The Complexity of Satisfiability for Sub-Boolean Fragments of $ALC$

Arne Meier$^1$ Thomas Schneider$^2$

$^1$ Leibniz Universität Hannover, Germany,
$^2$ University of Manchester, UK.

May 4th, 2010
Defining \textit{ALC} via operators

\begin{definition}(\textit{ALC})\end{definition}

Let $A$ be an atomic concept and $R$ a role. The set of concept descriptions is defined via

$$C \equiv_{\text{def}} A \mid C \sqcap C \mid \neg C \mid \exists R.C \mid \forall R.C,$$

with usual semantics.
TBoxes and Ontologies as usual

$C$, $D$ concepts, $R$ role, $x$ individual.

- **General concept inclusion (GCI):** $C \sqsubseteq D$
  
  (if also $D \sqsubseteq C$, then $C \equiv D$)
TBoxes and Ontologies as usual

$C, D$ concepts, $R$ role, $x$ individual.

- **General concept inclusion (GCI):** $C \sqsubseteq D$
  (if also $D \sqsubseteq C$, then $C \equiv D$)
- **TBox:** finite set of GCIs
- **ABox:** finite set of axioms $R(x, y), C(x)$
TBoxes and Ontologies as usual

- **C, D concepts, R role, x individual.**
  - **General concept inclusion (GCI):** $C \sqsubseteq D$
    (if also $D \sqsubseteq C$, then $C \equiv D$)
  - **TBox:** finite set of GCIs
  - **ABox:** finite set of axioms $R(x, y), C(x)$
  - **Ontology:** union of a TBox and an ABox
Status quo

Consider the following decision problems for $\mathcal{ALC}$

- Concept Satisfiability (CSAT)
  PSPACE-complete
  [Ladner '77, Schmidt-Schauß and Smolka '91]
Status quo

Consider the following decision problems for $\mathcal{ALC}$

- Concept Satisfiability ($\text{CSAT}$)
  \text{PSPACE}-complete
  [Ladner '77, Schmidt-Schauß and Smolka '91]

- TBox Satisfiability ($\text{TSAT}$), concept sat. w.r.t. TBoxes ($\text{TCSAT}$)
  \text{EXPTIME}-complete
  [Pratt '78, Vardi and Wolper '86, Schild '91]
Status quo

Consider the following decision problems for $\mathcal{ALC}$

- Concept Satisfiability (CSAT)  
PSPACE-complete  
  [Ladner '77, Schmidt-Schauß and Smolka '91]

- TBox Satisfiability (TSAT), concept sat. w.r.t. TBoxes (TCSAT)  
EXPTIME-complete  
  [Pratt '78, Vardi and Wolper '86, Schild '91]

- Ontology Satisfiability (OSAT), concept sat. w.r.t. ontologies (OCSAT)  
EXPTIME-complete  
  (confer item above)
Motivation

What about the complexity if ...

- ... we allow only the Boolean function $\land$ for concepts?
- ... we allow only monotone functions in axioms?
Motivation

What about the complexity if ...

- ... we allow only the Boolean function $\land$ for concepts?
- ... we allow only monotone functions in axioms?

$\rightsquigarrow$ Parameterization of the five decision problems by a set $B$ of Boolean functions
Motivation

What about the complexity if ...

- ... we allow only the Boolean function $\land$ for concepts?
- ... we allow only monotone functions in axioms?

Parameterization of the five decision problems by a set $B$ of Boolean functions

We need a suitable characterisation for sets of Boolean functions.
Boolean fragments of $ALC$

Restrict the set of *Boolean* operators!

**Why?**

- Propositional SAT becomes tractable, e.g., without negation. [Lewis '79]
- SAT for Modal Logic or LTL becomes tractable for certain restrictions. [Bauland et al. '06/07]
- SAT for many sub-Boolean description logics is tractable. [Baader et al. '98/05/08, Calvanese et al. '05–07]

**Our goal:**

- Find border between tractable and intractable fragments.
- Find tight complexity bounds.
**Definition**

- A **clone** is a set $B$ of Boolean functions that contains all projections and is closed under composition.

- For a set $B$ of Boolean functions, we denote by $[B]$ the smallest clone containing $B$.

- $B$ is called a **base** for $[B]$. 

---

**Example:**

$\oplus \in [\{\land, \neg\}]$ because $x \oplus y \equiv \neg(\neg(x \land \neg y) \land \neg(\neg x \land y))$. 

---

**Preliminaries**

- Motivation

**Post's Lattice**

**Results**
A Little Bit of Universal Algebra

Definition

- A clone is a set $B$ of Boolean functions that contains all projections and is closed under composition.
- For a set $B$ of Boolean functions, we denote by $[B]$ the smallest clone containing $B$.
- $B$ is called a base for $[B]$.

Thus:

- $[B]$ consists of those functions that can be defined by a propositional formula with connectives from $B$. 
A Little Bit of Universal Algebra

### Definition

- A **clone** is a set $B$ of Boolean functions that contains all projections and is closed under composition.
- For a set $B$ of Boolean functions, we denote by $[B]$ the smallest clone containing $B$.
- $B$ is called a **base** for $[B]$.

**Thus:**

- $[B]$ consists of those functions that can be defined by a propositional formula with connectives from $B$.

**Example:** $\oplus \in \{\land, \neg\}$ because $x \oplus y \equiv \neg(\neg(x \land \neg y) \land \neg(\neg x \land y))$
So what is the benefit?

\[ \Pi \] – computational problem defined over propositional formulas

\[ \Pi(B) \] – the restriction of \( \Pi \) to formulas with connectives from \( B \), e.g., \( \text{SAT}(B) \), the satisfiability problem for formulas with connectives from \( B \)
So what is the benefit?

Π – computational problem defined over propositional formulas

Π(\(B\)) – the restriction of Π to formulas with connectives from \(B\),
e.g., \(\text{SAT}(B)\), the satisfiability problem for formulas with
connectives from \(B\)

Then: If \(B \subseteq [B']\) then \(\Pi(B) \leq \Pi(B')\)
So what is the benefit?

\( \Pi \) – computational problem defined over propositional formulas

\( \Pi(B) \) – the restriction of \( \Pi \) to formulas with connectives from \( B \), e.g., \( SAT(B) \), the satisfiability problem for formulas with connectives from \( B \)

Then: If \( B \subseteq [B'] \) then \( \Pi(B) \leq \Pi(B') \)

Upper bounds carry over downwards.
Lower bounds carry over upwards.
So what is the benefit?

\( \Pi \) – computational problem defined over propositional formulas

\( \Pi(B) \) – the restriction of \( \Pi \) to formulas with connectives from \( B \),
   e.g., \( SAT(B) \), the satisfiability problem for formulas with
   connectives from \( B \)

Then: If \( B \subseteq [B'] \) then \( \Pi(B) \leq \Pi(B') \)

Upper bounds carry over downwards.
Lower bounds carry over upwards.

Complexity of \( \Pi(B) \) is determined by the clone \([B]\).
So what is the benefit?

Π – computational problem defined over propositional formulas

Π(\(B\)) – the restriction of Π to formulas with connectives from \(B\), e.g., SAT(\(B\)), the satisfiability problem for formulas with connectives from \(B\)

Then: If \(B \subseteq [B']\) then \(\Pi(B) \leq \Pi(B') \sim (\text{SAT}(B) \leq^p_m \text{SAT}(B'))\).

Upper bounds carry over downwards.
Lower bounds carry over upwards.

Complexity of \(\Pi(B)\) is determined by the clone \([B]\).

Caveat: Explosion of formula size!
(Only happens if there is a formula \(f(x_1, \ldots, x_n) \in [B]\) which has not a short representation in \([B']\), i.e., there is no \(h \in [B']\) s.t. \(f \equiv h\) and each \(x_i\) appears only once in \(h\).)
Properties of Boolean Functions

Some Properties of Boolean Functions

- $f$ is $c$-reproducing if $f(c, \ldots, c) = c$, $c \in \{\bot, \top\}$.
- $f$ is monotone if $a_1 \leq b_1, \ldots, a_n \leq b_n$ implies $f(a_1, \ldots, a_n) \leq f(b_1, \ldots, b_n)$.
- $f$ is $c$-separating if there exists an $i \in \{1, \ldots, n\}$ such that $f(a_1, \ldots, a_n) = c$ implies $a_i = c$, $c \in \{\bot, \top\}$.
- $f$ is self-dual if $f \equiv \text{dual}(f)$, where $\text{dual}(f)(x_1, \ldots, x_n) = \neg f(\neg x_1, \ldots, \neg x_n)$.
- $f$ is linear if $f \equiv x_1 \oplus \cdots \oplus x_n \oplus c$ for a constant $c \in \{0, 1\}$ and variables $x_1, \ldots, x_n$. 
# Important Boolean Clones

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Base</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>All Boolean functions</td>
<td>${\land, \neg}$</td>
</tr>
<tr>
<td>$R_0$</td>
<td>${f : f$ is 0-reproducing$}$</td>
<td>${\land, \not\land}$</td>
</tr>
<tr>
<td>$R_1$</td>
<td>${f : f$ is 1-reproducing$}$</td>
<td>${\lor, \rightarrow}$</td>
</tr>
<tr>
<td>M</td>
<td>${f : f$ is monotone$}$</td>
<td>${\lor, \land, \bot, \top}$</td>
</tr>
<tr>
<td>$S_0$</td>
<td>${f : f$ is 0-separating$}$</td>
<td>${\rightarrow}$</td>
</tr>
<tr>
<td>$S_1$</td>
<td>${f : f$ is 1-separating$}$</td>
<td>${\not\land}$</td>
</tr>
<tr>
<td>D</td>
<td>${f : f$ is self-dual$}$</td>
<td>${(x \land \overline{y}) \lor (x \land \overline{z}) \lor (\overline{y} \land \overline{z})}$</td>
</tr>
<tr>
<td>L</td>
<td>${f : f$ is linear$}$</td>
<td>${\oplus, \top}$</td>
</tr>
<tr>
<td>V</td>
<td>${f : f \equiv c_0 \lor \bigvee_{i=1}^{n} c_i x_i}$</td>
<td>${\lor, \bot, \top}$</td>
</tr>
<tr>
<td>E</td>
<td>${f : f \equiv c_0 \land \bigwedge_{i=1}^{n} c_i x_i}$</td>
<td>${\land, \bot, \top}$</td>
</tr>
<tr>
<td>N</td>
<td>${f : f$ depends on only one variable$}$</td>
<td>${\neg, \bot, \top}$</td>
</tr>
<tr>
<td>I</td>
<td>${f : f$ is a projection or constant$}$</td>
<td>${\text{id}, \bot, \top}$</td>
</tr>
</tbody>
</table>
Refers to Emil Post, 1941.
- Refers to Emil Post, 1941.
- Each node corresponds to a finite set of Boolean functions (base).
Post's Lattice

- Refers to Emil Post, 1941.
- Each node corresponds to a finite set of Boolean functions (*base*).
- Many new decision problems arise (for each clone).
Post's Lattice

- Refers to Emil Post, 1941.
- Each node corresponds to a finite set of Boolean functions (base).
- Many new decision problems arise (for each clone).
- Hardness results carry over upwards.
Post's Lattice

- Refers to Emil Post, 1941.
- Each node corresponds to a finite set of Boolean functions (base).
- Many new decision problems arise (for each clone).
- Hardness results carry over upwards.
- Membership results carry over downwards.
Post's Lattice and $\mathcal{ALC}$

The "new" decision problems

- We restrict the allowed Boolean operators $o$ for composing concepts $C$ to functions from $[B]$, i.e.,

\[ C = \text{def} \ A \mid f(C, \ldots, C) \mid \exists R \cdot C \mid \forall R \cdot C, \]

where $f$ is a Boolean operator in $[B]$. 
The "new" decision problems

- We restrict the allowed Boolean operators $\circ$ for composing concepts $C$ to functions from $[B]$, i.e.,

\[
C = \text{def } A \mid f(C, \ldots, C) \mid \exists R.C \mid \forall R.C,
\]

where $f$ is a Boolean operator in $[B]$.

- $\text{CSAT}(B)$ is the concept satisfiability problem restricted to concepts using only functions in $[B]$. 

Post's Lattice and $\text{ALC}$

The “new” decision problems

- We restrict the allowed Boolean operators $\circ$ for composing concepts $C$ to functions from $[B]$, i.e.,

$$C \equiv \text{def } A \mid f(C, \ldots, C) \mid \exists R. C \mid \forall R. C,$$

where $f$ is a Boolean operator in $[B]$.

- $\text{CSAT}(B)$ is the concept satisfiability problem restricted to concepts using only functions in $[B]$.

- $\text{TSAT}(B)$, $\text{TCSAT}(B)$, $\text{OSAT}(B)$ and $\text{OCSAT}(B)$ are defined analogously.
Getting a Feeling of it

Some examples

**CSAT: Concept Satisfiability**

\[ A \land B \in \text{CSAT}([\{\land\}]) = \text{CSAT}(E_0), \]
Getting a Feeling of it

Some examples

**CSAT: Concept Satisfiability**

\[ A \land B \]

\[ (\forall R. A) \oplus (B \land C) \]

\[ \in \text{CSAT}([\land]) \quad \in \text{CSAT}([\lor, \land]) \]

\[ = \text{CSAT}(E_0), \quad = \text{CSAT}(R_0), \]
Getting a Feeling of it

Some examples

**CSAT: Concept Satisfiability**

\[ A \land B \in \text{CSAT}([[\land]]) = \text{CSAT}(E_0), \]
\[ (\forall R. A) \oplus (B \land C) \in \text{CSAT}([\{\oplus, \land\}]) = \text{CSAT}(R_0), \]
\[ \exists R.(C \rightarrow A) \in \text{CSAT}([\{\rightarrow\}]) = \text{CSAT}(S_0), \]

\[ \neg A \in \text{CSAT}([\{\neg\}) = \text{CSAT}(N_2), \]

\[ \{\top \sqcup \exists R_1. A, A \equiv B, \forall R_2. B \sqsubseteq \bot\} \in \text{TSAT}([\{\top, \bot\}) = \text{TSAT}(I). \]
Getting a Feeling of it

Some examples

CSAT: Concept Satisfiability

\[ A \land B \in \text{CSAT}([\{\land\}]) = \text{CSAT}(E_0), \]
\[ (\forall R. A) \oplus (B \land C) \in \text{CSAT}([\{\oplus, \land\}]) = \text{CSAT}(R_0), \]
\[ \exists R. (C \rightarrow A) \in \text{CSAT}([\{\rightarrow\}]) = \text{CSAT}(S_0), \]
\[ \neg A \in \text{CSAT}([\{\neg\}]) = \text{CSAT}(N_2), \]
Getting a Feeling of it

Some examples

CSAT: Concept Satisfiability, TSAT: TBox Satisfiability

\[ A \land B \in \text{CSAT}([\land]) = \text{CSAT}(E_0), \]
\[ (\forall R.A) \oplus (B \land C) \in \text{CSAT}([\oplus, \land]) = \text{CSAT}(R_0), \]
\[ \exists R.(C \rightarrow A) \in \text{CSAT}([\rightarrow]) = \text{CSAT}(S_0), \]
\[ \neg A \in \text{CSAT}([\neg]) = \text{CSAT}(N_2), \]
\[ \{\top \sqsubseteq \exists R_1.A, A \equiv B, \forall R_2.B \sqsubseteq \bot\} \in \text{TSAT}([\top, \bot]) = \text{TSAT}(I) \]
Complexity Results for $\text{CSAT}(B)$

Theorem (Hemaspaandra et al. '08)

Let $B$ be a finite set of Boolean functions.

1. If $S_{11} \subseteq [B]$, then $\text{CSAT}(B)$ is PSPACE-complete.
2. If $[B] \in \{E, E_0\}$, then $\text{CSAT}(B)$ is coNP-complete.
3. If $[B] \subseteq R_1$, then $\text{CSAT}(B)$ is trivial.
4. Otherwise $\text{CSAT}(B) \in P$.

\[
S_1 : \{\neg\}
\]
\[
S_{11} : \text{monotone and in } S_1
\]
\[
R_1 : \text{1-reproducing, i.e., } f(\top, \ldots, \top) = \top
\]
\[
E_0 : \{\land\}
\]
\[
E : \{\land, \top, \bot\}
\]
Interreducibility

Lemma

\[ \text{TSAT}(B) \leq_{m}^{\log} \text{TCSAT}(B) \leq_{m}^{\log} \text{OSAT}(B) \equiv_{m}^{\log} \text{OCSAT}(B). \]
Complexity of $\text{TSAT}(B)$

Theorem

Let $B$ be a finite set of Boolean functions. If

- $\{T, \land\} \subseteq [B]$, or
- $\{T, \lor\} \subseteq [B]$, or
- $\neg \in [B]$, or
- $\{T, \bot\} \subseteq [B]$, or
- all self-dual functions are in $[B]$, or

then $\text{TSAT}(B)$ is intractable. Otherwise $\text{TSAT}(B)$ is tractable.
Complexity of TSAT(\(B\))

- EXPTIME-hard
- PSPACE-hard
- trivial
Complexity of $\text{TCSAT}(B)$

Theorem (Lower Bounds $\text{TCSAT}(B)$)

Let $B$ be a finite set of Boolean functions. If

- $\land \in [B]$, or
- $\lor \in [B]$, or
- $\neg \in [B]$, or
- $\bot \in [B]$, or
- all self-dual functions are in $[B]$,

then $\text{TCSAT}(B)$ is intractable. Otherwise $\text{TCSAT}(B)$ is tractable. $\text{OSAT}(B)$ and $\text{OCSAT}(B)$ are the same.
Complexity of TCSAT($B$), of OSAT($B$), of OCSAT($B$)
Conclusion

Results

- Complete classification for tractable and intractable fragments.
- Found hard fragments for clones near the bottom of Post's lattice.
## Conclusion

### Results

- Complete classification for tractable and intractable fragments.
- Found hard fragments for clones near the bottom of Post's lattice.

### Outlook

- Find matching upper bounds for the hardness results.
- Look at fragments with only \( \exists \) or \( \forall \).
- Classify concept subsumption.
Conclusion

Results

- Complete classification for tractable and intractable fragments.
- Found hard fragments for clones near the bottom of Post's lattice.

Outlook

- Find matching upper bounds for the hardness results.
- Look at fragments with only $\exists$ or $\forall$.
- Classify concept subsumption.

Thank you.