Nelson Oppen Integration Schema

Seminar

Joel Charles
Supervision: Gereon Kremer

SS 2017

Zusammenfassung

Das sogenannte *SAT-solving*, also die Erfüllbarkeitsprüfung einer aussagenlogischen Formel, ist in der Lage Probleme mit Millionen von Variablen zu lösen. Jedoch kann die Aussagenlogik allein zu schwach für die Modellierung sein. Der Wechsel zur Prädikatenlogik der ersten Stufe erlaubt es, ausdrucksstärkere Probleme zu formulieren. Um ihre Erfüllbarkeit zu prüfen, muss das *SAT-solving* hin zum SMT (*SAT-modulo-theories*) erweitert werden. In modernen *SMT-solvern* kollaboriert eine SAT-Einheit mit einem *theory-solver*. In der Praxis kann eine Theorie aus mehreren Theorien zusammengesetzt sein ($T = T_1 \cup ... \cup T_n$). Folglich ist ein Entscheidungsverfahren für die Vereinigung mehrerer Theorien erforderlich. Diese Arbeit stellt zwei verschiedene Ansätze für SMT_T vor und wägt sie gegeneinander ab.

1 Einleitung

Das sogenannte *SAT-solving*, also die Erfüllbarkeitsprüfung einer aussagenlogischen Formel, ist in der Lage Probleme mit Millionen von Variablen zu lösen. Daher findet es sowohl in der Industrie eine weite Verbreitung (z.B. Design eines Schaltplans), als auch in verschiedenen Forschungsbereichen (z.B. Analyse, Synthese und der Optimierung), in denen es sich etabliert hat.
Jedoch kann die Aussagenlogik allein zu schwach für die Modellierung sein. Der Wechsel zum quantifizierungsfreien Fragment der Prädikatenlogik der ersten Stufe (*quantifier-free fragments of first-order logic over various theories*) erlaubt es, ausdrucksstärkere Probleme zu formulieren. Um ihre Erfüllbarkeit zu prüfen, muss das *SAT-solving* hin zum *SMT-solving* (*SAT-modulo-theories*) erweitert werden.

In modernen *SMT-solvern* kollaboriert eine SAT-Einheit mit einem *theory-solver*. Die SAT-Einheit arbeitet auf einer booleschen Abstraktion der Eingabeformel, um die notwendige Erfüllbarkeit der booleschen Struktur zu testen, ehe ein *theory-solver* die Konsistenzprüfung der möglichen Lösung gegen die zugrundeliegende Theorie T übernimmt. In der Praxis kann sie aus mehreren Theorien zusammengesetzt sein ($T = T_1 \cup ... \cup T_n$). Folglich ist ein Entscheidungsverfahren für die Vereinigung mehrerer Theorien erforderlich.

Diese Arbeit stellt zwei verschiedene Ansätze für SMT_T vor und wägt sie gegeneinander ab. Einführend rekapituliert der Abschnitt 2 nochmals die Grundidee des *SAT-solvings*, um es Teil 3 zu ermöglichen, *SMT-solving* (SAT-Modulo Theories) zunächst mit einer nicht zusammengesetzten Theorie einzuführen.

Mithilfe dieser Wissensbasis stellt Kapitel 4 die Nelson-Oppen Integration vor [4]. Dabei handelt es sich um ein Integrationsschema, welches mittels eines strukturierten Informationsaustausches zwischen den $T_1, ..., T_n$ ein Entscheidungsverfahren für ihre Kombination erstellt.

Im direkten Kontrast hierzu steht der Delayed Theory Combination Ansatz, den Abschnitt 5 vorstellt [1]. Er vermeidet ein Integrationsschema gänzlich und schafft es auf diese Weise die *theory-solver* unabhängig voneinander arbeiten zu lassen. Stattdessen wird die Verbindung zwischen den *theory-solvern* und dem *SAT-solver* verstärkt, wodurch die Zusammenführung der einzelnen Modelle für SMT_{T_i} zu einem vollständigen Modell für SMT_T, nur falls nötig durchgeführt wird. Hierdurch sind nicht nur nicht benötigte komplexe Interaktionen zwischen den Theorien vermeidbar, sondern es erlaubt es mit der Konvexität einer Theorie leichter umzugehen. Zum Schluss zieht Kapitel 6 ein abschließendes Fazit.

2 SAT

Das *SAT-solving* legt den Grundstein für das *SMT-solving* in Kapitel 3. Die Beantwortung der Frage nach der Erfüllbarkeit einer aussagenlogischen Formel ϕ, ist das Ziel des *SAT-solvings*.

Als Eingabe sei eine aussagenlogische Formel ϕ in konjunktiver Normalform (KNF) gegeben. Beispielsweise sei

$$\phi_{Skelett} = (a \vee b) \wedge \neg c \wedge d \tag{1}$$

Das Verfahren entscheidet nun, ob eine erfüllende Belegung α der Variablen in der Menge der möglichen Belegungen *Assign* existiert. Hierfür wird über alle Belegungen iteriert und jeweils auf Erfüllbarkeit evaluiert. Wenn ein solches α existiert, ist ϕ erfüllbar, andernfalls unerfüllbar. Die Formel 1 ist erfüllbar, da die Belegung welche allen Variablen bis auf c den Wahrheitswert *True* zuweist, ϕ erfüllt.
Für das obige Beispiel 1 muss jede erfüllende Belegung α, der Variable d den Wert *True* und c den Wert *False* zuordnen, damit α die Formel erfüllen kann. Weiterhin gilt, wenn α der Variable b den Wert *False* zuweist, muss a den Wert *True* erhalten, andernfalls ist α keine erfüllende Belegung. Die sogenannte *Boolean Constraint Propagation (BCP)* verfolgt den Ansatz, die während des Suchprozesses nach einer erfüllenden Belegung erlangten Informationen umgehend in die Teilbelegung einfließen zu lassen. Indem sie die Gültigkeit der Implikationen, die aus dem Formelaufbau und bereits aus Teilbelegungen von ϕ folgen erzwingt, lenkt sie die Suche in die richtige Richtung.
Daher wird meist über die *BCP* versucht, den Suchraum im Vorhinein zu beschränken, indem innerhalb von ϕ nach *unit Klauseln* gesucht wird. Eine Klausel ist genau dann eine *unit Klausel*, wenn allen Variablen, bis auf einer, ein Wahrheitswert zugewiesen wurde, aber die Klausel noch nicht erfüllt ist. Aufgrund der Intention, der Suche nach einer erfüllenden Belegung α für ϕ, impliziert eine *unit Klausel* bereits den Wahrheitswert der letzten Variable. Wenn es nicht möglich ist die Implikationen, die direkt aus ϕ folgen[1], zu erfüllen, dann ist die Formel unerfüllbar (Bsp. $X \wedge \neg X$).
Anschließend werden der Reihe nach weitere Variable mit einem geratenen Wert belegt. Durch eine erfolgreiche Zuweisung wiederum, können neue *unit Klauseln* entstehen, weshalb wieder eine *Boolean Constraint Propagation* durchgeführt wird. Hierbei können Konflikte auftreten, die über Konfliktresolution und Backtracking gelöst werden sollen [3].

[1] Bevor Entscheidungen für die Belegung getroffen sind.

Wenn der Konflikt nicht auflösbar ist, dann ist ϕ unerfüllbar. Andernfalls wird die nächste Variable zugewiesen. Existiert jedoch keine solche Variable, ist ϕ erfüllbar.

3 SAT-Modulo Theories

Das *SMT-solving* erweitert die grundlegende Fragestellung des *SAT-solvings*.
Indem zusätzlich zur Erfüllbarkeit einer Σ-Formel ϕ eine *Signatur* Σ festgelegt wird, welche die Menge der nicht-logischen Symbole bestimmt. ϕ liegt in First-Order-Logik vor.

Definition 3.1 (First-order Logik) *Die First-order Logik gibt einen Rahmen für die erlaubte Ausdrucksstärke von Formeln vor. Sie enthält die folgenden Bestandteile:*

1. *Theorie Symbole:*
   - *Konstanten [z.B. 0, 1],*
   - *Variablen [z.B. x, y]*
   - *Funktionssymbole [z.B. −]*
2. *Logische Symbole: Logische Verbindungen [z.B. $\wedge$, $\vee$] und Quantoren*
3. *Prädikat Symbole: Heben die Formel von dem Theorie Level auf das logische Level [z.B. $<, >, =$]*

Weiterhin ist die Konsistenz zu einer zugrundeliegenden Σ-*theory* T gefordert (T-*satisfiable*).
Der Einfachheitshalber ist eine Σ-Theorie im Folgenden nur noch als Theorie bezeichnet.
Eine Theorie ist als ein Regelwerk zu verstehen, welches die Interpretation der nicht-logischen Symbole in Σ definiert.
Als Beispiel: die Theorie der reellen Zahlen $\mathbb{R}$, mit der Signatur $\Sigma = (\mathbb{R}, +, -, \leq, 0, 1, =)$, wobei die nicht-logischen Symbole wie üblich zu interpretieren sind. Oder die Theorie der *Equality Logic with uninterpreted functions* $\mathbb{E}$, die als Theorie Symbole Variablen mit einer beliebigen Domäne, Konstanten der selben Domäne, Funktionssymbole und die Gleichheit als Prädikat Symbol enthält. Die zwei eingeführten Theorien, werden im Verlauf der Arbeit durchgehend Verwendung finden. $x = y$ und $x \leq y$ sind somit $\mathbb{R}$-Literale[2] und $f(x) = f(y)$ ein $\mathbb{E}$-Literal, wobei f ein Funktionssymbol ist.

3.1 Arbeitsweise des SMT-solvings

Nachfolgend soll die Arbeitsweise eines *SMT-solvers* anhand der Formel

$$\phi = (f(x) = f(y) \vee f(x) = f(z)) \wedge x \neq z \wedge x = y$$

erläutert werden.
Das *SMT-solving* setzt mit einer booleschen Abstraktion von ϕ an und unterteilt die Problemlösung in die zwei Komponenten *SAT-solving* und *theory-solving*.
Hierbei resultiert die Abstraktion in der Formel aus Beispiel 1. Im Unterschied zum klassischen *SAT-solving* drücken die Wahrheitswerte der Variablen jetzt die Gültigkeit der Theorie-Einschränkung aus, die sie repräsentieren.
Zunächst überprüft ein klassischer *SAT-solver*, dieses boolesche Skelett auf Erfüllbarkeit. Wenn bereits die Abstraktion nicht erfüllbar ist, kann ϕ nicht erfüllbar sein aufgrund der geforderten Konsistenz zu T, welche noch strengere Anforderungen hinzufügt. Auf diese

[2] Ein Literal ist eine atomare Formel oder seine Negation.

Weise wird der *theory-solver* nicht unnötig aufgerufen. Findet der *SAT-solver* eine potentielle Lösung, also eine Belegung α der Variablen welche die Abstraktion erfüllt, muss diese auf Konsistenz mit der Theorie überprüft werden. Hierzu erhält der *theory-solver* die zu α korrespondierenden Einschränkungen und prüft sie gegen die Theorie.

Für $\phi_{Skelett}$ könnte beispielsweise die Lösung $(False \vee True) \wedge False \wedge True$ gefunden werden. Anschließend wird der *theory-solver* befragt, ob $(f(x) \neq f(y) \vee f(x) = f(z)) \wedge x \neq z \wedge x = y$ unter Beachtung der Theorie-Einschränkungen ϕ erfüllt.
Auf Grund der Form der Eingabe (KNF) ist es bereits ausreichend, dass eine Teilmenge von α unerfüllbar ist, um zu beweisen, dass α ϕ nicht erfüllt. Lassen sich keine Theoriekonflikte finden, ist ϕ erfüllbar, andernfalls weist der *theory-solver* den *SAT-solver* an, eine andere Belegung der Abstraktion zu finden (Backtracking) [3].

Gegebenenfalls ist der *theory-solver* auch in der Lage eine Erklärung zu liefern, warum die Menge der Einschränkungen falsifiziert. Die Begründung *(Infeasible Subset)* ist hierbei eine Teilmenge der Einschränkungen, die bereits unerfüllbar ist [1]. Anschließend wird sie wird vom *SAT-solver* gelernt, um die Kombination der Einschränkungen/ (Un-)Gleichungen zu vermeiden. Eine detaillierte Beschreibung des genauen Vorgehens ist [1] zu entnehmen. Das Prozedere ist in Abbildung **??** zu erkennen und zeigt ein stark vereinfachtes Entscheidungsverfahren von einem *SMT-solver*, auf dessen Arbeitsweise eingegangen wurde.

4 Nelson-Oppen Integration

In der praktischen Anwendung des *SMT-solvings* zeigt sich, dass die zugrundeliegende Theorie aus mehreren Theorien T (z.B. $T = T_1 \cup T_2$) zusammengesetzt sein kann. Dies bedeutet für den *theory-solver*, dass er nicht nur mit den einzelnen Theorien umgehen muss, sondern ein Entscheidungsverfahren für T (mit $T = T_1 \cup ... \cup T_n$) bieten muss. Mit dem Nelson-Oppen Integrationsschema wurde 1979 ein Verfahren von G. Nelson und D. Oppen geschaffen, um die Kombination von Theorien in einem einzelnen Entscheidungsverfahren zu vereinigen [4].

Im Folgenden ist in Abschnitt 4.1 zunächst ein naiver Ansatz für SMT_T gegeben, der sich als falsch herausstellt. Der nachfolgende Teil 4.2 erläutert die Gründe. Der nächste Abschnitt (4.3) befasst sich zunächst mit den Anforderungen die das Nelson-Oppen Integrationsschema an die Theorien stellt. An ihn knüpft der Abschnitt 4.4 an, der die Arbeitsweise des Nelson-Oppen Verfahrens vorstellt.

4.1 Naiver Ansatz

Gegeben sei das SMT-Problem $SMT_T(\phi)$, wobei die jeweiligen Σ_i-Theorien jeweils paarweise disjunkte Signaturen (= ausgenommen) haben $(1 \leq i \leq n \,; 2 \leq n)$.
Nachfolgend soll in einem ersten Ansatz versucht werden, das SMT Problem für eine zusammengesetzte Theorie zu lösen. Die Arbeitsweise soll anhand einer geringfügigen Modifikation der Formel aus 3.1 erläutert werden. Die Formel wird derart verändert, dass sie aus mehreren Theorien zusammengesetzt ist. Sie sei mit

$$\phi = f(x) \leq f(y) \wedge f(x) \neq f(z) \wedge y = x - 1 \wedge z = y + 1 \tag{2}$$

gegeben.

Es ist erkennbar, dass sich T aus $\mathbb{R}$ und $\mathbb{E}$ zusammensetzt. Aufgrund existierender Entscheidungsverfahren für $\mathbb{R}$ ($SMT_{\mathbb{R}}$) und $\mathbb{E}$ ($SMT_{\mathbb{E}}$) ist es naheliegend ϕ derart zu separieren, dass die Bestandteile von dem jeweiligen *theory-solver* gelöst werden können ($\phi' = \phi_{\mathbb{R}} \wedge \phi_{\mathbb{E}}$). Die ϕ'_i enthalten somit jeweils nur Symbole aus einer einzigen Theorie. Hierzu werden neue Variablen eingeführt und über Gleichungen definiert. Beispielsweise bedarf das Literal $f(x) \leq f(y)$ der Umwandlung, da weder *uninterpreted functions* in $\mathbb{R}$ enthalten sind, noch $\mathbb{E}$ mit $\leq$ umgehen kann. Daher werden die Hilfsvariablen g_1 und g_2 eingeführt, welche den Funktionswert von $f(x)$ beziehungsweise $f(y)$ repräsentieren.

Die erfüllbarkeitsäquivalente Umformung resultiert in,

$$\phi' := g_1 \leq g_2 \wedge f(x) \neq f(z) \wedge y = x - 1 \wedge z = y + 1 \wedge f(x) = g_1 \wedge f(y) = g_2 \tag{3}$$

dessen Bestandteile von den bestehenden Entscheidungsverfahren akzeptiert werden, wenn sie in $\phi_{\mathbb{R}}$ und $\phi_{\mathbb{E}}$ mit

$$\begin{aligned} \phi_{\mathbb{R}} &= g_1 \leq g_2 \wedge y = x - 1 \wedge z = y + 1 \\ \phi_{\mathbb{E}} &= f(x) \neq f(z) \wedge f(x) = g_1 \wedge f(y) = g_2 \end{aligned} \tag{4}$$

gruppiert werden.

Der naive Ansatz würde abschließend prüfen, ob $SMT_{\mathbb{R}}(\phi_{\mathbb{R}}) \wedge SMT_{\mathbb{E}}(\phi_{\mathbb{E}})$ erfüllbar ist. Sowohl $\phi_{\mathbb{R}}$ als auch $\phi_{\mathbb{E}}$ sind erfüllbar, daher attestiert der naive Ansatz die Erfüllbarkeit von 2.

4.2 Interaktion zwischen den Theorien

Bei genauer Betrachtung des Beispiels aus 4.1 ist festzustellen, dass zwar jedes ϕ'_{i} für sich allein erfüllbar ist, aber ihre Konjunktion ϕ' nicht. Denn $y = x - 1 \wedge z = y + 1 \implies x = z$, aber aus $f(x) \neq f(z)$ folgt mittels der funktionalen Kongruenz $x \neq z$. Aufgrund des Widerspruchs arbeitet das Entscheidungsverfahren nicht korrekt. Daher muss es eine Interaktion zwischen den Theorien geben, die die Formel unerfüllbar macht. Wobei die Interaktion nur an den Kommunikations-Schnittstellen (Interface) zwischen den Theorien erfolgen kann. Diese Erkenntnis verwendet die Nelson-Oppen Integration, um ein Entscheidungsverfahren für die Kombination der Theorien zu liefern.

Das Verfahren verfolgt denselben Separationsansatz wie Abschnitt 4.1, der dem Integrationsschema als Aufbereitung der Eingabe für SMT_T dient. Nachfolgend ist der Ansatz etwas technischer formuliert und als Purifikation bezeichnet.

Die Separation von ϕ in ein erfüllbarkeitsäquivalentes ϕ' wird genau dann als Purifikation bezeichnet, wenn jedes Literal in ϕ' i-pure ist, für ein $i \in 1, ..., n$. Anders ausgedrückt: die Literale enthalten nur Symbole aus einer einzigen Theorie. Daher ist ein Literal L genau dann *i-pure*, wenn es nur *i-pure Terme* enthält[3]. Ein *i-Term* $\mathrm{T_i}$ (siehe Def. 4.1) ist wiederum i-pure, wenn $\mathrm{T_i}$ keinen Subterm enthält, der ein *j-Term* ($i \neq j$) ist (Variablen ausgenommen). Das resultierende ϕ' nennt sich dann *pure*. Die Purifikation ermöglicht es nun ϕ' als Konjunktion von i-pure Literalen ($\phi'_1 \wedge ... \phi'_{\mathrm{n}}$) ausdrücken.

[3]Beachte, dass ein Literal (logisches Level) mehrere Terme enthalten kann. Nach Definition 4.1 enthält ein Term keine Prädikat Symbole, weshalb Terme erst durch die Prädikat Symbole auf ein logisches Level gehoben werden.

Definition 4.1 (i-Term) *Seien $\Sigma_1, ..., \Sigma_n$ bis auf = disjunkte Signaturen ($n > 1$) und sei T_i eine Σ_i-theory. Dann ist ein $\Sigma_1 \cup ... \cup \Sigma_n$-Term genau dann ein i-Term, wenn er eine Variable aus Σ_i ist oder der Form $f(t_1, ..., t_m)$ ($m > 1$) entspricht, wobei f aus Σ_i stammt.*
Ein Term ist wiederum eine Konstanten beziehunsweise eine Variablen, oder wenn t_1, ..., t_n ($n > 0$) Terme sind und f ein n-Stelliges Funktionssymbol ist, dann ist $f(t_1, ..., t_n)$ ein Term (endliche Anwendung der Regel).

Bei einer genaueren Betrachtung der Gestalt ist zu erkennen, dass das Interface auf die Gleichungen zwischen den Variablen reduziert wurde, welche sowohl in einem i-pure Literal vorkommen, als auch in mindestens einem weiteren j-pure Literal ($i \neq j$), sie werden *interface Variablen* genannt. Folgerichtig wird die Gleichung $e_{\mathrm{ij}} := c_{\mathrm{i}} = c_{\mathrm{j}}$ *interface equality* genannt (c_{i}, c_{j} *interface Variablen*), welche das Verfahren zwischen den Theorien propagiert und so ein Entscheidungsverfahren für STM_T liefert.
Die Problematik des intuitiven Ansatzes liegt nun darin, dass die Interaktion der Theorien schlicht nicht über das Interface kommuniziert wird und es daher zu keinem Informationsaustausch zwischen $SMT_{\mathbb{R}}(\phi_{\mathbb{R}})$ und $SMT_{\mathbb{E}}(\phi_{\mathbb{E}})$ kommt. Es findet keine Theorie übergreifende Konsistenzprüfung für die Belegung statt.

4.3 Nelson-Oppen Theorien

Damit das Verfahren korrekt arbeitet, fordert es von allen Entscheidungsverfahren SMT_{T_i} die Deduktion-Vollständigkeit. Gemeint ist, dass jedes Entscheidungsverfahren für das jeweilige T_i alle Gleichungen aufdeckt die aus ϕ_i' folgen, insbesondere auch sämtliche *interface equalities* auffindet.
Wenn $\phi_{\mathbb{R}}$ aus 4 beispielsweise an $\mathbb{R}$ übergeben wird, dann muss das Entscheidungsverfahren für $\mathbb{R}$ erkennen, dass $x = z$ enthalten ist.
Die e_{ij} werden im weiteren Verlauf der Nelson-Oppen Integration zwischen den Entscheidungsverfahren ausgetauscht und propagiert, um SMT_T zu lösen.

Durch die Deduktion-Vollständigkeit wird sichergestellt, dass sowohl alle implizit, als auch explizit kodierten Einschränkungen von ϕ_i' an eine erfüllende Belegung α gefunden werden. Diesen Anforderungen muss α genügen, da sie direkt aus den ϕ_i' abgeleitet sind.
Auf diese Weise ist sichergestellt, dass alle relevanten Informationen (*interface equalities*) im Laufe des Verfahrens exploriert werden. Anschließend werden sie über das auf *interface equalities* eingeschränkte Interface zwischen den SMT_{T_i} kommuniziert. Die Spezifikation des Interfaces ermöglicht es, die Kommunikation einzuschränken und damit zu kontrollieren [1].
Weiterhin fordert das Nelson-Oppen Integrationsschema, dass jede Theorie *stably-infinite* ist. Eine Theorie ist genau dann *stably-infinite*, wenn jede quantifizierungsfreie T-erfüllbare Formel in einem unendlichen Modell von T erfüllbar ist. Theorien die beide Anforderungen erfüllen nennen sich Nelson-Oppen Theorien [2].

4.4 Arbeitsweise des Nelson-Oppen Integration Verfahrens

Ein *SMT-solver*, der das Nelson-Oppen Integration Verfahren umsetzt erhält eine Formel ϕ (in KNF) als Eingabe und trifft eine Aussage über ihre Erfüllbarkeit. Er ist in der Lage die Entscheidungsverfahren für die zugrundeliegenden Theorien über das Equality Propagation Verfahren zu vereinen. Zur besseren Veranschaulichung ist das nachfolgend beschriebene Verfahren als Pseudocode gegeben.

```
function Equality Propagation Verfahren(φ)
    φ'_1 ∧ ... ∧ φ'_n = φ' ← purifikation(φ)
    while !konvergenz_erreicht do
        for all φ'_i (1 ≤ i ≤ n) do
            if SMT(φ'_i == UNSAT) then return UNSAT
            else
                for all φ'_i do
                    if φ'_i ≠ konvex then
                        kommuniziere rekursiv jeden Fall der Disjunktion
                    end if
                    kommuniziere interface equalities
                end for
            end if
        end for
    end while
    return SAT
end function
```

Zunächst ist eine *Purifikation* durchzuführen, die ϕ während der Vorverarbeitung, in die geeignetere (erfüllbarkeitsäquivalente) Form ϕ' überführt. Aufgrund des Vorliegens von ϕ' als *pure*-Formel, lässt sich ϕ' folglich als Konjunktion von i-pure Literalen ($\phi'_1 \wedge ... \phi'_n$) ausdrücken. Anschließend sind die ϕ'_i den entsprechenden Entscheidungsverfahren SMT_{T_i} zu übergeben. Stellt eines von ihnen die Unerfüllbarkeit fest, so ist ϕ auch unerfüllbar und das Verfahren endet.
Andernfalls startet die Kommunikation zwischen den SMT_{T_i}, indem sie jeweils über Deduktion alle *interface equalities* explorieren, woraufhin sie die Information an die anderen propagieren. Ist durch sie ein Erkenntnisgewinn möglich, wird sie dem jeweiligen ϕ'_i hinzugefügt. Mit dessen Hilfe kann die Deduktion weiter vorangetrieben werden.
Während jedes Propagations-Schrittes, testet das Verfahren jedes ϕ'_i auf Unerfüllbarkeit. Kommt es hierbei zu Widersprüchen, sprich, es existiert ein i, sodass ein ϕ'_i durch die Equality Propagation unerfüllbar geworden ist, dann ist auch ϕ' unerfüllbar und damit ϕ. In diesem Fall kann das Verfahren frühzeitig mit *UNSAT* abbrechen.
Anzumerken ist, dass die Umkehrung wenn ϕ' unerfüllbar ist, dann ist ein ϕ'_i unerfüllbar, jedoch im Allgemeinen nicht gilt (siehe 4.1). Dieser Vorgang wiederholt sich bis zur Konvergenz. Also solange, bis während eines Propagations-Schrittes keine neuen *interface equalities* mehr ausgetauscht werden.

Es folgt für jedes ϕ'_i eine Prüfung nach einer Fallunterscheidung. Sie ist genau dann notwendig, wenn ein ϕ'_i existiert, welchem eine nicht konvexe Theorie zugrunde liegt. Das heißt die Deduktion findet für ein c_i Disjunktionen von *interface equalities* $e_{i,j} \vee e_{i,k}$ (mit c_i, c_j, c_k paarweise verschieden), aber in keinem Disjunktionsterm sind $e_{i,j}$ oder $e_{i,k}$ allein enthalten. Alle so in ϕ'_i aufgefundenen *interface equalities* verursachen eine Fallunterscheidung. Falls für nicht-konvexe ϕ_i keine Fallunterscheidung durchgeführt wird, werden nicht alle Verknüpfungen entdeckt. Das Verfahren testet nun rekursiv jede dieser *interface equalities* nacheinander auf Konsistenz mit den anderen Theorien. Falls ein Zweig erfüllbar ist, ist ϕ erfüllbar und unerfüllbar andernfalls.

Es genügt daher allgemein nicht die ϕ'_i einzeln zu testen, sondern es sind die *interface equalities* und die Konvexität der ϕ'_i zu betrachten. Da die Unerfüllbarkeit durch die Verknüpfung der Theorien entstehen kann.

5 Delayed Theory Combination

Mit der Delayed Theory Combination steht ein weiterer Ansatz bereit, um die Erfüllbarkeit von zusammengesetzten Nelson-Oppen-Theorien zu prüfen [1]. Das Verfahren verfolgt einen anderen Ansatz und vermeidet eine direkte Kommunikation zwischen den *theory-solvern*. Es verschmelzen also nicht mehrere *theory-solver* über ein Integrationsschema zu einem Entscheidungsverfahren für die Vereinigung.
Stattdessen existiert wie im klassischen SMT (mit einer Theorie) ein SAT solver, der nun mit den jeweiligen *theory-solvern* kommuniziert. Durch die Vermeidung eines direkten Informationsaustausches wird erreicht, dass die *theory solver* unabhängig und ohne Kenntnis voneinander arbeiten können.
Damit $SMT_{T_1} \wedge ... \wedge SMT_{T_n}$ konsistent zu SMT_T arbeitet, also keine falschen Rückschlüsse auf die Erfüllbarkeit wie in Abschnitt 4.1 entstehen, wird eine Formel ϕ_e konstruiert und der Eingabe ϕ hinzugefügt. ϕ_e enthält alle konstruierbaren *Interface equalities*, insbesondere die e_{ij} die nicht in ϕ enthalten sind. Nach [5] ist dann SMT_T erfüllbar, wenn keiner der *theory-solver* einen Konflikt aufdeckt. Den Namen Delayed Theory Combination trägt das Verfahren, da zunächst alle SMT_{T_i} für sich arbeiten und ihre Synchronisation, also die Zusammenführung der Modelle zu einem Gesamtmodell für ϕ, verzögert abläuft [1]. Die Synchronisation erfolgt erst zu dem Zeitpunkt, indem den e_{ij} ein konkreter Wert zugewiesen wird.

Im Folgenden ist die Arbeitsweise der Delayed Theory Combination aufgezeigt, welche als Pseudocode abgebildet ist.

```
function DTC(φ)
    n ← AnzahlTheorien(φ)
    φ' ← purifikation(φ)
    φ'' ← φ' ∧ interface-equalities(φ')
    φ_Skelett ← boolSkelett(φ'')
    while SAT(φ_Skelett) do
        α_1 ∧ ... ∧ α_n ∧ α_e = α ← wähle-Belegung(φ_Skelett)
        for i = 1; i ≤ n do
            (ρ, π) ← theorySolver_{T_i}(sklett2theory(α_i ∧ α_e))
            if π == UNSAT then
                φ_Skelett ← φ_Skelett ∧ ¬ρ
                return UNSAT
            end if
        end for
    end while
    return SAT
end function
```

Abbildung 1: Delayed Theory Combination - Pseudocode

Die Eingabe ist für beide Verfahren eine Formel in KNF (ϕ). Ebenso wie bei Nelson-Oppen ist eine Vorverarbeitung von ϕ durchzuführen, diese findet sich in den Zeilen 2 bis 4 wieder. Auf den Purifikationsschritt folgt der Hauptunterschied zwischen den Verfahren. Die Abbildung 2 veranschaulicht nochmals die konzeptionellen Unterschiede. Anstatt die ϕ'_i den SMT_{T_i} zu übergeben, identifiziert der DTC Ansatz zunächst alle *interface Variablen*. Mit ihrer Hilfe konstruiert es sämtliche *interface equalities* und fügt ihre Konjunktion ϕ_e

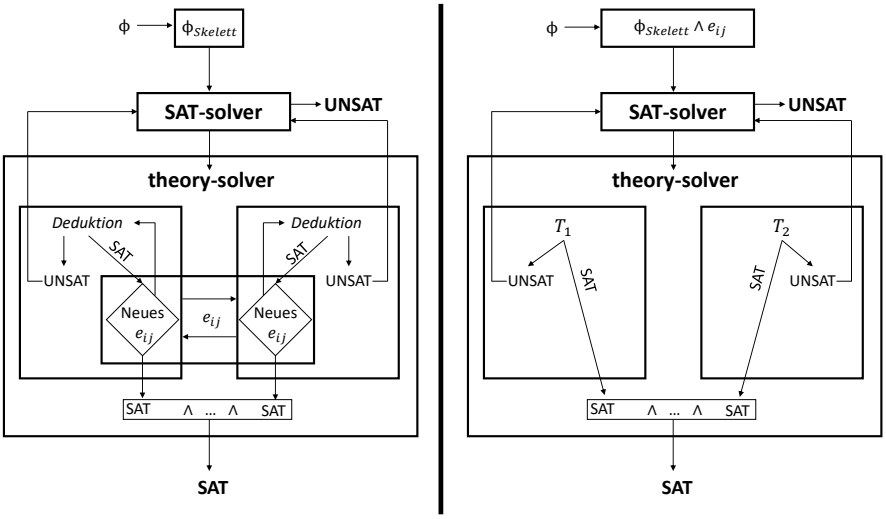

Abbildung 2: Konzept: Nelson-Oppen vs. DTC

zu ϕ' hinzu ($\phi'' = \phi' \wedge \phi_e$). Beachte, dass selbst die e_{ij} erstellt werden, welche nicht in ϕ' auftauchen[4].
Anschließend prüft ein *SAT-solver* die boolesche Abstraktion $\phi_{Skelett}$ von ϕ'' auf Erfüllbarkeit. Solange der *SAT-solver* $\phi_{Skelett}$ als erfüllbar ansieht, wird gegen die Konsistenz zur Theorie geprüft. Wie bereits in Kapitel 3.1 erwähnt, ist eine weitere Suche sonst nicht mehr nötig.
Nach [5] ist es unter diesen Voraussetzungen möglich aus $SMT_{T_1}(\phi'_1 \wedge \phi_e) \wedge ... \wedge SMT_{T_1}(\phi'_n \wedge \phi_e)$ auf SMT_T zu schließen. Es genügt somit den einzelnen SMT_{T_i} eine Belegung (für den sie betreffenden Formelteil), inklusive der Belegung für die *interface equalities*, zu übergeben. Wichtig anzumerken ist, dass alle SMT_{T_i}, die gleiche Belegung übermittelt bekommen, daher erhalten die e_{ij} in allen Theorien dieselben Wahrheitswerte. Im Prinzip beruht die Wahl der Belegung auf Raten, jedoch unter Beachtung von *BCP* und kann wie in [3] mit weiteren Techniken optimiert werden.
Mittels der Funktion *sklett2theory* werden die Theorie-Einschränkungen extrahiert, die zur booleschen Belegung korrespondieren und dem jeweiligen *theory-solver* übergeben. Findet keiner von ihnen einen Theorie-Konflikt, ist $SMT_{T_1}(\phi'_1 \wedge \phi_e) \wedge ... \wedge SMT_{T_1}(\phi'_n \wedge \phi_e)$ erfüllbar und damit auch $SMT_T(\phi)$. Andernfalls muss die nächste erfüllende Belegung getestet werden. Zusätzlich kann, je nach Fähigkeiten der *theory-solver* eine Begründung ρ für die Unerfüllbarkeit gegeben werden, welche der SAT-solver nutzt, um den Konflikt bei der Wahl einer Belegung zu umgehen.

6 Fazit/ Ausblick

Im Folgenden werden die beiden vorgestellten Verfahren gegenübergestellt. Zunächst ist zu erwähnen, dass das DTC Verfahren deutlich einfacher gestaltet ist, da die *theory-solver* unabhängig und ohne Kenntnis voneinander arbeiten können. Dies erleichtert den Implementierungsaufwand deutlich, da Integrationsschemata gänzlich vermieden werden.

Wenn die Implementierung beispielsweise um neue Theorien erweitert werden soll, ist

[4]Für Formel 3 bsp. $x = y, x = z, x = g_1, x = g_2, y = z, y = g_1, y = g_2, z = g_1, y = g_2$

dies nur bei dem DTC Ansatz mit einem näherungsweise Konstanten Aufwand möglich, da keine Kommunikation zwischen den *theory-solvern* existiert.
Ausserdem setzt das Nelson-Oppen Integrationsverfahren die Deduktion-Vollständigkeit einer Theorie voraus, welche aufgrund verschiedener Gesichtspunkte problematisch sein kann. Zwar ist die Fähigkeit zur Deduktion meist entscheidend für die Effizienz eines Entscheidungsverfahren, dennoch kann es für komplexe Theorien schwer sein, diese Eigenschaft zu erfüllen [1]. Außerdem ist es möglich, dass diese Anforderung eine hohe Rechenkomplexität in das Entscheidungsverfahren einführt. Das DTC Verfahren rückt daher von dieser strikten Voraussetzung ab und erlaubt es je nach Theorie abzuwägen, ob die Implementierung einer Deduktion-Vollständigkeit sinnvoll ist oder eine partielle Deduktion aus Effizienzgründen besser ist.
Weiterhin zeigt das Kapitel 4.4 auf, dass die Konvexität einer Theorie den Rechenaufwand des Verfahrens durch Fallunterscheidungen deutlich steigern kann. Zu beachten ist, dass im Nelson-Oppen Verfahren nicht klar ist, welcher Zweig der Suche zum Erfolg führt und daher quasi willkürlich Disjunktionen von *interface equalities* ausgetauscht werden. Wohingegen der DTC Ansatz dies gänzlich vermeidet, da keine Kommunikation zwischen den *theory-solvern* stattfindet, sondern diese lediglich mit dem *SAT-solver* in Kontakt treten und DTC damit auf alle Optimierungstechniken der *SAT-solver* zurückgreifen kann [1].

Dennoch hat auch DTC nicht nur Vorteile. Zwar fordern beide Verfahren die Purifikation, dennoch fügt nur DTC zu Beginn alle *interface equalities* hinzu ($O(n^2)$). Nach [1] sind einige der e_{ij} jedoch nicht in ϕ' vorhanden und das Anwachsen des potentiell größeren Suchraums kann durch Optimierungstechniken des *SAT-solvers* eingeschränkt werden.
Weiterhin stützt die experimentelle Evaluation in [1] die Auffassung der Autoren, dass es sich bei der DTC um ein effizienteres Verfahren handelt.
Abschließend sei zu erwähnen, dass beide Verfahren nur mit Nelson-Oppen-Theorien umgehen können, wohingegen sich [6] mit der Kombination von nicht Nelson-Oppen-Theorien befasst.

Literatur

[1] M. Bozzano, R. Bruttomesso, A. Cimatti, T. Junttila, S. Ranise, P. Van Rossum, and R. Sebastiani. Efficient satisfiability modulo theories via delayed theory combination. In *Computer Aided Verification*. Springer, 2005.

[2] R. Bruttomesso, A. Cimatti, A. Franzen, A. Griggio, and R. Sebastiani. Delayed theory combination vs. nelson-oppen for satisfiability modulo theories: a comparative analysis. *Annals of Mathematics and Artificial Intelligence*, 2009.

[3] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient sat solver. In *Proceedings of the 38th annual Design Automation Conference*. ACM, 2001.

[4] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. Program. Lang. Syst.*, 1979.

[5] C. Tinelli and M. Harandi. A new correctness proof of the nelson-oppen combination procedure. In *Frontiers of Combining Systems*. Springer, 1996.

[6] C. Tinelli and C. G. Zarba. Combining non-stably infinite theories. *Electronic Notes in Theoretical Computer Science*, 2003.