Interaktive Vervollständigung der Szenariobasierten Spezifikation eines Parkassistenzsystems

Fadi Chabarek
Technische Universität Berlin
Daimler Center for Automotive IT Innovations
fadi.chabarek@tu-berlin.de

Ulrike Golas
Konrad-Zuse-Zentrum
für Informationstechnik Berlin
golas@zib.de

Abstract: Szenariobasierte Anforderungsanalyse ist einer der verbreitesten Ansätze zur Gewinnung und Dokumentation von Anforderungen. Sie erlaubt eine einfache und fokussierte Betrachtung von Systemaufgaben, führt aber zu stark fragmentierten und schwierig zu vervollständigenden Spezifikationen. Existierende Ansätze zur Verbesserung der Vollständigkeit konzentrieren sich auf Simulation oder synthetisieren Zustandsautomaten, die anschließend manuell geprüft werden müssen.

In diesem Papier wollen wir beispielhaft anhand eines Parkassistenzsystems ein interaktives Verfahren zur Vervollständigung von Szenariospezifikationen vorstellen. Basierend auf einem induktiven, heuristischen Lernverfahren werden Szenarien über Ereignismuster iterativ rekombiniert. Die resultierenden szenariobasierten Anfragen werden von Autoren oder Gutachtern akzeptiert oder verworfen. Das Ergebnis ist ein Verfahren, das eine hochwertige, interaktive Vervollständigung von Szenariokompositionen ermöglicht und systematisch fehlendes Verhalten erkennt.

1 Einleitung

Heutzutage wird ein maßgeblicher Anteil der Funktionalität eines Automobils durch Software bestimmt. Circa 40% der Fehler, die durch die Verwendung softwarebasierter Funktionen im Automobil entstehen, sind auf fehlerhafte und unvollständige Anforderungen zurückzuführen [Gri03]. Damit die kontinuierlich steigende Größe und Komplexität entwickelter Systeme auch in Zukunft beherrscht werden kann, verlagert sich die Anforderungsanalyse im Automobilbereich von einer bauteilorientierten Sichtweise hin zu der Betrachtung kundenwahrnehmbarer Fahrzeugfunktionen [Die08].

In der *funktionsorientierten Entwicklung* beschreibt eine Fahrzeugfunktion die Reaktion eines Fahrzeugs auf Ereignisse durch Szenarien. Fehlen Szenarien, können unspezifizierte Fahrzeugfunktionen in der Entwicklung von Lastenheften bis hin zu ihrer Realisierung unbemerkt bleiben. Durch einen höheren Vollständigkeitsgrad der Spezifikationen kann daher entscheidende Funktionalität gewonnen und erheblicher Mehraufwand durch nachträgliche Änderungen vermieden werden.

Ein Nebeneffekt der Verwendung von Szenariospezifikationen ist jedoch eine verringerte Übersicht über das Gesamtverhalten. Zum einen werden während der Anforderungsanalyse Zusammenhänge zwischen den einzelnen Teilverhalten häufig nicht durch explizite

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

Komposition erfasst. Zum anderen erschwert der partielle Charakter von Szenarien die Betrachtung von Alternativ- und Ausnahmefällen sowie von ähnlichen und selten auftretenden Verhalten [WK04]. Darüber hinaus fällt es Autoren und Gutachtern schwer, intuitiv über die Vollständigkeit partieller Spezifikationen zu entscheiden [AE03]. Fehlende Szenarien haben nicht nur einen großen Einfluss auf den Mehraufwand in der Entwicklung eines Produkts, sondern auch auf die Testbarkeit eines Systems. Ein System kann dadurch nur eingeschränkt validiert werden, so dass Fehler im unspezifizierten Verhalten einer Szenariospezifikation unentdeckt bleiben.

Szenariospezifikationen können unvollständig sein, weil Szenarien fehlen oder Zusammenhänge zwischen Szenarien nicht beschrieben wurden. Im Gegensatz zum manuellen Design können durch automatisierte Syntheseverfahren Eigenschaften von Szenariospezifikationen bewahrt werden. Zusätzlich sollen unpräzise Anforderungen durch Interaktion aufgedeckt und ergänzt werden, so dass die Korrektheit eines synthetisierten Gesamtverhaltensmodells per Konstruktion sichergestellt wird. Die entstehende Systembeschreibung kann dann gegenüber Sicherheitseigenschaften verifiziert, zur modellbasierten Entwicklung genutzt [CFGK05] und weiter verfeinert werden. Eine interaktive Vervollständigung von Szenariokompositionen ermöglicht daher einen effizienten und nahtlosen Übergang von der Analyse zum Design. Sowohl die Fehlerrate als auch der Entwicklungsaufwand können dadurch reduziert werden.

In diesem Papier wollen wir ein Verfahren vorstellen, dass eine szenariobasierte Interaktion verwendet, um sowohl gezielt fehlendes Verhalten als auch eine umfassende Szenariokomposition zu erschließen. Diese Analyse soll bereits in frühen Phasen des Systementwicklungsprozesses einsetzbar sein, durch die Vervollständigung vermeidbare Nachbesserungen reduzieren sowie vorbereitend einen nahtlosen Übergang von der Anforderungsanalyse zum Design mit bestehenden Syntheseverfahren ermöglichen.

Konkret realisieren wir dieses Vorhaben, indem wir positive und negative Szenarien einer funktionsorientierten Szenariospezifikation aus dem Automobilbereich über Wörter darstellen und diese in den Grammatik-Inferenz-Algorithmus Blue-Fringe-EDSM [LPP98] eingeben. Dieser versucht, aus den Beispielwörtern heuristisch einen Automaten als Repräsentanten einer regulären Grammatik induktiv zu erschließen. Wir stellen das erschlossene Verhalten und die Kompositionen über szenariobasierte Anfragen dar und vervollständigen die Eingabespezifikation in Interaktion mit Ingenieuren. In Anlehnung an die spielerische Vervollständigung des Verhaltens nennen wir das Verfahren Puzzleorientierte Vervollständigung von Szenariospezifikationen über Interaktion (engl. Jigsaw-oriented Scenario specification Completion by Interaction, JigSCI). Der vollständige Algorithmus ist in [Cha11] beschrieben.

Der Algorithmus wurde prototypisch implementiert. Auf der Benutzeroberfläche des JigSCI-Verfahrens werden Anfragen grafisch dargestellt und durch Ingenieure beantwortet. Die Präsentationsschicht besteht des Weiteren aus einem Funktionsstruktur-Editor und einer Komponente zur Konfiguration des Verfahrens.

In diesem Papier wollen wir anhand eines Parksystems demonstrieren, dass durch das JigSCI-Verfahren umfangreich fehlendes Verhalten und eine annähernde Szenariokomposition bestimmt wird. Wir zeigen dadurch, dass mit dem JigSCI-Verfahren mit angemesse-

nem Aufwand nicht-triviale Szenariospezifikationen zu einem hohen Grad vervollständigt werden können.

In Abschnitt 2 stellen wir dieses Parksystem vor. Grundlegende Begriffe bezüglich der Vollständigkeitsanalyse werden in Abschnitt 3 definiert. In Abschnitt 4 zeigen wir dann die strukturelle Vervollständigung des Beispiels, während in Abschnitt 5 die funktionale Vollständigkeit untersucht wird. Die Arbeit schließt mit einer Zusammenfassung in Abschnitt 6. Für die theoretischen Aspekte dieses Papiers sollte der Leser gelabelte Transitionssystem (LTS) kennen und grundlegende Kenntnisse über Zustandsautomaten haben.

2 Das Parkassistenzsystem

Als durchgängiges Beispiel verwenden wir eine Szenariospezifikation eines automatischen Parkassistenzsystems, welches aus zwei Teilsystemen besteht. Die Parklückenvermessung (engl. Parking Space Surveillance, PSS) vermisst während der Vorwärtsfahrt die Umgebung eines Fahrzeugs nach möglichen Parklücken mittels Radarsensoren. Eine geeignete Lücke wird dem Fahrer durch ein Parksymbols im Kombiinstrument des Fahrzeugs angezeigt. Durch das Einlegen des Rückwärtsgangs und Einstellen des Blinkers kann der Fahrer dann die Parkführung des Systems (engl. Automatic Parking Guidance, APG) aktivieren, die ihn durch Lenkwinkelvorschläge kollisionsfrei in die Parklücke dirigiert. Im Folgenden werden wir die Fahrzeugfunktionen durch natürlichsprachliche Szenarien beschreiben, für die wir eine Anforderungsvorlage aus der Automobilindustrie – eine sogenannte *Funktionsstruktur* – verwenden.

Die Parklückenvermessung Die Funktionsstruktur des Teilsystems PSS besteht aus vier positiven Fahrzeugfunktionen *F1*, ..., *F4* und einer negativen *FN1*. Entdeckt das System in Vorwärtsfahrt eine mögliche Parklücke, überprüft es deren Größe und stellt fest, ob ein kollisionsfreies Einparken möglich ist. Ist dies der Fall, weist das Fahrzeug den Fahrer im Kombiinstrument auf die Parklücke hin (*F1*), ansonsten wird sie verworfen (*F2*). Registriert das System ein Hindernis, wird die vorgeschlagene Parklücke zurückgezogen (*F3*). Mit *F4* erkennt das System, dass ein Fahrer eine Parklücke nicht nutzen möchte, da er vorbei gefahren ist, und zieht seinen Vorschlag zurück. Die negative Funktion *FN1* schließt aus, das auf Parklücken hingewiesen wird, die durch ein Hindernis versperrt werden.

Diese Funktionen werden durch Ingenieure im Anforderungsmanagementwerkzeug DOORS beschrieben und liegen so gegliedert vor, dass aus den Funktionen Sequenzen von Ein- und Ausgabeereignissen des Fahrzeugs ableitbar sind. Wir behandeln diese Ereignissequenzen als Szenarien, welche wir der Übersicht halber als gerichtete Graphen darstellen. Die Spezifikation $S_{PSS} = (S_+, S_-)$ besteht damit aus den positiven und negativen Szenarien $S_+ = \{S1, \ldots, S4\}$ bzw. $S_- = \{SN1\}$ (Abb. 1 links). Auf der rechten Seite der Abbildung ist den Szenarien die vollständige, *gewünschte* Funktionalität des PSS-Systems als Zustandsautomat A_{PSS} gegenübergestellt. Zum Zeitpunkt der Entwicklung der Szenariospezifikation ist er aber unbekannt.

Die Parkführung Die Funktionsstruktur des Parkassistenten besteht neben den Funktionen der Parkvermessung *F1* bis *F4* und *FN1* aus weiteren Funktionen, von denen die

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

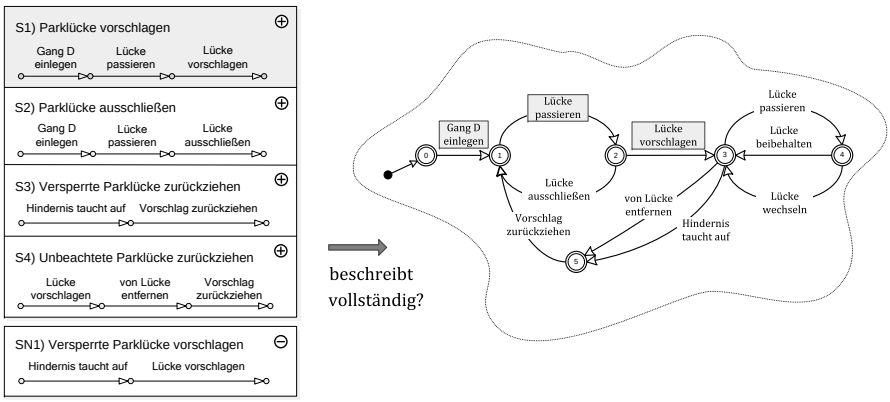


Abbildung 1: Die Szenarien und das angestrebte Gesamtverhalten des Teilsystems PSS

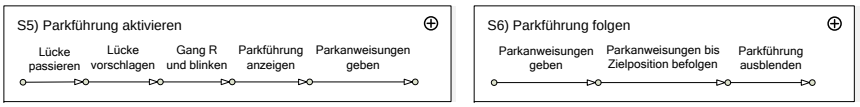


Abbildung 2: Die wesentlichen Szenarien des Parkassistenten APG

zwei Szenarien in Abb. 2 das Standardverhalten darstellen. Der Parkassistent wurde zur Demonstration der Konzepte dieser Arbeit auf diese wesentliche Funktionalitäten eingegrenzt, welche durch den Automaten in Abb. 3 beschrieben werden. Wir werden feststellen, dass dieses System nicht vollständig durch die gegebenen Funktionen beschrieben wird. Zur Verdeutlichung der Schwierigkeit einer Inspektion sollte der Leser versuchen, die gegebene Funktionsstruktur auf Vollständigkeit zu überprüfen. Selbst auf diesem hohen Level der Abstraktion und mit einem gegebenen Zielautomaten bereitet die Vollständigkeitsanalyse Schwierigkeiten.

3 Strukturelle und funktionale Vollständigkeit

Eine vollständige Spezifikation eines Systems grenzt das erwünschte vom unerwünschten Verhalten eindeutig ab. Als Ziel der Vervollständigung verwenden wir eine *Referenz-Szenariospezifikation* $V^{Ziel} = (V_+^{Ziel}, V_-^{Ziel})$ mit $V_+^{Ziel} \uplus V_-^{Ziel} = \Sigma^*$, die jedes positive und negative Verhalten des gesuchten Systems über dem Ereignisalphabet Σ definiert. Das Referenzverhalten enthält unendlich viele mögliche Szenarien und soll durch die Komposition einer endlichen Spezifikation beschrieben werden. Zur Optimierung der Komposition nehmen wir an, dass das Referenzverhalten *präfixabgeschlossen* ist, d. h. das jeder Präfix eines positiven Szenarios ebenfalls positiv ist. Ein LTS ist *kompatibel* zu einer Szenariospezifikation, wenn es unter Berücksichtigung der Präfixabgeschlossenheit des Referenzverhaltens die positiven Szenarien der Spezifikation akzeptiert und die negativen verwirft. Damit ein LTS kompatibel zu einer Szenariospezifikation sein kann, müssen die

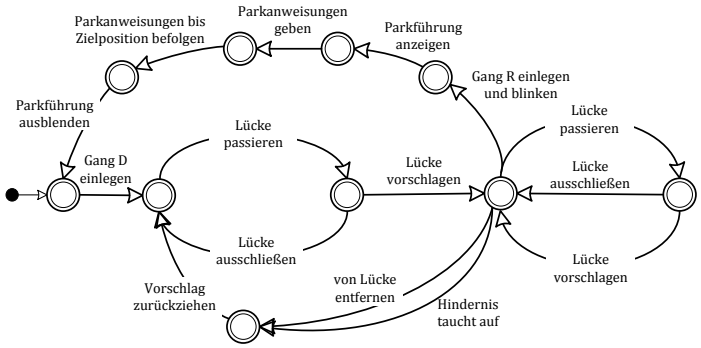


Abbildung 3: Das durch die Funktionsstruktur beschriebene Parksystem A_{Park}

Szenarien der Spezifikation im initialen Zustand des Transitionssystems beginnen. Eine Szenariospezifikation, deren Szenarien diese Bedingung erfüllen, nennen wir *normalisiert*.

Das kleinste zum Referenzverhalten kompatible LTS nennen wir das *gesuchte* LTS. Eine Szenariospezifikation ist *strukturell vollständig*, wenn das gesuchte LTS zu ihr kompatibel ist und die positiven Szenarien über alle Transitionen des gesuchten LTS verlaufen. Dies stellt sicher, dass eine Szenariospezifikation weder zu viel Verhalten enthält noch gültiges Verhalten durch negative Szenarien ausschließt.

Eine *Kompositionsabbildung* erzeugt aus jeder Szenariospezifikation ein kompatibles Transsitionssystem. Durch die Eingabe einer strukturell vollständigen Szenariospezifikation wird ein Transitionssystem erzeugt, das die Transitionen des gesuchten LTS enthält, aber nicht zwingend kompatibel zum Referenzverhalten ist. Eine Szenariospezifikation ist *funktional vollständig* bezüglich einer gegebenen Kompositionsabbildung, wenn die Spezifikation zu einem LTS führt, das das Referenzverhalten beschreibt.

Ein *angereichertes LTS* (ALTS) unterscheidet zwischen akzeptierenden, neutralen und verwerfenden Zuständen und berücksichtigt die Präfixabgeschlossenheit akzeptierter und verworfener Szenarien. Demzufolge werden Szenarien, die in einem verwerfenden Zustand oder einem seiner Nachfolgezustände enden, verworfen, während Szenarien, die in Vorgängerzuständen akzeptierender Zustände enden, akzeptiert werden. Über Szenarien, die in den neutralen Zuständen enden, wird keine Aussage getroffen. Der Blue-Fringe-EDSM-Algorithmus [LPP98] synthetisiert ein ALTS aus gegebenen Beispielwörtern. In einem ersten Schritt werden die positiven Wörter der Eingabe in ein ALTS umgewandelt, das genau die positiven Beispiele akzeptiert und die negativen verwirft. Es entsteht ein baumförmiges ALTS aus positiven, neutralen und negativen Zuständen. Anschließend beginnt die Generalisierung des ALTS. Dafür werden drei Zustandsmengen definiert. Der initiale Zustand, rot gefärbt, repräsentiert einen konsolidierten Zustand. Die zweite Zustandsmenge, blau gefärbt, enthält alle Zustände, die durch eine ausgehende Transition von den konsolidierten Zuständen aus erreichbar sind. Die übrigen Zustände werden nicht eingefärbt (s. Abschnitt 4). Nun wird ein Zustandspaar aus einem blauen und einem roten Zustand auf Basis ihrer Ähnlichkeit selektiert und vereinigt. Ist das entstehende ALTS kom-

patibel zur Spezifikation, wird es determinisiert und das nächste Zustandspaar untersucht. Der Algorithmus terminiert, wenn alle Zustandspaare des aktuellen Lösungsautomaten auf diese Weise miteinander verglichen wurden, und gibt den Automaten aus.

In [Ang88] wird das *Anfragegestützte Lernen* eines Konzepts unter Zuhilfenahme von Interaktion untersucht. Inferenz-Algorithmen können spezifische Anfragen an ein Orakel – in unserem Fall an einen Ingenieur – stellen, um unzulängliche Beispieldaten zu kompensieren. Es gibt verschiedene Anfragearten für eine reguläre Sprache L mit einem kanonischen Automaten A:
Elementanfragen Eine Elementanfrage hat die Form „$w \in L$?“. Die Antwort des Orakels ist *ja*, wenn $w \in L$ oder andernfalls *nein*.
Teilmengenanfrage Ist eine Menge L_* Teilmenge der Sprache L, d.h. $L_* \subseteq L$, antwortet das Orakel mit *ja*, ansonsten mit *nein* und gibt ein Gegenbeispiel $v \in L_* \backslash L$ an.
Komplementärmengenanfragen Ist die Sprache L disjunkt zu einer Menge L_*, d.h. $L \cap L_* = \varnothing$, antwortet das Orakel mit *ja*, ansonsten mit *nein* und der Angabe eines Gegenbeispiels $v \in L \cap L_*$.

4 Strukturelle Vervollständigung von Szenariospezifikationen

Die Szenariospezifikation S_{PSS} ist strukturell unvollständig bezüglich des LTS A_{PSS}, da dieses weder die Szenarien *S3*, *S4* akzeptiert noch die Reaktion auf das mehrmalige Passieren von Parklücken durch die Spezifikation beschrieben ist. Im JigSCI-Verfahren kombinieren wir den Blue-Fringe-EDSM-Algorithmus mit Elementanfragen, um interaktiv die Szenariospezifikation strukturell zu vervollständigen.

In Abb. 4 ist eine Übersicht über die Scenario-Puzzling-Technik dargestellt. Gegeben ist eine zu vervollständigende Szenariospezifikation $SSpec_I$, deren positive Szenarien wir im ersten Schritt des JigSCI-Verfahrens auf Erreichbarkeit prüfen. Der Algorithmus erfragt gültige Szenarien, die zur Aktivierung der unerreichbaren Szenarien führen. Mit diesen ergänzen wir $SSpec_I$ zu der normalisierten Szenariospezifikation $\| SSpec_I \|$.

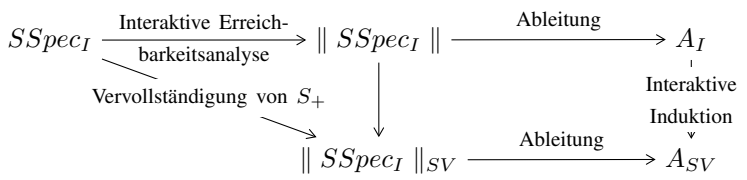


Abbildung 4: Eine Übersicht über die Scenario-Puzzling-Technik

Aus dieser Spezifikation leiten wir ein LTS A_I ab. Zur Entdeckung strukturell fehlenden Verhaltens verwenden wir A_I als Lösungstransitionssystem des Blue-Fringe-EDSM-Algorithmus. Wir überprüfen über Interaktion, ob das durch die Komposition dieser Zustände entstehende Verhalten strukturell fehlt. Dieses wird der Szenariospezifikation $\| SSpec_I \|$ hinzugefügt und das selektierte Zustandspaar anschließend vereinigt. Die re-

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

sultierende Spezifikation $\| SSpec_I \|_{SV}$ enthält dann die durch die Erreichbarkeitsanalyse aufgedeckten Präfixe sowie Szenarien, aus denen das im finalen Lösungstransitionssystem A_{SV} enthaltene strukturell fehlende Verhalten eindeutig rekonstruierbar ist.

Interaktive Erreichbarkeitsanalyse Ein Szenario ist in einer Szenariospezifikation erreichbar, wenn eine Szenariokomposition existiert, die im initialen Zustand des Systems beginnt und das Verhalten des Szenarios enthält. Die Grundidee der Erreichbarkeitsanalyse ist es, durch Elementanfragen die normalisierten und dadurch direkt erreichbaren Szenarien einer Eingabespezifikation zu erkennen und diese über Szenarioüberlappungen und -fortsetzungen zu erweitern. Zur Demonstration der Erreichbarkeitsanalyse erweitern wir das System PSS (s. Abb. 1) um die Funktion *F6* aus Abb. 2 und überprüfen die resultierende Funktionsstruktur $FS_{PSS'} = (\{F1, \ldots, F4, F6\}, \{FN1\})$ bezüglich des Referenzverhaltens $V^{Ziel} = (L(A_{Park}), \Sigma^* \backslash L(A_{Park}))$ aus Abb. 3 auf Vollständigkeit.

Die Erreichbarkeitsanalyse beginnt mit den Anfragen aus Tab. 1. Mit $\cdot$ bezeichen wir dabei die Konkatenation von Ereignissen zu einem Szenario und heben die Kombination von Szenarien durch $\bullet$ hervor. Die Anfragen identifizieren die Szenarien *S1* und *S2* als normalisiert. Die noch unerreichbaren Szenarien können in jedem Zustand des zugehörigen ALTS außer q_0 ausführbar sein. Wir fragen im zweiten Schritt der Analyse nach Szenarioüberlappungen und -fortsetzungen zwischen Ereignissen. Akzeptierende Zustände werden priorisiert, um Fortsetzungen ganzer Szenarien zuerst zu prüfen. Dies führt zu der Elementanfrage

Gang D einlegen $\cdot$ Lücke passieren $\cdot$ Lücke vorschlagen $\bullet$ von Lücke entfernen $\cdot$ Vorschlag zurückziehen $\in L(A_{APG}))$?

Diese wird bejaht und das Szenario hinzugefügt. Das Verfahren überprüft dann Fortsetzungen durch die Szenarien *S3* und *S6*. Die erste Elementanfrage nach dem Verhalten

Gang D $\cdot$ Lücke passieren $\cdot$ Lücke vorschlagen $\bullet$ Hindernis taucht auf $\cdot$ Vorschlag zurückziehen$\in L(A_{APG}))$?

führt zu einem LTS, das das positive Verhalten der PSS-Funktionen beschreibt. Die verbleibenden Anfragen werden verneint und identifizieren damit das Szenario *S6* als unerreichbar. Der Ingenieur wird daraufhin nach einem gültigen Präfix befragt. Er entscheidet, dass das Einlegen des Rückwärtsgangs und das Blinken auf der Seite der Parklücke einen Einparkvorgang genügend stark kennzeichnen. Diese Aktivierungsbedingung wird der Funktionsstruktur in Form der Funktion *F5* hinzugefügt. Das LTS wird abschließend automatisch durch das negative Verhalten des Szenarios *SN1* zum ALTS aus Abb. 5 ergänzt, welches dann aus den weißen Zuständen des LTS und dem in grau gehaltenen negativen Zustand `14` besteht.

Strukturelle Vervollständigung Das ALTS bildet nach der Erreichbarkeitsanalyse den Zustandsraum, der durch den JigSCI-Algorithmus nach fehlendem Verhalten durchsucht wird. Zur Suche nach strukturell fehlendem Verhalten bewerten wir die Zustandspaare

Anfrage	Antwort	erreichbar
Gang D einlegen $\in L(A_{APG})$?	Ja	
Gang D einlegen $\cdot$ Lücke passieren $\cdot$ Lücke vorschlagen $\in L(A_{APG})$?	Ja	S1
Gang D einlegen $\cdot$ Lücke passieren $\cdot$ Lücke ausschließen $\in L(A_{APG})$?	Ja	S1, S2
Hindernis taucht auf $\in L(A_{APG})$?	Nein	
Lücke vorschlagen $\in L(A_{APG})$?	Nein	
Parkanweisungen geben $\in L(A_{APG})$?	Nein	

Tabelle 1: Die zur Identifikation normalisierter Szenarien gestellten Elementanfragen

ISBN 978-3-88579-286-4

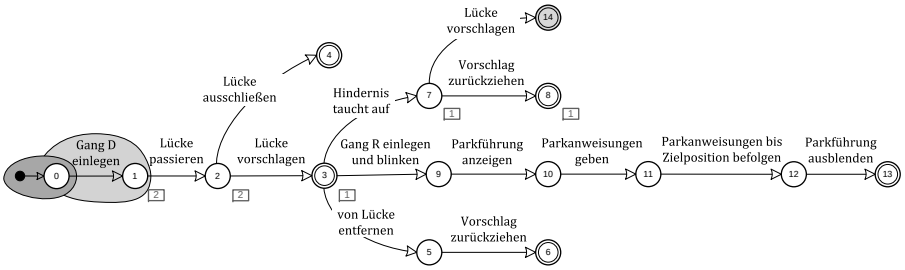


Abbildung 5: Das durch die Erreichbarkeitsanalyse ermittelte ALTS A_0

über die Ähnlichkeit ihrer Suffixe und selektieren nur solche, die besonders starke gemeinsame Verhaltensmuster aufweisen. Wir untersuchen das durch die Vereinigung entstehende Verhalten durch *Kürzeste-Präfix-Anfragen*, die als Elementanfragen durch die Verkettung des kürzesten Präfix eines der Zustände mit den Szenario-Suffixen des jeweils anderen Zustands entstehen. Diese werden solange gestellt, bis entweder alle Anfragen positiv beantwortet wurden oder eine Grenze an negativen Antworten erreicht ist.

Die wesentliche Kernaufgabe des Algorithmus ist es, jene Zustandspaare möglichst schnell auszusortieren, deren Vereinigung zu keinem fehlenden Verhalten führt, und das strukturell fehlende Verhalten möglichst effektiv zu identifizieren. Wir reduzieren den notwendigen Interaktionsaufwand durch Konfiguration eines EDSM-Minimalwerts und der Interaktionssteuerung. Zusätzlich erhalten Ingenieure die Möglichkeit, durch die Angabe optionaler Prioritäten und einer zu konfigurierenden Prioritätsgrenze P bestimmte Ereignisse oder Szenarien zu gewichten und damit die Untersuchung des Verfahrens zu steuern.

Wir wenden den JigSCI-Algorithmus auf die Ausgabespezifikation der Erreichbarkeitsanalyse $\| S_{PSS'} \|$ an. Das Ergebnis ist das in Abb. 5 dargestellte ALTS A_0 mit dem roten Zustand `0` und dem blauen Zustand `1`. Sei $p_{PSS'}$(Lücke passieren) $= 2$, $p_{PSS'}$(Hindernis taucht auf $\cdot$ Vorschlag zurückziehen) $= 1$ eine Priorisierungsfunktion, mit der die Ingenieure die auf Vollständigkeit zu untersuchenden Szenarien priorisiert haben. Die *Priorität eines Zustands* ist dann die maximale Priorität der Szenarien, die über den Zustand verlaufen, oder die maximale Priorität der angrenzenden Transitionsereignisse und wird rechts unterhalb des Zustands angezeigt.

Zuerst untersucht der Algorithmus die Paare `(1,0)`, `(2,0)` und `(2,1)`. Das Paar `(1,0)` wird selektiert und der Präfix mit der kürzesten Länge eines der Zustände, z.B. λ für den Zustand `0`, mit dem ersten Ereignis Lücke passieren des Suffix des anderen Zustands `1` kombiniert und als *Kürzeste-Präfix-Anfrage* gestellt (s. Tab. 2). Die Elementanfrage wird vom Ingenieur verneint, so dass die Überprüfung des Paares abgebrochen

(q_b, q_r)	Anfrage	Antwort
`(1,0)`	Lücke passieren $\in V_+^{Ziel}$?	Nein
`(2,0)`	Gang D einlegen $\cdot$ Lücke passieren $\bullet$ Gang D einlegen $\in V_+^{Ziel}$?	Nein
`(2,1)`	Gang D einlegen $\cdot$ Lücke passieren $\bullet$ Lücke passieren $\in V_+^{Ziel}$?	Nein

Tabelle 2: Die für die Zustandspaare `(1,0)`, `(2,0)` und `(2,1)` gestellten Anfragen

Anfrage	p	Antwort
Gang D einlegen · Lücke passieren · Lücke vorschlagen • Lücke passieren $\in V_+^{Ziel}$?	2	Ja
Gang D einlegen · Lücke passieren · Lücke vorschlagen • Lücke passieren · Lücke vorschlagen $\in V_+^{Ziel}$?	2	Ja
Gang D einlegen · Lücke passieren · Lücke vorschlagen • Lücke passieren · Lücke ausschließen $\in V_+^{Ziel}$?	2	Ja
Gang D einlegen • Hindernis taucht auf $\in V_+^{Ziel}$?	1,5	Nein
Gang D einlegen • von Lücke entfernen $\in V_+^{Ziel}$?	0	Nein
Gang D einlegen • Gang R einlegen und blinken $\in V_+^{Ziel}$?	0	Nein

Tabelle 3: Die zur Überprüfung des Zustandspaares `(3,1)` gestellten Anfragen

und der Zustand `1` in die Menge der roten Zustände befördert wird. Analog werden die Paare `(2,0)` bzw. `(2,1)` verworfen und führen zur Konsolidierung des Zustands `2`.

Als nächstes wird das Zustandspaar `(3,1)` selektiert und die Kürzeste-Präfix-Anfragen in Tab. 3 generiert. Die Reihenfolge der Anfragen wird auf Basis der Prioritätsmetrik p festgelegt. Zunächst werden die Anfragen mit auf das erste Ereignis verkürztem Szenario-Suffix gestellt und bei positiver Beantwortung auf den vollen Szenario-Suffix erweitert.

Durch die gebildete Reihenfolge sowie durch die Interaktionsintensivierung des JigSCI-Verfahrens führen diese und die darauffolgende Anfrage zur erfolgreichen Aufdeckung weiterer strukturell fehlender Szenarien. Diese werden in der Lösungsszenariospezifikation ergänzt und durch die Vereinigung der Zustände im ALTS $A_1 = A_0/_{3=1}$ abgebildet (s. Abb. 6). Das entstandene ALTS ist zwar übergeneralisiert, da es das Szenario „Gang D einlegen · Gang R einlegen und blinken · Parkführung anzeigen“ akzeptiert. Im Gegenzug erhält der Zustand $\{1, 3\}$ in A_1 jedoch sowohl die Eigenschaft des Zustands `3`, aus F_+ zu sein, als auch die Suffixe des Zustands `1`.

Die schrittweise Analyse aller möglichen weiteren Zustandspaare führt dann zum ALTS A_5 (s. Abb. 7). Dieses ist das Lösungstransitionssystem des Scenario Puzzlings und enthält das positive Verhalten der Lösungsszenariospezifikation $\| S_{PSS'} \|_{SV}$, die sich nun aus den Szenarien der normalisierten Eingabespezifikation und den positiv und negativ klassifizierten Szenarien der gestellten Kürzeste-Präfix-Anfragen zusammensetzt. Die Interaktion des Scenario Puzzling beträgt für die in diesem Beispiel verwendeten Parameter 42

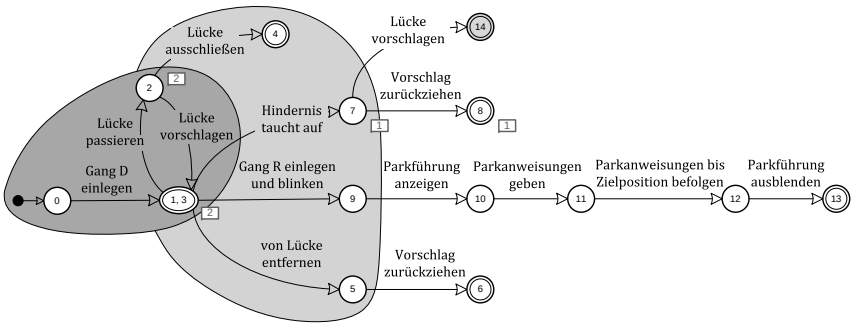


Abbildung 6: Das übergeneralisierte ALTS A_1

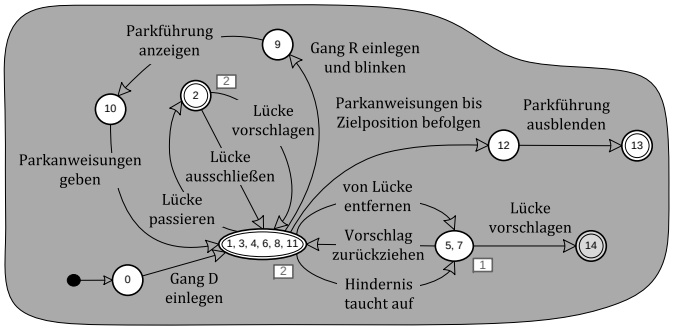


Abbildung 7: Das resultierende Lösungs-ALTS A_5

Kürzeste-Präfix-Anfragen zur Induktion fehlenden Verhaltens und erzeugt durch die entdeckten positiven Szenarien der Tab. 3 eine strukturell vollständige Szenariospezifikation.

5 Funktionale Vervollständigung von Szenariospezifikationen

Eine strukturell vollständige Szenariospezifikation soll nun funktional vervollständigt werden. Dazu verwenden wir die Merge-Validation-Technik, die das JigSCI-Verfahren um Teilmengen- und Komplementärmengenanfragen erweitert und durch die Interaktion Gegenbeispiele für ungültige Kompositionen feststellt. Aus den positiven Szenarien S_+ wird ein ALTS A_{SV} abgeleitet, das zu einem ALTS A_{FV} unter Berücksichtigung der negativen Szenarien S_- generalisiert wird (s. Abb. 8). Die Zustandsvereinigungen des Algorithmus spiegeln die möglichen Kompositionen der positiven Szenarien wider, die durch negative Szenarien als Kompositionsinformation eingeschränkt werden. Nach [Dup96] ist die

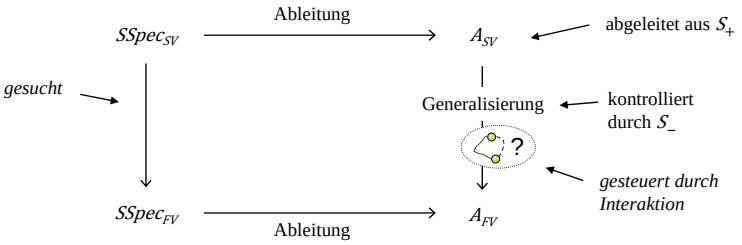
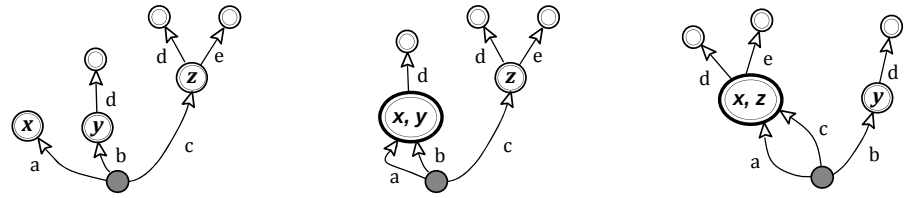


Abbildung 8: Der Prozess der Merge-Validation-Technik

verbleibenden Zustandspaar Teilmengenanfragen. Diese werden gestellt, bis das vollständige Verhalten der Vereinigung durch die Ingenieure bestätigt oder ein Gegenbeispiel bestimmt wird. Bei der Angabe eines Gegenbeispiels wird das Zustandspaar verworfen und das negative Szenario der Szenariospezifikation hinzugefügt. Dieses Vorgehen wird für die verbleibenden Zustandspaare wiederholt, bis alle Vereinigungen bestätigt oder ausgeschlossen worden sind.

Untergeneralisierungen können durch lokal optimale Partitionen entstehen, die in Abb. 9 veranschaulicht sind. Das ALTS auf der linken Seite enthält drei hervorgehobene Zustände x, y und z. Das Referenzsystem auf der rechten Seite zeigt, dass die Zustände x und z vereinigt werden müssen, um das exakte Referenzverhalten zu erschließen. Zuerst wird allesdings das Zustandspaar (y, x) ausgewählt. Da x und y auf Basis ihrer Suffixe zueinander kompatibel sind, wird das Lösungstransitionssystem durch eine Vereinigung der Zustände nicht übergeneralisiert, was zu dem mittleren Transitionssystem führt. In diesem sind die Zustände $\{x, y\}$ und z nicht mehr zusammenfassbar, da der Zustand y durch das Szenario $\mathsf{b{\cdot}e} \in V_{-}^{Ziel}$ die Vereinigung von x und z ausschließt. Als eine Konsequenz lokal optimaler Partitionen kann der Blue-Fringe-EDSM-Algorithmus nicht garantieren, dass bei Eingabe eines strukturell vollständigen Beispiels eine exakte Lösung durch Teilmengenanfragen erschlossen werden kann. Aufgrund des hohen Interaktionsaufwands, der zur Überprüfung jedes Suffixes notwendig wäre, stellen lokal optimale Zustandsvereinigungen daher ein

[dɪsiː aɪtiː]



(a) Ein initialer APTA A_0

(b) Eine lokal optimale Zustandsvereinigung von x und y

(c) Das gesuchte Transitionssystem

Abbildung 9: Die Ursache der Untergeneralisierung

ISBN 978-3-88579-286-4

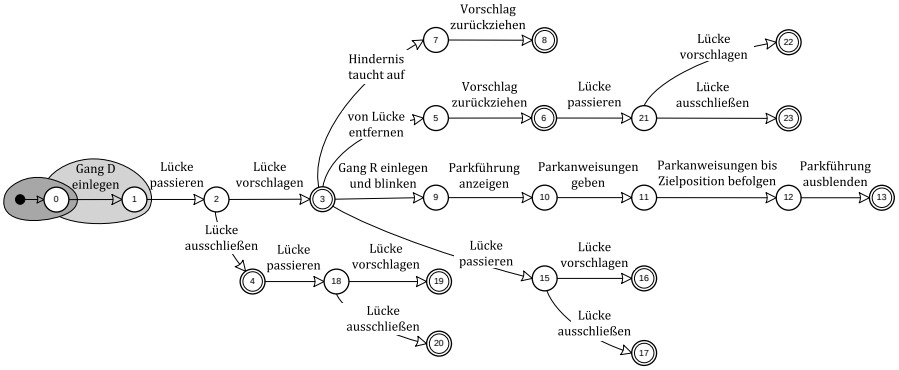


Abbildung 10: Das aus dem Ergebnis des Scenario Puzzling abgeleitete ALTS A_0

Der Blue-Fringe-Delta-EDSM-Algorithmus erzeugt aus der strukturell vollständigen Spezifikation als initiale Lösung das ALTS A_0 aus Abb. 10. Das ALTS wird in der Abbildung nur durch seine positiven Zustände repräsentiert und das negative Verhalten in der Darstellung aus Gründen der Prägnanz ausgelassen. Auf Basis der roten und blauen Menge werden analog zum Scenario Puzzling Zustandspaare ausgewählt. Die Paare `(1,0)`, `(2,0)` und `(3,0)` scheiden aufgrund ihrer zu geringen Ähnlichkeit aus. Das Zustandspaar `(3,1)` wird aufgrund des negativen Szenarios „Gang D einlegen • Hindernis taucht auf“ des Scenario Puzzlings ohne Interaktion verworfen.

Das erste validierte Zustandspaar ist daher das Paar `(4,1)`. Die Interaktion der Merge-Validation-Technik setzt sich aus Kürzeste-Präfix-, Komplementärelement-, und Zyklische-Verhaltens-Anfragen zusammen, welche in dieser Reihenfolge durch das JigSCI-Verfahren gestellt werden (s. Tab. 4). Die Kürzeste-Präfix-Anfragen des Zustandspaares werden bereits durch das Transitionssystem A_0 akzeptiert und automatisch beantwortet. Die negativen Suffixe des Zustands `1` führen daraufhin zu drei Komplementärelementanfragen, welche durch den Ingenieur verneint werden. Die Antworten bestätigen das komplementäre Verhalten der Anfragen, so dass ein lokales Optimum in der Vereinigung ausgeschlossen wird. Die anschließenden Zyklische-Verhaltens-Anfragen der Zeilen 4 und 5 der Tabelle überprüfen den Zyklus Lücke passieren · Lücke ausschließen, der durch die

Anfrage	Antwort
Gang D einlegen · Lücke passieren · Lücke ausschließen • Lücke passieren· Hindernis taucht auf · Lücke vorschlagen $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke ausschließen • Lücke passieren· Gang D einlegen $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke ausschließen • Lücke passieren· Lücke passieren $\in V_+^{Ziel}$?	Nein
Gang D einlegen · (Lücke passieren · Lücke ausschließen)* • Lücke passieren· Lücke vorschlagen $\subseteq V_+^{Ziel}$?	Ja
Gang D einlegen · (Lücke passieren · Lücke ausschließen)* • Lücke passieren· Lücke ausschließen $\subseteq V_+^{Ziel}$?	Ja

Tabelle 4: Die zur Validierung der Zustandsvereinigung `(4,1)` gestellten Anfragen

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html

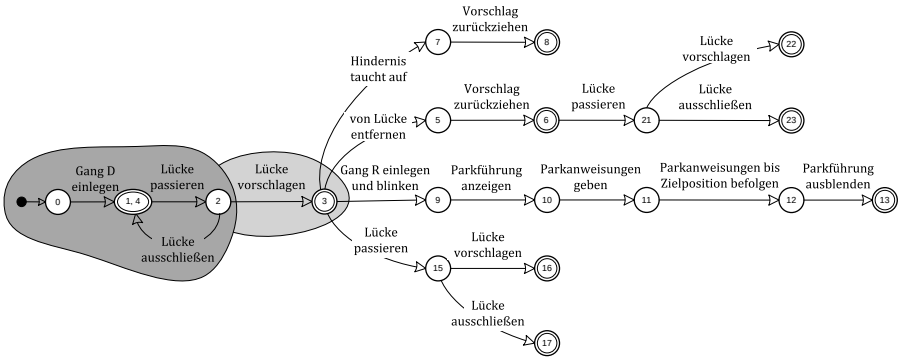


Abbildung 11: Das durch die Vereinigung der Zustände `(4,1)` entstehende ALTS A_1

Zustandsvereinigung gebildet wird. Die Anfragen werden durch den Ingenieur bejaht und der Zyklus dadurch akzeptiert. Da die verbleibende Bestätigungsanfrage äquivalent zur ersten Zyklische-Verhaltens-Anfrage ist, wird keine weitere Anfrage gestellt und die Vereinigung durchgeführt. Das resultierende Lösungstransitionssystem $A_1 = A_0/_{1=4}$ ist in Abb. 11 dargestellt.

Das nächste zu validierende Zustandspaar ist das Paar `(15,2)`. Die Kürzeste-Präfix-Anfragen wurden bereits im Scenario Puzzling beantwortet und entfallen daher auch bei diesem Zustandspaar. Anschließend decken die Komplementärelementanfragen ein lokales Optimum durch die Überprüfung der determinisierten Zustände `(17,1)` mit der Anfrage „Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Lücke ausschließen · Hindernis taucht auf $\in V_+^{Ziel}$?“ auf (s. Tab. 5). Die Anfrage wird vom Ingenieur mit Ja beantwortet und das lokale Optimum verworfen.

Anfrage	Antwort
Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Hindernis taucht auf · Lücke vorschlagen $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Gang D einlegen $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Lücke passieren $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Vorschlag zurückziehen $\in V_+^{Ziel}$?	Nein
Gang D einlegen · Lücke passieren · Lücke vorschlagen · Lücke passieren • Lücke ausschließen · Hindernis taucht auf $\in V_+^{Ziel}$?	Ja

Tabelle 5: Die Komplementäranfragen zur Aufdeckung der lokal optimalen Vereinigung

Analog werden alle weiteren Zustandspaare überprüft, bis das gesuchte Transitionssystem aus Abb. 3 erschlossen wurde. Insgesamt wurden 51 Anfragen benötigt. Das Transitionssystem ist vollständig konsolidiert und stellt das resultierende Lösungstransitionssystem der Merge-Validation-Technik dar. Es kann durch die Eingabe der vervollständigten Szenariospezifikation durch den Blue-Fringe-Delta-EDSM-Algorithmus wiederhergestellt werden. Weil das Transitionssystem äquivalent zum Transitionssystem A_{Park} ist, konnten wir daher die Szenariospezifikation erfolgreich funktional vervollständigen.

5.1 Das vollständige Parkassistenzsystem

Während wir in diesem Papier nur einen Teil des Parkassistenten analysiert haben, wurde in [Cha11] des komplette System modelliert und vervollständigt. In der kompletten Fallstudie wurde das System in vier Iterationen durch einen Ingenieur mit Hilfe des JigSCI-Verfahrens untersucht. Während der Vervollständigung wurden die Antworten des Ingenieurs aufgezeichnet und die relative strukturelle und funktionale Vollständigkeit bezüglich der einzelnen Subsysteme durch zufällig generierte Testspezifikationen im Verhältnis zum Interaktionsaufwand gemessen.

Zu Beginn der Fallstudie betrug die funktionale Vollständigkeit der Eingabespezifikation $FV_{PSS} = 55,1\%$ bezüglich des PSS-Systems und $FV_{APG} = 50,7\%$ im Kontext des Gesamtsystems APG. Die Szenarien der Spezifikation haben weiterhin vier der 34 Transitionen des gesuchten LTS A_{APG} abgedeckt. Die strukturelle Vollständigkeit war daher $SV_{PSS} = 40\%$ bzw. $SV_{APG} = 12,5\%$.

Die insgesamt 78 gestellten Anfragen (s. Tabelle 6) und die zu Beginn jeder Iteration hinzugefügten Szenarien steigerten die strukturelle Vollständigkeit der PSS-Spezifikation insgesamt von 12,5% auf 94,1% um 81,6 Prozentpunkte bezüglich des APG-Systems. Von 34 Transitionen wurden lediglich zwei Transitionen nicht erkannt. Die Merge-Validation-Technik vervollständigte die resultierende Szenariospezifikation soweit, das aus dieser ein Transitionssystem eindeutig abgeleitet werden kann, das nicht übergeneralisiert ist und nur noch eine lokal optimale Zustandsvereinigung aufweist. Die funktionale Vollständigkeit wurde dabei insgesamt von 50,7% auf 58,7% bezüglich des APG-Systems gesteigert. Diese geringe Anhebung des Vollständigkeitsmaßes wurde auf eine fehlende Transition zurückgeführt, über die im statistischen Mittel die Hälfte aller positiven Szenarien einer zufälligen Testspezifikation verlaufen. Vernachlässigen wir diese Transition, wurde die funktionale Vollständigkeit der Spezifikation um 41,4 Prozentpunkte auf 92,1% gesteigert. Der Ingenieur benötigte zur Beantwortung dieser Anfragen etwa 2,5 Stunden.

	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Σ
Scenario-Puzzling-Anfragen	26	24	28	0	78
Mergevalidation-Anfragen	11	53	81	56	201

Tabelle 6: Der Interaktionsaufwand der kompletten Fallstudie

6 Zusammenfassung und Ausblick

Wir haben in diesem Papier gezeigt, wie mit Hilfe einer automatisiertes Vollständigkeitsanalyse die Szenariospezifikation eines Parkassistenten vervollständigt werden kann. Mit angemessenem Aufwand können fehlendes Verhalten und unbeschriebene Kompositionsinformationen nicht-trivialer Szenariospezifikationen zu einem hohen Grad vervollständigt werden. Der Algorithmus rekombiniert während seiner Induktion das Verhalten der Szenarien auf Basis derer Ähnlichkeit und erschließt dadurch neues, fehlendes Verhalten und Szenariokompositionen. Wir haben das erschlossene Verhalten des Algorithmus in sze-

ISBN 978-3-88579-286-4

nariobasierten Anfragen dargestellt und die Szenariospezifikationen durch die Antworten von Autoren und Gutachtern um fehlende Szenarien ergänzt. Auf Basis der Antworten der Ingenieure werden Spezifikationslücken erkannt und sowohl Über- als auch Untergeneralisierung verhindert.

Durch den Einsatz des JigSCI-Verfahrens kann eine Szenariospezifikation annähernd vervollständigt und aus dieser Spezifikation automatisch ein Transitionssystem abgeleitet werden. Wie wir in automatisierten Experimenten gezeigt haben [Cha11], liegt der Aufwand für Spezifikationen mit 400 Ereignissen für die Ingenieure bei bis zu zwei Arbeitstagen, wobei die Anfragen korrekt beantwortet werden müssen. Das Verfahren sollte weiter verbessert werden, um den benötigten Aufwand zu reduzieren und inkorrekte Antworten von Ingenieuren zu berücksichtigen. Weiterhin existieren zahlreiche Einsatzgebiete, wie zum Beispiel die Unterstützung im Entwurf von Testfällen oder zum Reverse Engineering von Zustandsautomaten, die von einer Vervollständigung des JigSCI-Verfahrens profitieren können. In zukünftigen Forschungsarbeiten sollte daher die Erweiterung und Verbesserung, die Berücksichtigung von fehlerhaften Eingaben sowie weitere Einsatzgebiete des JigSCI-Verfahrens betrachtet werden.

Literatur

[AE03] Daniel Amyot und Armin Eberlein. An Evaluation of Scenario Notations and Construction Approaches for Telecommunication Systems Development. *Telecommunication Systems*, 24(1):61–94, 2003.

[Ang88] Dana Angluin. Queries and Concept Learning. *Machine Learning*, 2(4):319–342, 1988.

[CFGK05] Mirko Conrad, Ines Fey, Matthias Grochtmann und Torsten Klein. Modellbasierte Entwicklung eingebetteter Fahrzeugsoftware bei DaimlerChrysler. *Informatik - Forschung und Entwicklung*, 20:3–10, 2005.

[Cha11] Fadi Chabarek. *Interaktive Vervollständigung von Szenariospezifikationen*. Dissertation, Technische Universität Berlin, 2011.

[Die08] Björn Dietrich. Funktionsorientierung nimmt der Automobilindustrie Komplexität. *Economic Engineering*, 6:28–31, 2008.

[Dup96] Pierre Dupont. *Utilization et Apprentissage de Modulès de Langage pour la Reconnaissance de la Parole Continue*. Dissertation, Ecole Nationale Supérieure des Télécommunications, 1996.

[Gri03] Klaus Grimm. Software technology in an automotive company: major challenges. In *ICSE '03: Proceedings of the 25th International Conference on Software Engineering*, Seiten 498–503. IEEE Computer Society, 2003.

[LPP98] Kevin J. Lang, Barak A. Pearlmutter und Rodney Price. Results of the Abbadingo One DFA Learning Competition and a New Evidence-Driven State Merging Algorithm. In *ICGI '98: Proceedings of the 4th International Colloquium on Grammatical Inference*, Jgg. 1433 of *LNCS*, Seiten 1–12. Springer-Verlag, 1998.

[WK04] Jon Whittle und Ingolf H. Krüger. A Methodology for Scenario-Based Requirements Capture. In *SCESM '04: Proceedings of the 3rd International Workshop on Scenarios and State Machines: Models, Algorithms and Tools, ICSE'04*, Seiten 2–7, 2004.

erschienen im Tagungsband der INFORMATIK 2011
Lecture Notes in Informatics, Band P192
ISBN 978-3-88579-286-4
weitere Artikel online:
http://informatik2011.de/519.html