$\mathcal{T}$-Conflict. This is because in this case, Conflict would be applicable. Analogous to the rules Backtrack and $\mathcal{T}$-Propagate, it would be possible to define $\mathcal{T}$-Conflict such that L is immediately propagated, i.e., the resulting trail would be M_1L instead of M_1.

Note that the clause $C \lor L$ used to justify $\mathcal{T}$-Propagate as well as the clause $C \lor L$ used for $\mathcal{T}$-Conflict and the clause C added by $\mathcal{T}$-Learn are tautologies in the theory. The calculus would be complete even without the rules $\mathcal{T}$-Propagate, $\mathcal{T}$-Atom and $\mathcal{T}$-Learn. In addition to the normal CDCL rules, only $\mathcal{T}$-Success to get from a model M of N to a final state and $\mathcal{T}$-Conflict to reject a $\mathcal{T}$-unsatisfiable model M are needed. The remaining rules are just for efficiency. There is a tradeoff between at any time checking M with respect to the theory and thus avoiding $\mathcal{T}$-Conflicts, called *eager theory consideration*, and computing with respect to the boolean structure and thus taking into account eventual extra $\mathcal{T}$-Conflict, called *lazy theory consideration.* Similarly, it is not obvious whether the applicability of $\mathcal{T}$-Conflict should be checked eagerly because this might be expensive.

The minimal requirement for $\mathcal{T}$ is a decision procedure that checks for a conjunction of literals whether it is satisfiable, and that, if not, ideally provides a minimal unsatisfiable subset.

Example 7.2.2. This example illustrates different strategies as to when to consider the theory. Consider the clause set $N = \{P, Q, R \lor S\}$ with $V = \{P, Q, R, S\}$ and $\text{atr}^{-}1(P) = (a \geq 0), \text{atr}^{-}1(Q) = (a + b < 0), \text{atr}^{-}1(R) = (b > 0), \text{atr}^{-}1(S) = (b = 0)$. A run using only the rules used in the completeness proof [7.2.8], serving as an example for lazy theory consideration, might look as follows.

$$\begin{array}{ll}
 & (V; \epsilon; N; \emptyset; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P; N; \emptyset; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P Q^Q; N; \emptyset; 0; \top) \\
\Rightarrow^{\text{Decide}}_{\text{CDCL(T)}} & (V; P^P Q^Q R^1; N; \emptyset; 1; \top) \\
\Rightarrow^{\mathcal{T}\text{-Conflict}}_{\text{CDCL(T)}} & (V; P^P Q^Q; N; \{C := \neg P \lor \neg Q \lor \neg R\}; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P Q^Q \neg R^C; N; \{C\}; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P Q^Q \neg R^C S^{S \lor R}; N; \{C\}; 0; \top) \\
\Rightarrow^{\mathcal{T}\text{-Conflict}}_{\text{CDCL(T)}} & (V; P^P Q^Q \neg R^C; N; \{C, D := \neg P \lor \neg Q \lor \neg S\}; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P Q^Q \neg R^C S^{S \lor R}; N; \{C, D\}; 0; \top) \\
\Rightarrow^{\text{Conflict}}_{\text{CDCL(T)}} & (V; P^P Q^Q \neg R^C S^{S \lor R}; N; \{C, D\}; 0; D = \neg P \lor \neg Q \lor \neg S)
\end{array}$$

From this point on, the clause set will be refuted by four applications of the rule Resolve. In the second application of $\mathcal{T}$-Conflict, as mentioned earlier, $S^{S \lor R}$ could have been left on the trail if the rule was defined differently.

Here, it was removed by $\mathcal{T}$-Conflict and then immediately added again by Propagate.

An alternative run with more eager theory consideration on the same clause set could look as follows.

$$
\begin{array}{ll}
 & (V; \epsilon; N; \emptyset; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P; N; \emptyset; 0; \top) \\
\Rightarrow^{\text{Propagate}}_{\text{CDCL(T)}} & (V; P^P Q^Q; N; \emptyset; 0; \top) \\
\Rightarrow^{\mathcal{T}\text{-Atom}}_{\text{CDCL(T)}} & (V' := V \cup \{T : \text{atr}^- 1(T) = (b < 0)\}; P^P Q^Q; N; \emptyset; 0; \top) \\
\Rightarrow^{\mathcal{T}\text{-Propagate}}_{\text{CDCL(T)}} & (V'; P^P Q^Q T^C; N; \{C := \neg P \vee \neg Q \vee T\}; 0; \top) \\
\Rightarrow^{\mathcal{T}\text{-Propagate}}_{\text{CDCL(T)}} & (V'; P^P Q^Q T^C \neg R^D; N; \{C, D := \neg T \vee \neg R\}; 0; \top) \\
\Rightarrow^{\mathcal{T}\text{-Propagate}}_{\text{CDCL(T)}} & (V'; P^P Q^Q T^C \neg R^D \neg S^E; N; \{C, D, E := \neg T \vee \neg S\}; 0; \top) \\
\Rightarrow^{\text{Conflict}}_{\text{CDCL(T)}} & (V'; P^P Q^Q T^C \neg R^D \neg S^E; N; \{C, D, E\}; 0; R \vee S)
\end{array}
$$

From this point on, the clause set will be refuted by five applications of the rule Resolve. This run could have been obtained by feeding the trail into the theory solver literal by literal. After adding P and Q, the theory solver might have deduced $b < 0$, which was added to the CDCL(T) state as the propositional variable T using $\mathcal{T}$-Atom and $\mathcal{T}$-Propagate. Given the atoms R and S already present in the CDCL(T) state, the theory solver might have found that $\neg T \vee \neg R$ and $\neg T \vee \neg S$, two facts that were added to the CDCL(T) state using $\mathcal{T}$-Propagate.

Note that the initial clause set N is not modified by any rule, so we can assume that N is constant in any run. For all other objects occurring in the statements and proofs of the following lemmas and theorems, we use a notation like $\hat{C}$ to refer to variables introduced in the respective lemma, theorem or proof. Variables without a hat like C refer to the variables used in the definitions of the rules.

Lemma 7.2.3 (Invariants)**.** If $(\hat{V}; \epsilon; N; \emptyset; 0; \top) \Rightarrow^*_{\text{CDCL(T)}} (\hat{V}'; \hat{M}; N; \hat{U}; \hat{k}; \hat{D})$, then the following hold:

1. there are no literals $\hat{L}, \hat{K}$ such that $\hat{M} = \hat{M}_1 \hat{L} \hat{M}_2 \hat{K} \hat{M}_3$ and $\text{atom}(\hat{L}) = \text{atom}(\hat{K})$,
2. $\hat{k}$ is the highest level of any literal $\hat{L}$ in $\hat{M}$, or $\hat{k} = 0$ if $\hat{M} = \epsilon$,
3. and for all atoms $\hat{A} \in \text{atom}(N \cup \hat{U} \cup \hat{M} \cup \hat{D})$, $\hat{A} \in \hat{V}$.

Proof. All properties are shown by induction on the length of the derivation. For the start state, all properties obviously hold.

1. We only have to show something if the last applied rule was Backtrack,