Ein Werkzeug zur Analyse von Feature-Interaktionen in IN

Joachim Thees und Jan Bredereke

Universität Kaiserslautern, FB Informatik, Pf. 3049, D-67653 Kaiserslautern
E-Mail: {thees,bredereke}@informatik.uni-kl.de
World-Wide-Web: http://uklirb.informatik.uni-kl.de/aggotz

Zusammenfassung
In diesem Aufsatz wird die Arbeitsweise eines Werkzeuges dargestellt, mit dessen Hilfe die Analyse von Feature-Interaktionen in Intelligenten (Telefon-)Netzwerken unterstützt wird. Dieses Werkzeug basiert auf einem von uns entwickelten formalen Lösungsansatz, der aus einem geeigneten Spezifikationsstil, aus einem formalen Kriterium zur Erkennung von Feature-Interaktionen und aus einer Methode zur Auflösung der erkannten Feature-Interaktionen besteht. Das Werkzeug führt eine statische Analyse von Estelle-Spezifikationen durch und erkennt dabei potientielle Feature-Interaktionen sowie nichtausführbare Transitionen. Darüberhinaus kann es die erkannten nichtausführbaren Transitionen zur Optimierung aus der Spezifikation entfernen. Wir erläutern zunächst kurz den zugrundeliegenden Ansatz und beschreiben danach die Anwendung auf Estelle anhand der Funktionsweise des Werkzeuges.

1. Einleitung

Telefonvermittlungssysteme sind ein klassisches Beispiel für langlebige und sich ständig weiterentwickelnde Software im Kommunikationsbereich. Nach Einführung der speicherprogrammierten Vermittlungsstellen stellten diese zunächst weiterhin den einfachen alten Telefondienst (plain old telephone service, POTS) zur Verfügung. Nach und nach wurden dann immer neue Leistungsmerkmale (Features) hinzugefügt, die den Kunden (z.B. bei Anrufweiterleitung) und/oder dem Dienstanbieter (z.B. bei verbesserter Abrechnung) einen zusätzlichen Nutzen bringen sollten.[1] Der Begriff des Features wird dabei recht verschieden benutzt, abhängig von der Sichtweise. Cameron und Velthuijsen [CaVe93] etwa beschreiben ein Feature aus der Marketing-Sicht als etwas, wofür man Gebühren erheben kann, und aus der Sicht des Implementierers als ein beliebiges Inkrement an Funktionalität für ein existierendes System.

Inzwischen gibt es so viele zusätzliche Features (einige hundert, [Bo$^+$89]), insbesondere bei den amerikanischen Telefongesellschaften. Daher ist die Wahrscheinlichkeit sehr groß, daß das Hinzufügen eines weiteren Features zum System ein anderes Feature beeinflussen wird, insbesondere auch negativ. Dies bezeichnet man als Feature-Interaktion. Sowohl der Begriff des Features als auch der der Feature-Interaktion werden recht unscharf und informell verwendet, was eine Lösung des Problems nicht erleichtert. (Wir arbeiten z.Z.

[1]Beispiel: Die ITU-T-Standards zu Intelligenten Netzwerken (IN) [ITU93].

an einer Formalisierung, siehe [Bre95].) Man kann aber auf jeden Fall festhalten (siehe [CaVe93]), daß eine Feature-Interaktion dann auftritt, wenn das Verhalten eines Features durch den Gebrauch eines anderen Features verändert wird.

Im nächsten Kapitel skizzieren wir kurz einen von uns entwickelten Lösungsansatz ([BrGo94a, BrGo94b]) für das Feature-Interaktions-Problem in Intelligenten Netzwerken (IN). In Kapitel 3 beschreiben wir unsere Umsetzung dieses Ansatzes in das Werkzeug Confine. Dieses Werkzeug analysiert Spezifikationen von Telefonvermittlungssystemen, die in der Formalen Beschreibungstechnik Estelle ([ISO89]) abgefaßt sind. In Kapitel 4 werden kurz seine Implementation und erste Erfahrungen mit ihm beschrieben, gefolgt von einer Zusammenfassung in Kapitel 5.

2. Das Feature-Interaktions-Problem in IN

Basierend auf dem in Kapitel 1 beschriebenen Begriff des Verhaltens und seiner Veränderung haben wir einen Lösungsansatz entwickelt ([BrGo94a, BrGo94b]). Er besteht darin, bei der Entwicklung eines Systems einen geeigneten Spezifikationsstil zu verwenden, der es dann erlaubt, ein formales Kriterium anzuwenden, um zu prüfen, ob Feature-Interaktionen möglich sind. Anschließend werden die erkannten Feature-Interaktionen aufgelöst. Die Grundidee des angesprochenen Spezifikationsstils ist,

1. Erweiterungen eines Systems stets nur mit einer groben Granularität vorzunehmen und

2. nur etwas hinzuzufügen, aber niemals vorhandenes zu modifizieren oder gar zu entfernen.

Bei dem Verhalten eines Systems beschränken wir uns auf die funktionalen Aspekte. Die Vorhersage von Feature-Interaktionen z.B. im Bereich der Performance ist eine noch weitgehend offene Frage für die Forschung. Damit beschreiben wir Verhalten als eine Menge von Berechnungsfolgen, repräsentiert durch einen globalen Automaten. Praktisch spezifiziert wird dieser globale Automat mit einer Estelle-Spezifikation.

Bisher haben wir unseren Ansatz auf Modulebene und für die FDT Estelle ausgearbeitet, eine Verallgemeinerung auf ganze Systeme ist in Vorbereitung ([Bre95]). Auf Modulebene bedeutet „grobe Granularität", daß wir stets nur um ganze Transitionen erweitern und niemals z.B. einzelne Anweisungen innerhalb einer Transition modifizieren. Punkt 2 bedeutet entsprechend, daß wir nur Transitionen hinzufügen, aber keine verändern oder entfernen. Wenn letzteres notwendig werden sollte, ist dies ein Fall einer Auflösung von Feature-Interaktionen; hierzu kommen wir weiter unten.

Indem wir beim Hinzufügen eines Features zu einem System stets nur ganze Transitionen einfügen, können wir umgekehrt jede Transition eindeutig einem Feature zuordnen, was wir auch syntaktisch tun. (Den Basisdienst betrachten wir aus Gründen der Orthogonalität ebenfalls als Feature, das immer vorhanden sein muß.) Damit ist es möglich, zu jeder Berechnungsfolge von Transitionen eines Moduls die beteiligten Features anzugeben.

Wenn wir die obigen Stilregeln (sowie einige speziellere, hier nicht angeführte) beachten, erreichen wir, daß das Hinzufügen eines Features zum System stets nur neue, zusätzliche Berechnungsfolgen möglich macht, ohne alte unmöglich zu machen. Die einzige Art, auf

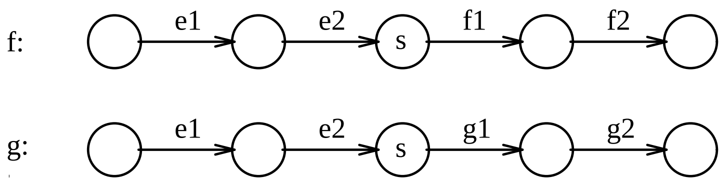

Abbildung 1. Indeterminismus bei Zustand s zwischen der neuen Transition g_1 und der alten Transition f_1.

die das Verhalten eines vorhandenen Systems (mit einer Familie F_i untereinander interaktionsfreier Features) durch ein neues, hinzugefügtes Feature G somit verändert werden kann, ist damit die folgende: Sei g eine Berechnungsfolge des *erweiterten* Systems, zu der (u.a.) das Feature G gehört. Dann muß es einen Zustand s geben, von dem aus eine Transition g_1 von G schaltet, und vor dem keine Transition von G mehr vorkommt (siehe Abb. 1). Wir nehmen an, daß das System auch ohne das Feature G niemals terminiert[2], so daß es eine andere Berechnungsfolge f geben muß, deren Präfix bis s genauso aussieht, die aber mit einer Transition f_1 (eines Features F_i) des bisherigen Systems fortsetzt. Damit muß es im Zustand s eine indeterministische Entscheidung zwischen einer Transition, die zum Feature G gehört, und einer Transition eines anderen Features geben. Voraussetzung hierfür ist, daß wir keine Prioritäten zwischen Transitionen definiert haben, was in Estelle möglich wäre, aber durch eine unserer Stilregeln untersagt wird.

Damit haben wir ein *formales Kriterium* für Feature-Interaktionen: Eine Feature-Interaktion kann nur auftreten, wenn es einen (erreichbaren) Zustand gibt, in dem mindestens zwei Transitionen aus verschiedenen Features gleichzeitig schalten können, d.h. wenn es Indeterminismus zwischen verschiedenen Features gibt.

Man beachte, daß dieses Kriterium eine hinreichende, aber mehr als notwendige Bedingung für *unerwünschte* Feature-Interaktionen ist. Es ist möglich, daß unser Kriterium eine Warnung für eine Stelle erzeugt, die sich bei manueller Inspektion als harmlos erweist. Eine vollständig automatische Analyse ist aber i.a. schon aus prinzipiellen Gründen unmöglich, da sie voraussetzen würde, daß die Wünsche und Vorstellungen der Dienstnutzer und -anbieter vollständig formalisiert sind. Gerade im Bereich der Telefonvermittlungssysteme stößt man bei Erweiterungen aber immer wieder auf das Problem, daß früher implizite Annahmen (z.B. über den Basisdienst) gemacht wurden, die nun nicht mehr gelten, und daß selbst Grundbegriffe sich plötzlich in ihrer Bedeutung wandeln.[3] Mit geeigneten Erkennungskriterien können wir allerdings erreichen, daß wir nur noch eine relativ kurze Liste kritischer Punkte von Hand analysieren und ggf. auflösen müssen. Darüberhinaus können wir diese Arbeit sogar auf verschiedene Experten für die jeweils betroffenen Features verteilen, wir benötigen hier keinen Experten mehr, der das gesamte, große System sicher überblickt.

Eine Methode zur Auflösung von unerwünschten Feature-Interaktionen ist, für jedes Paar von Features zu definieren, welches vor welchem Vorrang hat. Dies kann man als

[2]Wir vernachlässigen hier den Fall endlicher Berechnungsfolgen, da er für reaktive Systeme uninteressant ist.

[3]Beispiel: Der Begriff des Anrufers. Wenn Kunde A den Kunden B anruft, ist A der Anrufer. Wenn aber B das Feature der Anrufweiterleitung nutzt, und eine weitere Verbindung zu C aufbaut, wer ist dann für C der Anrufer? Ist es B, der diese Verbindung aufgebaut hat, oder A, mit dem er sprechen wird?

eine Entwurfsentscheidung bei der Entwicklung von Features ansehen, sie benötigt Hintergrundwissen über die Strategien und Ansichten des Dienstanbieters, folglich kann sie nicht automatisiert werden.

Sehr wohl kann allerdings die Umsetzung dieser Entscheidungen automatisiert werden. Dazu haben wir ein Hilfswerkzeug entwickelt, das aus einer Präzedenzmatrix automatisch den notwendigen Estelle-Code erzeugt, um die Transitionen der verschiedenen Features mit jeweils geeigneten Prioritätswerten zu versehen. In die Präzedenzmatrix müssen nur die tatsächlich getroffenen Entwurfsentscheidungen eingetragen werden. Falls zwischen zwei Features keine Präzedenz notwendig ist, wird möglicherweise auch die gleiche Prioritätsstufe generiert.

Manchmal reicht es nicht aus, eine Präzedenz zwischen zwei in Konflikt stehenden Features zu definieren. Es kann notwendig sein, völlig neues Verhalten einzuführen, um die komplexe Situation richtig zu behandeln. Dieses Hinzufügen neuen Verhaltens behandeln wir formal genauso wie das Hinzufügen eines weiteren Features. Dieses „Feature“ muß genau dann selektiert sein, wenn auch die beiden in Konflikt stehenden Features selektiert sind, und es muß eine (etwas) höhere Priorität als die beiden haben.

Zur Zeit entwickeln wir weitere, verfeinerte Erkennungskriterien ([Bre95]). Sie liefern detailliertere Aussagen für den Fall, daß durch ein neues Feature G die Berechnung nicht nur von einem Feature F weggelenkt wird, wie in Abbildung 1 gezeigt, sondern indirekt auch von einem weiteren Feature H, dessen Transitionen aber erst einige Schritte nach dem Zustand s aus Abbildung 1 zum Zuge kommen.

Außerdem erweitern wir derzeit das obige Indeterminismus-Erkennungskriterium insofern, daß es auch anwendbar wird, wenn bereits eine Runde von Erkennung und Auflösung stattgefunden hat, wodurch Prioritäten zwischen Transitionen eingeführt werden. Durch die totale Ordnung der Prioritätswerte ist es nicht immer möglich, dann noch ein neues Feature einzuführen, das die gleiche Priorität wie alle bereits vorhandenen Features hat und für das dementsprechend per Indeterminismus mögliche Feature-Interaktionen aufgezeigt werden können. Dieses Problem wird derzeit dadurch gelöst, daß unser Analysewerkzeug Confine (Kap. 3 und 4) nicht nur nach Indeterminismus sucht, sondern auch nach *Überlappungen*, also Fällen, wo zwischen zwei Transitionen verschiedener Features Inderminismus nur dadurch vermieden wird, daß ihnen verschiedene Prioritäten zugeordnet sind.

Als weitere Verfeinerung ist geplant, bei den Beziehungen zwischen den verschiedenen Features nicht mehr die Estelle-Prioritäten zu beachten, sondern die ursprüngliche Präzedenzrelation. Dies ist im im folgenden beschriebenen Analysewerkzeug Confine allerdings noch nicht realisiert.

3. Erkennung von Indeterminismus und Überlappungen

Zur Unterstützung des Analyseverfahrens aus Kapitel 2 haben wir unter anderem das Werkzeug Confine („reCOgNition of Feature Interactions in intelligent Networks with Estelle“) entwickelt ([The95]), mit dessen Hilfe Estelle-Spezifikationen, die dem dortigen Spezifikationsstil entsprechen, statisch auf folgende Punkte untersucht werden können:

- Erkennung von möglichem **Indeterminismus** zwischen Transitionen aus verschiedenen Features
- Erkennung von möglichen **Überlappungen** zwischen Transitionen aus verschiedenen Features
- Erkennung von vollständig überlappten und damit nicht ausführbaren Transitionen

Transitionen, die gemäß des letzten Punktes als nicht ausführbar erkannt wurden, können zudem durch CONFINE aus der Spezifikation entfernt werden. Dadurch ist eine Optimierung der Spezifikation möglich, da derartige Transitionen bei der Ausführung einer Implementation der Spezifikation i.a. regelmäßig auf ihre Schaltbarkeit geprüft werden, ohne jedoch jemals ausgewählt werden zu können. Vollständig überlappte Transitionen ergeben sich unter dem obigen Spezifikationsstil immer dann, wenn Transitionen durch Hinzufügen neuer Transitionen höherer Priorität vollständig ersetzt werden sollen.

CONFINE verfolgt dabei zwei grundlegende Strategien: (Potentielle) Überlappungen und (potentieller) Indeterminismus werden immer dann gemeldet, wenn sie nicht *sicher* ausgeschlossen werden können (**offensive Meldestrategie**) und Transitionen werden nur dann als nichtausführbar erkannt und gemeldet, wenn sie *sicher* nicht ausgeführt werden können (**defensive Optimierungsstrategie**). Während der erste Punkt garantiert, daß sich sämtliche Feature-Interaktionen in Überlappungs- und Indeterminismusmeldungen widerspiegeln, sichert der zweite Punkt die Korrektheit[4] des Optimierungsschritts. Diese beiden Strategien werden es später ermöglichen, die anfangs genannten Analyseziele auf die Frage nach der *(prädikatenlogischen) Erfüllbarkeit existenzquantifizierter Aussagen* zurückzuführen.

Aufgrund der mangelnden Berechenbarkeit vieler Teilaspekte und der hohen Komplexität der gesamten Problemstellung sind einige Einschränkungen bei der Analyse unerläßlich. So ist u.a. nicht entscheidbar, ob ein beliebiger globaler Zustand erreichbar ist. Daher schließen wir zunächst zum einen die Transitionsblöcke[5] aus der Untersuchung aus und führen zum anderen die Analyse nur **modullokal** durch, d.h. wir betrachten die Transitionen jedes Modulrumpfs separat.

Bei der Analyse werden daher die einzelnen Module isoliert voneinander und lediglich anhand der Klauseln ihrer Transitionen untersucht. Im nächsten Abschnitt wird gezeigt, wie dabei die gewünschten Bedingungen für Indeterminismus, Überlappungen und die Nichtausführbarkeit von Transitionen gewonnen werden können.

3.1. Ableitung von Bedingungen

Bei der (modullokalen) Analyse müssen die Transitionsklauseln von Transitionen verschiedener Features miteinander in Beziehung gesetzt werden. Dabei sollen zunächst möglicher Indeterminismus und mögliche Überlappungen erkannt werden. *Indeterminismus* im hier gebrauchten Sinne liegt in einem Modul vor, wenn mindestens zwei Transitionen (verschiedener Features) zugleich schaltbar sind und daher nicht feststeht, welche

[4] Korrektheit hier: Keine Semantikänderung durch die Optimierung.

[5] Transitionsblöcke beschreiben Schaltwirkungen, welche über die Veränderung des Hauptzustandes hinausgehen.

der Transitionen ausgewählt wird. Wenn die Transitionen verschiedene Prioritäten haben und nur dieser Umstand Indeterminismus in einem Zustand verhindert, so liegt eine *Überlappung* vor.

Mit einem **Zustand** wird hier die Gesamtheit aller Hauptzustände, Variablenwerte, Modulinstanziierungen, Verbindungsstrukturen und Warteschlangeninhalte der gesamten Spezifikation bezeichnet.[6] Ein konkreter Zustand bestimmt also, welche Transitionen bereit bzw. schaltbar werden.

Die **lokale Schaltbedingung einer Transition** beschreibt eine Teilmenge des Zustandsraumes, in der die Transitionsklauseln `WHEN`, `FROM` und `PROVIDED` erfüllt sind. In Abbildung 2 wurde der Zustandsraum eindimensional als horizontale Achse dargestellt. Die Transitionen belegen eine Teilmenge[7] dieser Achse. Auf der vertikalen Achse wurde die Priorität der Transitionen angegeben. Jede Transition hat dabei genau eine Priorität.

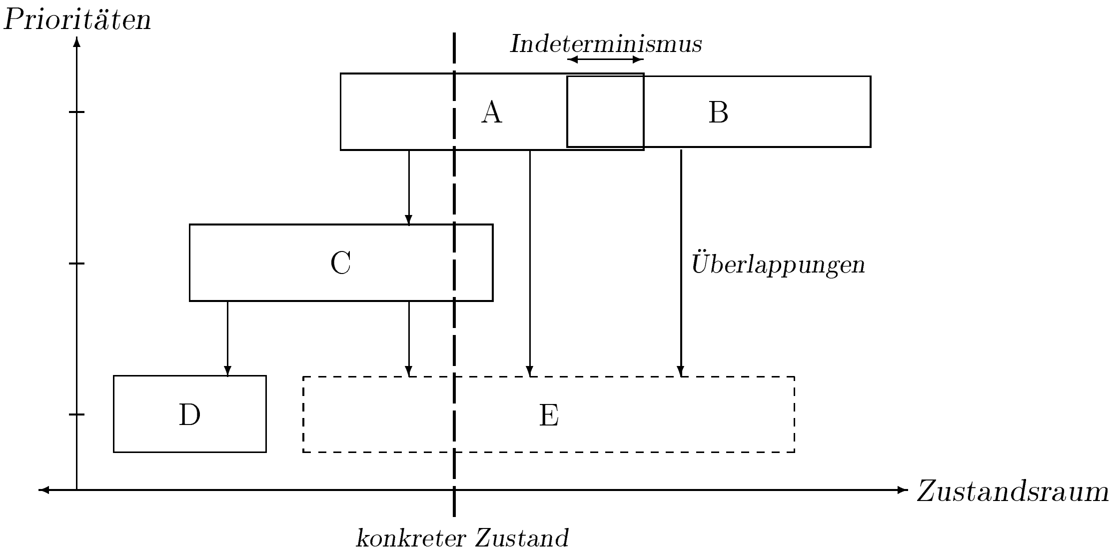

Abbildung 2. Überlappungen und Indeterminismus zwischen Transitionen

Man erkennt leicht, daß Indeterminismen (bzw. Überlappungen) sich an gemeinsam erfüllbaren lokalen Schaltbedingungen zeigen, wobei die Prioritäten gleich (bzw. verschieden) sein müssen. In der Abbildung besteht Indeterminismus zwischen A und B, (partielle) Überlappungen gibt es auf C (durch A), D (durch C) und E (durch A, B, C).

Im folgenden bezeichnen wir mit *Interferenz* entweder Indeterminismus oder eine Überlappung. Anhand von Abbildung 2 kann man sich einige Eigenschaften solcher Interferenzen verdeutlichen: Gibt es einen Zustand, in dem mehrere Transitionen gleicher Priorität zugleich schaltbar werden können, so besteht der Indeterminismus zwischen *jedem Paar* dieser Transitionen. Ebenso besteht zu einer Transition, die in einem Zustand durch mehrere andere Transitionen überlappt wird, die Überlappungsbeziehung *paarweise* mit jeder einzelnen dieser Transitionen höherer Priorität.

Es genügt also zur Erkennung von Interferenzen, die *Transitionen nur paarweise* zu *vergleichen*. Es ist daher nicht notwendig, komplexe Interaktionen zwischen drei oder mehr Transitionen zu untersuchen. Dies ist eine wesentliche Vereinfachung der Analyse.

[6] Auf die Modellierung von Zeitintervallen gehen wir hier nicht ein.

[7] Im Bild nur zusammenhängend dargestellt, im allgemeinen jedoch fragmentiert.

Vollständige Überlappungen sind dagegen durch den paarweisen Vergleich von Transitionen nicht immer unmittelbar zu erkennen.[8] Schränkt man die lokale Schaltbedingung einer Transition jedoch um all diejenigen Zustände ein, in denen sie überlappt wird, so erhält man die **effektive Schaltbedingung**. Diese kann durch *sukzessiven paarweisen Vergleich* der Transitionen ermittelt werden. Vollständige Überlappungen manifestieren sich dann (indirekt) als unerfüllbare effektive Schaltbedingungen.

Somit kann die modullokale Analyse vollständig auf den paarweisen Vergleich von Transitionen reduziert werden. Dazu werden jeweils die Transitionsklauseln gleicher Art paarweise miteinander verglichen[9] und dabei bestimmt, ob diese überhaupt *zueinander passen* (d.h. ob sie überhaupt gleichzeitig erfüllbar sind)[10] und welche Bedingungen ggfs. dazu erfüllt sein müssen[11]. Anhand der PRIORITY-Klauseln wird zudem festgestellt, ob es sich um eine Überlappung oder um Indeterminismus handeln kann. Passen alle Klauseln des Transitionspaares zueinander, so liefert die Konjunktion der aus den Klauseln abgeleiteten Bedingungen schließlich die Bedingung für den potentiellen Indeterminismus bzw. die potentielle Überlappung zwischen den beiden Transitionen. Diese Bedingungen werden dabei auf boolwertige Ausdrücke und Hauptzustände eingeschränkt. (Die übrigen Aspekte der Interferenzbedingungen werden bereits vorab behandelt.) Kann kein Widerspruch in dieser abgeleiteten Bedingung erkannt werden (s.u.), so wird eine entsprechende Meldung in das Analyseprotokoll aufgenommen. Diese Vorgehensweise erfüllt die offensive Meldestrategie, da so alle Interferenzen gemeldet werden, die nicht sicher widerlegt werden konnten.

Schränkt man die Schaltbedingungen einer Transition um all die Bedingungen ein, in denen die Transition überlappt wird, so erhält man die bereits erwähnte effektive Schaltbedingung (diese wird ebenfalls auf boolwertige Ausdrücke und Hauptzustände eingeschränkt). Kann sie als sicher unerfüllbar nachgewiesen werden, so ist die Transition nicht ausführbar und kann zur Optimierung aus der Spezifikation entfernt werden. Dies entspricht der geforderten defensiven Optimierungsstrategie.

3.2. Auswertung der Bedingungen

Es wurde gezeigt, wie aus der Spezifikation Bedingungen für Indeterminismus, Überlappungen und die Ausführbarkeit von Transitionen gewonnen werden. Es muß nun für jede Bedingung geprüft werden, ob der durch die Spezifikation beschriebene globale Automat überhaupt einen Zustand erreichen kann, in dem diese Bedingung erfüllt ist.

Aufgrund der modullokalen Analyse ohne Berücksichtigung der Transitionsblöcke stehen jedoch nur wenige[12] Informationen über die Erreichbarkeit eines (globalen oder lokalen) Zustandes zur Verfügung. Die Bedingungen werden daher auf ein schwächeres Prädikat, die *prädikatenlogische Erfüllbarkeit*[13], geprüft.

[8] Siehe Abbildung 2: Vollständige Überlappung von E durch A, B und C *gemeinsam*.
[9] Nicht explizit spezifizierte Transitionsklauseln werden dabei natürlich auch berücksichtigt.
[10] Z.B. passen zwei WHEN-Klauseln, die verschiedene Interaktionen aus der selben Warteschlange erwarten, nicht zueinander.
[11] Z.B. bei PROVIDED-Klauseln die Konjunktion ihrer Bedingungen.
[12] In CONFINE werden nur Informationen über die Erreichbarkeit von Hauptzuständen selbständig ermittelt.
[13] Erfüllbarkeit hier immer in der durch die Estelle-Semantik definierten Algebra.

Um dennoch die Erreichbarkeit eines Zustandes mit in die Analyse einfließen lassen zu können, haben wir die Möglichkeit geschaffen, bestimmte invariante Bedingungen explizit formulieren zu können. Dazu kann jeder Transition eine argumentlose boolwertige Funktion zugeordnet werden, die Bedingungen beschreibt, die beim Schalten der Transition garantiert werden (siehe Abbildung 3). Diese **Zusicherungen** werden von CONFINE berücksichtigt, haben sonst jedoch Kommentarcharakter. Es besteht auch die Möglichkeit, die Korrektheit der spezifizierten Zusicherungen mit Hilfe automatisch erzeugter Prototypimplementationen der Spezifikation zu testen.[14]

```
VAR x: ARRAY [1..10] OF INTEGER;

TRANS
   WHEN ipx.msgx(a, b)
   FUNCTION trans_assertion:BOOLEAN;
      BEGIN
      IF (a >= 1) AND (a <= 10) THEN
         trans_assertion := (b>=0) AND (b<=x[a])
      ELSE
         trans_assertion := TRUE
      END;
   NAME t:
      BEGIN  { Zu Beginn des Schaltens der Transition wird garantiert: }
      ....   {   Wenn a im Indexbereich von x liegt, dann }
      END    {   gilt (b>=0) und (b<=x[a]). }
```

Abbildung 3. Beispiel für eine Zusicherungsfunktion in einer Transition

Die Untersuchung auf prädikatenlogische Erfüllbarkeit erfolgt in drei Stufen: Zunächst werden die ermittelten Bedingungen (in Form boolwertiger Estelle-Ausdrücke) mit Hilfe eines *Termersetzungssystems* bearbeitet. Dabei werden u.a. Funktionsaufrufe soweit wie möglich zu geschlossenen Ausdrücken expandiert. Die Termersetzung dient dabei in erster Linie dazu, die Ausdrücke für die darauffolgende *aussagenlogische Untersuchung* aufzubereiten.[15] Als Ergebnis dieses zweiten Schritts liegen die Bedingungen als *Disjunktive Normalformen* (DNF) vor. Anschließend werden die einzelnen Konjunktionen der DNFs separat auf bestimmte Klassen von *prädikatenlogischen Widersprüchen*[16] untersucht.

Konnte eine Bedingung nicht als widersprüchlich nachgewiesen werden, so gilt sie als *potentiell erfüllbar*, und eine entsprechende Interferenzmeldung wird ausgegeben (potentiell erfüllbare Interferenzbedingung) bzw. die Transition kann nicht entfernt werden (potentiell erfüllbare effektive Schaltbedingung). Diese Vorgehensweise entspricht den o.g. Analysestrategien.

[14] Z.B. durch eine geringfügige Erweiterung von DINGO (siehe Kapitel 4).

[15] Z.B. werden AL-Atome wie $b_1 = b_2$ (mit boolwertigen Ausdrücken b_1 und b_2) durch äquivalente AL-zerlegbare Ausdrücke ersetzt (hier: $(b_1 \wedge b_2) \vee (\neg b_1 \wedge \neg b_2)$).

[16] In erster Linie die Verletzung der Relationseigenschaften von $=, \neq, >, \geq, \ldots$

4. Implementation

Das Analysewerkzeug Confine ([The95]) wurde aufbauend auf das PET-DINGO-Toolkit in C++ realisiert (ca. 7800 Zeilen, 35 Klassen). PET-DINGO ([SiSt90]) ist ein Werkzeug zur Erzeugung von (verteilten) Prototyp-Implementationen von Estelle-Spezifikationen. Die wichtigsten Komponenten sind der Estelle-Parser PET („Portable Estelle Translator“) und der C++-Codegenerator DINGO („Distributed Implementation Generator“).

Confine erhält als Eingabe eine Spezifikation in Form einer PET-Objektdatei und liefert neben dem Analyseprotokoll eine optimierte Spezifikation (ebenfalls in Form einer PET-Objektdatei, siehe Abbildung 4). Diese kann dann direkt als Eingabe für DINGO dienen oder mit Hilfe des Werkzeugs PDRESTORE in einen Estelle-Quelltext zurückgewandelt werden.

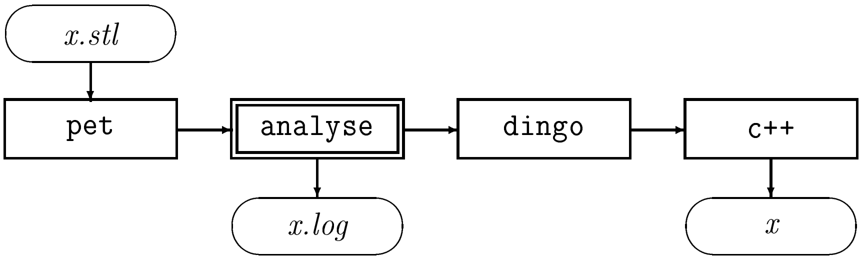

Abbildung 4. Das Analysewerkzeug Confine im Datenfluß von PET-DINGO

Erste Erfahrungen mit dem Einsatz unseres Werkzeugs Confine liegen bereits vor. Es wurde auf eine stark vereinfachte Estelle-Spezifikation eines Telefonvermittlungssystems angewandt. Dabei wurden nicht nur die bereits vorher bekannten Feature-Interaktionen gefunden, sondern auch zwei Interferenzen, die uns beim Spezifizieren dieses einfachen Beispiels entgangen waren. Die manuelle Analyse der Situation ergab anschließend, daß beide Fälle von Interferenzen harmlos sind, sofern die Implementation genügend schnell abläuft. Trotzdem haben wir dabei ein tieferes Verständnis für einige Zusammenhänge gewonnen. Weiterhin konnte Confine automatisch die inaktiven Transitionen erkennen und entfernen, die durch die Auflösung der bekannten Feature-Interaktionen entstanden waren. Zur Zeit sind wir dabei, ein detaillierteres, auf den ITU-T-Standards für Intelligente Netzwerke [ITU93] basierendes Telefonvermittlungssystem zu spezifizieren und werden in diesem anschließend ebenfalls die Feature-Interaktionen analysieren.

5. Zusammenfassung

Das Problem der Feature-Interaktionen in Intelligenten (Telefon-)Netzwerken behindert derzeit immer stärker die Weiterentwicklung von Telefonvermittlungssystemen. Wir haben kurz einen Lösungsansatz skizziert, der aus einem geeigneten Spezifikationsstil, aus einem formalen Kriterium zur Erkennung von Feature-Interaktionen und aus einer Methode zur Auflösung der erkannten Feature-Interaktionen besteht. Anschließend haben wir die Umsetzung des Erkennungskriteriums in das Werkzeug Confine beschrieben, das Estelle-Spezifikationen automatisch analysieren kann. Aufgrund der mangelnden Berechenbarkeit vieler Teilaspekte wird eine offensive Meldestrategie verfolgt, die alle potentiellen

Feature-Interaktionen anzeigt. CONFINE ist darüberhinaus in der Lage, Transitionen, die als nichtausführbar erkannt wurden, aus der Estelle-Spezifikation zu entfernen. Dabei wird entsprechend eine defensive Optimierungsstrategie verfolgt. Um die Analyse zu unterstützen, wurde außerdem ein Verfahren entwickelt, wie man Estelle-Spezifikationen formal um Zusicherungen über Invarianten erweitern kann. Derartige Zusicherungen über Eigenschaften, die ansonsten nur indirekt aus dem Spezifikationstext ableitbar wären, sind auch weit über das Anwendungsfeld dieses Aufsatzes hinaus verwendbar, um abstraktere Eigenschaften des in Estelle spezifizierten Systems explizit, formal und eigenschaftsorientiert auszudrücken. Es liegen bereits erste Erfahrungen mit dem Einsatz von CONFINE vor. In einer einfachen Fallstudie wurden nicht nur die bereits vorher bekannten Feature-Interaktionen gefunden, sondern auch zwei (relativ harmlose) Interferenzen, die uns beim Spezifizieren entgangen waren. CONFINE konnte wie erwartet nichtausführbare Transitionen erkennen und entfernen. Zur Zeit arbeiten wir an einer umfangreicheren Fallstudie.

Literatur

[Bo+89] Bowen, T. F. u. a.. *The feature interaction problem in telecommunication systems*. In „Seventh IEEE International Conference on Software Engineering for Telecommunication Systems“ (Juli 1989).

[BoVe94] Bouma, L. G. und Velthuijsen, H. (Hrsg.). *Feature Interactions in Telecommunications Systems*. IOS Press, Amsterdam (1994).

[Bre95] Bredereke, J. *Automata-theoretic criteria for feature interactions*. Int. Ber., Univ. Kaiserslautern, FB Informatik (1995). In Vorbereitung.

[BrGo94a] Bredereke, J. und Gotzhein, R. *A case study on specification, detection and resolution of IN feature interactions with Estelle*. Int. Ber. 245/94, Univ. Kaiserslautern, FB Informatik (Mai 1994).

[BrGo94b] Bredereke, J. und Gotzhein, R. *Specification, detection and resolution of IN feature interactions with Estelle*. In Hogrefe, D. und Leue, S. (Hrsg.), „FORTE '94, Proceedings“, S. 366–368, Bern, Schweiz (4.–7. Okt. 1994).

[CaVe93] Cameron, E. J. und Velthuijsen, H. *Feature interactions in telecommunications systems*. IEEE Commun. Mag. **31**(8), 18–23 (Aug. 1993).

[CGL+94] Cameron, E. J., Griffeth, N. D., Lin, Y.-J., Nilson, M. E., Schnure, W. K. und Velthuijsen, H. *A feature interaction benchmark in IN and beyond*. In Bouma und Velthuijsen [BoVe94], S. 1–23.

[ISO89] ISO/TC 97/SC 21, ISO 9074. *Information Processing Systems — Open Systems Interconnection — Estelle: A Formal Description Technique Based on an Extended State Transition Model* (1989).

[ITU93] ITU-T. *Q12xx-Series Intelligent Network Recommendations* (1993).

[SiSt90] Sijelmassi, R. und Strausser, B. *PET DINGO: An integrated tool set for Estelle*. In Quemada, J., Maños, J. und Vazquez, E. (Hrsg.), „FORTE '90“, S. 661–666, Madrid, Spanien (5.–8. Nov. 1990). North-Holland.

[The95] Thees, J. *Entwurf und Implementierung eines Werkzeugs zur Analyse von Feature-Interaktionen in Estelle-Spezifikationen*. Diplomarbeit, Univ. Kaiserslautern, FB Informatik (Apr. 1995).