Compactness Theorem
Compactness Theorem

(⇐) Let $A_1, A_2, A_3 \cdots$ be the atomic formulas in $\Gamma$. 
Compactness Theorem

(⇐) Assume that we defined an assignment \( \nu \) of truth-values to \( A_1, \ldots, A_n \) such that each finite subset of \( \Gamma \) has a model in which \( A_1, \ldots, A_n \) assume the values \( \nu(A_1), \ldots, \nu(A_n) \).
Compactness Theorem

\[(\Leftarrow)\] Assume that we defined an assignment \(v\) of truth-values to \(A_1, \cdots, A_n\) such that each finite subset of \(\Gamma\) has a model in which \(A_1, \cdots, A_n\) assume the values \(v(A_1), \cdots, v(A_n)\). Suppose, to fix ideas, that when assigning \(v(A_{n+1}) = 0\) the above condition does not hold;
Compactness Theorem

\((\Leftarrow)\) Assume that we defined an assignment \(v\) of truth-values to \(A_1, \cdots, A_n\) such that each finite subset of \(\Gamma\) has a model in which \(A_1, \cdots, A_n\) assume the values \(v(A_1), \cdots, v(A_n)\). I.e. there exists a finite set \(\Delta' \subseteq \Gamma\), that has no model in which \(A_1, \cdots, A_n, A_{n+1}\) assume the values \(v(A_1), \cdots, v(A_n)\) and \(v(A_{n+1}) = 0\).
Compactness Theorem

\[ \iff \]
Assume that we defined an assignment \( v \) of truth-values to \( A_1, \ldots, A_n \) such that each finite subset of \( \Gamma \) has a model in which \( A_1, \ldots, A_n \) assume the values \( v(A_1), \ldots, v(A_n) \). I.e. there exists a finite set \( \Delta' \subseteq \Gamma \), that has no model in which \( A_1, \ldots, A_n, A_{n+1} \) assume the values \( v(A_1), \ldots, v(A_n) \) and \( v(A_{n+1}) = 0 \). We show that each finite subset of \( \Gamma \) has a model in which \( A_1, \ldots, A_n, A_{n+1} \) assume the values \( v(A_1), \ldots, v(A_n) \) and \( v(A_{n+1}) = 1 \).
Compactness Theorem

\[(\Leftarrow)\] Assume that we defined an assignment \(v\) of truth-values to \(A_1, \ldots, A_n\) such that each finite subset of \(\Gamma\) has a model in which \(A_1, \ldots, A_n\) assume the values \(v(A_1), \ldots, v(A_n)\). I.e. there exists a finite set \(\Delta' \subseteq \Gamma\), that has no model in which \(A_1, \ldots, A_n, A_{n+1}\) assume the values \(v(A_1), \ldots, v(A_n)\) and \(v(A_{n+1}) = 0\).

Let \(\Delta\) be any finite subset of \(\Gamma\);
Compactness Theorem

(⇐) Assume that we defined an assignment $v$ of truth-values to $A_1, \ldots, A_n$ such that each finite subset of $\Gamma$ has a model in which $A_1, \ldots, A_n$ assume the values $v(A_1), \ldots, v(A_n)$. I.e. there exists a finite set $\Delta' \subseteq \Gamma$, that has no model in which $A_1, \ldots, A_n, A_{n+1}$ assume the values $v(A_1), \ldots, v(A_n)$ and $v(A_{n+1}) = 0$.

Let $\Delta$ be any finite subset of $\Gamma$; by i.h. $\Delta \cup \Delta'$ has a model in which $A_1, \ldots, A_n$ assume the values $v(A_1), \ldots, v(A_n)$ and, due to $\Delta'$, $v(A_{n+1})$ is necessarily 1.
(⇐) Assume that we defined an assignment $v$ of truth-values to $A_1, \cdots, A_n$ such that each finite subset of $\Gamma$ has a model in which $A_1, \cdots, A_n$ assume the values $v(A_1), \cdots, v(A_n)$. I.e. there exists a finite set $\Delta' \subseteq \Gamma$, that has no model in which $A_1, \cdots, A_n, A_{n+1}$ assume the values $v(A_1), \cdots, v(A_n)$ and $v(A_{n+1}) = 0$.

Let $\Delta$ be any finite subset of $\Gamma$; by i.h. $\Delta \cup \Delta'$ has a model in which $A_1, \cdots, A_n$ assume the values $v(A_1), \cdots, v(A_n)$ and, due to $\Delta'$, $v(A_{n+1})$ is necessarily 1.

The limit $v$ of this construction is a model of $\Gamma$. 
SAT and TAUT
SAT and TAUT

SAT: is the problem of determining whether any arbitrary proposition is satisfiable.
SAT and TAUT

- **SAT**: is the problem of determining whether any arbitrary proposition is satisfiable.
- **TAUT**: is the problem of determining whether any arbitrary proposition is a tautology.
NP-Completeness of SAT
NP-Completeness of SAT

NP-problem: A decision problem is in NP if a non-deterministic Turing machine can solve it in polynomial time.
NP-Completeness of SAT

- **NP-problem**: A decision problem is in NP if a non-deterministic Turing machine can solve it in polynomial time.

- **NP-complete**: A decision problem is NP-complete if it is in NP and every problem in NP can be converted into it by a transformation of the inputs in polynomial time.
NP-Completeness of SAT
NP-Completeness of SAT

Proof

Suppose that a problem in NP is solved by a non-deterministic Turing machine $M = (Q, \Sigma, s, F, \delta)$ and that $M$ accepts or rejects an instance of the problem in time $p(n)$, where $n$ is the size of the instance and $p$ is a polynomial function.
NP-Completeness of SAT

Proof

Suppose that a problem in NP is solved by a non-deterministic Turing machine $M = (Q, \Sigma, s, F, \delta)$ and that $M$ accepts or rejects an instance of the problem in time $p(n)$, where $n$ is the size of the instance and $p$ is a polynomial function.

For each instance $I$ of the problem we construct a formula $B_I$ of classical logic such that $B_I$ is satisfiable iff the $M$ accepts $I$. 
NP-Completeness of SAT

\[ M = (Q, \Sigma, s, F, \delta), \text{ where} \]

- \( Q \) \ldots finite set of states, \( \Sigma \) \ldots finite tape alphabet
- \( s \in Q \) \ldots initial state, \( F \subseteq Q \) \ldots set of accepting states
- \( \delta \) \ldots set of transitions (\( \delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{-1, +1\} \))
NP-Completeness of SAT

\[ M = (Q, \Sigma, s, F, \delta), \text{ where} \]

\begin{itemize}
  \item $Q$ \ldots finite set of states, $\Sigma$ \ldots finite tape alphabet
  \item $s \in Q$ \ldots initial state, $F \subseteq Q$ \ldots set of accepting states
  \item $\delta$ \ldots set of transitions ($\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{-1, +1\}$
\end{itemize}

\[
q \in Q \quad -p(n) \leq i \leq p(n) \\
j \in \Sigma \quad 0 \leq k \leq p(n)
\]

\[
T_{ijk} \quad \text{true iff tape cell } i \text{ contains symbol } j \text{ at step } k \\
H_{ik} \quad \text{true iff the } M\text{'s r/w head is at tape cell } i \text{ at step } k \\
Q_{qk} \quad \text{true iff } M \text{ is in state } q \text{ at step } k \text{ of the computation}
\]
Twenty Questions game

Someone thinks of a number between 1 and 1 million ($< 2^{20}$). Another person is allowed to ask up to 20 questions, to each of which the first person is supposed to answer only yes or no.
Twenty Questions game

Someone thinks of a number between 1 and 1 million ($< 2^{20}$). Another person is allowed to ask up to 20 questions, to each of which the first person is supposed to answer only yes or no.

- The conjunction of two equal answers to the same repeated question needs not to be equivalent to the same answer.
Twenty Questions game

Someone thinks of a number between 1 and 1 million ($< 2^{20}$). Another person is allowed to ask up to 20 questions, to each of which the first person is supposed to answer only yes or no.

- The conjunction of two equal answers to the same repeated question needs not to be equivalent to the same answer.
- The conjunction of two opposite answers to the same repeated question needs not to lead to a contradiction.