Outline

Saturation Algorithms
How to Establish Unsatisfiability?

Completeness is formulated in terms of derivability of the empty clause $\square$ from a set $S_0$ of clauses in an inference system $I$. However, this formulations gives no hint on how to search for such a derivation.
How to Establish Unsatisfiability?

Completeness is formulated in terms of derivability of the empty clause □ from a set $S_0$ of clauses in an inference system $I$. However, this formulations gives no hint on how to search for such a derivation.

Idea:

- Take a set of clauses $S$ (the search space), initially $S = S_0$. Repeatedly apply inferences in $I$ to clauses in $S$ and add their conclusions to $S$, unless these conclusions are already in $S$.
- If, at any stage, we obtain □, we terminate and report unsatisfiability of $S_0$. 

How to Establish Satisfiability?

When can we report satisfiability?
How to Establish Satisfiability?

When can we report satisfiability?

When we build a set $S$ such that any inference applied to clauses in $S$ is already a member of $S$. Any such set of clauses is called saturated (with respect to $I$).
How to Establish Satisfiability?

When can we report **satisfiability**?

When we build a set $S$ such that any inference applied to clauses in $S$ is already a member of $S$. Any such set of clauses is called **saturated** (with respect to $I$).

In first-order logic it is often the case that all saturated sets are infinite (due to undecidability), so in practice we can never build a saturated set.

The process of trying to build one is referred to as **saturation**.
Let $\mathcal{I}$ be an inference system on formulas and $S$ be a set of formulas.

- $S$ is called saturated with respect to $\mathcal{I}$, or simply $\mathcal{I}$-saturated, if for every inference of $\mathcal{I}$ with premises in $S$, the conclusion of this inference also belongs to $S$.

- The closure of $S$ with respect to $\mathcal{I}$, or simply $\mathcal{I}$-closure, is the smallest set $S'$ containing $S$ and saturated with respect to $\mathcal{I}$.
Inference Process

Inference process: sequence of sets of formulas $S_0, S_1, \ldots$, denoted by

$$S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$$

$(S_i \Rightarrow S_{i+1})$ is a step of this process.
Inference Process

Inference process: sequence of sets of formulas $S_0, S_1, \ldots$, denoted by

$$S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$$

($S_i \Rightarrow S_{i+1}$) is a step of this process.

We say that this step is an $I$-step if

1. there exists an inference

$$\frac{F_1 \ldots F_n}{F}$$

in $\Pi$ such that $\{F_1, \ldots, F_n\} \subseteq S_i$;

2. $S_{i+1} = S_i \cup \{F\}$. 
Inference Process

Inference process: sequence of sets of formulas $S_0, S_1, \ldots$, denoted by

$$S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$$

$(S_i \Rightarrow S_{i+1})$ is a step of this process.

We say that this step is an $I$-step if

1. there exists an inference

$$\frac{F_1 \ldots F_n}{F}$$

in $I$ such that $\{F_1, \ldots, F_n\} \subseteq S_i$;

2. $S_{i+1} = S_i \cup \{F\}$.

An $I$-inference process is an inference process whose every step is an $I$-step.
Let $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ be an $\mathbb{I}$-inference process and a formula $F$ belongs to some $S_i$. Then $S_i$ is derivable in $\mathbb{I}$ from $S_0$. In particular, every $S_i$ is a subset of the $\mathbb{I}$-closure of $S_0$. 

Property
The limit of an inference process $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ is the set of formulas $\bigcup_i S_i$. 
The limit of an inference process $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ is the set of formulas $\bigcup_i S_i$.

In other words, the limit is the set of all derived formulas.
The **limit** of an inference process \( S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots \) is the set of formulas \( \bigcup_i S_i \).

In other words, the limit is the set of all derived formulas.

Suppose that we have an infinite inference process such that \( S_0 \) is **unsatisfiable** and we use the **binary resolution inference system**.
The limit of an inference process $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ is the set of formulas $\bigcup_i S_i$.

In other words, the limit is the set of all derived formulas.

Suppose that we have an infinite inference process such that $S_0$ is unsatisfiable and we use the binary resolution inference system.

**Question**: does completeness imply that the limit of the process contains the empty clause?
Let $S_0 \Rightarrow S_1 \Rightarrow S_2 \Rightarrow \ldots$ be an inference process with the limit $S_\infty$. The process is called fair if for every \(\Pi\)-inference

$$
\begin{array}{c}
F_1 \ldots F_n \\
\hline \\
F
\end{array},
$$

if \(\{F_1, \ldots, F_n\} \subseteq S_\infty\), then there exists \(i\) such that \(F \in S_i\).
Completeness, reformulated

**Theorem** Let II be an inference system. The following conditions are equivalent.

1. II is complete.
2. For every unsatisfiable set of formulas $S_0$ and any fair II-inference process with the initial set $S_0$, the limit of this inference process contains $\Box$. 
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- given clause

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- given clause
- candidate clauses
- search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- **search space**
- **candidate clauses**
- **given clause**
- **children**
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space

children
Fair Saturation Algorithms: Inference Selection by Clause Selection

\[ \text{children} \]

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- given clause

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- given clause
- candidate clauses
- search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

- search space
- given clause
- candidate clauses
- children
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space

children
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space

children
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

search space
Fair Saturation Algorithms: Inference Selection by Clause Selection

MEMORY

search space
A **saturation algorithm** tries to **saturate** a set of clauses with respect to a given inference system.

**In theory** there are three possible scenarios:

1. At some moment the empty clause $\square$ is generated, in this case the input set of clauses is unsatisfiable.
2. Saturation will terminate without ever generating $\square$, in this case the input set of clauses in satisfiable.
3. Saturation will run **forever**, but without generating $\square$. In this case the input set of clauses is **satisfiable**.
Saturation Algorithm in Practice

In practice there are three possible scenarios:

1. At some moment the empty clause □ is generated, in this case the input set of clauses is unsatisfiable.
2. Saturation will terminate without ever generating □, in this case the input set of clauses is satisfiable.
3. Saturation will run until we run out of resources, but without generating □. In this case it is unknown whether the input set is unsatisfiable.
Saturation Algorithm

Even when we implement inference selection by clause selection, there are too many inferences, especially when the search space grows.
Saturation Algorithm

Even when we implement inference selection by clause selection, there are too many inferences, especially when the search space grows. **Solution:** only apply inferences to the selected clause and the previously selected clauses.
Saturation Algorithm

Even when we implement inference selection by clause selection, there are too many inferences, especially when the search space grows.

**Solution:** only apply inferences to the selected clause and the previously selected clauses.

Thus, the search space is divided in two parts:

- **active clauses**, that participate in inferences;
- **passive clauses**, that do not participate in inferences.

**Observation:** the set of passive clauses is usually considerably larger than the set of active clauses, often by 2-4 orders of magnitude (depending on the saturation algorithm and the problem).