeiten der Systemkomponenten werden dabei über logische und- bzw. oder-Verknüpfungen so miteinander verknüpft, dass sich daraus die Ausfallwahrscheinlichkeit des Systems ergibt.

Genauso werden auch die Ausfallwahrscheinlichkeiten der Systemkomponenten wieder durch die Ausfälle der Systemkomponenten modelliert. Dies wird so lange fortgesetzt, bis nur noch die Ausfallwahrscheinlichkeiten einzelner, voneinander unabhängiger Komponenten übrigbleiben. Insgesamt beschreiben Fehlerbäume also die Unverfügbarkeit sicherheitsrelevanter SSC des zu analysierenden Anlagensystems. Zusammengefasst wird jede postulierte Ausfallursache einer SSC mit einer Wahrscheinlichkeit als ein Basisereignis spezifiziert und ermöglicht durch logische Kombinatorik in Form einer Fehlerbaumtopologie, die Berechnung der Unverfügbarkeit der SSC.

Das Konzept der Fehlerbaumanalyse geht zurück in das Jahr 1960 und wurde 1970 von Vesely um die Methode der ‘Kinetic Tree Theory‘ ergänzt /VES 81/, mit Hilfe derer sich exakte und detaillierte probabilistische Aussagen über Fehlerbäume ableiten lassen. Die grundlegende Annahme ist dabei, dass die Unverfügbarkeit der modellierten Komponen-

ten voneinander unabhängig ist, d. h. es ist wichtig, die zusammengesetzten Komponenten in einer Fehlerbaummodellierung so weit aufzuschlüsseln, dass dies in der Tat der Fall ist. Dass gleiche Komponenten möglicherweise an verschiedenen Stellen eines Fehlerbaums vorkommen, widerspricht nicht der ‘Kinetic Tree Theory‘. Übergreifende Einwirkungen von innen und außen (Englisch: Hazards) können dazu führen, dass Abhängigkeiten zwischen den Ausfällen der einzelnen Komponenten entstehen, die vorher nicht bei den Analysen betrachtet werden mussten. Diese Abhängigkeiten müssen bei der Anpassung eines existierenden Fehlerbaumes unter Berücksichtigung einer einzelnen übergreifenden Einwirkung oder einer Einwirkungskombination (entsprechend der Definition im IAEA Specific Safety Guide SSG-64 /IAE 21/) so modelliert werden, dass die Unabhängigkeit der neuen Komponenten gegeben ist. Dies bedeutet, dass an verschiedenen Stellen einer PSA die gleichen Änderungen durchgeführt werden müssen.

Moderne Methoden und Werkzeuge und zur Erstellung einer PSA, wie beispielsweise RiskSpectrum® /RIS 24/ oder SAPHIRE /SMI 16/ bieten typischerweise eine graphische Oberfläche, die die nutzerfreundliche Erstellung einer PSA ermöglicht. Aufgrund der komplexen Datenmenge eines PSA-Modells (bestehend aus Ereignisbäumen, darin eingebetteten Fehlerbaumtopologien und den spezifizierten, probabilistischen Annahmen) speichern etablierte PSA-Programme, wie z. B. RiskSpectrum®, diese Information unter Verwendung relationaler Datenbanksysteme. Bedingt durch die Größe und die Menge der Fehlerbäume, die für die PSA eines komplexen technischen Systems (z. B. eines Kernkraftwerks) notwendig sind, ist die konsistente Modifikation einer bestehenden PSA durch solche Änderungen eine große Herausforderung, für welche die üblichen graphischen Nutzeroberflächen nicht geeignet sind. Die Betrachtung übergreifender Einwirkungen verstärkt dieses Problem noch einmal. In der GRS wurde deshalb die auf der Ruby-Programmiersprache basierte Software RiskRobot entwickelt /HER 12/, die es erlaubt, Skripte zu erstellen, um diese komplexen und aufwändigen topologischen Operationen automatisiert durchführen zu können. Hierbei wurde der direkte Zugriff (mittels SQL-Kommandos) genutzt, um die zugrundeliegenden Datenbankinformationen konsistent zu modifizieren. Diese Software wurde im Forschungs- und Entwicklungsvorhaben RS1539 in die Python-basierte Software pyRiskRobot überführt, die modularisiert und schichtbasiert aufgebaut ist und so leichter eine Erweiterung und Wartbarkeit der Software ermöglicht. Die Software pyRiskRobot nutzt etablierte, frei verfügbare Python-Bibliotheken, wie beispielsweise SQLAlchemy /MYE 15/, NetworkX /HAG 08/ und PyTables /PYT 24/. Durch Vergleichsstudien, u. a. eines anlageninternen Brandszenarios /BER 16/ und eines anlagenexternen Überflutungsszenarios /BER 17/, wurde pyRiskRobot validiert. Im

Vorhaben RS1556 wurde die gewonnene Flexibilität genutzt, um die Software methodisch weiterzuentwickeln und weitere Modifikationsklassen zu ergänzen. Eine neuere Anwendung von pyRiskRobot für eine interne Überflutung ist in /BER 23/ beschrieben. Auf der Basis von Erfahrungen bei der Integration übergreifender Einwirkungen in PSA-Modelle bietet pyRiskRobot drei generische, topologische Modellierungsklassen an: Generierung, Modifikation und Duplizierung von Fehlerbäumen.

Bisher wurde pyRiskRobot von der GRS immer zusammen mit der PSA-Software RiskSpectrum® genutzt. pyRiskRobot hat dabei mittels des Python-Softwaretools SQLAlchemy auf die MSSQL-Datenbank zugegriffen, die den PSA-Ausgabedateien des elektronischen Anlagenmodells in RiskSpectrum® zugrunde liegt. Der direkte Zugriff von pyRiskRobot auf die Datenbank erfolgt in der untersten Schicht des Schichtmodells von pyRiskRobot. Dieses Schichtmodell bildet die Grundlage dafür, dass der Zugriff prinzipiell auf andere Datenbanken erweiterbar ist, ohne Funktionen in weiter oben liegenden Schichten verändern zu müssen.

Es wurde bereits bei der ersten Anwendung ein methodischer Ansatz erarbeitet, bei dem die Räume bzw. Anlagenbereiche, auch als Hazard Compartments bezeichnet, und deren gegenseitige Abhängigkeiten als einfache Netzwerkgraphen visualisiert wurden /BER 17/. Dabei sind Hazard Compartments als zusammenhängende Bereiche aller im Raum befindlichen, relevanten Komponenten definiert, die gleichartig und gleichzeitig von einer einzelnen übergreifenden Einwirkung oder eine Einwirkungskombination unzulässig beeinträchtigt bzw. beeinflusst werden. Eine Erweiterung dieses Ansatzes erfolgte, indem die Hazard Compartments als Knoten und ihre Abhängigkeiten als gerichtete, gewichtete Verbindungen eines komplexen Netzwerkes interpretiert wurden /BER 19/. Aufbauend auf der Darstellung der Raumabhängigkeiten eines individuellen Hazards wurde ein generischer Ansatz erarbeitet, bei welchem eine multidimensionale Netzwerktopologie verwendet wird, um die Raumabhängigkeiten mehrerer Einwirkungen in einem Anlagensystem zu repräsentieren /BER 20/. Dabei entspricht jeweils eine Netzwerkdimension (Englisch als Layer für Schicht bezeichnet) dem in Abb. 1.1 dargestellten Raumabhängigkeitsnetzwerk für die jeweilige übergreifende Einwirkung. Prinzipiell können auf diese Weise die Raumabhängigkeiten eines individuellen Hazards in den einzelnen Schichten untersucht und über verschiedene Schichten hinweg potenzielle Abhängigkeiten zwischen den Räumen bzw. Raumbereichen in Bezug gesetzt werden.

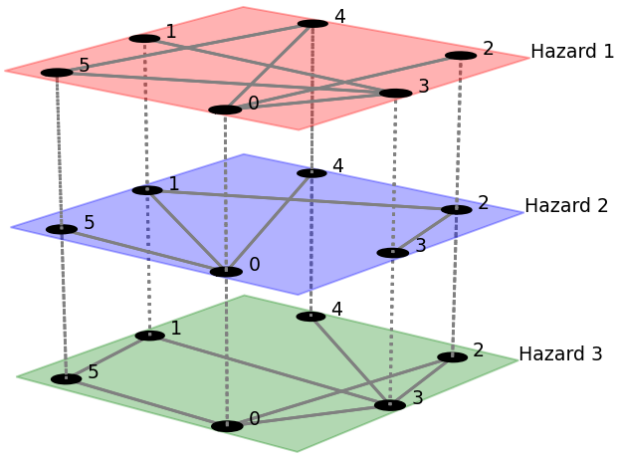

Abb. 1.1 Darstellung multipler Raumabhängigkeiten für übergreifende Eiwirkungen als Schichten eines Multiplex-Netzwerks /BER 20b/

2 Weiterentwicklung von Werkzeugen zur generischen automatisierten Modifikation von PSA-Modellen

Die verschiedenen im Rahmen dieses Vorhabens durchgeführten Erweiterungen an pyRiskRobot machen eine saubere Strukturierung der Kernfunktionalitäten von pyRiskRobot nötig. Die neue Struktur, die in Abschnitt 2.1 beschrieben ist, erleichtert den Überblick über die Funktionalitäten von pyRiskRobot. Die vorgenommenen Anpassungen der unteren Softwareschicht von pyRiskRobot um den Zugriff auch auf Datenbanken nicht nur in RiskSpectrum® zu erweitern sind in Abschnitt 2.2 dargestellt. Diese Anpassungen wurden dazu genutzt, pyRiskRobot auch für SAPHIRE PSA-Anlagenmodelle anwendbar zu machen. Die weiteren, notwendigen Schritte, um SAPHIRE PSA-Modelle mittels pyRiskRobot automatisiert zu modifizieren, sind in Abschnitt 2.3 beschrieben. In diesem Vorhaben wurde pyRiskRobot zusätzlich um neue Funktionsgruppen zur Modellierung von Ausfällen aus gemeinsamer Ursache (GVA, bzw. Englisch: CCFs für common cause failures) ergänzt, wie in Abschnitt 2.4 dargestellt. Um den Informationsaustausch zwischen pyRiskRobot und der Netzwerkanalyse zu verbessern wurde eine Schnittstelle geschaffen, die den Datenaustausch zwischen pyRiskRobot und dem Netzwerkanalyse Modul ermöglicht, dies ist in Abschnitt 2.5 beschrieben. Eine Möglichkeit der Anwendung einer Kombination von pyRiskRobot, der Netzwerkanalyse und maschinellem Lernen für die Vorbereitung einer PSA für übergreifende Einwirkungen findet sich in Abschnitt 2.6. Das Netzwerkanalysemodul selbst wurde ebenso überarbeitet. In Abschnitt 2.7 ist beschrieben wie das Netzwerkanalysemodul um Analysemethoden und Visualisierungen für multidimensionale Netzwerke erweitert wurde. Insbesondere wurde auch verschiedene internationale Ansätze zur Dynamisierung von Fehlerbäumen (siehe Abschnitt 2.8) betrachtet und verglichen.

2.1 Umstrukturierung des Werkzeugs pyRiskRobot

Da sich sowohl Python als auch die genutzten Python-Bibliotheken, wie SQLAlchemy, weiterentwickeln, wurde pyRiskRobot im Rahmen des Forschungs- und Entwicklungsvorhabens RS1596 angepasst, z. B. durch Überführung von Python 2 in Python 3, um auch weiterhin nutzbar zu sein. Des Weiteren erfolgten Umstrukturierungen der Software pyRiskRobot, um die Wartbarkeit und Qualitätssicherung zu verbessern. So wurden z. B. Test-Skripte für die verschiedenen Funktionalitäten von pyRiskRobot erstellt. Diese Test-Skripte erlauben eine schnelle Überprüfung dieser Funktionalitäten. Sinnvollerweise sollten diese Skripte zukünftig in eine kontinuierliche Integrationsumgebung ein-

gefügt werden. Damit wird sichergestellt, dass Weiterentwicklungen, die mit pyRiskRobot erzielten Ergebnisse nicht verfälschen. Darüber hinaus wurde pyRiskRobot weiter modularisiert und in einzelne Pakete unterteilt. So wurden die Kernfunktionalitäten, wie der Zugriff auf die zugrundeliegende Datenbank eines PSA-Modells, von den verschiedenen höheren Funktionalitäten, wie beispielsweise der Modellierung übergreifender Einwirkungen und der Duplizierung von Fehlerbäumen, getrennt.

Bisher waren alle Python-Module, die für die verschiedenen Schichten in pyRiskRobot gebraucht wurden, im Verzeichnis pylibs gespeichert. Durch die in diesem Vorhaben durchgeführten Erweiterungen von pyRiskRobot wurde es notwendig, die vorhandenen Module weiter zu unterteilen, um durch eine bessere Übersichtlichkeit Wartungen zu vereinfachen. Die folgenden Unterverzeichnisse von pylibs wurden angelegt:

- **Core:** In diesem Verzeichnis befinden sich Module, die für die Anbindung und die direkte Kommunikation zwischen pyRiskRobot und der Datenbank verantwortlich sind.
- **Cloning:** In diesem Verzeichnis befinden sich Module, die für das Kopieren von Fehlerbäumen genutzt werden können.
- **Impact:** In diesem Verzeichnis befinden sich Module, die für das Modellieren von Abhängigkeiten bei übergreifenden Einwirkungen benötigt werden.
- **Saphire:** In diesem Verzeichnis befinden sich Module, die das Einlesen eines SAPHIRE PSA-Anlagenmodells im MARD-Format und die Erzeugung eines modifizierten MARD-Ausgabedateienzuständig sind.
- **CCF:** In diesem Verzeichnis befinden sich Module, welche die Modellierung von gemeinsam verursachten Ausfällen vereinfachen.
- **Utilities:** Dieses Verzeichnis enthält Hilfsfunktionen, die in den anderen Modulen benötigt werden.

2.2 Erweiterung der Kernfunktionalitäten des Werkzeugs pyRiskRobot

Der Kern von pyRiskRobot baut auf der SQLAlchemy-Bibliothek auf. Diese erlaubt den Zugriff und die Modifikation von SQL-basierten Datenbanken über Python-Skripte. Datenbanktabellen werden dabei in Python-Objekte umgewandelt und mittels eines sogenannten Object Relational Mappings (ORMs) miteinander verknüpft. Verantwortlich für

diese Aufgaben waren bisher die beiden Python-Modulen SQLDB.py und ORM.py. Im Rahmen des beschriebenen Vorhabens wurden diese Module überarbeitet und erweitert.

Das SQLDB-Modul wurde unterteilt in eine Basisklasse, welche die von der Art der SQL-basierten Datenbank unabhängigen Arbeiten durchführt, und zwei abgeleiteten Klassen, die für die Anbindung an die RiskSpectrum®-Datenbank bzw. die Anbindung an eine SQLITE-Datenbank verantwortlich sind. Dieses Modell sorgt dafür, dass die höheren Schichten, die z. B. für das ORM verantwortlich sind, unabhängig davon sind, welche abgeleitete Klasse genutzt wird, solange in dieser Klasse die vorgegebene Schnittstelle implementiert ist.

Im Zuge verschiedener, mit pyRiskRobot durchgeführter Anwendungen /BER 23/ und /BER 23a/ wurde deutlich, dass der Zugriff auf die Datenbanktabellen erheblich erweitert werden musste, um Zugriff auf alle benötigten Informationen zu haben. So war bisher der Zugriff auf die Tabelle mit den Anwendern von RiskSpectrum® und auf die Fehlerbaumtabelle und die zugehörigen verknüpften Tabellen zu Fehlerbaumknoten und Ereignissen beschränkt. Das ORM hat die Verknüpfung zwischen diesen Tabellen abgebildet. Im Laufe des Vorhabens wurden zusätzlich noch das Auslesen der folgenden Tabellen (vgl. Anhang Tab. A.1) ermöglicht:

- **Attribute:** Die Attribute-Tabelle beinhaltet zusätzliche Ereignisattribute wie die Art des Attributs, die Kennung (ID) oder auch zugehörigen Text.
- **EventAttribute:** Dabei handelt es sich um eine Tabelle, die für die Verknüpfung zwischen Ereignissen (Events) und Attributen (Attribute) zuständig ist.
- **EventExchange:** Dies ist eine Tabelle, die die notwendige Information für sogenannte Exchange Events enthält. Exchange Events sind Einträge im PSA-Modell, die Fehlerbäume miteinander verknüpfen. Dies vereinfacht die Modellierung, da komplizierte Unter-Fehlerbäume separat eingegeben und mittels eines Exchange Events mit anderen Fehlerbäumen verknüpft werden können. Die Tabelle beinhaltet die Information, welche Events in den verschiedenen Ereignisbäumen über das Exchange Event miteinander verknüpft sind.
- **Parameter:** Die Parameter-Tabelle beinhaltet Informationen über die zum Parameter zugehörige Verteilung, wie Typ der Verteilung, Parameter der Verteilung, Einheit, Mittelwert, Median, 5 %-Quantil und 95 %-Quantil.

- **EvPar:** Hierbei handelt es sich um eine Tabelle, die die notwendige Information für die Verknüpfung zwischen Events und Parameter enthält, also welche Ereignisse mit welchen Parametern verknüpft sind.
- **CCFEvPar:** Diese Tabelle enthält die notwendigen Informationen für die Verknüpfung zwischen GVA-Ereignissen und den zugehörigen Parametern. Zur Modellierung von GVA wurden von dieser Tabelle abgeleiteten Objekte genutzt.

Manche dieser Tabellen sind miteinander verknüpft. Diese Verknüpfung wird bei der Erstellung des ORM berücksichtigt, so dass diese Verbindung auch für die abgeleiteten Python-Objekte besteht. Dabei wurde die Struktur angepasst, damit die einzelnen Tabellen nur dann ausgelesen werden, wenn sie entweder von einer höheren Schicht aus angefragt werden oder wenn eine verknüpfte Tabelle angefragt wird. Auf diese Weise wird die benötigte Zeit für eine Bearbeitung eines PSA-Modells mit pyRiskRobot reduziert, da nur die notwendigen Informationen ausgelesen werden. Ist eine Tabelle einmal ausgelesen worden, wird diese Information gespeichert, so dass ein erneutes Auslesen nicht notwendig ist.

Das ursprüngliche ORM-Modul wurde in verschiedene Einzelmodule unterteilt, deren Zusammenhang in Abb. 2.1 verdeutlicht wird. Hierbei werden mit jedem Beerbungsschritt die Funktionalitäten erweitert. Gleichzeitig wurden für den Zugriff auf die verschiedenen abgeleiteten Datenbankklassen sogenannte ‘RiskRobots‘ implementiert. Jeder ‘RiskRobot‘ ist wiederum ein Objekt einer von einer Default RiskRobot Klasse abgeleiteten Klasse. Diese erlauben es, korrekt verbundene Objekte der abgeleiteten Datenbankklassen zu erzeugen und zu modifizieren. Dabei enthält jeder ‘RiskRobot‘ erst einmal nur einen Verweis zu der zugehörigen Datenbankklasse. Ein Zugriff auf die entsprechende Tabelle erfolgt erst, wenn diese angefragt wird.

Die Klasse pyRiskRobot fasst diese verschiedenen Unterklassen zusammen und dient als Schnittstelle zum Nutzer. Wird ein Objekt der Klasse pyRiskRobot aufgerufen, so wird in der Nutzerdatenbank ein neuer Nutzer-RiskRobot angelegt. Alle Änderungen, die nun von pyRiskRobot durchgeführt werden, können diesem Nutzer zugeordnet werden. Die Nutzer-Datenbank ist mit der Ereignis-, der Fehlerbaum- und der Editor-Datenbank verknüpft, d. h. es gibt für jeden Nutzer bzw. Anwender zugeordnete Ereignisse, Fehlerbäume und Editoren. Für diese Datenbanken erfolgt damit ebenso wie für die Nutzer-Datenbank ein Zugriff, wenn ein Objekt der Klasse pyRiskRobot initialisiert wird.

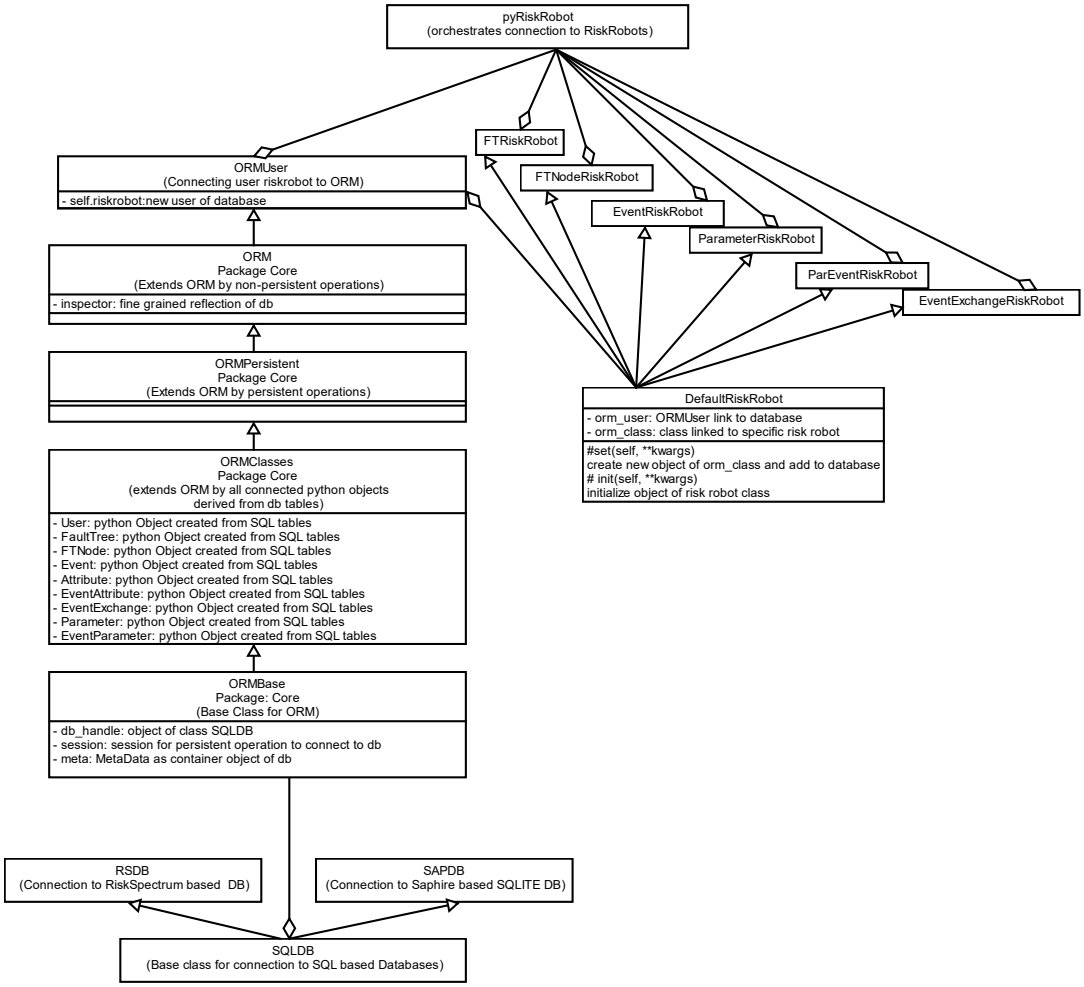

Abb. 2.1 Unified Modelling Language (UML)-Klassendiagramm /KEC 06/ der verschiedenen, für die Kernfunktionalitäten von pyRiskRobot benötigten Klassen

Das Werkzeug pyRiskRobot wurde um sogenannte Jupyter Notebooks ergänzt. Dabei handelt es sich um webbasierte interaktive Computerplattformen, die es dem Nutzer ermöglichen, interaktiv eine Analyse nachzuvollziehen. In den zur Verfügung gestellten Notebooks sind exemplarisch die Kernfunktionalitäten von pyRiskRobot verdeutlicht, der Nutzer kann die Notebooks Schritt für Schritt ausführen und sofort die Ergebnisse jedes Prozessschritts sehen. Abb. 2.2 zeigt einen Ausschnitt solch eines Notebooks.

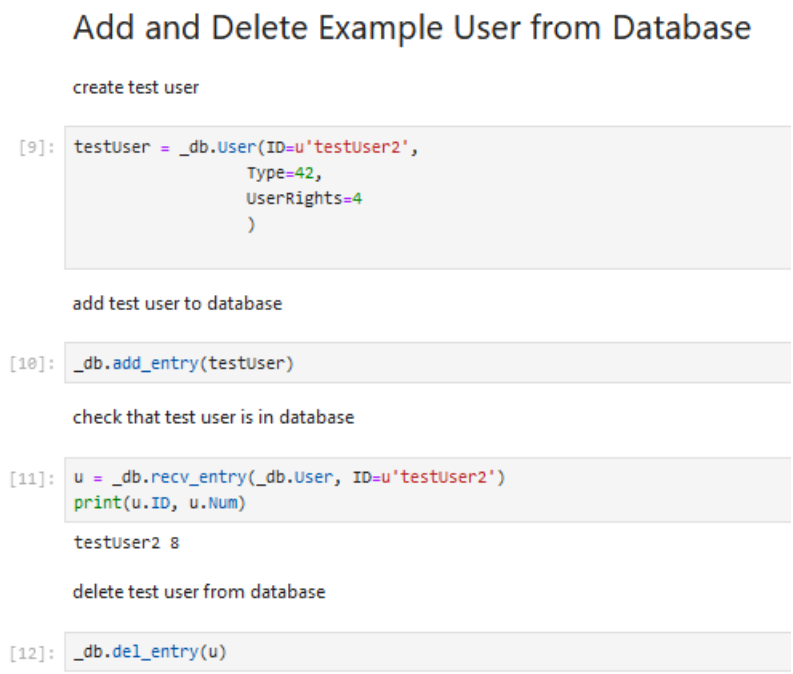

Abb. 2.2 Ausschnitt eines Jupyter Notebooks zur Illustration der exemplarischen Nutzung der pyRiskRobot Kernfunktionalitäten

2.3 Anbindung von pyRiskRobot an SAPHIRE

Während die GRS bisher grundsätzlich das in Europa am weitesten verbreitete, kommerziell vertriebene PSA-Programm RiskSpectrum® /RIS 24/ zur Modellierung einer PSA der Stufe 1 genutzt hat, wird insbesondere für die von U.S.-amerikanischen Herstellern von Small Modular Reactors (SMRs) in den USA und Kanada im Rahmen der Genehmigungsverfahren durchgeführten PSA häufig das grundsätzlich frei verfügbare Programm SAPHIRE /SMI 16/ genutzt, welches von den Idaho National Laboratories (INL) entwickelt und kontinuierlich gepflegt und verbessert wird. Im Rahmen des Forschungs- und Entwicklungsvorhabens RS1596, dessen Schwerpunkt auf der Weiterentwicklung und Anpassung von PSA-Methoden und -Werkzeugen, vor allem für neuartige Reaktoren vom Typ SMR, liegt, sollte dementsprechend auch SAPHIRE bei der Modellierung genutzt werden.

Damit in SAPHIRE PSA-Anlagenmodellen Änderungen mittels pyRiskRobot durchgeführt werden können, wurde in pyRiskRobot eine Schnittstelle eingefügt um auf von SAPHIRE erzeugten, flachen ASCII-basierten Dateien im MARD-Format zugreifen zu können. MARD-Format definiert eine relationale Datenbankstruktur, der einheitliche Standard soll den Datenaustausch zwischen verschiedenen PSA-Werkzeugen erlauben.

Die Struktur der von SAPHIRE erzeugten Datenbank-Dateien wie auch deren Beschreibungen finden sich in /SMI 16/. Um den Austausch mit anderen PSA-Programmen zu ermöglichen, erzeugt SAPHIRE Dateien, welche Beschreibungen der verschiedenen Bestandteile eines PSA-Anlagenmodells enthalten. Diese befinden sich in einem Unterordner, dessen Name sich aus dem Projektnamen und der Endung _Subs zusammensetzt. Zusätzlich wird eine Datei mit der Endung .MARD erzeugt, welche die Pfade zu diesen Beschreibungsdateien enthält (siehe nachfolgende Abb. 2.3).

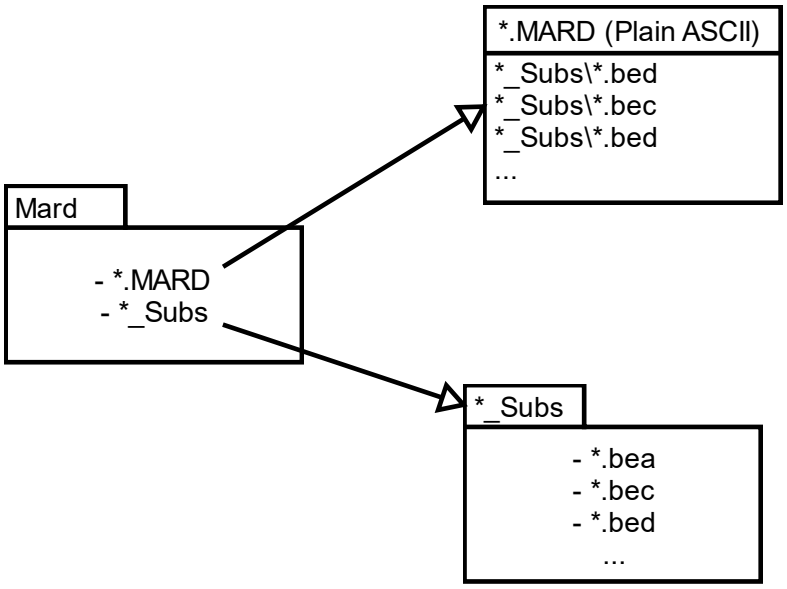

Abb. 2.3 Verbindung zwischen den von SAPHIRE erzeugten Verzeichnissen und Dateien für den Export im MARD-Format

Um zu garantieren, dass SAPHIRE die mit pyRiskRobot bearbeiteten Dateien wieder einlesen und korrekt auswerten kann, ist es wichtig, die Zusammenhänge zwischen den verschiedenen, von SAPHIRE erzeugten Dateien zu verstehen und Änderungen konsistent durchzuführen. Dies ist insbesondere dann von Bedeutung, wenn Fehlerbäume erweitert werden, da dies zu gleichzeitigen Änderungen an mehreren Dateien führt.

Die folgenden Klassen wurden für die Anbindung von SAPHIRE an pyRiskRobot implementiert:

- **SAPDB:** SAPDB ist die bereits in Abschnitt 2.2 beschriebene, von der SQLDB Basisklasse abgeleitete Klasse, die den Zugriff von pyRiskRobot auf eine SQLite-Datenbank ermöglicht.

- **MARD:** Die MARD genannte Klasse erleichtert die Verwaltung der verschiedenen von SAPHIRE erzeugten Dateien im MARD-Ausgabeformat. Diese liest die ‘MARD‘-Datei ein und speichert die Pfade zu den SAPHIRE-Dateien, welche die Beschreibungen enthalten. Darüber hinaus stellt sie eine Funktion zur Verfügung, um zu prüfen, ob in der in dem in der ‘MARD‘-Datei angegebenen Verzeichnis alle benötigten Dateien vorhanden sind.

- **Description:** Für jedes in einer PSA wichtige Bestandteil wird in SAPHIRE eine spezielle Ausgabedatei erzeugt. Dazu wurde pyRiskRobot für jede dieser Ausgabedateitypen um eine spezielle Klasse erweitert. Diese Klassen leiten sich alle von der gleichen Basisklasse ab, um eine einheitliche Schnittstelle zur Verfügung zu stellen. Die Beziehung zwischen der Basis-‘Description‘-Klasse und den abgeleiteten Klassen ist in Abb. 2.4 dargestellt. In den abgeleiteten ‘Description‘-Klassen ist das jeweilige, zu erwartende Format der zughörigen SAPHIRE-Datei gespeichert. Damit bilden diese Klassen die Schnittstelle zwischen dem textbasierten MARD-Format und pyRiskRobot. Objekte dieser Klassen können die entsprechenden SAPHIRE-Dateien lesen und in diese schreiben, zudem können sie mit in pyRiskRobot gespeicherten Informationen gefüllt werden. Zu diesen Description-Klassen gehört z. B. die Klasse BEADescription für Basic Event Attribute (BEA). Diese liest die von SAPHIRE erzeugte ‘BEA…‘-Datei ein. In dieser Datei ist beispielsweise die Information gespeichert, ob es sich bei dem Ereignis um ein sogenanntes Template Event handelt. Template Ereignisse können in SAPHIRE genutzt werden, wenn an verschiedenen Stellen der PSA Ereignisse mit den gleichen Parametern eingefügt werden sollen.

- **DescriptionContainer:** Ein Objekt der DescriptionContainer-Klasse fasst alle verfügbaren Beschreibungen (Descrptions) zusammen, es koordiniert das Erstellen der Descriptions und das Schreiben der neuen SAPHIRE-Ausgabedateien.

- **SaphireRiskRobot:** SaphireRiskRobot leitet sich von der pyRiskRobot-Klasse ab und erweitert sie um die Möglichkeit, basierend auf den oben beschriebenen Descriptions, einen Fehlerbaum in pyRiskRobot zu erstellen und in einer SQLITE-Datenbank zu speichern. Dies beinhaltet auch das Speichern aller mit dem Fehlerbaum verbundenen Informationen über Ereignisse, Attribute und Parameter.

- **Klassen im Distribution-Modul:** Die Klassen in diesem Modul dienen der Übersetzung der verschiedenen Verteilungsfunktionen wie z. B. Normal- und Gammaver-

teilungen zwischen pyRiskRobot und SAPHIRE. Für die folgenden Verteilungen sind Übersetzungen implementiert:

  - Default Option, falls keine Verteilung angegeben ist,
  - Lognormal,
  - Beta,
  - Gamma,
  - Normal,
  - Uniform.

- **Klassen im FailureModel-Modul:** Die Klassen in diesem Modul dienen der Modellierung der verschiedenen, in SAPHIRE implementierten Ausfallmodelle:

  - Ausfälle bei Anfrage,
  - Ausfälle, bei denen keine Reparatur berücksichtigt wird,
  - Ausfälle, bei denen Reparaturen mitberücksichtigt werden,
  - Ausfälle einer periodisch getesteten Komponente,
  - Ausfälle mit und ohne Reparatur,
  - Sogenannte House Events: diese Spezialereignisse werden häufig in einer PSA als eine Art Schalter genutzt, um Teile eines Fehlerbaums variabel mit in die Analyse einzubeziehen oder abzutrennen,

  und als Schnittstelle zu den in RiskSpectrum® verwendeten Ereignis- und Parameterarten. Diese Ausfallmodelle unterscheiden sich im Hinblick auf die Gleichung, auf die die Wahrscheinlichkeit eines Ausfalls, aber auch auf die notwendigen Parameter.

Im pyRiskRobot-Verzeichnis ipynotebooks\SAPHIRE finden sich verschiedene Jupyter Notebooks, welche die Verwendung von pyRiskRobot zur Modifikation von SAPHIRE PSA-Modellen interaktiv verstehbar machen. Abb. 2.5 zeigt einen Ausschnitt eines solchen Notebooks.

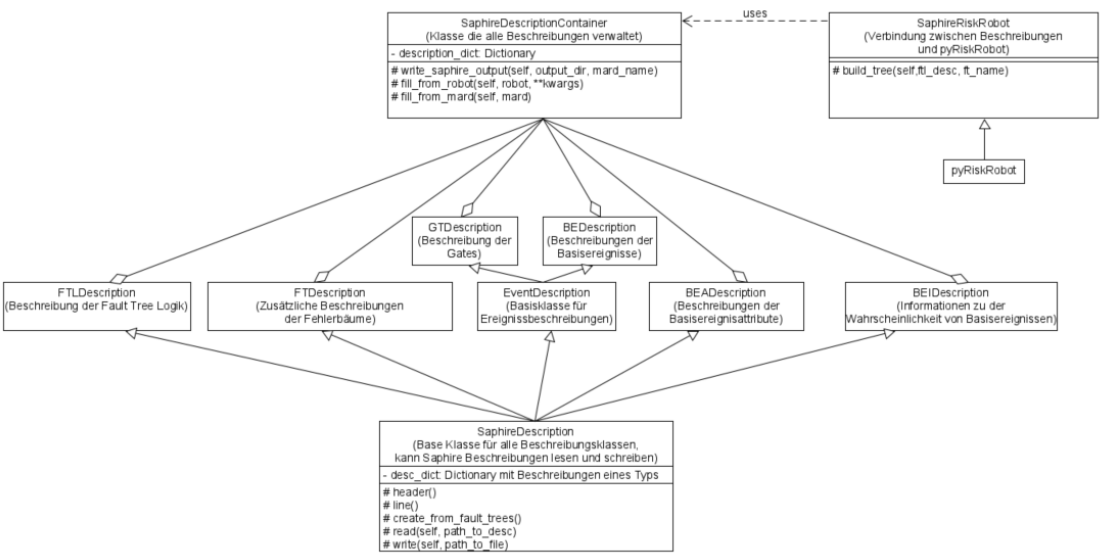

Abb. 2.4 UML-Klassendiagramm /KEC 06/ für die Anbindung von SAPHIRE an py-RiskRobot

Create instance of pyRiskRobot for sqlite database

```
[ ]: rb = SaphireRiskRobot(databasename = 'test_database',rename='test_sqlite',databasetype='sqlite')
```

Read Flat Saphire Output

```
[ ]: path_to_mard = 'C:/Users/ERA/ModelleSaphire/SaphireProj/Mard'
     #path_to_mard = 'C:/Users/ERA/Saphire8'

     mard_name = 'TEST.MARD'

     sdc = SaphireDescriptionContainer(project='ERA',saphire_version='8.3.2')

     mard = MARD(path_to_mard=path_to_mard,mard_name=mard_name)

     sdc.fill_from_mard(mard)
```

```
[ ]: sp_tree = rb.build_tree(sdc.description_dict['FTL'].desc_dict,'CCSFT')

     rb.print_ftstruct(sp_tree)
```

```
[ ]: ftq  = rb.get_entry(rb.FaultTree,ID=u'CCSFT')
```

Print Fault Tree Structure

```
[ ]: rb.print_ftstruct(ftq)
```

Abb. 2.5 Beispiel eines Jupyter-Notebooks für die Anbindung von SAPHIRE an py-RiskRobot

2.4 Neue Funktionsgruppen zur Modellierung gemeinsam verursachter Ausfälle

Der Begriff ‘gemeinsam verursachte Ausfälle‘ (GVA) bezeichnet systematische Ausfälle von technischen Einrichtungen, die gleichzeitig oder im engen zeitlichen Zusammenhang aufgrund einer gemeinsamen Ursache auftreten. Da GVA zu einer gleichzeitigen Unverfügbarkeit mehrerer redundanten Einrichtungen des Sicherheitssystems führen können, haben sie eine hohe sicherheitstechnische Bedeutung. Entsprechend haben GVA einen großen Einfluss auf das PSA-Ergebnis und dominieren es in vielen Fällen. Somit kommt der Modellierung von GVA eine hohe Bedeutung zu /FAK 05/, /FAK 16/. Die Ursachen von GVA sind vielfältig und umfassen z. B. Instandhaltungsfehler, Auslegungsfehler, Herstellungsfehler, unerwartete Alterung sowie nicht vorhergesehene Auswirkungen spezieller Umgebungsbedingungen /HOM 23/. In der jüngsten Vergangenheit sind aufgrund mehrerer entsprechenden Ereignisse in Kernkraftwerken systematische Ausfälle von sicherheitsrelevanten Komponenten aufgrund von Asymmetrien im elektrischen Stromnetz in den Fokus der Aufmerksamkeit gerückt /BRU 17/, /BER 20a/, /BER 23/, /BER 23b/ und /STI 23/.

Zwei Ansätze zur Modellierung der korrelierten Ausfälle aufgrund von Asymmetrien im elektrischen Stromnetz /BER 23b/ konnten nicht direkt in RiskSpectrum® realisiert werden, sondern wurden zusammen mit pyRiskRobot und SUSA umgesetzt. In diesen beiden Ansätzen hängen die Parameter der GVA von den Ausfallwahrscheinlichkeiten der zugehörigen Basisereignisse ab, was nicht in RiskSpectrum® abgebildet werden kann. Um diese Abhängigkeiten zu modellieren, wurden zunächst in einer Monte Carlo-Simulation mit SUSA 1.000 Stichproben aus den Zufallsverteilungen und Abhängigkeiten der Ausfallwahrscheinlichkeiten und GVA-Parameter gezogen. Für jede dieser Stichproben wurde mit pyRiskRobot ein PSA-Anlagenmodell, basierend auf einem Template PSA-Anlagenmodell, erzeugt, dass die jeweiligen Ausfallwahrscheinlichkeiten und die davon abhängenden GVA-Parameter enthält.

Als nächstes wurden mit jedem der 1.000 PSA-Anlagenmodelle eine Konsequenzanalyse mit den festen Ausfallwahrscheinlichkeiten und Parametern durchgeführt. Anschließend wurden die Ergebnisse der Analysen zusammengeführt und ausgewertet. Die Monte Carlo-Simulation der Ausfallwahrscheinlichkeiten und GVA-Parameter wurde somit außerhalb von RiskSpectrum® durch SUSA durchgeführt, wodurch die gegenseitigen Abhängigkeiten berücksichtigt werden konnten. Um das Einlesen der SUSA-Ausgabedatei und das Ersetzen der GVA-Parameter in das jeweilige Anlagenmodell effizient

automatisiert zu bewerkstelligen, wurde pyRiskRobot um nachfolgend aufgeführte Funktionen ergänzt, die auf den in Abschnitt 2.2 beschriebenen Kernfunktionalitäten aufbauen:

- Eine Funktion, die eine SUSA-Ausgabedatei und eine Liste von Parameter-Namen einliest sowie eine Liste von Dictionaries, also Zuordnungen von Schlüsselbegriffen zu Objekten ausgibt. Dabei entspricht jedes Dictionary einem von SUSA erzeugten Parameterset, jeder Parametername wird also einem Parameterwert zugeordnet.

- Eine Funktion, die die Namen aller Einträge in einer Datenbanktabelle aussucht, die ein bestimmtes Muster erfüllen. Hier können variabel die einzulesende Datenbank, der Name der Datenbanktabelle, wie auch die Definition des zu suchenden Musters als ‘regular expression‘ vom Nutzer gesetzt werden. Im Fall einer GVA-Analyse wurden durch diese Funktion alle Ereignisse herausgesucht, die mit einer bestimmten Bezeichnung enden.

- Eine Funktion, die eine Liste von Parametern auf einen fixen Wert setzt. Diese übernimmt ein Objekt der Klasse pyRiskRobot als Schnittstelle zum PSA-Modell sowie eines der Parameterset-Dictionaries und eine Liste von Parameternamen. Alle Parameter, die sowohl in der Liste als auch im PSA-Modell und im Dictionary vorkommen, werden auf die im Dictionary festgelegten Werte gesetzt. So kann der Nutzer mittels der übergebenen Liste von Parameternamen bestimmen, welche der im Dictionary angegebenen Parameter von ihrem Wert im PSA-Modell auf den zugehörigen Wert im Dictionary gesetzt werden sollen.

- Eine Funktion, welche die von SUSA ausgegebenen Parametersets nutzt, um die zugehörigen GVA-Ausfallwahrscheinlichkeiten zu berechnen.

Die geeignete Modellierung gemeinsam verursachter Ausfälle ist auch in Bezug auf Multi-Modul-Anlagen relevant, wie sie oft bei SMRs vorgesehen sind. So hat man, falls n der Module gleich aufgebaut sind, viele gleiche Komponenten. Dies kann dazu führen, dass diese Komponenten in den verschiedenen Modulen in etwa zeitgleich ausfallen, (z. B. bei übergreifenden Einwirkungen wie Erdbeben oder bei Notstromfällen), was wiederum parallele Handmaßnahmen in mehreren Modulen erfordern würde und so z. B. zu einer Überlastung des Personals und damit zu einer erhöhten Fehlerrate führen könnte.

Diese Ergänzungen zu pyRiskRobot erlauben es, zukünftig vergleichbare Analysen für GVA schneller durchführen zu können.

2.5 Schnittstelle zur Netzwerkanalyse

Um ausgewählte übergreifende Einwirkungen auf ein komplexes Anlagensystem abzubilden, wird ein allgemeiner Ansatz genutzt, welcher auf der Identifikation relevanter, voneinander abhängiger Räume bzw. Raumbereiche (Englisch: Compartments) basiert, die von den jeweils zu berücksichtigenden Einwirkungen unterschiedlich betroffen sind. Eine PSA für übergreifende Einwirkungen, auch als Hazards PSA bezeichnet, lässt sich dabei durch Nutzung des GRS-Werkzeugs pyRiskRobot möglichst effizient mittels einer automatisierten Integration der abgeleiteten relevanten Raumbereiche (als Hazard Compartments, kurz HCs, bezeichnet) in die Fehlerbäume für die von der jeweiligen Einwirkung betroffenen sicherheitsrelevanten Bauteile, Systeme und Komponenten erstellen. Für übergreifende Einwirkungen lassen sich die Zusammenhänge zwischen Komponenten über diese Hazard Compartments modellieren. Bei einer Brand-PSA sind die Hazard Compartments häufig die physischen Räume, in denen sich Komponenten befinden, da oft angenommen wird, dass ein Feuer in einem Raum auf alle Komponenten in diesem Raum einwirkt. Ausnahmen hiervon sind z. B. Kabel, bei denen im Falle eines Versagens auch Komponenten außerhalb des Raums betroffen wären. Bei einer Überflutungs-PSA könnte man noch die Höhe, auf der sich Komponenten befinden, für die Definition der jeweiligen Hazard Compartments hinzuziehen.

Um die Abhängigkeiten zwischen Komponenten korrekt zu modellieren, müssen die Abhängigkeiten zwischen verschiedenen Hazard Compartments berücksichtigt werden.

In der in /BER 20/, Abschnitt 3.1.2 vorgestellten Studie zur Untersuchung der Raumabhängigkeiten für Brandausbreitungen verschiedener Modellierungstiefe wurde dargelegt, dass die Pfadtiefe bei der Berücksichtigung der Brandausbreitung für das vorgelegte Beispiel beschränkt werden muss, um die Umsetzbarkeit der PSA zu gewährleisten. Wenn alle Abhängigkeiten mittels pyRiskRobot modelliert wurden, dann wurde das entstehende Modell so komplex, dass die mit einem Standard-Werkzeug durchgeführte PSA nicht mehr konvergierte. In solchen Fällen ist es wichtig, in einem vorbereitenden Schritt zu entscheiden, welche Abhängigkeiten im Modell mitgenommen werden müssen, um das Ergebnis der PSA nicht zu sehr zu verfälschen und dennoch das Konvergieren zu ermöglichen. Um dem Nutzer diesen Präprozessierungsschritt zu erleichtern, wurde als Erweiterung zu pyRiskRobot ein Modul zur Netzwerkanalyse eingeführt /BER 20/, /BER 20b/. Dieses Modul unterstützt die Darstellung der Netzwerke von Hazard Compartments und deren Analyse mittels Netzwerkmaßen. Im Forschungs- und Entwicklungsvorhaben RS1596 wurde die Anbindung zwischen pyRiskRobot und

dem Netzwerkanalyse-Modul vertieft, und eine Schnittstelle zwischen beiden geschaffen.

In das pyRiskRobot-Modul zur Modellierung von übergreifenden Einwirkungen wurde eine Funktion eingefügt, welche die Abhängigkeit zwischen miteinander verbundenen Hazard Compartments in ein sogenanntes ‘Dictionary‘ im Node-Link-Format /HAG 08/ umwandelt. Als verbundene Hazard Compartments werden dabei benachbarte Räume bzw. Anlagenbereiche bezeichnet, zwischen denen sich Auswirkungen übergreifender Einwirkung (z. B. Hitze, heiße Rauchgase oder Ruß bei einem Brand, Wasser bei einer Überflutung etc.) über nicht verschlossene Verbindungen bzw. Öffnungen (wie Türen, Tore, Gitteroste, nicht geschlossene Klappen oder Ventile in Lüftungskanälen, nicht abgedichtete Kabeldurchführungen in Wänden oder Decken etc.) von einem Hazard Compartment in ein weiteres Hazard Compartment ausbreiten können. Das Dictionary enthält dabei zwei Einträge, einen für die Nodes, also die Knoten, d. h. im Fall von pyRiskRobot die Hazard Compartments, und einen für die Links, also die (ggf. offenen) Verbindungen zwischen den Hazard Compartments. Für die Nodes ist wiederum eine Liste mit Dictionaries gespeichert, jedes Dictionary besteht dabei aus einem Eintrag für die ID des jeweiligen Hazard Compartments und einem für die Wahrscheinlichkeit des Auftretens einer Einwirkung (bzw. Einwirkungskombination) von innen oder außen in diesem Hazard Compartment. Für die Links ist ebenfalls eine Liste von Dictionaries gespeichert, wobei jedes Dictionary aus den folgenden drei Einträgen besteht:

- einem für die ‘source‘, also die ID des Hazard Compartments, von dem sich die Auswirkungen der zu betrachtende übergreifende Einwirkung oder Einwirkungskombination ausbreiten,
- einem für das ‘Target‘, also die ID des Hazard Compartments, in welches hinein die Ausbreitung erfolgt, und
- einem ‘weight‘, d. h. der Ausbreitungswahrscheinlichkeit zwischen den beiden Hazard Compartments.

Diese zusammengefügten Dictionaries können leicht im json-Dateiformat gespeichert werden, die NetworkX Python Library /HAG 08/, auf welcher die Netzwerkanalyse aufbaut, kann diese einlesen und daraus ein Netzwerk erstellen. Mittels NetworkX kann auch ein bereits vorhandenes Netzwerk in diesem Format gespeichert werden. Hierbei ist anzumerken, dass die Dictionaries für die Nodes, also die einzelnen Hazard Compartments, um weitere Informationen ergänzt werden können, ohne dass dies die Les-

barkeit des Formats beeinträchtigt. Auf diese Weise können mittels der Netzwerkanalyse Informationen zu den einzelnen Nodes hinzugefügt werden, z. B. ihre Relevanz für eine PSA oder die für das zugehörige Hazard Compartment relevante Ausbreitungstiefe.

Zwei Funktionen wurden für die Aufgabe des Informationsaustauschs in pyRiskRobot eingefügt: eine Funktion zum Erstellen dieser Node-Link Dictionaries und eine Funktion, die ein Dictionary im Node-Link-Format einliest und es wieder in ein für das pyRiskRobot verarbeitbares Format übersetzt. Die Netzwerkanalyse wurde ebenfalls um die entsprechenden Funktionen ergänzt.

Zusätzlich wurde pyRiskRobot um eine Funktion ergänzt, die es erlaubt, statt einer Modellierung der Abhängigkeiten für alle Hazard Compartments bis zu einer festen Ausbreitungstiefe verschiedene Ausbreitungstiefen für die jeweiligen Hazard Compartments über ein Dictionary einzulesen und zu modellieren. So könnten die in einem Vorbereitungsschritt über die Netzwerkanalyse gewonnenen Informationen über die optimale Ausbreitungstiefe für die jeweiligen Hazard Compartments genutzt werden, um für jedes Hazard Compartment die Abhängigkeiten optimiert zu modellieren.

Somit wurde der Informationsaustausch zwischen pyRiskRobot und der Netzwerkanalyse ermöglicht. Die Anwendung dieser neuen Schnittstelle ist im Detail in Abschnitt 2.6 beschrieben.

2.6 Beispielhafte Nutzung von Netzwerkmaßen zur Vorbereitung einer Brand-PSA-Modellierung

Im Forschungs- und Entwicklungsvorhaben RS1556 /BER 20b/ wurde die Netzwerkanalyse zur qualitativen Betrachtung der miteinander verknüpften Hazard Compartments genutzt. Basierend auf der in /BER 17/ vorgestellten Studie zur benötigten Ausbreitungstiefe zur Modellierung einer Brand-PSA wurde im Vorhaben RS1596 nach einer Kombination von Netzwerkmaßen gesucht, welche in der Lage ist, die benötigte Ausbreitungstiefe zufriedenstellend vorherzusagen.

Dazu wurden zunächst die in /BER 20/ und /BER 20b/ vorgestellten Netzwerkmaße und Netzwerkvisualisierungsmöglichkeiten erweitert. Zuerst wurde die auf NetworkX /HAG 08/ basierenden Netzwerkvisualisierungen testweise um eine interaktive Visualisierung, basierend auf der Python-Bibliothek plotly Dash /HOS 19/, erweitert. Dash ist eine Python-Bibliothek, die es dem Nutzer erlaubt, interaktive Dashboards mit Python zu

erstellen. Die nachfolgende Abb. 2.6 zeigt die Visualisierung eines Raumnetzwerkes mit NetworkX.

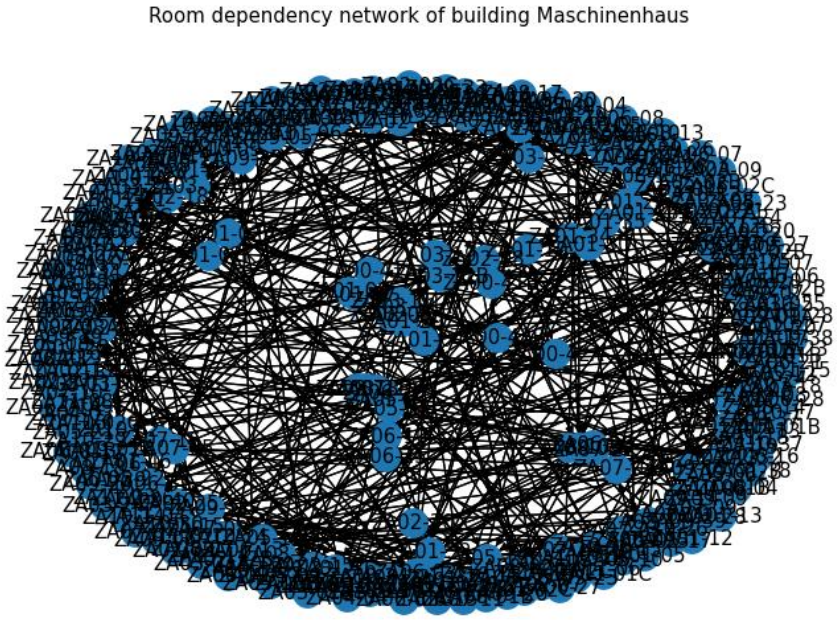

Abb. 2.6 Visualisierung von Raumabhängigkeiten, basierend auf der in /BER 20/, Abschnitt 3.1 vorgestellten Studie, erstellt mit NetworkX und einem Spring Layout

Das obere Eingabefeld in Abb. 2.7 erlaubt es, einen Knoten des Netzes (Raum) auszuwählen, um auf ihn zu fokussieren, das untere Eingabefeld bestimmt, welches Netzwerkmaß genutzt wird, um die Farbgebung der Knoten festzulegen.

Die nachfolgende Abb. 2.7 zeigt die Visualisierung des gleichen Raumnetzwerkes wie Abb. 2.6, jedoch so, wie sie testweise mit der plotly Dash-Bibliothek zusammen mit der visdcc-Erweiterung /BOW 22/ implementiert wurde. Hierbei ist zu beachten, dass in der Darstellung von plotly Dash für jede Verbindung zwei Pfeile dargestellt sind. Dies korrespondiert mit der möglicherweise unterschiedlichen Ausbreitungswahrscheinlichkeit, zum einen von Raum A zu Raum B und zum anderen von Raum B zu Raum A.

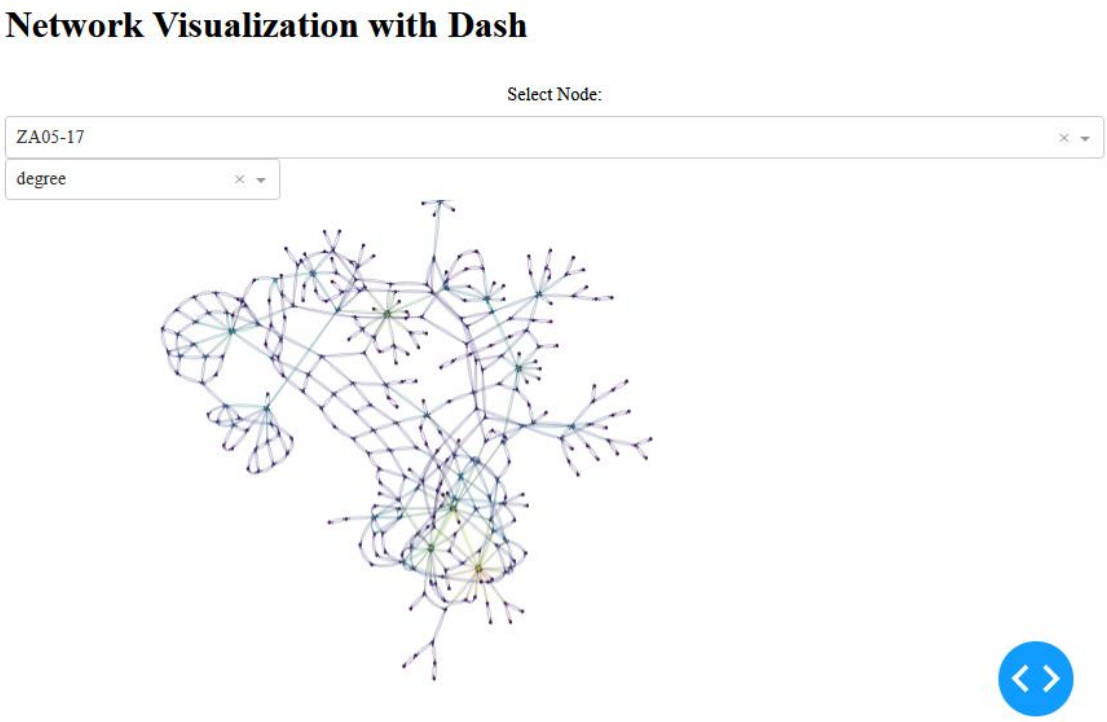

Abb. 2.7 Visualisierung der Raumabhängigkeiten mit der plotly Dash-Bibliothek und der visdcc-Erweiterung /BOW 22/

Die in Abb. 2.7 gezeigte Visualisierung ist hinsichtlich der Nutzerfreundlichkeit und Performance noch nicht ausgereift, erlaubt jedoch bereits

- einzelne Knoten im Netz zu greifen und zu bewegen, so dass man einen besseren Überblick über die Zusammenhänge bekommt,
- einzelne Knoten herauszusuchen und den Ausschnitt auf diese zu zentrieren (obere Eingabefläche),
- interaktiv die Farbe der Knoten anhand von knoten-basierten Netzwerkmaßen zu ändern (untere Eingabefläche).

Der letzte der vorgenannten Punkte war auch der Grund für die testweise Implementierung. Über das in plotly Dash dargestellte Netz ist es möglich, schnell zu erkennen, wie die verschiedenen, knotenbasierten Netzwerkmaße im Netz verteilt sind. Auch lässt sich sehr schnell erkennen, welche Hazard Compartments nur schwach oder gar nicht mit dem Hauptnetz verbunden sind.

Die Communities für das betrachtete Raumnetz bei der Brand-PSA sind in den Abbildungen Abb. 2.8 bis Abb. 2.11 dargestellt. In Abb. 2.9 sieht man einen Zoom auf zwei

der Greedy-Modularity-Communities. Dabei wird sichtbar, wie der Algorithmus schwach verbundene Bereiche in Communities einsortiert.

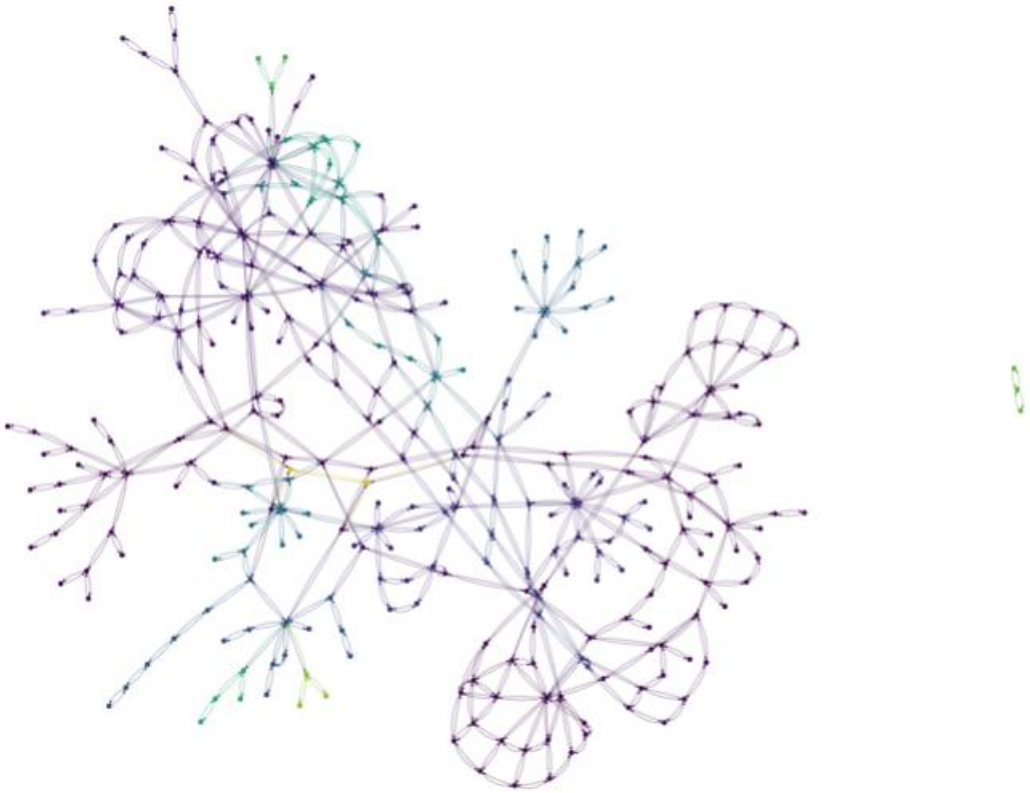

Abb. 2.8 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Greedy-Modularity Community der Knoten bestimmt wird

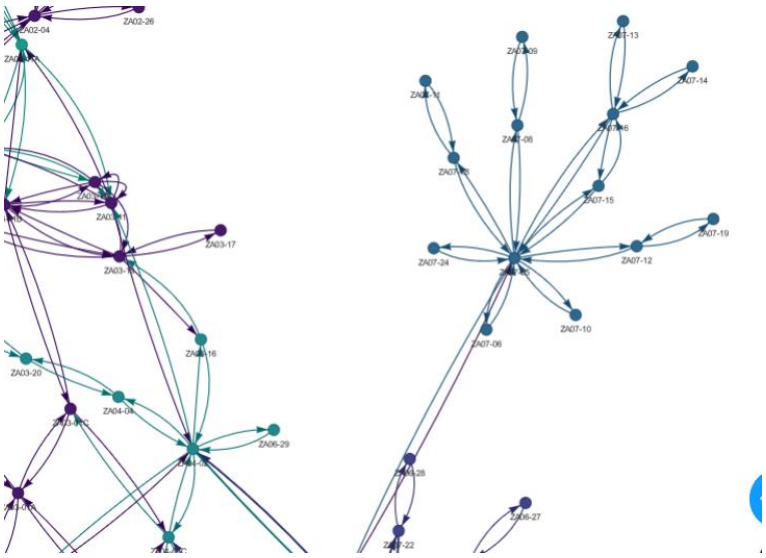

Abb. 2.9 Zoom auf zwei Greedy-Modularity Communities aus Abb. 2.8

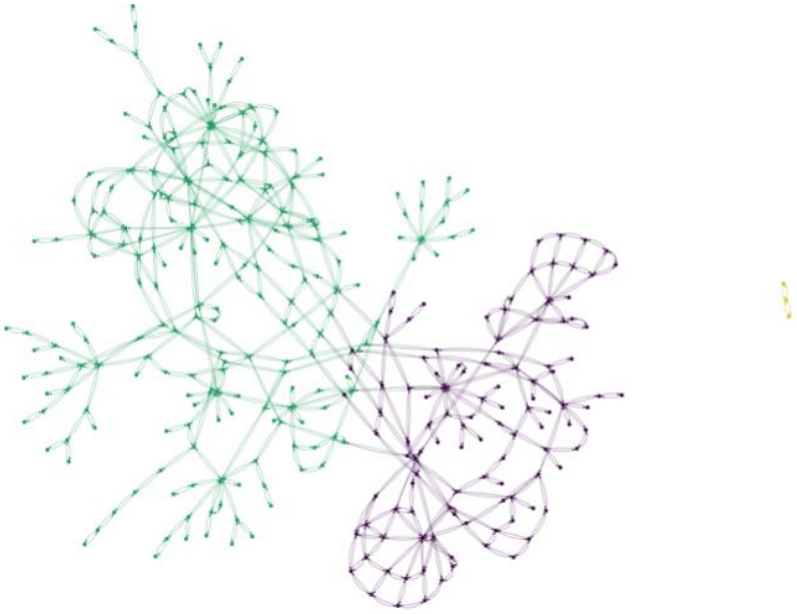

Abb. 2.10 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Girvan-Newman Community der Knoten bestimmt wird

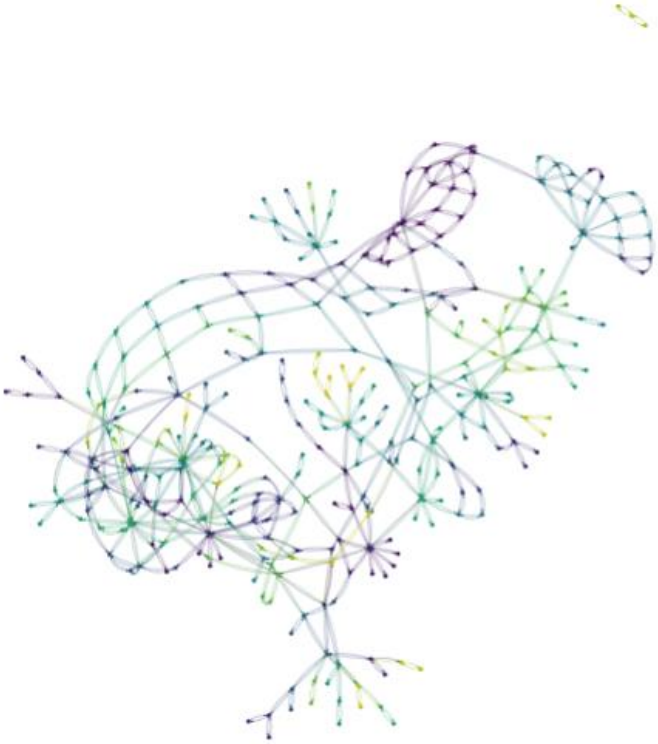

Abb. 2.11 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Label-Propagation Community der Knoten bestimmt wird

In einem nächsten Schritt wurden die Standard-Netzwerkmaße Gradzentralität, Nähezentralität und Zwischenzentralität um die Subset-Zwischenzentralität /ZAK 14/ und die Informationen über sogenannte Communities ergänzt. Die Subset-Zwischenzentralität erweitert das Konzept der Zwischenzentralität dadurch, dass nur ein Teil der Knoten bei der Bestimmung dieser Zentralität berücksichtigt wird. Communities im Bereich der Netzwerkanalyse sind verallgemeinernd Gruppen von Knoten im Netz, die miteinander dichter verwoben sind als mit Knoten außerhalb der Gruppe. In Hinblick auf eine Analyse übergreifender Einwirkungen ist hier die These, dass sich übergreifende Einwirkungen mit einer höheren Wahrscheinlichkeit innerhalb einer Gruppe ausbreiten, bevor sie zu Bereichen außerhalb der Gruppe propagieren. Drei verschiedene Community-Algorithmen wurden miteinander verglichen:

- Greedy-Modularity-Communities /RUS 24/: Dieser Algorithmus basiert auf dem Konzept der Modularität. Die Modularität misst hierbei, wie stark die Verbindungen in einer Gruppe im Vergleich zu den Verbindungen zwischen verschiedenen Gruppen sind. Hierbei beginnt jeder Knoten als seine eigene Gruppe, benachbarte Gruppen werden in jedem Schritt so miteinander verbunden, dass die Modularität des Netzes maximiert wird. Die Gewichtung jeder Verbindung ist durch die Ausbreitungswahrscheinlichkeit zwischen zwei Hazard Compartments bestimmt /NEW 10/.
- Girvan-Newman-Communities /GIR 02/: Dieser Algorithmus entfernt Schritt für Schritt Verbindungen vom Netz, und zwar typischerweise die Verbindungen mit der höchsten Zwischenzentralität. Dabei handelt es sich um die Verbindungen, die Teil der meisten kürzesten Pfade im Netz sind. Die sich ergebenden Unternetze (Communities) können dann beispielsweise in einem Dendogramm dargestellt werden.
- Asynchronous-Label-Propagation-Communities /RAG 07/: Bei diesem Algorithmus wird als Ausgangspunkt jedem Knoten ein Label zugewiesen. Dann werden asynchron für jeden Knoten die Label so angepasst, dass jeder Knoten das Label erhält, welches am häufigsten in seiner Nachbarschaft vertreten ist.

Eine mögliche zukünftige Studie könnte untersuchen, wie gut Ausfallwahrscheinlichkeiten vorhergesagt werden könnten, wenn nur die Ausbreitung zwischen Compartments in einer Community betrachtet werden, falls sich mindestens ein Ausgangsraum bzw. Anlagenbereich (Source Compartment) in der Community befindet.

Es ist erkennbar, dass zumindest mit den Standardeinstellungen der Girvan-Newman-Algorithmus größere Communities erzeugt als der Greedy-Modularity-Algorithmus und

der Asynchronous-Label-Propagation-Algorithmus kleinere Communities als der Greedy-Modularity-Algorithmus. Der Girvan-Newman Algorithmus unterteilt das Netz in drei Communities, wobei eine dieser Communities vollständig vom Rest des Netzes getrennt ist.

Die Darstellung in /RAG 07/ zeigt, dass die Community-Verteilung des Asynchronous-Label-Propagation-Algorithmus weniger intuitiv ist als die der anderen beiden Algorithmen, d. h. die Grenzen zwischen den Communities entsprechen weniger denen, die man aufgrund der unterschiedlichen Vernetzungsdichte erwarten würde. Da der Asynchronous-Label-Propagation-Algorithmus zudem keine weiteren Parameter außer dem Gewicht der Verbindungen und dem Random-Seed (d. h. dem Wert, mit dem ein Zufallszahlengenerator initialisiert wird) hat, ist zu überlegen, ob er für die Zwecke der Vorbereitung einer PSA geeignet ist. Der Girvan-Newman-Algorithmus erlaubt es, statt der Zwischenzentralität ein anderes Netzwerkmaß zu verwenden, um die wichtigsten Verbindungen zu identifizieren. Auch hier wäre eine weitere, vergleichende Studie zielführend, um zu prüfen, inwieweit diese Anpassungsmöglichkeit genutzt werden kann, um sinnvolle Communities zu finden.

In den nächsten Schritten der Analyse wird für jedes Hazard Compartment die Größe der zugehörigen Girvan-Newman-Modularität-Community als weiteres Merkmal für die Vorhersage der benötigten Ausbreitungstiefe verwandt. Zusätzlich zu den Communities und den Standardnetzwerkmaßen wurden weitere Maße berechnet und in der Vorhersage der notwendigen Ausbreitungstiefe verwandt. Dies erfolgte insbesondere, um der Tatsache Rechnung zu tragen, dass im Fall übergreifender Einwirkungen einige Räume als Entstehungsorte (Source Compartments) dienen können. Die Ausbreitungswahrscheinlichkeit zwischen den verschiedenen Source Compartments und den Target Compartments (d. h. benachbarten Räumen bzw. Raumbereichen mit sicherheitstechnisch relevanten SSC, in die sich die Auswirkungen der betrachteten Einwirkung (in diesem Fall des Brandes) ausbreiten können), ist daher für die Modellierung der Brandausbreitungswahrscheinlichkeit in einem Hazard Compartment von Bedeutung. Folgende Maße wurden für jedes Hazard Compartment berechnet:

- die Länge des kürzesten Pfades zum nächstgelegenen Source Compartment,
- die Länge des kürzesten Pfades zum stärksten Source Compartment im Netz.
- Subset-Zwischenzentralität: Bis zu einem Abstand von fünf Hazard Compartments vom betrachteten Hazard Compartment aus wird die Ausbreitungswahrscheinlichkeit

aller in dieser Tiefe entdeckten Source Compartments aufsummiert und durch die Zahl aller im Netz vorhandenen Source Compartments geteilt. Hierfür wurde das in /ZAK 14/ definierte Konzept der Subset-Zwischenzentralität für die Zwecke der Vorbereitung einer PSA für übergreifende Einwirkungen (Hazards PSA) angepasst. Hazard Compartments mit einer hohen Subset-Zwischenzentralität sind damit besonders eng mit potenziellen Source Compartments verbunden. Hier ist zu erwarten, dass für diese Hazard Compartments eine geringere Ausbreitungstiefe benötigt wird.

Folgende Schritte wurden bei der Analyse durchgeführt:

1. Die in /BER 20/ durchgeführte Analyse für eine Brand-PSA wurde zunächst reproduziert. Abb. 2.12 zeigt die normierte Ausfallwahrscheinlichkeit für die einzelnen Gebäude und Räume.

2. Wie in Abb. 2.12 zu erkennen ist, gibt es für die verschiedenen Hazard Compartments jeweils eine typische Ausbreitungstiefe, ab der die weitere Erhöhung der betrachteten Ausbreitungstiefe zu keiner signifikanten Änderung in der Brandausbreitungswahrscheinlichkeit für das Hazard Compartment führt. Hier wurde das Kriterium angewandt, dass eine signifikante Erhöhung zwischen zwei Ordnungen immer dann zu erkennen ist, wenn zwischen diesen Ordnungen die normierte Ausfallwahrscheinlichkeit um einen Wert von mehr als 0,5 ansteigt. Diese notwendige Ausbreitungstiefe wurde für jedes Hazard Compartment bestimmt und diese Information im json-Dateiformat gespeichert.

3. Die in /BER 20/ betrachteten Abhängigkeiten zwischen Hazard Compartments wurden in das in Abschnitt 2.6 beschriebene Node-Link-Format überführt.

4. Die Node-Link-Dictionaries wurden in Dateien im json-Format gespeichert und mittels der Nutzung der NetworkX-Bibliothek im Networksanalyzer wieder eingelesen.

5. Folgende Netzwerkmaße wurden für jedes Hazard Compartment im Networksanalyzer berechnet:

   a) Nähezentralität (Umkehrwert der Summe der Länge aller kürzesten Pfade zu allen anderen Hazard Compartments) geteilt durch die Zahl aller erreichbarer Hazard Compartments minus 1; je größer die Nähezentralität, desto besser ist das Hazard Compartment auch mit allen anderen (erreichbaren) Hazard Compartments vernetzt;

   b) Größe der zugeordneten Girvan-Newman-Community;

c) Länge des kürzesten Pfades zum nächstgelegenen Source Compartment im Netz;

d) Länge des kürzesten Pfades zum stärksten Source Compartment im Netz;

e) Subset-Zwischenzentralität.

6. Als Methode des maschinellen Lernens (ML) wurde ein Random Forest Regression-Algorithmus /BRE 01/ genutzt. Dieser hat im Vergleich zu anderen Methoden des maschinellen Lernens den Vorteil, dass die genutzten Variablen nicht vorher normiert werden müssen und dass er zudem im Vergleich zu einfachen Entscheidungsbäumen /WU 08/ weniger zum Übertrainieren neigt und dennoch, im Gegensatz zu neuronalen Netzen, eine bessere Nachvollziehbarkeit gegeben ist. Auch wird beim Random-Forest-Algorithmus automatisch die Wichtigkeit der einzelnen, betrachteten Parameter ermittelt. Dies dient auch dem Verständnis des Anwenders. Die über den Random-Forest-Algorithmus bestimmte Bedeutung eines Parameters ist dabei die mittlere Reduktion der ‘Impurity‘ pro Verzweigung (die Wahrscheinlichkeit eines neuen Datenpunktes falsch klassifiziert zu werden) durch diesen Parameter gemittelt über alle Entscheidungsbäume eines Random Forest.

7. Zwei Methoden wurden angewandt, um die Aussagekraft des erarbeiteten Modells zu testen.

   a) Es wurde ein Cross-Validation Score berechnet /HAS 09/. Dazu werden die Hazard Compartments jeweils in n verschiedene Blöcke unterteilt (hier wurde n = 5 gewählt), jeweils vier Blöcke dienen dem Training und ein Block dem Test, so kann insgesamt n-mal die Vorhersagekraft des Modells getestet werden. Die Zahl der Blöcke wurde als Kompromiss zwischen einer ausreichend großen Zahl an Tests und einer ausreichend großen Zahl an Hazard Compartments pro Block gewählt. Der Cross-Validation Score ist dann der Mittelwert der Scores dieser Tests.

   b) Der Cross-Validation Score hat beim betrachteten Modell den Nachteil, dass die Hazard Compartments miteinander verbunden sind und so eine willkürliche Unterteilung in n-Blöcke dazu führen kann, dass der Testblock nicht unabhängig vom Trainingsblock ist. Aus diesem Grund wurde das Modell zusätzlich einmal anhand eines Gebäudes (und damit einem Netz) trainiert und für die Vorhersage für ein anderes Gebäude verwendet.

8. Über die in Abschnitt 2.5 beschriebene Schnittstelle kann die Information, für welches Hazard Compartment eine Modellierung bis zu welcher Tiefe erfolgen sollte,

an pyRiskRobot zurückgeleitet werden. Dort kann die Information dazu genutzt werden, für jedes Hazard Compartment nur die angegebene Ausbreitungstiefe zu modellieren.

Vier verschiedene Raumnetze, basierend auf vier verschiedenen Gebäuden, werden im Folgenden verglichen. Die folgenden Erkenntnisse konnten dabei für das Raumnetz 1 gewonnen werden:

Die Random Forest Wichtigkeit der betrachteten Parameter ist:

- Minimale Distanz zum nächsten Hazard Source Compartment: 0,79,
- Subset-Zwischenzentralität: 0,11,
- Nähezentralität: 0,07,
- Minimale Distanz zur stärksten Quelle im Netz: 0,03,
- Größe der zugehörigen Girvan-Newman-Modularität-Community: 0,004.

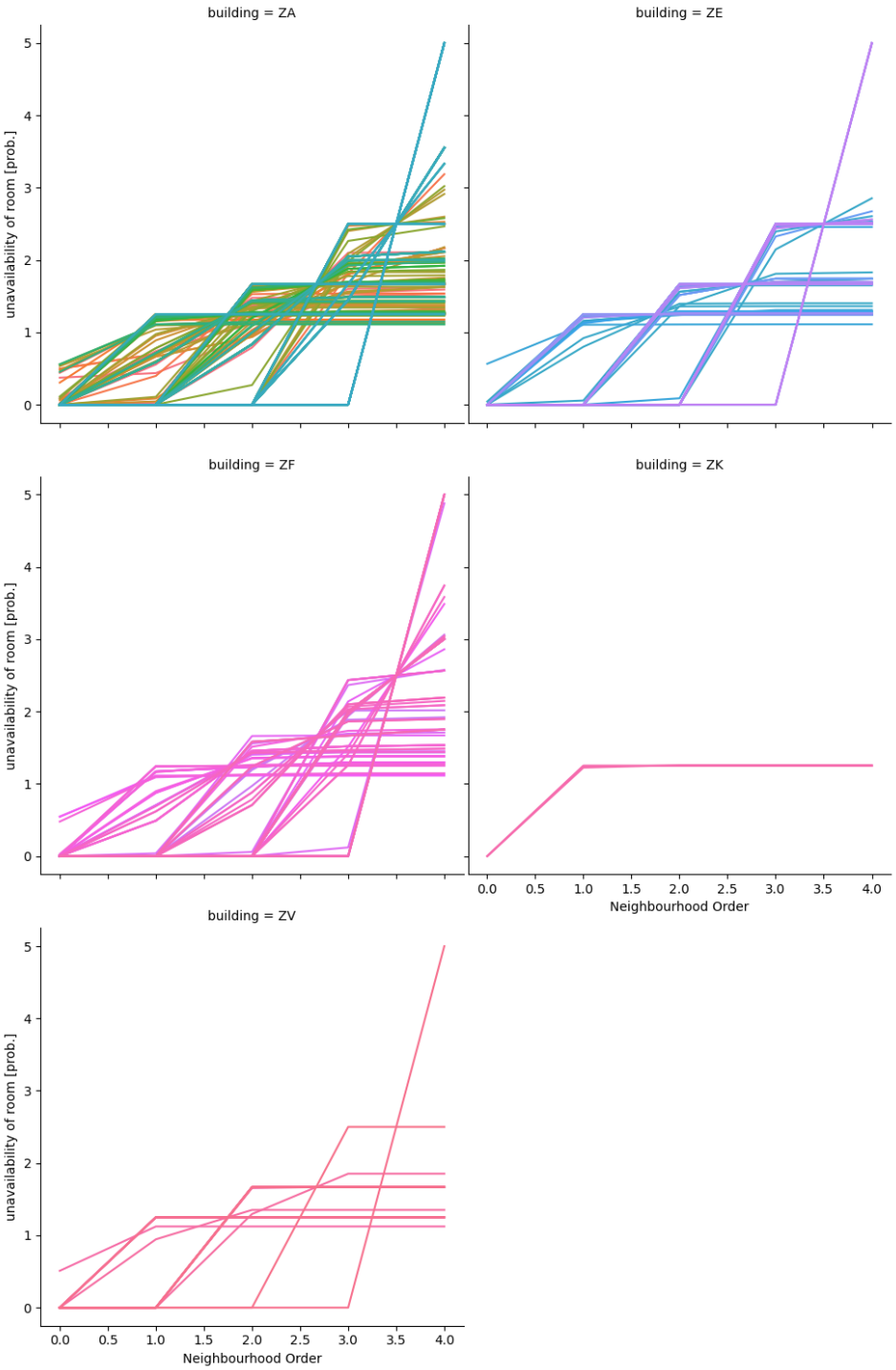

Abb. 2.12 Normalisierte Brandwahrscheinlichkeit von Brandräumen (d. h. Nichtverfügbarkeit durch Brand) pro Anlagengebäude in Abhängigkeit der modellierten Nachbarschaftsordnungen 0 bis 4 der spezifischen Brandausbreitungspfade, basierend auf /BER 17/

Die wichtigste Größe, um zu entscheiden, bis zu welcher Pfadtiefe die Abhängigkeiten eines Hazard Compartments modelliert werden sollten, ist damit der Abstand zum

nächsten Source Compartment. Die Subset-Zwischenzentralität hat die zweithöchste Wichtigkeit, da auch sie ein Maß dafür ist, wie gut ein Hazard Compartment mit möglichen Source Compartments verbunden ist. Die Pearson-Korrelation zwischen den verschiedenen betrachteten Größen ist in Abb. 2.13 dargestellt.

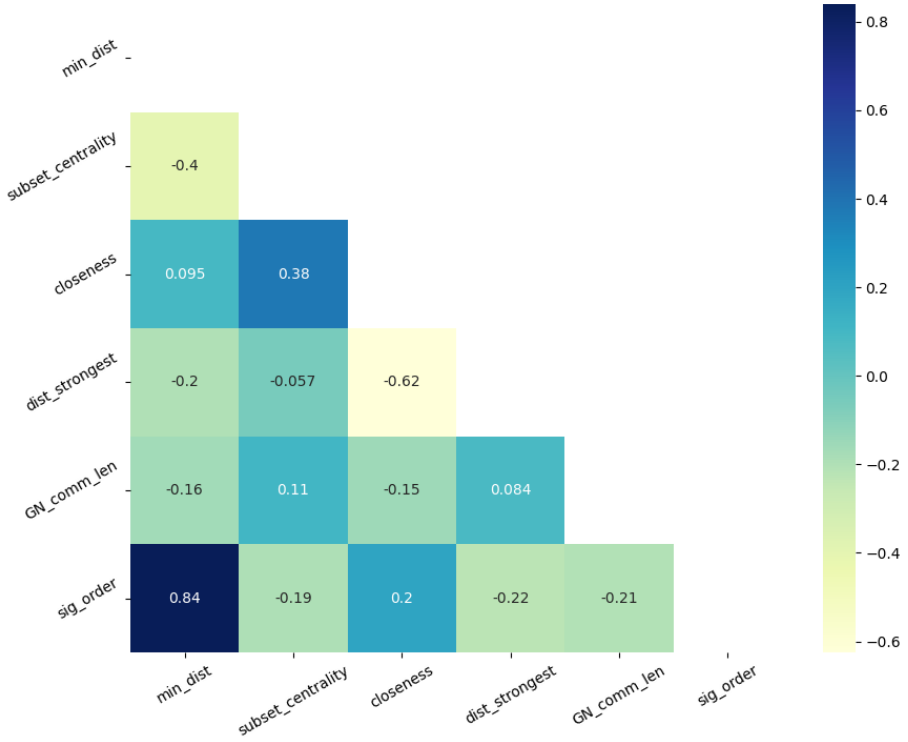

Abb. 2.13 Korrelation zwischen den verschiedenen betrachteten Größen zur Bestimmung der für die Analyse erforderlichen Ausbreitungstiefe

Es ist zu erkennen, dass der Abstand zum nächstgelegenen Source Compartment (min_dist) eine nicht unerhebliche Korrelation (- 0,4) zur Subset-Zwischenzentralität (subset_centrality) aufweist. Die Wichtigkeit der Subset-Zwischenzentralität könnte sich daher aus der Wichtigkeit der korrelierten minimalen Distanz zum nächsten Source Compartment ableiten. Die hohe Wichtigkeit dieser Distanz spiegelt sich in dem Korrelationskoeffizienten von 0,84 zu der benötigten Ausbreitungstiefe (sig_order). Der Korrelationskoeffizient der anderen Größen zur Ausbreitungstiefe liegt bei etwa 0,2. Der Wert mit der dritthöchsten Wichtigkeit ist die Nähezentralität (closeness), diese ist ein reines Netzwerkmaß ohne Berücksichtigung der Besonderheit von Source Compartments. Da sie eine geringe Korrelation zur minimalen Distanz zum nächsten Source Compartment aufweist, kann sie das sich ergebende Modell um neue Informationen ergänzen.

Der R2-Wert (d. h. das Bestimmtheitsmaß, Englisch: Coefficient of Determination), der über den Cross-Validation Score herausgefunden wurde, beträgt 0,76. Dies kann man so verstehen, dass sich 76 % der Variation der Daten über das gefundene Modell erklären lassen. Wie oben beschrieben, ist jedoch der über Cross-Validation bestimmte Wert nur beschränkt aussagekräftig, da die genutzten Hazard Compartments miteinander verbunden sind. Wird stattdessen das Raumnetz 1 genutzt, um ein Modell zu trainieren, und dies dann für andere Raumnetze verwendet, so ergeben sich folgende Resultate:

- Für das Raumnetz 2 ergibt sich ein R2-Wert von 0,69.
- Für das Raumnetz 3 ergibt sich ein R2-Wert von 0,66.
- Für das Raumnetz 4 ergibt sich ein R2-Wert von 0,30.

Während sich also für die Raumnetze 2 und 3 eine gewisse Vorhersagekraft feststellen lässt, so ist diese für Raumnetz 4 nicht in dem gleichen Maße gegeben. Das bedeutet, dass noch weitere Faktoren zu berücksichtigen sind, die den Unterschied zwischen den Gebäuden besser modellieren können.

Zusätzlich wurde die Vorhersage auch basierend allein auf der minimalen Distanz zum nächsten Source Compartment getroffen, um so den Nutzen der anderen Größen besser abschätzen zu können. Hier ergab sich, wiederum für das Raumnetz 1 als Testdatensatz, ein Cross-Validation R2-Score von 0,75 und gebäudeübergreifend veränderte sich der R2-Wert:

- von 0,69 auf 0,65 für das Raumnetz 2,
- von 0,66 auf 0,80 für das Raumnetz 3,
- von 0,30 auf 0,24 für das Raumnetz 4.

Die Berücksichtigung weiterer Parameter verbessert also die Vorhersagekraft des Modells.

Wird statt des Raumnetzes 1 das Raumnetz 4 zum Trainieren genutzt, ergeben sich folgende Resultate:

- Für das Raumnetz 1 ergibt sich ein R2-Wert von 0,48.
- Für das Raumnetz 2 ergibt sich ein R2-Wert von 0,74.
- Für das Raumnetz 3 ergibt sich ein R2-Wert von 0,31.

Auch hier ist zu erkennen, dass das Modell noch verbessert werden muss, um auch gebäudeübergreifend einsetzbar zu sein. Hierzu wäre es insbesondere sinnvoll, sich diejenigen Hazard Compartments anzusehen, für die die Vorhersage einen großen Abstand zum tatsächlichen Wert hat, um zu erkennen, welche Faktoren die Unterschiede zwischen Vorhersage und tatsächlichem Wert maßgeblich beeinflussen.

Insgesamt lässt sich festhalten, dass mit der Schnittstelle zwischen pyRiskRobot und der Berücksichtigung weiterer Größen die ersten Schritte erfolgt sind, um die Vorbereitung einer PSA für übergreifende Einwirkungen zu vereinfachen.

2.7 Neue Werkzeuge zur Visualisierung und Analyse multipler Hazards als mehrdimensionale Netzwerke und geeignete Netzwerkmaße

Im Forschungs- und Entwicklungsvorhaben RS1556 wurde ein generischer Ansatz erarbeitet, indem eine mehrdimensionale Netzwerktopologie verwandt wurde, um die Raumabhängigkeiten mehrerer Hazards in einem Anlagensystem zu repräsentieren /BER 20/. Hierbei entspricht jeweils eine Netzwerkdimension (Ebene) dem Raumabhängigkeitsnetzwerk für die jeweilige übergreifende Einwirkung.

Darauf aufbauend wurde im Vorhaben RS1596 eine Literaturstudie durchgeführt, um zu ermitteln, welche der verschiedenen, frei verfügbaren Bibliotheken sich am besten in den Networksanalyzer einbinden lassen, um die Visualisierung und Analyse mehrdimensionaler Netzwerke zu ermöglichen. Hierbei wurden unter anderem /OCK 23/ und /SKR 19/ betrachtet. Die nachfolgende Abb. 2.14 zeigt einen solchen Vergleich verschiedener Multi-Layer Netzwerkanalysen und Visualisierungsbibliotheken aus /SKR 19/.

Anhand dieser Tabelle wird deutlich, dass sowohl Pymnet /KIV 14/ als auch Py3plex /SKR 19/ mögliche Kandidaten für eine Nutzung durch den Networksanalyzer sind, da sie sich in ihren Funktionalitäten ergänzen. So ist Py3plex auf große Netzwerke optimiert und ermöglicht deren effiziente Darstellung und Analyse, während Pymnet auch die Darstellung von dreidimensionalen Netzwerken erlaubt und somit sinnvoll für eine anschaulichere Visualisierung der verschiedenen Arten der Auswirkung eines Hazards wäre. Zusätzlich zu den Informationen in Abb. 2.14 ist zu beachten, dass von den in Abb. 2.13 aufgelisteten Bibliotheken nur MuxViz und Py3plex nicht unter einer GNU General Public License (GPL) veröffentlicht werden. GPL /FSF 07/ ist eine offene Lizenz, die ein strenges „Copyleft“ Prinzip umsetzt. Ein strenges ‘Copyleft‘-Prinzip bedeutet, dass die

Kombination eines eigenen Programms mit einem Programm unter solch einer Lizenz ebenfalls nur unter dieser strengen Lizenz weitergegeben werden kann. Die Einbindung GPL-basierter Bibliotheken würde also erfordern, dass auch der Networksanalyzer nur unter der GPL-Lizenz weitergegeben werden darf, also als open-source-Programm. Dies ist zum jetzigen Zeitpunkt nicht der Fall, daher sollen Bibliotheken unter GPL und somit auch Pymnet nicht genutzt werden. Dies führt dazu, dass Py3plex ausgewählt wurde, um den Networksanalyzer um die Möglichkeit zur Analyse und Visualisierung mehrdimensionaler Netzwerke zu erweitern.

Table 1 Comparison of multilayer network analysis and visualization libraries

	Py3plex	Pymnet (Kivelä et al. 2014)	MuxViz (De Domenico et al. 2015)	MultinetX (Amato et al. 2017)
Core features				
Programming language	Python 3 and C (via Numpy and Cython)	Python 3 and C (via Numpy)	R	Python 3 and C (via Numpy)
Basic statistics	✓	✓	✓	✓
Visualization of large networks	✓	-	✓	-
Visualization in 3D	-	✓	-	✓
Aggregation/decomposition	✓	✓	✓	-
Random graph generators	✓	✓	✓	✓
Adjacency matrix manipulation	✓	✓	✓	✓
[3pt] Additional features				
[3pt] Node classification	✓	-	-	-
Isomorphisms	✓	✓	-	-
Community detection	✓	-	✓	-
GUI version	-	-	✓	-
Tensor manipulation	✓	✓	✓	✓
Node ranking	✓	✓	✓	✓
Semantic topology enrichment	✓	-	-	-
Temporal networks	-	-	✓	✓
Network embedding	✓	-	-	-

[4]For a comprehensive overview of visualization tools, we refer the reader to the recent survey (McGee et al. 2019).

Abb. 2.14 Vergleich der verschiedenen, verfügbaren Bibliotheken zur Visualisierung mehrdimensionaler Netzwerke mittels einer Tabelle aus /SKR 19/

Py3plex bietet u. a. die Möglichkeit, für das modellierte, mehrdimensionale Netzwerk Communities (vgl. Abschnitt 2.6) zu identifizieren. In Hinblick auf eine PSA-Modellierung der Brandausbreitungswahrscheinlichkeit in verschiedenen Hazard Compartments können Informationen über die zugehörige Community genutzt werden, um zu entscheiden,

welche anderen Hazard Compartments bei der Modellierung einer Ausbreitung der Auswirkungen eines Hazards mitberücksichtigt werden sollten.

Zusätzlich wurden folgende Netzwerkmaße untersucht, um Aussagen über die Bedeutung eines Hazard Compartments für verschiedene Arten von Hazards und vor allem für bedingt voneinander gleichzeitig auftretende Arten von Hazards geben zu können:

- Entropie der multiplexen Gradzentralität: Diese Entropie ist ein Maß, das nur für multiplexe Netzwerke bestimmt werden kann, also mehrdimensionale Netzwerke, die in allen Dimensionen die gleichen Knoten haben. Die multiplexe Gradzentralität erlaubt es, die Unterschiede in der Vernetzung eines Hazard Compartments über die verschiedenen Hazard-Typen zu quantifizieren. Ein Hazard Compartment mit geringer Entropie und hoher Gradzentralität wäre demnach wahrscheinlich ein wichtiges Compartment, das bei der Modellierung der Hazard Ausbreitung berücksichtigt werden sollte.
- Multiplex-Beteiligungskoeffizient: Der multiplexe Beteiligungskoeffizient ist ebenfalls ein Maß, das nur für multiplexe Netzwerke bestimmt werden kann. Wie die oben genannte Entropie gibt auch der Beteiligungskoeffizient an, ob die verschiedenen Verknüpfungen des Hazard Compartments hauptsächlich in einer Ebene auftreten oder ob das Hazard Compartment über verschiedene Ebenen gleichmäßig gut vernetzt ist. Der Multiplex-Beteiligungskoeffizient ist 0, wenn alle Verbindungen eines Knotens nur in einer Ebene liegen und 1, wenn der Knoten in allen Ebenen gleichviele Verbindungen hat.
- Durchschnittlicher Clustering-Koeffizient: Der durchschnittliche Clustering-Koeffizient pro Ebene ist ein Maß dafür, wie stark die Knoten dieser Ebene dazu tendieren, Cluster, also besonders gut vernetzte Bereiche im Netz zu bilden. Wobei der Cluster-Koeffizient eines Knotens der Anteil der realisierten geschlossenen Dreiecke im Netz ist, die diesen Knoten enthalten, im Vergleich zu allen möglichen geschlossenen Dreiecken basierend auf der Gradzentralität des Knotens.

Um die verschiedenen Anwendungen der mehrdimensionalen Netzwerkanalyse zu verdeutlichen, wurde der Networksanalyzer um zwei zusätzliche Jupyter-Notebooks ergänzt. Diese zeigen exemplarisch, wie Py3plex zur Netzwerkanalyse und zur Visualisierung von multiplen Hazards genutzt werden kann. Beispielsweise wird in diesen Notebooks die Entropieverteilung des multiplexen Grades untersucht wie in Abb. 2.15 dargestellt.

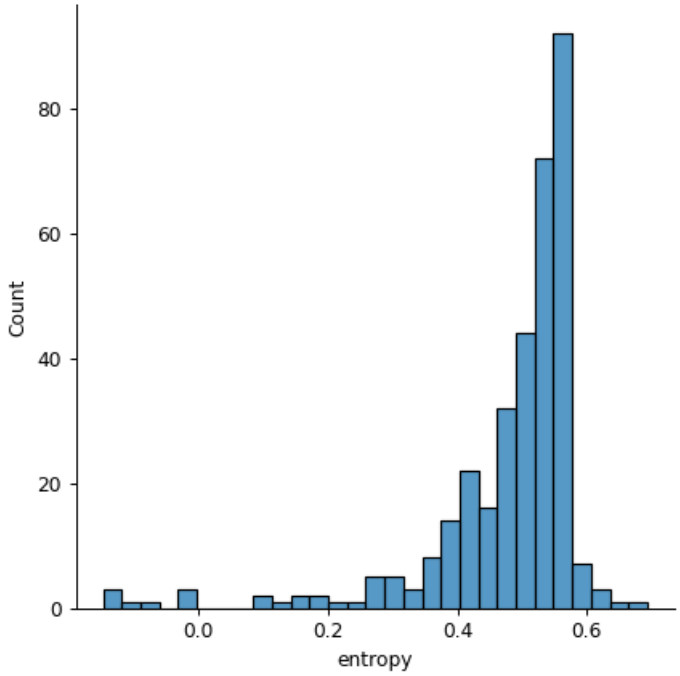

Abb. 2.15 Verteilung der Entropie eines multiplexen Grades, berechnet unter Nutzung der Py3plex-Bibliothek für das in Abschnitt 2.7 beschriebene mehrdimensionale multiplexe Netzwerk

Eines dieser Notebooks basiert auf dem gleichen Hazard Compartment-Netzwerk wie die in Abschnitt 2.6 vorgestellte Analyse. Um aus dem zugehörigen Hazard Compartment-Netzwerk für eine Brand-PSA ein mehrdimensionales, multiplexes Netzwerk zu erzeugen, wurden Auswirkungen eines weiteren Hazards, z. B. Wasserausbreitung durch ein Folgeereignis (anlageninterne Überflutung) des ersten Hazards (anlageninterner Brand) angenommen. Für die exemplarische Visualisierung wurden für diesen Hazard die Raumabhängigkeiten zufallsbasiert geändert, auch wurden für einige Hazard Compartments Abhängigkeiten zwischen den Ebenen geschaffen. Dies könnte z. B. dann der Fall sein, wenn durch Löschwassereinwirkung nach einem Brand in einem anderen Hazard Compartment eine Wassereinwirkung auf elektrische Einrichtungen zu einem weiteren Brandereignis führen würde. Hazard Compartments, für die eine solche Möglichkeit nicht auszuschließen ist, würden dann die Verbindung zwischen den beiden Ebenen darstellen.

Abb. 2.16 zeigt eine Visualisierung des mit Py3plex berechneten mehrdimensionalen Netzwerks und die Verteilung des multiplexen Entropiegrades über das Netzwerk. Eine Studie ähnlich der in Abschnitt 2.6 beschriebenen Untersuchung wäre sinnvoll, um die

wichtigsten Maße für eine PSA bei Kombinationen mehrerer übergreifender Einwirkungen bestimmen zu können.

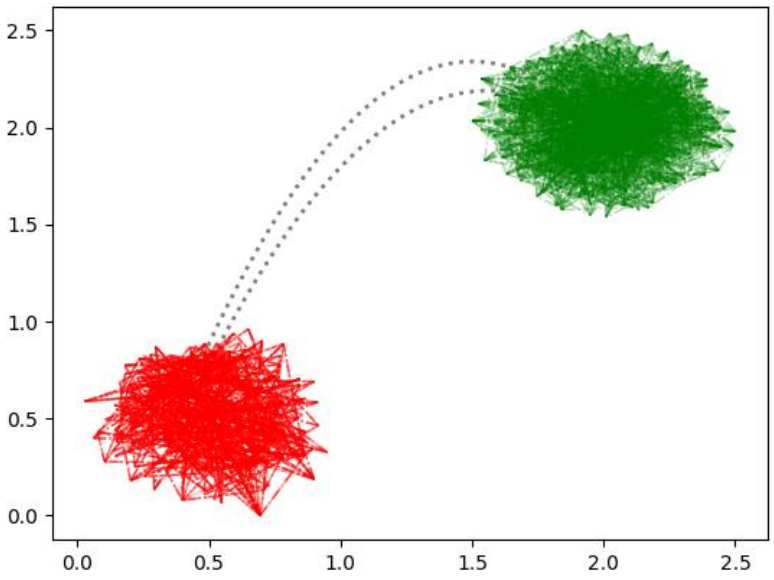

Abb. 2.16 Abbildung eines zweidimensionalen Netzwerkes mit py3plex /SKR 19/

2.8 Entwicklungen zur Modellierung und Analyse von Unfallabläufen als dynamische Netzwerke

In diesem Vorhaben wurden verschiedene Möglichkeiten zur Modellierung und Analyse von Unfallabläufen als dynamische Netzwerke in einer Literaturrecherche miteinander verglichen. Die bekanntesten dynamischen Methoden, die es erlauben, Unfallabläufe als Graph-Strukturen darzustellen, sind:

- Petri-Netze /REU 90/,
- dynamische Flussgraphen /GUA 18/,
- Ereignis-Sequenz-Diagramme /KEC 06/,
- Markov-Modelle /PES 91/ und
- die GO-FLOW Method /MAT 10/.

Die verschiedenen Ansätze unterscheiden sich in der Definition von Knoten und Kanten der dynamischen Netzwerke. Ziel des Vorhabens war es, herauszufinden, wie man vor-

handene PSA-Anlagenmodelle als dynamische Netzwerke darstellen könnte, um so neue Erkenntnisse aus den vielfältigen PSA-Datenbanken ziehen zu können.

Die vorgenannten Methoden eignen sich unterschiedlich gut für diesen Ansatz. Nach teilig sind insbesondere Methoden, bei denen viele unterschiedliche Arten von Knoten gebraucht werden, um den Unfallablauf zu modellieren. Dies trifft auf die GO-FLOW Method zu, bei der es knapp ein Dutzend verschiedener Knotentypen (Operatoren) gibt. Petri-Netze und dynamische Flussgraphen stehen mit ihren Transition- und Transferboxen ebenfalls einer automatisierten Übersetzung entgegen.

Ein gutes, PSA-nahes Beispiel für die Dynamisierung einer PSA ist in der von EDF entwickelten Software ANDROMEDA /FRI 14/ zu finden. ANDROMEDA hat insgesamt das Ziel, eine PSA zu ‘modularisieren‘, so dass die einzelnen Bestandteile einer PSA leicht wiederverwendet werden können. Dies ist ähnlich zu der in pyRiskRobot genutzten Behandlung von Auswirkungen eines Hazards (Ereignisse), bei denen auch vorgefertigte Fehlerbäume an den gewünschten Stellen eingefügt werden können. ANDROMEDA erlaubt es, Fehler- und Ereignisbäume, basierend auf der Eingabe eines Systems über den sogenannten Modeleditor, auf Grundlage der Programmiersprache Figaro /BOW 22/ automatisiert zu erstellen. Eine spezielle Erweiterung von ANDROMEDA erlaubt die Übersetzung des eingegebenen Systems in Ereignis-Sequenz-Diagramme. Diese Diagramme sind in der Art der Knoten den existierenden Fehler- und Ereignisbäumen ähnlich und sollten die Übersetzung vereinfachen.

Ereignis-Sequenz-Graphen sind Weiterentwicklungen von Ereignisablaufdiagrammen, sie zeigen die Pfade von einem auslösenden Ereignis zum Erfolgsendzustand an. Jeder Knoten in diesem Graph entspricht dabei einem Ereignis. Jedes Ereignis kann durch einen Systemgraph ersetzt werden. In diesem wird das Zusammenwirken verschiedener Systemkomponenten zur Behebung des auslösenden Systemfehlers modelliert. Im Gesamtgraph ergibt sich ein Erfolgsszenario, solange es mindestens einen nicht unterbrochenen Pfad vom auslösenden Ereignis zum Erfolgsendzustand gibt. In ANDROMEDA ist über den Model-Editor eine Übersetzung eines über die kb3-Software (siehe /BOW 22/) erstellten Systemmodells in Ereignis-Sequenz-Graphen implementiert, jedoch keine Übersetzung ausgehend von Ereignis- und Fehlerbäumen.

Der ANDROMEDA Model-Editor-Ansatz wurde in das kommerzielle PSA-Werkzeug RiskSpectrum® übernommen und dient dort als Basis des ModelBuilders. RiskSpectrum® untersucht dabei ebenfalls Ansätze, um die erzeugten Systeme zu dynamisieren

/KRC 22/, indem eine Monte-Carlo-Simulation verwendet werden kann, um die Ausfallraten zu berechnen. In /KRC 22/ wird beschrieben, wie diese Funktionalität verwendet werden kann, um beispielsweise die Ausfallraten von Systemen zu bestimmen, in denen digitale leittechnische Systeme (z. B. intelligente Voting-Systeme) genutzt werden. Hierbei ist zu beachten, dass der bei ANDROMEDA und dem ModelBuilder verwendete Ansatz nur dann vollumfänglich funktionieren kann, wenn das ganze System modelliert wird. Anders als bei pyRiskRobot wird dabei ein System von Grund auf neu modelliert, während in pyRiskRobot auf existierende Datenbanken zugegriffen wird.

Es hat sich herausgestellt, dass für eine existierende Datenbank keine vollständige, eindeutige Übertragung in Ereignissequenzen möglich ist. Es ist allerdings möglich, Fehlerbäume in sogenannte Reliability Block Diagrams /SIG 21/ zu übertragen. Reliability Block Diagramme wiederum können ebenfalls durch Monte-Carlo-Simulationen dynamisiert werden. Eine solche Übertragung wurde beispielhaft mit pyRiskRobot durchgeführt. Unter Nutzung der agentenbasierten Simulationssoftware NetLogo /WIL 99/ wurden eine Simulation durchgeführt und die erhaltene Ausfallrate mit der berechneten verglichen. Da für diese Übersetzung allerdings schon einige Werkzeuge existieren, z. B. ReliaSoft BlockSim@, wird empfohlen für solche Studien, sollten diese gewünscht sein, optimierte Algorithmen und Methoden zu verwenden. Grundsätzlich stellt dies ein Thema dar, bei dem eine Zusammenarbeit mit anderen Einrichtungen sinnvoll wäre, die bereits über systembasierte Ansätze für die Erstellungen von Fehlerbäumen verfügen.

3 Zusammenfassung und Ausblick

Die vorgestellten Weiterentwicklungen präsentieren und diskutieren Möglichkeiten, die Integration von übergreifenden Einwirkungen in eine bestehende probabilistische Sicherheitsanalyse (PSA) komplexer technischer Systeme noch effizienter zu machen. Dazu wurden zwei von der GRS entwickelte Softwarewerkzeuge erweitert und verbessert. Dabei handelt es sich zum einen um die Software pyRiskRobot, welche es erlaubt, komplexe Fehlerbaumtopologien automatisiert und nachverfolgbar zu erweitern, zum anderen handelt es sich um den sogenannten Networksanalyzer, der es ermöglicht komplizierte Raumabhängigkeiten übergreifender Einwirkungen als Netzwerke darzustellen und zu analysieren.

So wurde die pyRiskRobot-Software gewartet, um mit der Weiterentwicklung der eingebundenen python-Bibliotheken Schritt zu halten. Auch wurden die von pyRiskRobot zur Verfügung gestellten Funktionengruppen erweitert, um die Modellierung der Unsicherheiten von GVA-Bestimmungen zu erleichtern. Im Zuge dieses Vorhabens wurde pyRiskRobot zudem so erweitert, dass es zusätzlich zu RiskSpectrum® basierten Fehlerbäumen auch SAPHIRE basierte Fehlerbäume erweitern und modifizieren kann. Diese Erweiterung führte zu der Idee, pyRiskRobot zu nutzen, um Fehlerbäume von RiskSpectrum® in SAPHIRE und umgekehrt zu überführen. Diese Konvertierung ermöglicht eine Überprüfung vorhandener PSA-Modelle, die unabhängiger von der Software ist, die zur Erstellung der Modelle verwendet wird.

Eine erste Anwendung erfolgte im Forschungs- und Entwicklungsvorhaben 4721R1530 bei der Modellierung einer anlageninternen Überflutung des Reaktorgebäude-Ringraums eines Druckwasserreaktors /BER 24/. Mittels dieses Anwendungsfalls konnten wichtige Unterschiede zwischen einer Fehlerbaummodellierung in RiskSpectrum® im Vergleich zur Modellierung in SAPHIRE identifiziert werden. So wurde festgestellt, dass für die in RiskSpectrum® genutzten Exchange Events zur Verknüpfung verschiedener Fehlerbäume kein eindeutiges Äquivalent in SAPHIRE existiert. Die Vervollständigung der begonnenen Arbeit zur Übertragung von Fehlerbäumen zwischen verschiedenen PSA-Softwaremodellen wäre ein wichtiger Schritt in Richtung Modellunabhängigkeit einer PSA und eine sinnvolle Erweiterung von pyRiskRobot in zukünftigen Vorhaben.

Weitere, im Vorhaben RS1596 durchgeführte Arbeiten an pyRiskRobot sind die Umstrukturierung des Kerns von pyRiskRobot, um die oben beschriebenen Erweiterungen

sinnvoll in das Gesamtkonzept einzugliedern und die Wartbarkeit von pyRiskRobot aufrechtzuerhalten.

Zudem wurde der Teil von pyRiskRobot, der für die Modellierung übergreifender Einwirkungen zuständig ist, um eine Schnittstelle zum Networksanalyzer ergänzt. Die entsprechende Schnittstelle wurde auch im Networksanalyzer hinzugefügt. Unter Nutzung der neu eingefügten Schnittstelle konnte eine Analyse durchgeführt werden, welche eine probabilistische Analyse für übergreifende Einwirkungen exemplarisch vorführt. Im Rahmen dieser Analyse wurde der Networksanalyzer um neue Netzwerkmaße zur Bestimmung von Communities ergänzt. Die mehrdimensionale Netzwerkanalyse wurde ebenfalls um weitere Maße ergänzt.

Eine exemplarische Nutzung dieser Netzwerkmaße zur Präprozessierung einer PSA für eine Überlagerung mehrerer übergreifender Einwirkungen als Einwirkungskombinationen in der PSA analog zu der für eine einzelne Einwirkung durchgeführte probabilistische Analyse stellt eine sinnvolle Erweiterung dieses Analysewerkzeugs in möglichen zukünftigen Vorhaben dar.

Zur Nutzung dynamischer Netzwerke zur Ableitung von Risikoaussagen wurden eine Literaturrecherche und exemplarische Machbarkeitsstudien durchgeführt. Dabei stellte sich heraus, dass die Modellierung einer PSA als dynamisches Netzwerk dann am sinnvollsten ist, wenn das Modell neu erstellt wird, und dass dabei zu empfehlen ist, auf bereits vorhandene Analysewerkzeuge, wie den RiskSpectrum® ModelBuilder, aufzubauen.

Insgesamt erweist sich pyRiskRobot immer wieder als sinnvolles Werkzeug, um effizient eine erweiterte PSA entweder für übergreifende Einwirkungen durchzuführen oder eine PSA zur Bestimmung von Unsicherheiten durch gemeinsam verursachte Ausfälle zu nutzen. Die in diesem Vorhaben erfolgten Weiterentwicklungen tragen dazu bei, zukünftige PSA noch effizienter durchführen und auch zu komplexeren Problemstellungen, wie Kombinationen unterschiedlicher Arten von übergreifenden Einwirkungen, belastbare Aussagen treffen zu können.

So ist beispielswiese bei der Entwicklung von PSA-Modellen für neuartige Reaktorkonzepte ein schneller Informationsaustausch sinnvoll, um die Modelle gezielter entwickeln zu können. Dementsprechend sollte das GRS-Werkzeug pyRiskRobot zur automatisierten Bearbeitung von PSA-Modellen so erweitert werden, dass vollständige Fehlerbäume

eines PSA-Anlagenmodells, z. B. für einen multi-modularen SMR, zwischen mehreren PSA-Programmen wie RiskSpectrum® und SAPHIRE übertragen werden können. Weiterhin sollte ein Konzept zur Übertragung von Ereignisbäumen bereitgestellt werden, wodurch perspektivisch gesamte PSA-Modelle übertragen werden könnten.

Die Bewertung der Auswirkungen übergreifender Einwirkungen ist ein relevanter Aspekt probabilistischer Sicherheitsbewertungen, auch für sogenannte Small Modular Reactors (SMRs) mit mehreren Reaktormodulen an einem Standort. Ein weiteres Ziel zukünftiger Weiterentwicklungen besteht jedoch darin, das Werkzeug pyRiskRobot so anzupassen, dass die Modellierung übergreifender Einwirkungen effizienter als bisher hinsichtlich der Fehlerbaumgröße durchgeführt werden kann. Sowohl die Automatisierung als auch die Erhöhung der Übersichtlichkeit im PSA-Modell sollen eine fehlerfreie Modellierung begünstigen und die Überprüfung der Modelle erleichtern.

Literaturverzeichnis

/BER 16/ Berner, N., J. Herb: Generic framework for the automated integration of impacts from hazards in PSA models, Risk, Reliability and Safety: Innovation Theory and Practice, in: Walls, Revie and Bedford (Eds.): Proceedings of the 26th European Safety and Reliability Conference 2016 (ESREL 2016), Glasgow, Großbritannien, 2016.

/BER 17/ Berner, N., J. Herb: Weiterentwicklung der Methodik zur automatisierten Integration übergreifender Einwirkungen in PSA-Modelle der Stufe 1. GRS–454, ISBN 978-3-946607-36-6, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH, Köln, 2017, https://www.grs.de/sites/default/files/publications/grs-454.pdf.

/BER 19/ Berner, N., M. Utschick: Network-based analysis of hazard dependency patterns prior to the automated integration into PSA models, in: Beer, M., and E. Zio (Eds.): Proceedings of the 29th European Safety and Reliability Conference (ESREL 2019), Hannover, 22.-26. September 2019, http://rpsonline.com.sg/proceedings/9789811127243/html/0429.xml.

/BER 20/ Berner, N., J. Scheuer: A Multidimensional Network Approach for Analyzing Hazard Impact Dependencies, in: Proceedings of the 30th European Safety and Reliability Conference (ESREL 2020) & 15th Probabilistic Safety Assessment and Management Conference (PSAM15), Venedig, Research Publishing, Singapore, 2020.

/BER 20a/ Berner, N., et al.: Analyse von redundanzübergreifenden Ausfällen in der elektrischen Energieversorgung von Kernkraftwerken, GRS–538, ISBN 978-3-947685-23-3, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH, Köln, Juli 2000, https://www.grs.de/sites/default/files/2021-07/GRS-538.pdf.

/BER 20b/ Berner, N.: Automated Integration and Network-based Analysis of Hazard Impacts within Probabilistic Safety Analysis (PSA) Models, Technical Report, GRS–565, ISBN 978-3-947685-50-9, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH, Köln, Juli 2020, https://www.grs.de/sites/default/files/publications/grs-565.pdf .

/BER 23/ Berchtold, F., J. Stiller.: PSA Model Approaches for Asymmetries in the Electrical Power Supply for Nuclear Power Plants, in: Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023), Southampton, Großbritannien, 2023, https://www.rpsonline.com.sg/proceedings/esrel2023/pdf/P184.pdf.

/BER 23a/ Berchtold, F., T. Eraerds: Dynamic and Classic PSA Plant Model Comparison for a Plant Internal Flooding Scenario, in: Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023), Southampton, Großbritannien, 2023, https://rpsonline.com.sg/proceedings/esrel2023/pdf/P221.pdf.

/BER 23b/ Berchtold, F., et al.: Vertiefte Untersuchungen zu speziellen Fragestellungen zur Bewertung von redundanzübergreifenden Ausfällen in der Elektrotechnik, GRS-695, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH, Köln, 2023. https://www.grs.de/de/aktuelles/publikationen/grs-695.

/BER 24/ Berchtold, F., T. Eraerds: Dynamic Modelling of A Nuclear Power Plant Internal Flooding Scenario Under Severe Weather Conditions, Paper No.1265, in: Proceedings of 17th International Conference on Probabilistic Safety Assessment and Management & Asian Symposium on Risk Assessment and Management (PSAM17&ASRAM2024), Sendai, Myagi, Japan, 7.-11. Oktober 2024.

/BOW 22/ Bow, J.: Dash Core Components for Visualization, last update 2022, https://github.com/jimmybow/visdcc.

/BRE 01/ Breiman, L.: Random Forests, in: Machine Learning 45, S. 5-32, 2001, https://doi.org/10.1023/A:1010933404324.

/BRU 17/ Brück, B., et al.: Probabilistic Analysis of Electrical Faults Affecting Multiple Redundant Trains of the Electrical Power Supply System of Nuclear Power Plants, in: Proceedings of ANS PSA 2017 International Topical Meeting on Probabilistic Safety Assessment and Analysis, Pittsburgh, PA, USA, September 24-28, 2017, on CD-ROM, American Nuclear Society, LaGrange Park, IL, USA, 2017.

/FAK 05/ Facharbeitskreis Probabilistische Sicherheitsanalyse für Kernkraftwerke, Methoden zur probabilistischen Sicherheitsanalyse für Kernkraftwerke, BfS-SCHR-37/05, 2005Facharbeitskreis (FAK) Probabilistische Sicherheitsanalyse für Kernkraftwerke: Methoden zur probabilistischen Sicherheitsanalyse für Kernkraftwerke, Stand: August 2005, BfS-SCHR-37/05, ISBN 3-86509-414-7, Bundesamt für Strahlenschutz (BfS), Salzgitter, Germany, Oktober 2005, https://doris.bfs.de/jspui/bitstream/urn:nbn:de:0221-201011243824/1/BfS_2005_SCHR-37_05.pdf.

/FAK 16/ Facharbeitskreis (FAK) Probabilistische Sicherheitsanalyse für Kernkraftwerke: Methoden und Daten zur probabilistischen Sicherheitsanalyse für Kernkraftwerke, Stand: Mai 2015, BfS-SCHR-61/16, Bundesamt für Strahlenschutz (BfS), Salzgitter, Germany, September 2016, https://doris.bfs.de/jspui/bitstream/urn:nbn:de:0221-2016091314090/3/BfS-SCHR-61-16.pdf.

/FRI 14/ Friedlhuber, T.: Model Engineering in a Modular PSA, Thesis, Ecole Polytechnique, EDF R&D, Palaiseau, Frankreich, 2014, https://pastel.hal.science/tel-01110825.

/FSF 07/ Free Software Foundation: GNU General Public License, version 3, 2007, https://www.gnu.org/licenses/gpl-3.0.

/GIR 02/ Girvan, M., M, E. J. Newman: Community structure in social and biological networks, Proc. Natl. Acad. Sci. Vol. 99, S. 7821-7826, 2002, https://www.pnas.org/doi/abs/10.1073/pnas.122653799.

/GUA 18/ Guarro, S., M. Yau: Dynamic Flowgraph Methodology (DFM) Modelling of Nuclear and Advanced Technology System Risk and Reliability Scenarios, Advanced Concepts in Nuclear Energy Risk Assessment and Management, World Scientific, S. 353-425, 2018, https://www.worldscientific.com/doi/epdf/10.1142/9789813225619_0011.

/HAG 08/ Hagberg, A. A., D. A. Schult, P. J. Swart: Exploring network structure, dynamics, and function using NetworkX, in: Varoquaux, G., T. Vaught and J. Millman (Eds.): Proceedings of the 7th Python in Science Conference (SciPy2008), S. 11–15, Pasadena, CA, USA, 2008, https://www.osti.gov/biblio/960616.

/HAS 09/ Hastings, T., et al.: The elements of statistical learning, S. 219-259, Springer, New York, NY, USA, 2009.

/HER 12/ Herb, J., et al.: Fault tree auto-generator: How to cope with highly redundant systems, in: 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference 2012 (PSAM11 ESREL 2012), pp. 2704 – 2713, ISBN: 978-1-62276-436-5, Curran Associates, Inc., Red Hook, NY, USA, 2012.

/HOM 23/ Homann, M., et al.: Forschung zur aktualisierten Bewertung von Gemeinsam verursachten Ausfällen, Auswertung der Betriebserfahrung in deutschen Anlagen im Zeitraum 2011 bis 2018, GRS–676, ISBN 978-3-949088-67-4, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH, Köln, August, 2023, https://www.grs.de/sites/default/files/2023-08/GRS-676.pdf.

/HOS 19/ Hossain, S.: Visualization of Bioinformatics Data with Dash Bio, in: Proceedings of the 18th Python in Science Conference (SciPy2019), S. 126-133, 2019, Austin, TX, USA. July 8-14, 2019, ISSN: 2575-9752**,** https://doi.org/10.25080/issn.2575-9752**.**

/IAE 21/ International Atomic Energy Agency (IAEA): Protection against Internal Hazards in the Design of Nuclear Power Plants, Specific Safety Guide, IAEA Safety Standards Series No. SSG-64, STI/PUB/1947, ISBN 978-92-0-116121–5, Wien, August 2021, https://www-pub.iaea.org/MTCD/publications/PDF/Pub1947_web.pdf.

/KEC 06/ Kecher, C.: UML 2.0: Das umfassende Handbuch, Galileo Computing, Bonn, 2006.

/KIV 14/ Kivelä, M., et al.: Multilayer networks, in: Journal of Complex Networks, Volume 2, Issue 3, S. 203-271, September 2014, https://doi.org/10.1093/comnet/cnu016.

/KRC 22/ Krcal, P., et al.: Control Logic Encoding using RiskSpectrum ModelBuilder, Paper No. 131 in: Proceedings of 16th International Probabilistic Safety Assessment and Management Conference (PSAM16), Honolulu, HI, USA, 2022, https://www.iapsam.org/PSAM16/papers/PA131-PSAM16.pdf.

/MAT 10/ Matsuoka, T.: GO-FLOW methodology - Basic concept and integrated analysis framework for its applications, in: Nuclear Safety and Simulation, Vol. 1, Number 3, September 2010, http://mats121.world.coocan.jp/IJNS_GO-FLOW(2010).pdf.

/MYE 15/ Myers, J., R. Copeland: Essential SQLAlchemy: Mapping Python to Databases (2nd Edition), ISBN 9781491916551.O'Reilly Media, Boston, MA, USA, 2015.

/NEW 10/ Newman, M. E. J.: Networks: An Introduction, S. 224, ISBN 9780199206650, Oxford University Press, Oxford, Großbritannien, 2010.

/OCK 23/ Ocklind, C.: A Comparative Analysis of Multilayer Network Software, degree project, University of Uppsala, Uppsala, Sweden, 2023.

/PES 91/ Peschke, J: Erweiterung des Programmpakets für Zuverlässigkeitsberechnungen mit Markov-Modellen um eine Option zur Durchführung von Unsicherheits- und Sensitivitätsanalysen, GRS-A-1743, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, Garching, 1991.

/PYT 24/ PyTables Developers Team: PyTables: Hierarchical Datasets in Python, https://www.pytables.org/, letzter Zugriff: 08.08.2024.

/RAG 07/ Raghavan, U. N., A. Réka, K. Soundar: Near linear time algorithm to detect community structures in large-scale networks, Physical Review E 76.3, 2007, https://journals.aps.org/pre/abstract/10.1103/PhysRevE.76.036106.

/REU 90/ Reutenauer, C.: The Mathematics of Petri Nets, ISBN 0135618878, Prentice Hall International, Hoboken, NJ, USA, 1990, https://dl.acm.org/doi/abs/10.5555/79071.

/RIS 24/ RiskSpectrum®: Software RiskSpectrum PSA, https://www.riskspectrum.com/riskspectrum-psa, zuletzt abgerufen am 21.02.2024.

/RUS 24/ Rustamaji, H. C., et al.: Community detection with Greedy Modularity disassembly strategy, Sci Rep. 14, 4694, 2024, https://doi.org/10.1038/s41598-024-55190-7.

/SIG 21/ Signoret, J. P., A. Leroy: Reliability Block Diagrams (RBDs), Springer Series in Reliability Engineering, in: Reliability Assessment of Safety and Production Systems, S. 195-208, Springer, Sedzère, France, 2021, https://doi.org/10.1007/978-3-030-64708-7_15.

/SKR 19/ Skrlj, B.: Py3plex toolkit for visualizations and analysis of multilayer networks, Applied Network Science 4, S. 2364-8228, 2019, https://doi.org/10.1007/s41109-019-0203-7.

/SMI 16/ Smith, C., et al.: Advanced SAPHIRE 8 Modelling Methods for Probabilistic Risk Assessment via the Systems Analysis Program for Hands-On Integrated Reliability Evaluations (SAPHIRE) Software P-202 (No. INL/EXT-17-40996-Rev000), Idaho National Laboratories (INL), Idaho Falls, ID, USA, 2016.

/STI 23/ Stiller, J., et. al.: Probabilistic Modelling of Asymmetries in the Electrical Power Supply System of Nuclear Power Plants, in: Proceedings of ANS PSA 2023 International Topical Meeting on Probabilistic Safety Assessment and Analysis, Knoxville, TN, USA, July 15 – 20, 2023, on CD-ROM, American Nuclear Society (ANS), LaGrange Park, IL, USA, 2023.

/VES 81/ Vesely, W. E., et al.: Fault Tree Handbook, NUREG-0492, United States Nuclear Regulatory Commission (U.S. NRC), Washington, DC, USA, 1981, https://www.nrc.gov/docs/ML1007/ML100780465.pdf.

/WIL 99/ Wilensky, U.: NetLogo, Knowledge and Information Systems, Center for Connected Learning and Computer-Based Modelling, Northwestern University, Evanston, IL, USA, 1999.

/WU 08/ Wu, X., et al.: Top 10 algorithms in data mining, Knowledge and Information Systems, 14(1), S. 1-37, 2008, https://link.springer.com/article/10.1007/s10115-007-0114-2

/ZAK 14/ Zaki, M. J., W. Meira Jr.: Data Mining and Analysis: Fundamental Concepts and Algorithms, ISBN 9781108564175, Cambridge University Press, Cambridge, MA, USA, 2014.

Abkürzungsverzeichnis

BBN	Bayesian Belief Network
BE	Basisereignis
BEA	Basic Event Attribute
BMUV	Bundesministerium für Umwelt, Naturschutz, nukleare Sicherheit und Verbraucherschutz
BMWi	Bundesministerium für Wirtschaft und Energie
CCF	(Englisch: common cause failure, für Ausfall aus gemeinsamer Ursache)
CDF	Kernschadenshäufigkeit (Englisch: core damage frequency)
CDP	Kernschadenswahrscheinlichkeit (Englisch: core damage probability)
DL	Abhängigkeitsliste (Englisch: Dependency List)
ET	Ereignisbaum (Englisch: event tree)
FB	Fehlerbaum
FL-DL	Flooding Dependency List
FT	Englisch: fault tree, für Fehlerbaum
GPL	General Public License
GRS	Gesellschaft für Anlagen- und Reaktorsicherheit
GVA	gemeinsam verursachter Ausfall
H	übergreifende Einwirkung (Englisch: hazard)
HC	Raum bzw. Anlagenbereich einer übergreifenden Einwirkung (Englisch: hazard compartment)
HPSA	PSA für übergreifende Einwirkungen (Englisch: Hazards PSA)
IE	Auslösendes Ereignis (Englisch: initiating event)
IAEA	International Atomic Energy Agency
MSSQL	Microsoft® SQL Server
NB	Notebook
NLB	Nichtleistungsbetrieb
NRC	Nuclear Regulatory Commission
ORM	object relational mapping
PSA	Probabilistische Sicherheitsanalyse
p(...)	Wahrscheinlichkeit (Englisch: probability) für …
SMR	small modular reactor
SSC	Bauliche Anlagenteile, Systeme und Komponenten (Englisch: structures, systems, and components)
SQL	structured query language
SUSA	Software für Unsicherheits- und Sensitivitätsanalysen
UML	Unified Modelling Language

Abbildungsverzeichnis

Abb. 1.1 Darstellung multipler Raumabhängigkeiten für übergreifende Eiwirkungen als Schichten eines Multiplex-Netzwerks /BER 20b/ 4

Abb. 2.1 Unified Modelling Language (UML)-Klassendiagramm /KEC 06/ der verschiedenen, für die Kernfunktionalitäten von pyRiskRobot benötigten Klassen 9

Abb. 2.2 Ausschnitt eines Jupyter Notebooks zur Illustration der exemplarischen Nutzung der pyRiskRobot Kernfunktionalitäten 10

Abb. 2.3 Verbindung zwischen den von SAPHIRE erzeugten Verzeichnissen und Dateien für den Export im MARD-Format 11

Abb. 2.4 UML-Klassendiagramm /KEC 06/ für die Anbindung von SAPHIRE an pyRiskRobot 14

Abb. 2.5 Beispiel eines Jupyter Notebooks für die Anbindung von SAPHIRE an pyRiskRobot 14

Abb. 2.6 Visualisierung von Raumabhängigkeiten, basierend auf der in /BER 20/, Abschnitt 3.1 vorgestellten Studie, erstellt mit NetworkX und einem Spring Layout 20

Abb. 2.7 Visualisierung der Raumabhängigkeiten mit der plotly Dash-Bibliothek und der visdcc-Erweiterung /BOW 22/ 21

Abb. 2.8 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Greedy-Modularity Community der Knoten bestimmt wird 22

Abb. 2.9 Zoom auf zwei Greedy-Modularity Communities aus Abb. 2.8 22

Abb. 2.10 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Girvan-Newman Community der Knoten bestimmt wird 23

Abb. 2.11 Visualisierung der gleichen Raumabhängigkeiten wie in Abb. 2.6, wobei die Farbgebung der Knoten durch die Label-Propagation Community der Knoten bestimmt wird 23

Abb. 2.12 Normalisierte Brandwahrscheinlichkeit von Brandräumen (d. h. Nichtverfügbarkeit durch Brand) pro Anlagengebäude in Abhängigkeit der modellierten Nachbarschaftsordnungen 0 bis 4 der spezifischen Brandausbreitungspfade, basierend auf /BER 17/ 29

Abb. 2.13 Korrelation zwischen den verschiedenen betrachteten Größen zur Bestimmung der für die Analyse erforderlichen Ausbreitungstiefe 30

Abb. 2.14 Vergleich der verschiedenen, verfügbaren Bibliotheken zur Visualisierung mehrdimensionaler Netzwerke mittels einer Tabelle aus /SKR 19/ 33

Abb. 2.15 Verteilung der Entropie eines multiplexen Grades, berechnet unter Nutzung der Py3plex-Bibliothek für das in Abschnitt 2.7 beschriebene mehrdimensionale multiplexe Netzwerk 35

Abb. 2.16 Abbildung eines zweidimensionalen Netzwerkes mit py3plex /SKR 19/ 36

Tabellenverzeichnis

Tab. A.1 Spalten der Attributtabelle in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen 57

Tab. A.2 Spalten der Tabelle zu den Ereignisattributen in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen 57

Tab. A.3 Spalten der Tabelle zu den Exchange Events in der in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen 57

Tab. A.4 Spalten der Tabelle zu den Exchange Events in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen, weitere Spalten sind VarCoeff (int) und die Spalten EditDate, EditUid, ReviewDate, ReviewUid, ApprovedDate, ApprovedUid, wie in Tab. A.1 58

Tab. A.5 Spalten der Tabelle zu den Ereignisparametern in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen 58

Tab. A.6 Spalten der Tabelle zu den GVA-Ereignisparametern in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen 58

A Anhang A: Datenbanktabellen zum Auslesen von Ereignissen, Parametern und Attributen

Tab. A.1 Spalten der Attributtabelle in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen

Type	Num	ID	Text	Tag	EditDate	EditUid	ReviewDate	ReviewUid	ApprovedDate	ApprovedUID	flag
int	int	string	string	0	date	int	date	int	date	int	bool

Tab. A.2 Spalten der Tabelle zu den Ereignisattributen in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen

EventType	EventNum	AttType	AttNum	flag
int	int	int	int	0

Tab. A.3 Spalten der Tabelle zu den Exchange Events in der in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen

EventType	EventNum	CondType	CondNum	ExchType	ExchNum	flag
int	int	int	int	int	int	0

Tab. A.4 Spalten der Tabelle zu den Exchange Events in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen, weitere Spalten sind VarCoeff (int) und die Spalten EditDate, EditUid, ReviewDate, ReviewUid, ApprovedDate, ApprovedUid, wie in Tab. A.1

Type	Num	ID	Text	Tag	Mean	Dist-Type	Dist-Par1	Dist-Par2	Dist-Par3	Unit	Me-dian	P05	P95	…
int	int	string	string	0	float	int	float	float	float	int	float	float	float	…

Tab. A.5 Spalten der Tabelle zu den Ereignisparametern in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen

EventType	EventNum	ParType	ParNum	flag
int	int	int	int	0

Tab. A.6 Spalten der Tabelle zu den GVA-Ereignisparametern in der RiskSpectrum®-Datenbank und zugehörige Eintragstypen

IDNum	EventType	EventNum	ParType	ParNum	Value	flag
int	int	int	int	int	int	0

Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH

Schwertnergasse 1
50667 Köln
Telefon +49 221 2068-0
Telefax +49 221 2068-888

Boltzmannstraße 14
85748 Garching b. München
Telefon +49 89 32004-0
Telefax +49 89 32004-300

Kurfürstendamm 200
10719 Berlin
Telefon +49 30 88589-0
Telefax +49 30 88589-111

Theodor-Heuss-Straße 4
38122 Braunschweig
Telefon +49 531 8012-0
Telefax +49 531 8012-200

www.grs.de

ISBN 978-3-910548-76-3