Yale University
Department of Computer Science

Representing Reductions of NP-complete Problems in Logical Frameworks – A Case Study

Carsten Schürmann Jatin Shah

YALEU/DCS/TR-1254
August 2003

Representing Reductions of NP-complete Problems in Logical Frameworks – A Case Study*

Carsten Schürmann† Jatin Shah‡

Abstract

Under the widely believed conjecture P≠NP, NP-complete problems cannot be solved exactly using efficient polynomial time algorithms. Furthermore, any instance of a NP-complete problem can be converted to an instance of another problem in NP in polynomial time. Thus, identifying NP-complete problems is very important in algorithm design and can help computer scientists and engineers redirect their efforts towards finding approximate solutions to these problems. As a first step towards a digital library for NP-complete problems, we describe a case study involving two well-known NP-complete problems 3-SAT and CHROMATIC together with a reduction and the corresponding soundness proof in a logical framework.

1 Introduction

Logical frameworks are often used as meta-languages in logic and programming language design to describe deductive systems that are prevalent in type theories, proof theories, and programming language semantics. In this paper we apply logical framework technology to a different area, the domain of NP-complete problems, and study and evaluate one particular example: The reduction of 3-satisfiability (3-SAT) of conjunctive normal forms where each clause consists of exactly three literals to the chromatic number of a graph (CHROMATIC). We implement the reduction in the linear logical

*This work was supported in part by the National Science Foundation under grants CCR-0133502 and CCR-9985304

†Department of Computer Science, Yale University, CT

‡Department of Computer Science, Yale University, CT

framework LLF [CP96] which extends the logical framework LF [HHP93] by the concept of depletable resources.

Problems in complexity theory and their instances are usually described in terms of graphs, formulas, partitions, matchings, models, colorings etc., which lack in general deductive formulations that would render them directly implementable in a logical framework. Most problems of this kind are currently formalized, analyzed, and mechanically solved using logic. For example, in a second- or higher-order logic propositions such as "G is a graph", "p is in P", or "G is n colorable" may be expressed as types. Any general purpose proof assistant or theorem prover for such a logic can subsequently assist the reasoning process. Standard properties that need to be verified of a reduction include the preservation of solutions, and polynomial space and time complexity. On the other hand, since second-order and higher-order logics are designed for a general purpose, no special purpose facilities are provided to reason about graphs, reductions, running time, space requirements, and other properties innate to the question of reducibility.

In this paper we explore an alternative idea and employ special purpose type theories to represent complexity problems as types, instances of those problems as objects, and consequently reductions as mappings from well-typed objects to well-typed objects. We illustrate and evaluate this idea using the (3SAT) to (CHROMATIC) reduction. The reduction described in this paper is well-known [GJ79] and turns Boolean formulas in conjunctive normal form into graphs by inserting two vertices for each variable (one that corresponds to the positive form of the literal, the other to its negation) and a vertex for each individual clause. Depending on the literals contained in each clause, new edges are also inserted into the graph.

The existence of such a reduction entails that finding the chromatic number of the resulting graph is at least as complex to constructing a model for a original Boolean formula. Since 3-SAT is known to be NP-complete, and the reduction is polynomial-time computable (it maps objects of size n that encode formulas into objects of size $p(n)$ for some polynomial p that correspond to graphs), CHROMATIC is consequently an NP-complete problem.

The development presented in this paper gives a deductive account of all concepts involved, including formulas, semantical models, graphs, colorings, and reductions, and it describes the corresponding formalization in a prototype implementation of LLF, which is called linear Twelf. Unfortunately, linear Twelf is lacking automatic support for termination and coverage checking necessary to ensure that a relational encoding of a total function really constitutes a proof. We have, however, verified these

conditions by hand. The source code of this development is available from `http://www.cs.yale.edu/~jds58/chromatic.elf`.

This paper is organized in the following way. In Section 2, we describe the two central complexity theoretic problems of 3-SAT and CHROMATIC, respectively, and formulate them in the form of an inference system. How to convert 3-SAT to CHROMATIC is described in Section 3 followed by Section 4 on how to encode the problems and the respective reduction in LLF. In Section 5 we give a correctness proof of the reduction, and show that it is indeed total. We evaluate the case study in Section 6 and conclude with future work in Section 7.

2 Two NP-Complete Problems

The two fundamental problems in NP that are studied in this paper are 3-satisfiability (3-SAT) and chromatic number (CHROMATIC). We first proceed by presenting them in the familiar standard theoretical computer science discourse below, followed by a reformulation as an inference system. For more information about these and other NP-complete problems, the reader may refer to [GJ79].

Definition 2.1 (3-SAT) *Given a set* $U = \{u_1, u_2, \ldots, u_n\}$ *of Boolean variables and a conjunctive normal form formula* $f = c_1 \wedge c_2 \wedge \ldots c_m$ *on the Boolean variables in* U *such that* $c_i = l_{i1} \vee l_{i2} \vee l_{i3}, \forall i = 1, \ldots, m$ *and* $l_{i1}, l_{i2}, l_{i3} \in U \cup \overline{U}$ *where* $\overline{U} = \{\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_n\}$*. Is there a truth assignment to the Boolean variables such that every clause in* f *is satisfied?*

Definition 2.2 (CHROMATIC) *Given a graph* $G = (V, E)$ *where* V *is the set of vertices and* E *is the set of edges, and a positive integer* C*. Is* G C*-colorable, i.e., does there exist a function* $\chi : V \to \{1, 2, \ldots, C\}$ *such that* $\chi(u) \neq \chi(v)$ *whenever* $\{u, v\} \in E$*?*

Boolean variables		u, u_n
Boolean formulas	$f, f_n ::=$	$\mathsf{pos}\ u \mid \mathsf{neg}\ u \mid \mathsf{new}\ u.f \mid f_m \wedge f_n \mid f_m \vee f_n$
Vertex variables		$v, w, x, \ldots, v_n, w_n, x_n, \ldots$
Edges		e, e_n
Graphs	$G, G_n ::=$	$\# \mid \mathsf{newv}\ v.G \mid \mathsf{newe}\ e : (v_m, v_n).G \mid G_m \cup G_n$

Figure 1: Description of instances of Boolean formulas and graphs.

3-SAT's domain is that of propositional formulas with the connectives *and*, *or*, and *not* whereas CHROMATIC's is that of graphs consisting of vertices and edges. Both domains are depicted in Figure 1. Boolean variables are schematic and denoted with $u_1 \dots u_n$, and vertices and edges are represented by variables as well, being bound by new, newv, and newe. Without loss of generality, we assume that the individual conjuncts in a Boolean formula do not contain references to new. Colors C can be thought of as integers.

Next, we capture the essence of 3-SAT and CHROMATIC, formally. Of course, each definition can simply be expressed as a first-order formula enriched with predicates that describe formulas and graphs. When using a logical framework, however, it is easier to capture the respective meaning in form of two inference systems that are depicted in Figures 2 and 3.

$$\frac{}{\eta, u \to \mathsf{true} \vdash (\mathsf{pos}\ u)\ \mathsf{SAT}}\ satp \qquad \frac{}{\eta, u \to \mathsf{false} \vdash (\mathsf{neg}\ u)\ \mathsf{SAT}}\ satn$$

$$\frac{\eta \vdash F_1\ \mathsf{SAT} \quad \eta \vdash F_2\ \mathsf{SAT}}{\eta \vdash (F_1 \wedge F_2)\ \mathsf{SAT}}\ sat\wedge$$

$$\frac{\eta \vdash F_1\ \mathsf{SAT}}{\eta \vdash (F_1 \vee F_2)\ \mathsf{SAT}}\ sat\vee 1 \qquad \frac{\eta \vdash F_2\ \mathsf{SAT}}{\eta \vdash (F_1 \vee F_2)\ \mathsf{SAT}}\ sat\vee 2$$

$$\frac{\eta, u \to \mathsf{true} \vdash F\ \mathsf{SAT}}{\eta \vdash \mathsf{new}\ u.F\ \mathsf{SAT}}\ satt \qquad \frac{\eta, u \to \mathsf{false} \vdash F\ \mathsf{SAT}}{\eta \vdash \mathsf{new}\ u.F\ \mathsf{SAT}}\ satf$$

Figure 2: Inference rules for "Yes" instances of 3-SAT.

The statement that an instance of 3-SAT is a "Yes" instance, i.e. the Boolean formula F has a model, is written as $\eta \vdash F\ \mathsf{SAT}$ where environments η contain assignments for the free variables in F.

$$\text{Environments:} \quad \eta \quad ::= \quad \cdot \mid \eta, u \to \mathsf{true} \mid \eta, u \to \mathsf{false}$$

Environments also satisfy the standard properties of intuitionistic contexts, such as *weakening*, *strengthening* and *permutation*, allowing us to use higher-order encodings to represent them in a logical framework as discussed in Section 4.

Similarly, every "Yes" instance satisfying Definition 2.2 can be expressed as a derivation in the inference system depicted in Figure 3. For any graph G and color C, the judgment $\eta \vdash G\ C\ \mathsf{COLORING}$ means that the graph G

$$\frac{}{\eta \vdash \#\ C\ \mathsf{COLORING}}\ cgempty$$

$$\frac{C' \leq C \quad \eta, v \to C' \vdash G\ C\ \mathsf{COLORING}}{\eta \vdash \mathsf{newv}\ v.G\ C\ \mathsf{COLORING}}\ cgvertex$$

$$\frac{C_1 \leq C \quad C_2 \leq C \quad C_1 \neq C_2 \quad \eta, A \to C_1, B \to C_2 \vdash G\ C\ \mathsf{COLORING}}{\eta, A \to C_1, B \to C_2 \vdash \mathsf{newe}\ e : (A, B).G\ C\ \mathsf{COLORING}}\ cgedge$$

$$\frac{\eta \vdash G_1\ C\ \mathsf{COLORING} \quad \eta \vdash G_2\ C\ \mathsf{COLORING}}{\eta \vdash (G_1 \cup G_2)\ C\ \mathsf{COLORING}}\ cgunion$$

Figure 3: Inference rules for "Yes" instances of CHROMATIC.

can be colored with at most C colors where colors for free vertices in G are colored as defined in η, that is extended to also bind colors to vertices.

$$\text{Environment extension:} \quad \eta \quad ::= \quad \cdots \mid \eta, A \to C$$

No edge connects two vertices of same color. It is not hard to see that an instance of 3-SAT or CHROMATIC will have a deduction if and only if it is a "Yes" instance. As an example, the proof that the Boolean formula $\bar{u}_1 \wedge u_2$ is satisfiable is given in Figure 4.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{\cdot, u_1 \to \mathsf{false}, u_2 \to \mathsf{true} \vdash (\mathsf{neg}\ u_1)\ \mathsf{SAT}}\ satn \quad \cfrac{}{\cdot, u_1 \to \mathsf{false}, u_2 \to \mathsf{true} \vdash (\mathsf{pos}\ u_2)\ \mathsf{SAT}}\ satp}{\cdot, u_1 \to \mathsf{false}, u_2 \to \mathsf{true} \vdash (\mathsf{neg}\ u_1) \wedge (\mathsf{pos}\ u_2)\ \mathsf{SAT}}\ sat\wedge}{\cdot, u_1 \to \mathsf{false} \vdash \mathsf{new}\ u_2.(\mathsf{neg}\ u_1) \wedge (\mathsf{pos}\ u_2)\ \mathsf{SAT}}\ satt}{\cdot \vdash \mathsf{new}\ u_1.\mathsf{new}\ u_2.(\mathsf{neg}\ u_1) \wedge (\mathsf{pos}\ u_2)\ \mathsf{SAT}}\ satf$$

Figure 4: Proof that the Boolean formula $\bar{u}_1 \wedge u_2$ is satisfiable

However, it is important to note that the intractability of an NP-complete problem has a mirror image in the world of inference systems as well. A graph G and a color C forms an instance of Definition 2.2 if and only if a derivation of $\cdot \vdash G\ C\ \mathsf{COLORING}$ exists, which may involve checking all possible color assignments for vertices in the instance.

3 3-SAT CHROMATIC Reduction

A *polynomial time reduction* from 3-SAT to CHROMATIC consists in showing that there exists an algorithm which runs in time polynomial in the size

of the Boolean formula that converts every instance of 3-SAT to an instance of CHROMATIC such that all "Yes" instances of 3-SAT are mapped to "Yes" instances of CHROMATIC and vice-versa. Instead of mapping a Boolean formula F to a graph G, we shall use the inference system formulation from the previous section to represent the *polynomial time reduction* as a total function mapping derivation of $\eta \vdash F\ \mathsf{SAT}$ into derivations of $\eta \vdash G\ C\ \mathsf{COLORING}$.

Following [Kar72] and [Lew] we sketch the reduction first informally before formalizing it further. Suppose, we are given an instance of 3-SAT as described in Definition 2.1.

1. For every variable u_i, create vertices v_i, v_i' and x_i. For every clause c_j, create a vertex c_j in the graph.

2. Connect the edges between these vertices as below:

   (a) For every i, add an edge $\{v_i, v_i'\}$.

   (b) For every i and j, add an edge $\{x_i, x_j\}$ when $i \neq j$.

   (c) For every i and j, add an edge $\{v_i, x_j\}$ and $\{v_i', x_j\}$ when $i \neq j$.

   (d) For every i and j, add an edge $\{c_i, v_j\}$ if u_j does not appear in c_i and an edge $\{c_i, v_j'\}$ if $\bar{u}_j$ does not appear in c_i.

It is not hard to see that if the Boolean formula with n variables has a truth assignment then the graph has a $n+1$-coloring and vice versa. Essentially, the construction given above – connecting v_i's and v_i''s to the clique on x_i's – forces creation of n true colors and one false color.

A formalization of this reduction, again in form of a inference system is given in Figure 5. The main judgment is of the form $\Gamma; \Delta \vdash K \diamond F \Rightarrow_C C', G$, where Γ is a list of assumptions of the form (u, v, v', x) representing a relationship between a free Boolean variable u in F and its corresponding free graph vertices v, v' and x in G. Δ is a list of all distinct Boolean variables used in the Boolean formula (see rule *c_new*). Eventually, it will contain all free Boolean variables in F. We also maintain two colors C and C': C is incremented every time we see a new variable and C' corresponds to the total number of variables. All clauses that are contained in the Boolean formula prompt the insertion of edges into the graph corresponding to step (d) of the conversion algorithm. We achieve this by maintaining a "continuation" stack of clauses that were already encountered but not yet processed. The language of continuation stack is given below. Here $*$ is the initial continuation, indicating that we have no more clauses left. [Pfe01a] describes

$$\frac{\Gamma, (u, v, v', x); \Delta, u \vdash K \diamond F \Rightarrow_{C+1} C', G}{\Gamma; \Delta \vdash K \diamond \mathsf{new}\ u.F \Rightarrow_C C', \mathsf{newv}\ v\ v'\ x.\mathsf{newe}\ e : (v, v').G}\ c\_new$$

$$\frac{\Gamma; \Delta \vdash K; F \diamond F' \Rightarrow_C C', G}{\Gamma; \Delta \vdash K \diamond F \wedge F' \Rightarrow_C C', G}\ c\_\wedge$$

$$\frac{\Gamma; \Delta \vdash K; (F_1 \vee F_2 \vee F_3) \Rightarrow G_1 \quad \Gamma; \Delta \vdash G_2\ \mathsf{CLIQUE} \quad \Gamma; \Delta \vdash G_3\ \mathsf{VARS\text{-}TO\text{-}CLIQUE}}{\Gamma; \Delta \vdash K \diamond (F_1 \vee F_2 \vee F_3) \Rightarrow_C C, G_1 \cup G_2 \cup G_3}\ c\_\vee$$

$$\frac{}{\Gamma; \Delta \vdash * \Rightarrow \#}\ c'\_*$$

$$\frac{\Gamma; \Delta \vdash K \Rightarrow G_1 \quad \Gamma; \Delta \vdash F \Rightarrow G_2}{\Gamma; \Delta \vdash K; F \Rightarrow G_1 \cup G_2}\ c'\_;$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2')\ e_3 : (c, v_3').G}\ c''5.1$$

(39 similar rules. See Appendix A)

$$\frac{}{\Gamma; \cdot \vdash C \downarrow \#}\ c'''\_\#$$

$$\frac{\Gamma, (u, v, v', x); \Delta \vdash C \downarrow G}{\Gamma, (u, v, v', x); \Delta, u \vdash C \downarrow \mathsf{newe}\ e : (C, v)\ e' : (C, v').G}\ c'''\_v$$

$$\frac{}{\Gamma; \cdot \vdash \#\ \mathsf{CLIQUE}}\ clique\_\#$$

$$\frac{\Gamma, (u, v, v', x); \Delta \vdash G_1\ \mathsf{CLIQUE} \quad \Gamma, (u, v, v', x); \Delta \vdash \mathsf{CONNECTX}\ x\ G_2}{\Gamma, (u, v, v', x); \Delta, u \vdash (G_1 \cup G_2)\ \mathsf{CLIQUE}}\ clique\_v$$

$$\frac{}{\Gamma; \cdot \vdash \#\ \mathsf{VARS\text{-}TO\text{-}CLIQUE}}\ v2c\_\#$$

$$\frac{\Gamma, (u, v, v', x); \Delta \vdash G_1\ \mathsf{VARS\text{-}TO\text{-}CLIQUE} \quad \begin{array}{l}\Gamma, (u, v, v', x); \Delta \vdash \mathsf{CONNECTX}\ v\ G_2 \\ \Gamma, (u, v, v', x); \Delta \vdash \mathsf{CONNECTX}\ v'\ G_3 \\ \Gamma, (u, v, v', x); \Delta \vdash \mathsf{CONNECTV}\ x\ G_4\end{array}}{\Gamma, (u, v, v', x); \Delta, u \vdash (G_1 \cup G_2 \cup G_3 \cup G_4)\ \mathsf{VARS\text{-}TO\text{-}CLIQUE}}\ v2c\_v$$

$$\frac{}{\Gamma; \cdot \vdash \mathsf{CONNECTV}\ X\ \#}\ connectV\_\#$$

$$\frac{\Gamma, (u, v, v', x'); \Delta \vdash \mathsf{CONNECTV}\ X\ G}{\Gamma, (u, v, v', x'); \Delta, u \vdash \mathsf{CONNECTV}\ X\ \mathsf{newe}\ e : (X, v)\ e' : (X, v').G}\ connectV\_v$$

$$\frac{}{\Gamma; \cdot \vdash \mathsf{CONNECTX}\ X\ \#}\ connectX\_\#$$

$$\frac{\Gamma, (u, v, v', x'); \Delta \vdash \mathsf{CONNECTX}\ X\ G}{\Gamma, (u, v, v', x'); \Delta, u \vdash \mathsf{CONNECTX}\ X\ \mathsf{newe}\ e : (X, x').G}\ connectX\_v$$

Figure 5: Linear LF representation of 3-SAT CHROMATIC reduction.

continuations and their usage in compilation of expressions in considerable detail.

$$\text{Continuations} \quad K \quad ::= \quad * \mid K; f$$

Thus, these inference rules allow us to build a valid deduction for a judgment $\Gamma; \Delta \vdash K \diamond F \Rightarrow_C C', G$ if and only if the conversion algorithm given above converts the Boolean formula represented by combining the clauses in F and K to the graph G; C should always be more than the sum of the free Boolean variables in F and K; and C' is the total number of Boolean variables in F. Since C is updated every time we see a new variable, the invariant given in the lemma below always holds.

Lemma 3.1 (Invariant of the Reduction) *Given any continuation K, Boolean formula F, graph G and colors C, C': If $\mathcal{D} :: \Gamma \vdash K \diamond F \Rightarrow_C C', G$ then $C \leq C'$.*

Proof: A straightforward induction. The theorem is denoted by the LF type family `lemma3.1` and the encoding of the proof is given in Figure 19 and Appendix D. Note that `lemma3.1` is renamed as `conv_invariant` in Appendix D. □

The edges in step (a) are added immediately when we encounter a new variable in rule *c_new*, the edges in step (b) are added through the inference rules associated with judgment $\Gamma; \Delta \vdash G$ CLIQUE and the edges in step (c) are added through the inference rules associated with $\Gamma; \Delta \vdash G$ VARS-TO-CLIQUE.

In step (d), we create a vertex corresponding to every clause and add edges connecting the clause to vertices corresponding to literals not in the clause. These edges are added through the inference rules associated with $\Gamma; \Delta \vdash K; F \Rightarrow G$. We are only considering clauses with three literals and hence there are 40 different kinds of clauses: each of the 3 literals in a clause can have a variable appearing as itself or as its complement, giving us 8 choices and each clause can have up to 3 distinct variables, giving us 5 choices[1]. For the sake of conciseness we give only one representative rule *c"5.1* in Figure 5, the other 39 rules are given in Appendix A.

The predicate CONNECTX adds an edge between its first argument and every vertex among the resource in Δ. We note that once we access a vertex in Δ, it is automatically consumed (see for example rules *c"5.1*, *c'_;*,

[1]When variables appear only positively in each of the 3 literals, the 5 choices are: $(\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1)$, $(\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2)$, $(\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1)$, $(\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1)$, $(\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_3)$

clique_v, *v2c_v*, *connectV_v*, and *connectX_v*). Thus, Δ's properties are best described as those of the linear context in the sense of linear logic [Gir87].

Cliques are built recursively, using Δ as a structure over which to iterate. Every vertex in Δ is connected through an edge to every other vertex in that context defining a clique (see rule *clique_v*).

If a Boolean formula F has n variables, 0 free variables and m clauses, then the number of inference rules used in the derivation $\cdot;\cdot \vdash *\diamond F \Rightarrow_{\mathsf{Z}} C, G$ are $m + n + 1$ (each new variable corresponds to the inference rule *c_new*, each clause corresponds to the inference rule *conv*$\wedge$ and there is one base case). Further, the deductions for the judgments $\Gamma;\Delta \vdash K \Rightarrow G_1$, $\Gamma;\Delta \vdash G_2$ CLIQUE, and $\Gamma;\Delta \vdash G_3$ VARS-TO-CLIQUE have height $O(n)$. Hence, the total number of inference rules used in the derivation of the reduction is $O(m + n)$. Thus, the proposed reduction algorithm is in P. Moreover, this formulation renders it directly amenable to being implemented in a logical framework which we discuss next.

4 The Linear Logical Framework

A logical framework is a meta-language that serves the representation of deductive system defined in terms of judgments and inference rules in a type theory. Several frameworks are available; we mention only few such as Isabelle [Pau94], Lego [LP92], LF [HHP93] or LLF [CP96]. For a good overview about logical framework research, consult [Pfe99]. In pursuit of the overall goal of this work, the design of a digital library for complexity theoretic problems, special attention must be paid to the simplicity with which complexity theoretic problems are to be formulated by the user. Therefore, for this particular case study, our choice has fallen on LLF, mostly because the representation of Γ and Δ behave just like the intuitionistic and the linear context provided by the framework. And indeed, the entire development so far through Sections 2 and 3, and the correctness proof to be discussed below, has been implemented and type checked in a prototype implementation for LLF. It is given in Appendix B–E.

LLF is a conservative extension over LF and incorporates three connectives from linear logic, namely *multiplicative implication* ($\multimap$), *additive conjunction* ($\&$) and *additive truth* ($\top$). It is a two zone system that explicitly distinguishes between *intuitionistic* assumptions (that play the role of Γ), and *linear* assumptions (that play the role of resources Δ). In a derivation, linear assumptions can be used exactly once. The linear fragment of LLF is given in Figure 6. Note that the *additive conjunction* ($\&$) allows the

use of the same set of linear assumptions for proving both the conjuncts and *multiplicative implication* ($\multimap$) puts a linear assumption into the linear context. In the case of linear application, the argument may be reused several times and can therefore not refer to any linear assumptions. The rule *Iax* expresses that an intuitionistic assumption can only be used if no linear assumptions are present as opposed to the *Lax* rule that consumes one single linear assumption.

$$\frac{}{\Gamma, u : A; \cdot \vdash u : A}\ Iax \qquad \frac{}{\Gamma; \Delta, u : A \vdash u : A}\ Lax \qquad \frac{}{\Gamma; \Delta \vdash \langle\rangle : \top}\ Top$$

$$\frac{\Gamma; \Delta \vdash M_1 : A \quad \Gamma; \Delta \vdash M_2 : B}{\Gamma; \Delta \vdash \langle M_1, M_2\rangle : A \& B}\ \&I \qquad \frac{\Gamma; \Delta \vdash M : A \& B}{\Gamma; \Delta \vdash \mathsf{FST} M : A}\ \&E_1 \qquad \frac{\Gamma; \Delta \vdash M : A \& B}{\Gamma; \Delta \vdash \mathsf{SND} M : B}\ \&E_1$$

$$\frac{\Gamma; \Delta, x : A \vdash M : B}{\Gamma; \Delta \vdash \hat{\lambda} x : A.M : A \multimap B} \multimap I \qquad \frac{\Gamma; \Delta \vdash M_1 : B \multimap A \qquad \Gamma; \cdot \vdash M_2 : B}{\Gamma; \Delta \vdash M_1 \hat{} M_2 : A} \multimap E$$

Figure 6: A fragment of LLF.

LLF supports the *judgments-as-types* methodology for representation and incorporates the aforementioned linear connectives as type constructors. In addition, each rule is endowed with proof objects that correspond to the introduction and elimination forms as shown below.

$$\begin{array}{lll} \textit{Objects } M ::= & & \\ \quad \hat{\lambda} x : A.M \mid M_1 \hat{} M_2 & & \text{(Linear functions)} \\ \mid \quad \langle M_1, M_2\rangle \mid \mathsf{FST}\ M \mid \mathsf{SND}\ M & & \text{(Additive conjunction)} \\ \mid \quad \langle\rangle & & \text{(Additive unit)} \end{array}$$

Thus, a linear LF representation of the judgment $\Gamma; \Delta \vdash J$ is $\ulcorner \Gamma \urcorner \to \ulcorner \Delta \urcorner \multimap \ulcorner J \urcorner$ where $\ulcorner \Gamma \urcorner$, $\ulcorner \Delta \urcorner$ and $\ulcorner J \urcorner$ are linear LF representations of Γ, Δ and J respectively. And finding a proof is equivalent to finding an object of the corresponding type generated from the language of linear LF objects augmented as above.

Although LLF supports quite elegant representations of Boolean formulas and graphs given in Figure 1, there is something unsatisfying about the way the reduction is laid out in Figure 5. The linear context is used as an auxiliary concept for computing cliques, and to connect a vertex to all remaining vertices in a graph. Additive conjunction is used to pass this auxiliary information to the other relevant judgments. The graph isomorphism problem is not relevant for this particular case study, but might be in the general case, as are other operations such as the intersection of two graphs, or the expansion of a node in graph by another graph. To decide it in LLF

```
v  : type.    /\ : o -> o -> o.
b  : type.    \/ : o -> o -> o.
o  : type.    pos: v -> o.
              neg: v -> o.
              new: (v -> o) -> o.
```

Figure 7: Encoding of Boolean formulas

```
vertex : type.
edge   : vertex -> vertex -> type.
graph  : type.
newv   : (vertex -> graph) -> graph.
newe   : (edge A B -> graph) -> graph.
+      : graph -> graph -> graph.
```

Figure 8: Encoding of graphs

would require the user to encode it explicitly — a complicated and expensive operation that is prone for error and difficult to reason about. We can only speculate if the graph isomorphism problem can be directly incorporated into a logical framework. Our graph representation stores all vertices and edges in a graph together with the superfluous history in which order vertices and edges were inserted. Thus, adequacy of the representation is guaranteed.

The linear Twelf code for all inference rules shown so far can be directly derived from the *judgments-as-types* methodology underlying LLF. We represent Boolean variables, Boolean values, formulas and vertices by types `v`, `b`, `o` and `vertex` respectively. Colors are represented by the type family of natural numbers `nat`. We show the implementation of Boolean formulas in Figure 7 and graphs in Figure 8.

Let A_1 and A_2 be LLF types, and M an LLF object. In concrete syntax, we write $\{\texttt{x:}A_1\}A_2$ for the dependent type $\Pi x : A_1.A_2$, and $\texttt{[x:}A_1\texttt{]}M$ for the function $\lambda x : A_1.M$. As usual, where convenient we omit the leading block of Π quantifiers from constant declarations to improve readability. Furthermore we write A_1 `-o` A_2 for the linear function type $A_1 \multimap A_2$, and A_1 `&` A_2 for the additive conjunction $A_1 \& A_2$. Furthermore, we write `<T>` for $\top$, and `<>` for $\langle\rangle$.

4.1 Representation of 3-SAT and CHROMATIC

The Twelf code for the inference rules that encode 3SAT (Figure 2) is given in Figure 9 and for CHROMATIC (Figure 3) is given in Figure 11. Continuations are represented as objects of type `cont`. We write `*` for the initial continuation and $K\,;F$ for a continuation stack with F being the top element.

```
* : cont.
; : cont -> o -> cont.
```

The notion of satisfiability generalizes to continuations, and is given in Figure 10.

Each of the two problems is hypothetical in nature, 3SAT for example relies on correctly selecting a truth assignment for rules *satt* and *satf* and CHROMATIC on correctly assigning a color to a vertex in rules *cgvertex* and *cgedge*. In LLF, the encoding of these hypotheses gives rise to new two type families `hyp` and `colorvertex`, respectively.

4.2 Representation of the Reduction

The 3-SAT CHROMATIC reduction (Figure 5) is given in Figure 17 and encoded as a relation over Boolean formulas F, colors C and C', continuation K and graph G. It is implemented by the type family

```
conv : o -> nat -> nat -> cont -> graph -> type.
```

The assumptions (u, v, v', x) relating a Boolean variable u with graph vertices v, v' and x are implemented by the type family

```
relate : u -> vertex -> vertex -> vertex -> type.
```

Recall, that our reduction is based on the construction of cliques, which we encode in LLF in terms of a type families `clique` shown in Figure 12 and `vars2clique` in Figure 13. Note in these two figures, how the harmless looking `&` in `clique_v` is responsible for duplicating the context Δ in rule *clique_v* in Figure 5. In fact, all assumptions in Δ are treated as resources (to control iteration), and hence exclusive represented within the linear context by assumptions of type family `var : vertex -> type`.

The three auxiliary judgments $\Gamma, \Delta \vdash K \Rightarrow G$; $\Gamma, \Delta \vdash F \Rightarrow G$; and $\Gamma, \Delta \vdash C \downarrow G$ are given in Figures 14-16, respectively, leading up to the encoding of the reduction `conv` that is given in Figure 17.

```
hyp    : v -> b -> type.
sat    : o -> type.
store_var : {v:v}{b:b} hyp v b -> true.
satp   : sat (pos A)
          <- hyp A true.
satn   : sat (neg A)
          <- hyp A false.
sat/\  : sat (F1 /\ F2)
          <- sat F1
          <- sat F2.
sat\/1 : sat (F1 \/ F2)
          <- sat F1.
sat\/2 : sat (F1 \/ F2)
          <- sat F2.
satnewt: sat (new F)
          <- ({v:v} hyp v true
              -> sat (F v)).
satnewf: sat (new F)
          <- ({v:v} hyp v false
              -> sat (F v)).
```

Figure 9: Encoding of satisfiability.

5 Correctness

In this section, we show that the conversion from 3-SAT to CHROMATIC maps all "Yes" instances of 3-SAT to "Yes" instances of CHROMATIC and vice-versa. This section presents a sequence of lemmas cumulating in Theorem 5.7. All lemmas and theorems presented in this Section have been encoded as relations in LLF (see Appendix E and type checked in the prototype implementation of linear Twelf. Although type checking is not enough to guarantee that the source code constitutes a proof, we have convinced ourselves that all cases are covered and our induction principles are sound.

With the deductive description of the reduction algorithm from Figure 5, we first establish the invariant property about the assumptions in Γ with respect to the context η as used in Figure 2 and 3 (Definition 5.1). First, it relates the truth value of each Boolean variable in 3-SAT with the color assignment to each vertex in CHROMATIC and second it guarantees that all vertices in the context are assigned distinct colors. That the intuition-

```
satK  : cont -> type.
satK* : satK *.
satK; : satK (K ; F)
         <- satK K
         <- sat F.
```

Figure 10: Encoding of continuation satisfiability.

istic LLF context as image of Γ satisfies the second part at all times is a global property of the context and can currently only be enforced by manual inspection.

Definition 5.1 (Valid environments) *Given a set of Boolean variables $u_1, u_2, \ldots, u_n$ and sets of vertices $v_1, v_2, \ldots, v_n$; $v'_1, v'_2, \ldots, v'_n$ and $x_1, x_2, \ldots, x_n$: Let $\Gamma = (u_1, v_1, v'_1, x_1), (u_2, v_2, v'_2, x_2), \ldots, (u_n, v_n, v'_n, x_n)$ and $\eta = u_1 \to B_1, u_2 \to B_2, \ldots, u_n \to B_n$ where $B_i \in \{\mathsf{true}, \mathsf{false}\}$ and C be any color. An environment η' is said to be valid under environments Γ and η and color C if*

$$\eta' = \begin{cases} x_1 \to C_1, x_2 \to C_2, \ldots, x_n \to C_n \\ v_1 \to C'_1, v_2 \to C'_2, \ldots, v_n \to C'_n \\ v'_1 \to C''_1, v_2 \to C''_2, \ldots, v'_n \to C''_n \end{cases}$$

C_i's are distinct, $C_i \leq C$ and

$$C'_i = \begin{cases} C_i & \text{if } B_i = \mathsf{true} \\ \mathsf{c}_0 & \text{if } B_i = \mathsf{false} \end{cases} \qquad C''_i = \begin{cases} \mathsf{c}_0 & \text{if } B_i = \mathsf{true} \\ C_i & \text{if } B_i = \mathsf{false} \end{cases}$$

We begin now with the presentation of the individual lemmas. First, we describe an infrastructure lemma about properties of colors. If integers were available as constraint domain in linear Twelf as they are in the non-linear version of Twelf (which they are not), colors could have been encoded directly as integers rendering this lemma unnecessary.

Lemma 5.2 (Properties of colors) *Let C and C' be colors. The following properties hold: (1) If $C < C'$ then $C < C' + 1$. (2) If $C' \leq C$ then $C' \leq C + 1$. (3) If $C = C'$ then $C < C' + 1$. (4) $C \leq C$. (5) $C < C + 1$. (6) If $C < C'$ then $C \neq C'$. (7) If $C < C'$ then $C \leq C'$. (8) If $C < C'$ then $C < C' + 1$.*

The proofs are straightforward and hence omitted. In subsequent proofs, we will refer to these properties, and give therefore their encodings as type

```
colorvertex : vertex -> nat -> type.
coloring  : nat -> graph -> type.
store_color : {V:vertex}{C:nat}colorvertex V C -> type.
cg#     : coloring N #.
cgvertex : coloring C (newv [v] (G v))
            <- (C <= C')
            <- ({v:vertex} colorvertex v C'
                -> coloring C (G v)).
cgedge   : coloring C (newe [e: edge A B] G)
            <- colorvertex A C1
            <- colorvertex B C2
            <- C1 != C2
            <- C1 <= C
            <- C2 <= C
            <- coloring C G.
cgunion  : coloring C (G1 + G2)
            <- coloring C G1
            <- coloring C G2.
```

Figure 11: Encoding of coloring.

families in LLF (see Appendix C. In the Appendix, `lemma5.2(1)` is called `lemma1` and so on.).

```
lemma5.2(1) : C < C' -> C < (s C') -> type.
lemma5.2(2) : C <= C' -> C <= (s C') -> type.
lemma5.2(3) : C == C' ->  C < (s C') -> type.
lemma5.2(4) : C <= C -> type.
lemma5.2(5) : C < (s C) -> type.
lemma5.2(6) : C < C' -> C != C' -> type.
lemma5.2(7) : C < C' -> C <= C' -> type.
lemma5.2(8) : C < C' -> C < (s C') -> type.
```

Next, we prove the basic property of graph coloring that is a graph that is colorable with C colors is also colorable with $C+1$ colors.

Theorem 5.3 (Coloring preservation) *Let G be a graph and C a color. If $\mathcal{D} :: \eta \vdash G\ C$ COLORING then there exists $\mathcal{D}' :: \eta \vdash G\ (C+1)$ COLORING.*

Proof: The proof proceeds by induction over the height of derivation $\mathcal{D}$. We have four cases depending on whether the derivation $\mathcal{D}$ ends in *cg#*,

```
connectX   : vertex -> graph -> type.
connectX_# : connectX X #.
connectX_v : (var U -o
               connectX X (newe [e:edge X X'] G))
              <- relate U _ _ X'
              <- connectX X G.
clique   : graph -> type.
clique_# : clique #.
clique_v : (var U -o clique (G + G'))
                   <- clique G & connectX X G'
                   <- relate U _ _ X.
```

Figure 12: Encoding of CLIQUE.

cgvertex, *cgedge* or *cgunion* (see figure 3); the case *cg#* is the base case. The proof follows naturally using Lemma 5.2 (2). □

In LLF, this proof is represented as a relation over $\mathcal{D}$ and $\mathcal{D}'$, i.e. $\ulcorner\mathcal{D}\urcorner$ `:coloring C G` and $\ulcorner\mathcal{D}'\urcorner$ `:coloring (s C) G`. This relation is implemented as a type family

```
theorem5.3: coloring C G -> coloring (s C) G -> type
```

where each of the cases of the proof is represented as a declaration as given in Figure 18[2]. The declarations of `inc_v`, `inc_e` and `inc_+` correspond to the cases when the derivation $\mathcal{D}$ ends in *cgvertex*, *cgedge* and *cgunion* respectively.

The lemmas given below prove that the edges added in the steps (b), (c) and (d) of the conversion algorithm do not connect vertices assigned the same color.

Lemma 5.4 (Clique Coloring) *Let G be a graph; $x_1, x_2, \ldots, x_n$ free variables in G and $u_1, u_2, \ldots, u_n$ Boolean variables. Let $\Delta = u_1, u_2, \ldots, u_n$ and*

$$\Gamma = (u_1, \_, \_, x_1), (u_2, \_, \_, x_2), \ldots, (u_n, \_, \_, x_n)^3.$$

If $\mathcal{D} :: \Gamma; \Delta \vdash G$ CLIQUE and $C \geq \mathsf{c}_0 + n$ then there exists $\mathcal{G} :: \eta \vdash G\ C$ COLORING where $\eta = x_1 \to C_1, x_2 \to C_2, \ldots, x_n \to C_n$; C_i's are distinct and $C_i \leq C$.

[2] In Appendix C, the type family `theorem5.3` is renamed as `increase_color`

[3] We denote the variables whose values we do not care by the wild-card _.

```
connectV : vertex -> graph -> type.
connectV_# : connectV X #.
connectV_v : (var U -o
                connectV X (newe [e:edge V X]
                            newe [e':edge V' X] G))
                <- relate U V V' _
                <- connectV X G.
vars2clique : graph -> type.
v2c_# : vars2clique #.
v2c_v : (var U -o vars2clique (G1 + G2 + G3 + G4))
          <- vars2clique G1
             & connectX V G2
             & connectX V' G3
             & connectV X G4
          <- relate U V V' X.
```

Figure 13: Encoding of VARS-TO-CLIQUE.

```
conv''' : vertex -> graph -> type.
c'''_# : conv''' C #.
c'''_v : conv''' C (newe [e:edge C V]
                         newe [e':edge C V'] G)
                o- var U
                <- conv''' C G
                <- relate U V V' _.
```

Figure 14: Encoding of the $\Gamma, \Delta \vdash K \Rightarrow G$.

Proof: Since the inference rules for CLIQUE only connect the different x_i's and these vertices are colored with distinct colors, the proof follows by induction. The proof is implemented by the type family `lemma5.4`(renamed as `clique_color` in Appendix E) and an encoding of the individual cases is given in Appendix E. □

This lemma and its proof are encoded in LLF as type family

```
lemma5.4: {C:nat} clique G -> coloring C G -> type.
```

Lemma 5.5 *Let G be a graph. Let the free variables in G be denoted by $v_1, v_2, \ldots, v_n$; $v'_1, v'_2, \ldots, v'_n$ and $x_1, x_2, \ldots, x_n$, and $u_1, u_2, \ldots, u_n$ be Boolean*

```
conv'' : o -> graph -> type.
c''51 : conv'' ((pos U1) \/ (pos U2) \/ (pos U3))
            (newv [c]
             newe [e1:edge c V1']
             newe [e2:edge c V2']
             newe [e3:edge c V3'] (G c)))
          o- var U3 o- var U2 o- var U1
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2'_
          <- relate U3 V3 V3' _
          <- ({c:vertex} conv''' c (G c)).
```

Figure 15: Encoding of $\Gamma, \Delta \vdash F \Rightarrow G$ (case 5 of 40).

```
conv' : cont -> graph -> type.
c'_*  : conv' * # o- <T>.
c'_;  : conv' (K ; F) (G1 + G2)
          <- conv' K G1 & conv'' F G2.
```

Figure 16: Encoding of the $\Gamma, \Delta \vdash C \downarrow G$.

variables. Furthermore, let $\Delta = u_1, u_2, \ldots, u_n$*;*

$$\Gamma = (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), \ldots, (u_n, v_n, v_n', x_n)$$

and $\eta = u_1 \rightarrow B_1, u_2 \rightarrow B_2, \ldots, u_n \rightarrow B_n$ *where* $B_i \in \{\mathsf{true}, \mathsf{false}\}$.

If $\mathcal{D} :: \Gamma; \Delta \vdash G$ VARS-TO-CLIQUE *and* $C \geq \mathsf{c}_0 + n$ *then there exists* $\mathcal{G} :: \eta' \vdash G\ C$ COLORING *where* η' *is valid under environments* Γ*,* η *and color* C*.*

Proof: The inference rules for VARS-TO-CLIQUE only connect v_i to x_j and v_i' to x_j if $i \neq j$. These pairs of vertices are never assigned same colors if η' is a valid environment. The proof follows again by induction. The proof is implemented by the type family `lemma5.5` (renamed as `var2clique_color` in Appendix E) and an encoding of the individual cases is given in Appendix E. □

And again, this lemma and its proof has been implemented in LLF:

```
lemma5.5: {C:nat} vars2clique G
                    -> coloring C G -> type.
```

```
conv : o -> nat -> nat -> cont -> graph -> type.
c_new: conv (new F) C C' K
              (newv [v] newv [v'] newv [x]
                newe [e:edge v v'] (G v v' x))
         <- ({u:v}{v:vertex}{v':vertex}{x:vertex}
             relate u v v' x -> var u
             -o conv (F u) (s C) C' K (G v v' x)).
c_/\ : conv (F /\ F') C C' K G
          <- conv F' C C' (K ; F) G.
c_\/ : conv (F1 \/ F2 \/ F3) C C K (G1 + G2 + G3)
         <- conv' (K ; (F1 \/ F2 \/ F3)) G1
            & clique G2
            & vars2clique G3.
```

Figure 17: Encoding of the reduction

Lemma 5.6 (Coloring graph) *Under the same assumptions as given in Lemma 5.5 the following holds. Let K be a continuation and $u_1, u_2, \ldots, u_n$ be the free variables in K. If $\mathcal{D} :: \Gamma; \Delta \vdash K \Rightarrow G$ and $C \geq \mathsf{c}_0 + n$ then*

$$\mathcal{E} :: \eta \vdash K\ SAT \ \textit{ iff } \ \mathcal{G} :: \eta' \vdash G\ C\ \mathsf{COLORING}$$

where η' is valid under environments Γ, η and color C.

Proof: If the truth assignment in η satisfies all the clauses in K, then we color every clause vertex c_j with the color assigned to the literal which satisfies that clause, i.e. if the literal u_i appears in c_j and is true then we assign c_j with color of v_i (as defined in η') and if the literal $\bar{u}_i$ appears in c_i and is false then we assign c_j with color of v'_i (as defined in η'). In other words, we color each c_j with a true color. Since we have n distinct true colors and c_j is connected to vertices corresponding to literals not in c_j, we do not add any edges between vertices with same color. The other direction is similar. The theorem is implemented by the type family `lemma5.6` (renamed as `conv'_color` in Appendix E). The formal proof is by induction given in Appendix E. It uses auxiliary theorems which are implemented by the type families `conv''_color` and `conv'''_color`. □

The proof of Lemma 5.6 is encoded in LLF as type family

```
lemma5.6: {C:nat} conv' K G -> satK K
             -> coloring C G -> type.
```

```
inc_v : theorem5.3 (cgvertex CG E) (cgvertex CG' E')
          <- ({v:vertex}{c:colorvertex v C'}
                store_color v C' c ->
                theorem5.3 (CG v c) (CG' v c))
          <- lemma5.2(2) E E'.
inc_e : theorem5.3 (cgedge CG E2 E1 E3 C1 C2)
           (cgedge CG' E2' E1' E3 C1 C2)
          <- theorem5.3 CG CG'
          <- lemma5.2(2) E2 E2'
          <- lemma5.2(2) E1 E1'.
inc_+ : theorem5.3 (cgunion CG1 CG2)
           (cgunion CG1' CG2')
          <- theorem5.3 CG1 CG1'
          <- theorem5.3 CG2 CG2'.
```

Figure 18: Proof of Theorem 5.3.

The final step of the correctness proof relies on explicit appeals to the representation invariant as stated in Lemma 3.1. It is implemented as type family

```
lemma3.1: conv F C C' K G -> (C <= C') -> type.
```

An encoding of its proof is also given in Figure 19.

Theorem 5.7 (Main Theorem) *Under the same assumptions from Lemma 5.5 the following holds. Let F be a Boolean formula, K be a continuation and $u_1, u_2, \ldots, u_n$ be the free variables in F and K. If $\mathcal{D} :: \Gamma; \Delta \vdash K \diamond F \Rightarrow_C C', G$ and $C \geq \mathsf{c}_0 + n$ then*

$$\mathcal{E} :: \eta \vdash F\ \mathsf{SAT}, \mathcal{F} :: \eta \vdash K\ \mathsf{SAT}$$
$$\textit{iff}\ \ \mathcal{G} :: \eta' \vdash G\ C'\ \mathsf{COLORING}$$

where η' is valid under environments Γ and η and color C'[4].

Proof: (**Sketch**) We note that C' is at least n, so we always have n distinct colors. The proof follows by induction on the height of the derivation $\mathcal{D}$ using lemmas 5.4, 5.5 and 5.6. □

[4]Given a conjunct F (or continuation K), we choose the satisfiability proof $\mathcal{D} :: \Gamma \vdash F\ \mathsf{SAT}$ which selects the leftmost true literal with least indexed variable. For example, if $F = (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_3)$ and $\Gamma = u_1 \to \mathsf{false}, u_2 \to \mathsf{false}, u_3 \to \mathsf{true}$, then $\mathcal{D}$ consists of inference rules $sat\vee 2$ and $sat\vee 1$, i.e. the literal $(\mathsf{neg}\ u_2)$ is selected.

```
<= : nat -> nat -> type.
<=s : s M <= s N <- M <= N.
convinvr_base : lemma3.1 (c_\/ D) E
                 <- lemma4 E.
convinvr_/\ : lemma3.1 (c_/\ D) E
                <- lemma3.1 D E.
convinvr_new : lemma3.1 (c_new D) E
                <- ({u:v}{v:vertex}{v':vertex}{x:vertex}
                     {r:relate u v v' x}{var: var u}
                     lemma3.1 (D u v v' x r ^ var) (<=s E'))
                <- lemma5.2(2) E' E.
```

Figure 19: Enconding of the proof of the invariant given in Lemma 3.1 of the 3-SAT CHROMATIC Reduction.

The theorem is represented in LLF by a type family (renamed as `main_theorem` in Appendix E):

```
theorem5.7 : {C:nat} conv F C C' K G -> sat F
             -> satK K -> coloring C' G -> type.
```

It encodes the relation between color C, representations of the derivations $\mathcal{D}$, $\mathcal{E}$, $\mathcal{F}$ and $\mathcal{G}$, i.e. $\ulcorner\mathcal{D}\urcorner$ `: conv F C C' K G`, $\ulcorner\mathcal{E}\urcorner$ `: sat F`, $\ulcorner\mathcal{F}\urcorner$ `: satK K`, $\ulcorner\mathcal{G}\urcorner$ `: coloring C G`. Figure 20 gives the three representative cases of the proof (See Appendix E for the complete proof). The *if* and *only if* directions of the proof are verified by the Twelf mode checker with the following two mode specifications for the *if* and *only if* cases respectively:

```
%mode main_theorem +C +CV +CF +CK -CG.
%mode main_theorem +C +CV -CF -CK +CG.
```

We write `!= : nat -> nat -> type` for inequality on natural numbers, and `!=z : z != s N` for the proof that $0 \neq N + 1$ if N is positive, and `<= : nat -> nat -> type` for the $\leq$ relation on natural numbers, and `<=z : z <= s N` for that proof that $0 \leq N + 1$ for all natural numbers N.

In the case of `case1f`, the vertices `v'` and `x` are assigned a new unique `true` color `(s C)` and the vertex `v` is assigned the `false` color `z` and we prove the theorem recursively. The other two cases are a direct translation of the proof.

```
case1f : theorem5.7 C (c_new D) (satnewf E) F
         (cgvertex ([v] [cgv] (cgvertex ([v'][cgv']
           (cgvertex ([x][cgx]
             (cgedge (CG v v' x cgv cgv' cgx)
              H <=z !=z cgv' cgv)) H)) H)) <=z)
         <- ({u:v}{hypf: hyp u false}{v:vertex}
             {v':vertex}{x:vertex}
             {r:relate u v v' x}
             {var: var u}{cgv :colorvertex v z}
             {cgv':colorvertex v' (s C)}
             {cgx :colorvertex x (s C)}
             store_color v z cgv -> store_color v' (s C) cgv' ->
             store_color x (s C) cgx -> store_var u false hypf ->
             theorem5.7 (s C) (D u v v' x r ^ var)
               (E u hypf) F
               (CG v v' x cgv cgv' cgx))
         <- ({u:v}{v:vertex}{v':vertex}{x:vertex}
             {r:relate u v v' x}{var: var u}
             lemma3.1 (D u v v' x r ^ var) H).
case2 : theorem5.7 C (c_/\ D) (sat/\ E1 E2) F CG
         <- theorem5.7 C D E2 (satK; E1 F) CG.
case3 : theorem5.7 C (c_\/ (D1 , D2 , D3)) E F
         (cgunion CG1 (cgunion CG2 CG3))
         <- lemma5.6 C D1 (satK; E F) CG1
         <- lemma5.4 C D2 CG2
         <- lemma5.5 C D3 CG3.
```

Figure 20: Proof of Theorem 5.7

6 Evaluation

This case study has highlighted several advantages of using LF and a few future challenges that we face to be able to formalize NP-complete problems and their reductions successfully in a proof assistant based on a logical framework. Graphs play a prevalent role in complexity theory, and we evaluate thus our work with respect to the adequacy of their encoding, available operations, the expressibility of reductions, and the verification of the meta-theory.

Adequacy. The main challenge in representing graphs as objects lies in capturing isomorphisms in between graphs in type theory. Neither LF nor LLF provide a notion of definitional equality that is compatible with graph isomorphisms. The solution adopted in this paper is to consider two graphs equivalent if and only if the order in which vertices and edges were introduced into the graph coincides. Consequently, a single *graph* can correspond to several distinct but equivalent *graph representations*.

Operations. Many standard operations that are usually performed on a graph like addition or deletion of a vertex, iteration over subgraphs, edges and vertices are necessary to express basic reductions. Our solution towards incorporating iteration is based on linear constraints [CP96], where the linear context enforces complete traversal over the set of edges or vertices. Ideally, however, operations of this kind should be directly supported by the underlying logical framework.

Reductions. Any valid reduction between two NP complete problems must be a polynomial time reduction. The reduction of a Boolean formula F to a graph G is denoted by the derivation $\mathcal{D} :: \Gamma; \Delta \vdash K \diamond F \Rightarrow_C C', G$. Thus, given an appropriate notion of *size* of $\mathcal{D}$ and F, it is possible to argue that the reduction is polynomial time if the size of $\mathcal{D}$ is a polynomial in the size of F. Furthermore, since we are choosing a framework with higher-order terms, we need to develop notions of polynomial time when higher-order unification is permitted and these terms are reduced to their canonical forms. What is left to do is to develop polynomial-time checker for LF and LLF signatures, respectively. For this purpose, we believe that the research results of [BC92, Hof99, BNS00] show promise for use in logical frameworks.

Meta-Theory. The nature of this case study exposes a challenge for coverage checking algorithms [SP03] predominantly used in logical frame-

works as well. For example, an environment was defined to be valid only if all colors assigned to vertices are pairwise different. This is a global property that cannot easily be enforced locally as required in the proof of main theorem. Currently, invariants of this kind are not supported neither by the meta-logic for LF [Sch00] nor LLF [MS03]. Consequently, further research is necessary to provide coverage checking and automated theorem proving support for all lemmas and theorems contained in this paper.

7 Conclusion and Future Work

While, a lot of research has been undertaken to develop and implement logics dedicated to the formalization of mathematics [Con86, dB68, dB80, Nup03, Coq03, Pfe01b], not much attention has been paid to integrating logical framework technology as an enabling technology into proof assistants and automated theorem provers. We believe our case study to be a pioneering attempt to evaluate the potential of modern logical frameworks for representing and formalizing results in the general area of complexity theory, and the analysis of NP-complete problems, in particular.

In this paper, we have given a deductive account for a reduction from 3-SAT, the problem of deciding the satisfiability of a formulas in conjunctive normal form with clauses containing exactly three literals, into CHROMATIC, the problem of deciding if a graph has a given chromatic number. All inference systems, and all proofs presented in this paper have been formalized in the linear logical framework LLF and implemented in linear Twelf. We believe this to be the first attempt to employ logical framework technology for representing and reasoning with NP-complete problems. Eventually we plan to develop a digital library of NP-complete problems, that contains problems domains, reductions, and correctness proofs, and a powerful query language to retrieve them.

Acknowledgments We would like to thank Dana Angluin for raising our interest in studying the relationship between logical frameworks and complexity theory and the many discussions related to the subject.

References

[BC92] S. Bellantoni and S. Cook. A new recursion-theoretic characterization of the poly-time functions. *Computational Complexity*, pages

97–110, 1992.

[BNS00] Stephen J. Bellantoni, Karl-Heinz Niggl, and Helmut Schwichtenberg. Higher type recursion, ramification and polynomial time. *Annals of Pure and Applied Logic*, pages 17–30, 2000.

[Con86] R. L. Constable, et al. *Implementing Mathematics with the Nuprl Proof Development*. Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[Coq03] Coq, project home page. version 7.4. World Wide Web, `http://coq.inria.fr`, 2003.

[CP96] Iliano Cervesato and Frank Pfenning. A linear logical framework. In E. Clarke, editor, *Proceedings of the Eleventh Annual Symposium on Logic in Computer Science — LICS'96*, pages 264–275. IEEE Computer Society Press, 27–30 July 1996.

[dB68] N.G. de Bruijn. The mathematical language AUTOMATH, its usage, and some of its extensions. In M. Laudet, editor, *Proceedings of the Symposium on Automatic Demonstration*, pages 29–61, Versailles, France, December 1968. Springer-Verlag LNM 125.

[dB80] N.G. de Bruijn. A survey of the project AUTOMATH. In J.P. Seldin and J.R. Hindley, editors, *To H.B. Curry: Essays in Combinatory Logic, Lambda Calculus and Formalism*, pages 579–606. Academic Press, 1980.

[Gir87] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Co., 1979.

[HHP93] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the Association for Computing Machinery*, 40(1):143–184, January 1993.

[Hof99] Martin Hofmann. Linear types and non-size-increasing polynomial time computation. In *Logic in Computer Science*, pages 464–473, 1999.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of computer computations*, pages 85–103. Plenum Press, New York, NY, 1972.

[Lew] F. D. Lewis. *Solving NP-Complete Problems*. World Wide Web, http://cs.engr.uky.edu/ lewis/cs-heuristic/text/contents.html. This is web version of a work in progress.

[LP92] Zhaohui Luo and Robert Pollack. The LEGO proof development system: A user's manual. Technical Report ECS-LFCS-92-211, University of Edinburgh, May 1992.

[MS03] Andrew McCreight and Carsten Schürmann. A meta linear logical framework. Draft, 2003.

[Nup03] Nuprl, project home page. version 5. World Wide Web, `http://www.cs.cornell.edu/Info/Projects/NuPrl/nuprl.html`, 2003.

[Pau94] Lawrence C. Paulson. *Isabelle: A Generic Theorem Prover*. Springer-Verlag LNCS 828, 1994.

[Pfe99] Frank Pfenning. Logical frameworks. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers, 1999. In preparation.

[Pfe01a] Frank Pfenning. *Computation and Deduction*. Cambridge University Press, 2001.

[Pfe01b] Frank Pfenning. Logical frameworks. In *Handbook of Automated Reasoning*, pages 1063–1147. Elsevier Science Publishers, 2001.

[Sch00] Carsten Schürmann. *Automating the Meta-Theory of Deductive Systems*. PhD thesis, Carnegie Mellon University, 2000. CMU-CS-00-146.

[SP03] Carsten Schürmann and Frank Pfenning. A coverage algorithm for LF. In *Proceedings of Theorem Proving in Higher-order Logics (TPHOLs)*, Rome, 2003. To appear.

Appendix

A 3SAT-CHROMATIC Reduction in Linear LF

The 40 cases mentioned in Figure 5 are given below.

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1').G}\ c''1.1$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.2$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.3$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.4$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.5$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.6$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.G}\ c''1.7$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1); \Delta, u_1 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1).G}\ c''1.8$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2').G}\ c''2.1$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2).G}\ c''2.2$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''2.3$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''2.4$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''2.5$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''2.6$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2').G}\ c''2.7$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2).G}\ c''2.8$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2').G}\ c''3.1$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''3.2$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2).G}\ c''3.3$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''3.4$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''3.5$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2').G}\ c''3.6$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''3.7$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2).G}\ c''3.8$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2').G}\ c''4.1$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''4.2$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2').G}\ c''4.3$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2').G}\ c''4.4$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2).G}\ c''4.5$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''4.6$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_2 : (c, v_2).G}\ c''4.7$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2); \Delta, u_1, u_2 \vdash (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_1) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2).G}\ c''4.8$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2')\ e_3 : (c, v_3').G}\ c''5.1$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2')\ e_3 : (c, v_3).G}\ c''5.2$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2)\ e_3 : (c, v_3').G}\ c''5.3$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{pos}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1')\ e_2 : (c, v_2)\ e_3 : (c, v_3).G}\ c''5.4$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{pos}\ u_3) \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2')\ e_3 : (c, v_3').G}\ c''5.5$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\begin{array}{c}\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{pos}\ u_2) \vee (\mathsf{neg}\ u_3) \\ \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2')\ e_3 : (c, v_3).G\end{array}}\ c''5.6$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\begin{array}{c}\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{pos}\ u_3) \\ \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2)\ e_3 : (c, v_3').G\end{array}}\ c''5.7$$

$$\frac{\Gamma, (u_1, v_1, v_1', x_1), (u_2, v_2, v_2', x_2), (u_3, v_3, v_3', x_3); \Delta \vdash c \downarrow G}{\begin{array}{c}\Gamma, (u_1, v_1, v_1', x_1), \ldots, (u_3, v_3, v_3', x_3); \Delta, u_1, u_2, u_3 \vdash (\mathsf{neg}\ u_1) \vee (\mathsf{neg}\ u_2) \vee (\mathsf{neg}\ u_3) \\ \Rightarrow \mathsf{newv}\ c.\mathsf{newe}\ e_1 : (c, v_1)\ e_2 : (c, v_2)\ e_3 : (c, v_3).G\end{array}}\ c''5.8$$

B Encoding of 3-SAT

```
% Boolean variables
v : type.                                  % Variables
b : type.                                  % Boolean Domain
true : b.
false : b.

% Boolean Formulas
o : type.
/\ : o -> o -> o.                          %infix right 10 /\.
\/ : o -> o -> o.                          %infix right 10 \/.
pos : v -> o.
neg : v -> o.
new : (v -> o) -> o.

% Encoding 'Yes' instances of 3-SAT
hyp : v -> b -> type.
sat : o -> type.
satp : sat (pos A)
         <- hyp A true.
satn : sat (neg A)
         <- hyp A false.
sat/\: sat (F1 /\ F2)
        <- sat F1
        <- sat F2.
sat\/1: sat (F1 \/ F2)
         <- sat F1.
sat\/2: sat (F1 \/ F2)
         <- sat F2.
satnewt: sat (new F)
          <- ({v:v} hyp v true -> sat (F v)).
satnewf: sat (new F)
```

```
          <- ({v:v} hyp v false -> sat (F v)).

store_var : {v:v}{b:b} hyp v b -> type.
%mode store_var +V -B -H.
```

C Encoding of CHROMATIC

```
% Representation of graphs
vertex : type.
edge   : vertex -> vertex -> type.
graph  : type.
# : graph.                                          % Empty Graph
newv  : (vertex -> graph) -> graph.
newe  : (edge A B -> graph) -> graph.
+ : graph -> graph -> graph.                          %infix right 10 +.

% Colors
nat : type.
z : nat.
s : nat -> nat.
!= : nat -> nat -> type.                            %infix right 10 !=.
!=z1 : z != s N.
!=z2 : s N != z.
!=s : s N1 != s N2 <- N1 != N2.

< : nat -> nat -> type.                             %infix right 10 <.
<z : z < s N.
<s : s N1 < s N2 <- N1 < N2.

== : nat -> nat -> type.                            %infix right 10 ==.
=z : z == z.
=s : s N1 == s N2 <-  N1 == N2.

<= : nat -> nat -> type.                            %infix left 10 <=.
<=z : z <= N.
<=s : s N1 <= s N2 <- N1 <= N2.

% Some theorems about Colors
lemma1 : C < C' -> C < (s C') -> type.
%mode lemma1 +E1 -E2.
lemma1z: lemma1 <z <z.
lemma1s: lemma1 (<s D) (<s D')
  <- lemma1 D D'.

lemma2 : C <= C' -> C <= (s C') -> type.
%mode lemma2 +E1 -E2.
lemma2= : lemma2 <=z <=z.
lemma2< : lemma2 (<=s D) (<=s D')
```

```
   <- lemma2 D D'.

lemma3 : C == C' ->  C < (s C') -> type.
%mode lemma3 +E1 -E2.
lemma3z: lemma3 =z <z.
lemma3s: lemma3 (=s D) (<s D')
  <- lemma3 D D'.

lemma4 : C <= C -> type.
%mode lemma4 -E.
lemma4z: lemma4 <=z.
lemma4s: lemma4 (<=s D)
  <- lemma4 D.

lemma5 : C < (s C) -> type.
%mode lemma5 -E.
lemma5z: lemma5 <z.
lemma5s: lemma5 (<s D)
  <- lemma5 D.

lemma6 : C < C' -> C != C' -> type.
%mode lemma6 +E1 -E2.
lemma6z: lemma6 <z !=z1.
lemma6s: lemma6 (<s D) (!=s D')
  <- lemma6 D D'.

lemma7 : C < C' -> C <= C' -> type.
%mode lemma7 +E1 -E2.
lemma7z: lemma7 <z <=z.
lemma7s: lemma7 (<s D) (<=s D')
  <- lemma7 D D'.

lemma8 : C < C' -> C < (s C') -> type.
%mode lemma8 +E1 -E2.
lemma8z: lemma8 <z <z.
lemma8s: lemma8 (<s D) (<s D')
  <- lemma8 D D'.

% Encoding 'Yes' instances of CHROMATIC
colorvertex : vertex -> nat  -> type.
store_color : {V:vertex}{C:nat} colorvertex V C -> type.
%mode store_color +V -C -CGV.
coloring : nat -> graph -> type.
cg# : coloring N #.
cgvertex : coloring C (newv [v] (G v))
           <- (C' <= C)
           <- ({v:vertex} colorvertex v C' -> coloring C (G v)).
cgedge : coloring C (newe [e: edge A B] G)
          <- colorvertex A C1
```

```
          <- colorvertex B C2
          <- C1 != C2
          <- C1 <= C
          <- C2 <= C
          <- coloring C G.
cgunion : coloring C (G1 + G2)
           <- coloring C G1
           <- coloring C G2.

% A property of graph coloring
increase_color : coloring C G -> coloring (s C) G -> type.
%mode increase_color +CG1 -CG2.
inc_# : increase_color cg# cg#.
inc_v : increase_color (cgvertex CG E) (cgvertex CG' E')
            <- ({v:vertex}{c:colorvertex v C'} store_color v C' c ->
                          increase_color (CG v c) (CG' v c))
            <- lemma2 E E'.
inc_e : increase_color (cgedge CG E2 E1 E3 C1 C2) (cgedge CG' E2' E1' E3 C1 C2)
            <- increase_color CG CG'
            <- lemma2 E2 E2'
            <- lemma2 E1 E1'.
inc_+ : increase_color (cgunion CG1 CG2) (cgunion CG1' CG2')
            <- increase_color CG1 CG1'
            <- increase_color CG2 CG2'.
```

D Encoding of 3-SAT CHROMATIC Reduction

```
% Continuations
cont : type.
* : cont.
; : cont -> o -> cont.                          %infix right 10 ;.

% Satisfiability for continuations
satK : cont -> type.
satK* : satK *.
satK; : satK (K ; F)
         <- satK K
         <- sat F.

% Encoding of 3-SAT CHROMATIC reduction
var : v -> type.
relate : v -> vertex -> vertex -> vertex -> type.
conv : o -> nat -> nat -> cont -> graph -> type.
conv' : cont -> graph -> type.
conv'' : o -> graph -> type.
conv'''  : vertex -> graph -> type.
clique : graph -> type.
connectX : vertex -> graph -> type.
```

```
connectV : vertex -> graph -> type.
vars2clique : graph -> type.
c_new: conv (new F) C C' K (newv [v] newv [v'] newv [x] newe [e:edge v v'] (G v v' x))
        <- ({u:v} {v:vertex} {v':vertex} {x:vertex} relate u v v' x -> var u
              -o conv (F u) (s C) C' K (G v v' x)).
c_/\: conv (F /\ F') C C' K G
        <- conv F' C C' (K ; F) G.
c_\/: conv (F1 \/ F2 \/ F3) C C K (G1 + G2 + G3)
        <- conv' (K ; (F1 \/ F2 \/ F3)) G1 & clique G2 & vars2clique G3.

% Adding edges between clauses and all literals not in the clause
c'_*: conv' * # o- <T>.                  % Consume rest of the linear context
c'_;: conv' (K ; F) (G1 + G2)
            <- conv' K G1 & conv'' F G2.

c''11: (var U -o conv'' ((pos U) \/ (pos U) \/ (pos U))
                     (newv [c] newe [e:edge c V'] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U V V' _.

c''12: (var U -o conv'' ((pos U) \/ (pos U) \/ (neg U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
 <- relate U V V' _.

c''13: (var U -o conv'' ((pos U) \/ (neg U) \/ (pos U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U V V' _.


c''14: (var U -o conv'' ((pos U) \/ (neg U) \/ (neg U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U V V' _.

c''15: (var U -o conv'' ((neg U) \/ (pos U) \/ (pos U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U V V' _.

c''16: (var U -o conv'' ((neg U) \/ (pos U) \/ (neg U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U V V' _.

c''17: (var U -o conv'' ((neg U) \/ (neg U) \/ (pos U))
                     (newv [c] (G c)))
           <- ({c:vertex} conv''' c (G c))
```

```
          <- relate U V V' _.

c''18: (var U -o conv'' ((neg U) \/ (neg U) \/ (neg U))
                  (newv [c] newe [e:edge c V] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U V V' _.


c''21:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (pos U1) \/ (pos U2))
                          (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''22:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (pos U1) \/ (neg U2))
                          (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''23:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (neg U1) \/ (pos U2))
                          (newv [c] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''24:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (neg U1) \/ (neg U2))
                          (newv [c] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''25:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (pos U1) \/ (pos U2))
                          (newv [c] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''26:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (pos U1) \/ (neg U2))
                          (newv [c] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''27:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (neg U1) \/ (pos U2))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.
```

```
c''28:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (neg U1) \/ (neg U2))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.


c''31:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (pos U2) \/ (pos U1))
                          (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2'] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''32:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (pos U2) \/ (neg U1))
                          (newv [c] newe [e2:edge c V2'] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''33:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (neg U2) \/ (pos U1))
                          (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''34:  (var U1 -o var U2 -o conv'' ((pos U1) \/ (neg U2) \/ (neg U1))
                          (newv [c] newe [e2:edge c V2] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''35:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (pos U2) \/ (pos U1))
                          (newv [c] newe [e2:edge c V2'] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''36:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (pos U2) \/ (neg U1))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2'] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.

c''37:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (neg U2) \/ (pos U1))
                          (newv [c] newe [e2:edge c V2] (G c)))
           <- ({c:vertex} conv''' c (G c))
           <- relate U1 V1 V1' _
           <- relate U2 V2 V2' _.
```

```
c''38:  (var U1 -o var U2 -o conv'' ((neg U1) \/ (neg U2) \/ (neg U1))
                        (newv [c] newe [e1:edge c V1] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.


c''41:  (var U1 -o var U2 -o conv'' ((pos U2) \/ (pos U1) \/ (pos U1))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''42:  (var U1 -o var U2 -o conv'' ((pos U2) \/ (pos U1) \/ (neg U1))
                        (newv [c] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''43:  (var U1 -o var U2 -o conv'' ((pos U2) \/ (neg U1) \/ (pos U1))
                        (newv [c] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''44:  (var U1 -o var U2 -o conv'' ((pos U2) \/ (neg U1) \/ (neg U1))
                        (newv [c] newe [e1:edge c V1] newe [e2:edge c V2'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''45:  (var U1 -o var U2 -o conv'' ((neg U2) \/ (pos U1) \/ (pos U1))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''46:  (var U1 -o var U2 -o conv'' ((neg U2) \/ (pos U1) \/ (neg U1))
                        (newv [c] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.

c''47:  (var U1 -o var U2 -o conv'' ((neg U2) \/ (neg U1) \/ (pos U1))
                        (newv [c] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.
```

```
c''48:  (var U1 -o var U2 -o conv'' ((neg U2) \/ (neg U1) \/ (neg U1))
                        (newv [c] newe [e1:edge c V1] newe [e2:edge c V2] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _.


c''51:  (var U1 -o var U2 -o var U3 -o conv'' ((pos U1) \/ (pos U2) \/ (pos U3))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2']
                                  newe [e3: edge c V3'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.


c''52:  (var U1 -o var U2 -o var U3 -o conv'' ((pos U1) \/ (pos U2) \/ (neg U3))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2']
                                  newe [e3: edge c V3] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.

c''53:  (var U1 -o var U2 -o var U3 -o conv'' ((pos U1) \/ (neg U2) \/ (pos U3))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2]
                                  newe [e3: edge c V3'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.

c''54:  (var U1 -o var U2 -o var U3 -o conv'' ((pos U1) \/ (neg U2) \/ (neg U3))
                        (newv [c] newe [e1:edge c V1'] newe [e2:edge c V2]
                                  newe [e3: edge c V3] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.

c''55:  (var U1 -o var U2 -o var U3 -o conv'' ((neg U1) \/ (pos U2) \/ (pos U3))
                        (newv [c] newe [e1:edge c V1] newe [e2:edge c V2']
                                  newe [e3: edge c V3'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.
```

```
c''56:  (var U1 -o var U2 -o var U3 -o conv'' ((neg U1) \/ (pos U2) \/ (neg U3))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2']
                                    newe [e3: edge c V3] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.

c''57:  (var U1 -o var U2 -o var U3 -o conv'' ((neg U1) \/ (neg U2) \/ (pos U3))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2]
                                    newe [e3: edge c V3'] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.

c''58:  (var U1 -o var U2 -o var U3 -o conv'' ((neg U1) \/ (neg U2) \/ (neg U3))
                          (newv [c] newe [e1:edge c V1] newe [e2:edge c V2]
                                    newe [e3: edge c V3] (G c)))
          <- ({c:vertex} conv''' c (G c))
          <- relate U1 V1 V1' _
          <- relate U2 V2 V2' _
          <- relate U3 V3 V3' _.


c'''_v: (var U -o conv''' C (newe [e: edge C V] newe [e': edge C V'] G))
                <- (conv''' C G)
                <- relate U V V' _.
c'''_#: conv''' C #.


% Encoding of clique construction
clique_v : (var U -o clique (G1 + G2))
                   <- clique G1 & connectX X G2
                   <- relate U _ _ X.
clique_#  : clique #.

% Adds edges between the vertex X and all x-vertices in the context
connectX_v : (var U -o connectX X (newe [e:edge X X'] G))
                  <- relate U _ _ X'
                  <- connectX X G.
connectX_#      : connectX X #.

% Connect vertices corresponding to the variables to the clique
v2c_v : (var U -o vars2clique (G1 + G2 + G3 + G4))
                    <- vars2clique G1 & connectX V G2 & connectX V' G3 &
                       connectV X G4
                    <- relate U V V' X.
```

```
v2c_#   : vars2clique #.


connectV_v : (var U -o connectV X (newe [e:edge V X] newe [e':edge V' X] G))
                   <- relate U V V' _
                   <- connectV X G.
connectV_#      : connectV X #.


% A Property of 3-SAT CHROMATIC Reduction
conv_invariant : conv F C C' K G -> (C <= C') -> type.
%mode conv_invariant +CV -E.
convinvr_base : conv_invariant (c_\/ D) E
                <- lemma4 E.
convinvr_/\ : conv_invariant (c_/\ D) E
                <- conv_invariant D E.
convinvr_new : conv_invariant (c_new D) E
               <- ({u:v}{v:vertex}{v':vertex}{x:vertex}{r:relate u v v' x}{var: var u}
                     conv_invariant (D u v v' x r ^ var) (<=s E'))
               <- lemma2 E' E.                    % Proof of: (s C) <= C' ==> C <= C'
```

E Correctness of Encoding of 3-SAT CHROMATIC Reduction

```
% Correctness of CONNECTX: CONNECTX does not add edges between vertices
% having same color
connectX_color : {C:nat} colorvertex X C' -> (C < C') -> connectX X G
                                          -> coloring C' G -> type.
%mode connectX_color +C +CGV +E +CV -CG.
connectXc_base:  connectX_color z CGX E connectX_# cg#.
connectXc_cont:  {R:relate _ _ _ X1}
                 connectX_color (s C) CGX (<s E) (connectX_v D R ^ V)
                                                    (cgedge CG F2 F1 NE CGX1 CGX)
                 <- lemma8 E E'
                 <- connectX_color C CGX E' D CG
                 <- store_color X1 (s C) CGX1
                 <- lemma4 F1
                 <- lemma6 (<s E) NE
                 <- lemma7 (<s E) F2.
connectX_color' : {C:nat} colorvertex X z -> connectX X G -> coloring (s C) G -> type.
%mode connectX_color' +C +CGV +CV -CG.
connectXc'_base: connectX_color' z CGX connectX_# cg#.
connectXc'_cont: {R:relate _ _ _ X1}
                 connectX_color' (s C) CGX (connectX_v D R ^ V)
                                              (cgedge CG E' <=z !=z1 CGX1 CGX)
                 <- connectX_color' C CGX D CG'
                 <- increase_color CG' CG
```

```
                <- store_color X1 (s C) CGX1
                <- lemma5 E
                <- lemma7 E E'.

% Correctness of encoding of Clique
clique_color : {C:nat} clique G -> coloring C G -> type.
%mode clique_color +C +CV -CG.
cliquec_base : clique_color z clique_# cg#.
cliquec_cont : {R:relate _ _ _ X}
                clique_color (s C) (clique_v R (D1 , D2) ^ V) (cgunion CG1 CG2)
                <- clique_color C D1 CG1'
                <- increase_color CG1' CG1
                <- store_color X (s C) CGX
                <- lemma5 E                        % Proof of C < (s C)
                <- connectX_color C CGX E D2 CG2.

% Correctness of CONNECTV
connectV_color : {C:nat} colorvertex X C' -> (C < C') -> connectV X G
                                                  -> coloring C' G -> type.
%mode connectV_color +C +CGV +E +CV -CG.
connectVc_base: connectV_color z CGX E connectV_# cg#.
connectVc_cont1: {R:relate _ V1 V1' X1}
                connectV_color (s C) CGX (<s E) (connectV_v D R ^ V)
                   (cgedge (cgedge CG F2 <=z !=z1 CGX CGV1') F2 F1 NE CGX CGV1)
                 <- lemma8 E E'
                 <- connectV_color C CGX E' D CG
                 <- store_color V1 (s C) CGV1
                 <- store_color V1' z CGV1'
                 <- lemma6 (<s E) NE
                 <- lemma7 (<s E) F2
                 <- lemma4 F1.

connectVc_cont2: {R:relate _ V1 V1' X1}
                connectV_color (s C) CGX (<s E) (connectV_v D R ^ V)
                      (cgedge (cgedge CG F2 F1 NE CGX CGV1') F2 <=z !=z1 CGX CGV1)
                 <- lemma8 E E'
                 <- connectV_color C CGX E' D CG
                 <- store_color V1 z CGV1
                 <- store_color V1' (s C) CGV1'
                 <- lemma6 (<s E) NE
                 <- lemma7 (<s E) F2
                 <- lemma4 F1.

% Correctness of vars2clique (VARS-TO-CLIQUE): Edges added between x_j and v_i,
% and x_j and v_i' do not connect vertices with same color when i!=j
vars2clique_color : {C:nat} vars2clique G -> coloring C G -> type.
%mode vars2clique_color +C +CV -CG.
vars2cliquec_base : vars2clique_color z v2c_# cg#.
vars2cliquec_cont1 : {R:relate U V V' X}
```

```
                        vars2clique_color (s C)
                            (v2c_v R (D1 , D2, D3, D4) ^ VAR)
                            (cgunion CG1 (cgunion CG2 (cgunion CG3 CG4)))
                        <- vars2clique_color C D1 CG1'
                        <- increase_color CG1' CG1
                        <- store_color V z CGV
                        <- store_color V' (s C) CGV'
                        <- store_color X (s C) CGX
                        <- lemma5 E
                        <- connectX_color' C CGV D2 CG2
                        <- connectX_color C CGV' E D3 CG3
                        <- connectV_color C CGX E D4 CG4.
vars2cliquec_cont2 : {R:relate U V V' X}
                        vars2clique_color (s C)
                            (v2c_v R (D1 , D2, D3, D4) ^ VAR)
                            (cgunion CG1 (cgunion CG2 (cgunion CG3 CG4)))
                        <- vars2clique_color C D1 CG1'
                        <- increase_color CG1' CG1
                        <- store_color V (s C) CGV
                        <- store_color V' z CGV'
                        <- store_color X (s C) CGX
                        <- lemma5 E
                        <- connectX_color C CGV E D2 CG2
                        <- connectX_color' C CGV' D3 CG3
                        <- connectV_color C CGX E D4 CG4.


% For verifying that environment is valid
necolor : {M:nat} {N:nat} M != N -> type.
%mode necolor +M +N -MN.
necolor1: necolor z (s M) !=z1.
necolor2: necolor (s M) z !=z2.
necolor3: necolor (s M) (s N) (!=s D)
           <- necolor M N D.
lecolor : {M:nat}{N:nat} M <= N -> type.
%mode lecolor +M +N -MN.
lecolor1: lecolor z M <=z.
lecolor2: lecolor (s M) (s N) (<=s D)
          <- lecolor M N D.

% Correctness of coloring of the graph created by connecting a clause vertex with
% all literals not in the clause
conv'''_color : {C:nat} {C':nat} {C'':nat} colorvertex V (s C'') ->
                                           conv''' V G -> coloring C' G -> type.
%mode conv'''_color +C +C' +C'' +CGV +CV -CG.
conv'''c_base: conv'''_color z C' C'' CGV c'''_# cg#.
conv'''c_cont1:{R:relate U1 V1 V1' _}
               conv'''_color (s C) C' C'' CGV (c'''_v R D ^ V)
                         (cgedge (cgedge CG <=z F !=z2 CGV1' CGV) F1 F NE CGV1 CGV)
```

```
                <- conv'''_color C C' C'' CGV D CG
                <- store_color V1 (s C1) CGV1
                <- store_color V1' z CGV1'
                <- necolor (s C'') (s C1) NE
                <- lecolor (s C'') C' F
                <- lecolor (s C1) C' F1.

conv''_color : {C:nat} conv'' F G -> sat F -> coloring C G -> type.
%mode conv''_color +C +CV +CF -CG.
%%                                    %%
% Case: F=(pos U) \/ (pos U) \/ (pos U)%
%       U=True                         %
%%                                    %%
conv''c_11 : {R:relate U V V' _}
            conv''_color (s C) (c''11 R D ^ VAR) (sat\/1 (satp H))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc) <=z F !=z2 CGV' cgc)) F)
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- store_var U true H
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc)).
%%
% Case: F=(pos U) \/ (pos U) \/ (neg U)
%       U=True

conv''c_12A:{R:relate U V V' _}
            conv''_color (s C) (c''12 R D ^ VAR) (sat\/1 (satp H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- store_var U true H
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U) \/ (pos U) \/ (neg U)
%       U=False

conv''c_12B:{R:relate U V V' _}
            conv''_color (s C) (c''12 R D ^ VAR) (sat\/2 (sat\/2 (satn H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'

            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
```

```
                    conv'''_color C (s C) C' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U) \/ (neg U) \/ (pos U)
%       U=True

conv''c_13A:{R:relate U V V' _}
            conv''_color (s C) (c''13 R D ^ VAR) (sat\/1 (satp H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U true H
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U) \/ (neg U) \/ (pos U)
%       U=False

conv''c_13B:{R:relate U V V' _}
            conv''_color (s C) (c''13 R D ^ VAR) (sat\/2 (sat\/1 (satn H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).
%%
% Case: F=(pos U) \/ (neg U) \/ (neg U)
%       U=True

conv''c_14A:{R:relate U V V' _}
            conv''_color (s C) (c''14 R D ^ VAR) (sat\/1 (satp H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U true H
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U) \/ (neg U) \/ (neg U)
%       U=False
```

```
conv''c_14B:{R:relate U V V' _}
            conv''_color (s C) (c''14 R D ^ VAR) (sat\/2 (sat\/1 (satn H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U) \/ (pos U) \/ (pos U)
%       U=True

conv''c_15A:{R:relate U V V' _}
            conv''_color (s C) (c''15 R D ^ VAR) (sat\/2 (sat\/1 (satp H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U true H
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U) \/ (pos U) \/ (pos U)
%       U=False

conv''c_15B:{R:relate U V V' _}
            conv''_color (s C) (c''15 R D ^ VAR) (sat\/1 (satn H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U) \/ (pos U) \/ (neg U)
%       U=True

conv''c_16A:{R:relate U V V' _}
            conv''_color (s C) (c''16 R D ^ VAR) (sat\/2 (sat\/1 (satp H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U true H
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
```

```
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U) \/ (pos U) \/ (neg U)
%       U=False

conv''c_16B:{R:relate U V V' _}
            conv''_color (s C) (c''16 R D ^ VAR) (sat\/1 (satn H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U) \/ (neg U) \/ (pos U)
%       U=True

conv''c_17A:{R:relate U V V' _}
            conv''_color (s C) (c''17 R D ^ VAR) (sat\/2 (sat\/2 (satp H)))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U true H
            <- store_color V (s C') CGV
            <- store_color V' z CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U) \/ (neg U) \/ (pos U)
%       U=False

conv''c_17B:{R:relate U V V' _}
            conv''_color (s C) (c''17 R D ^ VAR) (sat\/1 (satn H))
                     (cgvertex ([c] [cgc] (CG c cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- lecolor (s C') (s C) F
            <- ({c:vertex}{cgc:colorvertex c (s C')} store_color c (s C') cgc ->
                     conv'''_color C (s C) C' cgc (D c) (CG c cgc)).
```

```
%%
% Case: F=(neg U) \/ (neg U) \/ (neg U)
%       U=False

conv''c_18 : {R:relate U V V' _}
            conv''_color (s C) (c''18 R D ^ VAR) (sat\/1 (satn H))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc)  <=z F !=z2 CGV cgc)) F)
            <- store_var U false H
            <- store_color V z CGV
            <- store_color V' (s C') CGV'
            <- ({c:vertex}{cgc:colorvertex c (s C')}
                      conv'''_color C (s C) C' cgc (D c) (CG c cgc))
            <- lecolor (s C') (s C) F.


%%                                      %%
% Case: F=(pos U1) \/ (pos U1) \/ (pos U2)%
%       U1=True & U2=True                 %
%%                                      %%
conv''c_21A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''21 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1' cgc)) F)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U1) \/ (pos U2)
%       U1=False & U2=True

conv''c_21B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''21 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H2)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1' cgc)) F2)
             <- store_var U2 true H2
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C2') (s C1') NE
```

```
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                     -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U1) \/ (pos U2)
%       U1=True & U2=False
conv''c_21C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''21 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') (s C2') NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U1) \/ (pos U1) \/ (neg U2)
%       U1=True & U2=False

conv''c_22A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''22 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1' cgc)) F)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U1) \/ (neg U2)
%       U1=False & U2=False

conv''c_22B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''22 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satn H2)))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1' cgc)) F2)
             <- store_var U2 false H2
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
```

```
            <- store_color V2 z CGV2
            <- store_color V2' (s C2') CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                     -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U1) \/ (neg U2)
%       U1=True & U2=True

conv''c_22C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''22 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') (s C2') NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U1) \/ (pos U2)
%       U1=True & U2=_

conv''c_23A : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''23 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V2' C2' CGV2'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U1) \/ (neg U1) \/ (pos U2)
%       U1=False & U2=_

conv''c_23B : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
```

```
              conv''_color (s (s C)) (c''23 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2' C2' CGV2'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U1) \/ (neg U1) \/ (neg U2)
%       U1=True & U2=_

conv''c_24A : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''24 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V2 C2' CGV2
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(pos U1) \/ (neg U1) \/ (neg U2)
%       U1=False & U2=_

conv''c_24B : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''24 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 C2' CGV2
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U1) \/ (pos U2)
%       U1=True & U2=_

conv''c_25A : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
```

```
              conv''_color (s (s C)) (c''25 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V2' C2' CGV2'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U1) \/ (pos U1) \/ (pos U2)
%       U1=False & U2=_

conv''c_25B : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''25 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2' C2' CGV2'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U1) \/ (neg U2)
%       U1=True & U2=_

conv''c_26A : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''26 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V2 C2' CGV2
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U1) \/ (neg U2)
%       U1=False & U2=_
```

```
conv''c_26B : {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''26 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
              <- store_var U1 false H1
              <- store_color V1' (s C1') CGV1'
              <- store_color V2 C2' CGV2
              <- lecolor C2' (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C1') C2' NE
              <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (neg U1) \/ (pos U2)
%       U1=False & U2=True

conv''c_27A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''27 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1 cgc)) F)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U1) \/ (pos U2)
%       U1=True & U2=True

conv''c_27B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''27 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H2)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1 cgc)) F2)
             <- store_var U2 true H2
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C2') (s C1') NE
             <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).
```

```
% Case: F=(neg U1) \/ (neg U1) \/ (pos U2)
%       U1=False & U2=False

conv''c_27C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''27 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') (s C2') NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U1) \/ (neg U1) \/ (neg U2)
%       U1=False & U2=False

conv''c_28A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''28 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1 cgc)) F)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U1) \/ (neg U2)
%       U1=True & U2=False

conv''c_28B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''28 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satn H2)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1 cgc)) F2)
             <- store_var U2 false H2
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
```

```
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                     -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U1) \/ (neg U2)
%       U1=False & U2=True

conv''c_28C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''28 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
                         (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1 cgc)) F1)
            <- store_var U1 false H1
            <- store_color V1 z CGV1
            <- store_color V1' (s C1') CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%                                      %%
% Case: F=(pos U1) \/ (pos U2) \/ (pos U1)%
%       U1=True & U2=True                %
%%                                      %%
conv''c_31A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''31 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
                         (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1' cgc)) F)
            <- store_var U1 true H1
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C1') (s (s C)) F
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U2) \/ (pos U1)
%       U1=False & U2=True

conv''c_31B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''31 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H2)))
```

```
                    (cgvertex ([c] [cgc] (cgedge (cgedge
                        (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1' cgc)) F2)
              <- store_var U2 true H2
              <- store_color V1 z CGV1
              <- store_color V1' (s C1') CGV1'
              <- store_color V2 (s C2') CGV2
              <- store_color V2' z CGV2'
              <- lecolor (s C2') (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C2') (s C1') NE
              <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U2) \/ (pos U1)
%       U1=True & U2=False
conv''c_31C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''31 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1' cgc)) F1)
              <- store_var U1 true H1
              <- store_color V1 (s C1') CGV1
              <- store_color V1' z CGV1'
              <- store_color V2 z CGV2
              <- store_color V2' (s C2') CGV2'
              <- lecolor (s C2') (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C1') (s C2') NE
              <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (pos U2) \/ (neg U1)
%       U1=True & U2=_

conv''c_32A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''32 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
              <- store_var U1 true H1
              <- store_color V1 (s C1') CGV1
              <- store_color V2' C2' CGV2'
              <- lecolor C2' (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C1') C2' NE
              <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).
```

```
%%
% Case: F=(pos U1) \/ (pos U2) \/ (neg U1)
%       U1=False & U2=_

conv''c_32B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''32 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2' C2' CGV2'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U2) \/ (pos U1)
%       U1=True & U2=False

conv''c_33A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''33 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1' cgc)) F)
             <- store_var U1 true H1
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (neg U2) \/ (pos U1)
%       U1=False & U2=False

conv''c_33B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''33 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H2)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1' cgc)) F2)
             <- store_var U2 false H2
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
```

```
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                     -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (neg U2) \/ (pos U1)
%       U1=True & U2=True

conv''c_33C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''33 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
                         (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1' cgc)) F1)
            <- store_var U1 true H1
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U2) \/ (neg U1)
%       U1=True & U2=_

conv''c_34A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''34 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
            <- store_var U1 true H1
            <- store_color V2 C2' CGV2
            <- store_color V1 (s C1') CGV1
            <- lecolor C2' (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') C2' NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U2) \/ (neg U1)
%       U1=False & U2=_

conv''c_34B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''34 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satn H1)))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
            <- store_var U1 false H1
```

```
            <- store_color V2 C2' CGV2
            <- store_color V1' (s C1') CGV1'
            <- lecolor C2' (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') C2' NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U2) \/ (pos U1)
%       U1=True & U2=_

conv''c_35A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''35 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H1)))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
            <- store_var U1 true H1
            <- store_color V2' C2' CGV2'
            <- store_color V1 (s C1') CGV1
            <- lecolor C2' (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') C2' NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U2) \/ (pos U1)
%       U1=False & U2=_

conv''c_35B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''35 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                     (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
            <- store_var U1 false H1
            <- store_color V2' C2' CGV2'
            <- store_color V1' (s C1') CGV1'
            <- lecolor C2' (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') C2' NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U2) \/ (neg U1)
%       U1=False & U2=True

conv''c_36A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''36 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
```

```
                    (cgvertex ([c] [cgc] (cgedge (cgedge
                        (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1 cgc)) F)
              <- store_var U1 false H1
              <- store_color V1 z CGV1
              <- store_color V1' (s C1') CGV1'
              <- store_color V2 (s C2') CGV2
              <- store_color V2' z CGV2'
              <- lecolor (s C1') (s (s C)) F
              <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (pos U2) \/ (neg U1)
%       U1=True & U2=True

conv''c_36B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''36 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H2)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1 cgc)) F2)
              <- store_var U2 true H2
              <- store_color V1 (s C1') CGV1
              <- store_color V1' z CGV1'
              <- store_color V2 (s C2') CGV2
              <- store_color V2' z CGV2'
              <- lecolor (s C2') (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C2') (s C1') NE
              <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (pos U2) \/ (neg U1)
%       U1=False & U2=False

conv''c_36C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
              conv''_color (s (s C)) (c''36 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1 cgc)) F1)
              <- store_var U1 false H1
              <- store_color V1 z CGV1
              <- store_color V1' (s C1') CGV1'
              <- store_color V2 z CGV2
              <- store_color V2' (s C2') CGV2'
              <- lecolor (s C2') (s (s C)) F2
              <- lecolor (s C1') (s (s C)) F1
              <- necolor (s C1') (s C2') NE
              <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).
```

```
%%
% Case: F=(neg U1) \/ (neg U2) \/ (pos U1)
%       U1=True & U2=_

conv''c_37A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''37 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 true H1
             <- store_color V2 C2' CGV2
             <- store_color V1 (s C1') CGV1
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (neg U2) \/ (pos U1)
%       U1=False & U2=_

conv''c_37B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''37 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V2 C2' CGV2
             <- store_color V1' (s C1') CGV1'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (neg U2) \/ (neg U1)
%       U1=False & U2=False

conv''c_38A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''38 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1 cgc)) F)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C1') (s (s C)) F
```

```
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U2) \/ (neg U1)
%       U1=True & U2=False

conv''c_38B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''38 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H2)))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1 cgc)) F2)
             <- store_var U2 false H2
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C2') (s C1') NE
             <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                      -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U2) \/ (neg U1)
%       U1=False & U2=True

conv''c_38C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''38 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
                          (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') (s C2') NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                      -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%                                      %%
% Case: F=(pos U2) \/ (pos U1) \/ (pos U1)%
%       U1=True & U2=True                 %
%%                                      %%
conv''c_41A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''41 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H2))
                      (cgvertex ([c] [cgc] (cgedge (cgedge
```

```
                    (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1' cgc)) F)
            <- store_var U2 true H2
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C1') (s (s C)) F
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                    -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U2) \/ (pos U1) \/ (pos U1)
%       U1=False & U2=True

conv''c_41B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''41 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H2))
                    (cgvertex ([c] [cgc] (cgedge (cgedge
                        (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1' cgc)) F2)
            <- store_var U2 true H2
            <- store_color V1 z CGV1
            <- store_color V1' (s C1') CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                    -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U2) \/ (pos U1) \/ (pos U1)
%       U1=True & U2=False
conv''c_41C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''41 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                    (cgvertex ([c] [cgc] (cgedge (cgedge
                        (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1' cgc)) F1)
            <- store_var U1 true H1
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 z CGV2
            <- store_color V2' (s C2') CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                    -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).
```

```
%%
% Case: F=(pos U2) \/ (pos U1) \/ (neg U1)
%       U1=True & U2=_

conv''c_42A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''42 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V2' C2' CGV2'
             <- store_color V1 (s C1') CGV1
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U2) \/ (pos U1) \/ (neg U1)
%       U1=False & U2=_

conv''c_42B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''42 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 false H1
             <- store_color V2' C2' CGV2'
             <- store_color V1' (s C1') CGV1'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U2) \/ (neg U1) \/ (pos U1)
%       U1=True & U2=_

conv''c_43A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''43 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V2' C2' CGV2'
             <- store_color V1 (s C1') CGV1
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).
```

```
%%
% Case: F=(pos U2) \/ (neg U1) \/ (pos U1)
%       U1=False & U2=_

conv''c_43B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''43 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 false H1
             <- store_color V2' C2' CGV2'
             <- store_color V1' (s C1') CGV1'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U2) \/ (neg U1) \/ (neg U1)
%       U1=False & U2=True

conv''c_44A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''44 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2' cgc) <=z F !=z2 CGV1 cgc)) F)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U2) \/ (neg U1) \/ (neg U1)
%       U1=True & U2=True

conv''c_44B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''44 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satp H2))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2' cgc) F1 F2 NE CGV1 cgc)) F2)
             <- store_var U2 true H2
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
```

```
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                     -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U2) \/ (neg U1) \/ (neg U1)
%       U1=False & U2=False
conv''c_44C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''44 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
                         (CG c cgc) F2 F1 NE CGV2' cgc) <=z F1 !=z2 CGV1 cgc)) F1)
            <- store_var U1 false H1
            <- store_color V1 z CGV1
            <- store_color V1' (s C1') CGV1'
            <- store_color V2 z CGV2
            <- store_color V2' (s C2') CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U2) \/ (pos U1) \/ (pos U1)
%       U1=True & U2=False

conv''c_45A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''45 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
                         (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1' cgc)) F)
            <- store_var U1 true H1
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 z CGV2
            <- store_color V2' (s C2') CGV2'
            <- lecolor (s C1') (s (s C)) F
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U2) \/ (pos U1) \/ (pos U1)
%       U1=False & U2=False

conv''c_45B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''45 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H2))
                     (cgvertex ([c] [cgc] (cgedge (cgedge
```

```
                    (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1' cgc)) F2)
            <- store_var U2 false H2
            <- store_color V1 z CGV1
            <- store_color V1' (s C1') CGV1'
            <- store_color V2 z CGV2
            <- store_color V2' (s C2') CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C2') (s C1') NE
            <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                    -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U2) \/ (pos U1) \/ (pos U1)
%       U1=True & U2=True

conv''c_45C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''45 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                    (cgvertex ([c] [cgc] (cgedge (cgedge
                        (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1' cgc)) F1)
            <- store_var U1 true H1
            <- store_color V1 (s C1') CGV1
            <- store_color V1' z CGV1'
            <- store_color V2 (s C2') CGV2
            <- store_color V2' z CGV2'
            <- lecolor (s C2') (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')}
                    conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U2) \/ (pos U1) \/ (neg U1)
%       U1=True & U2=_

conv''c_46A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
            conv''_color (s (s C)) (c''46 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                    (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
            <- store_var U1 true H1
            <- store_color V2 C2' CGV2'
            <- store_color V1 (s C1') CGV1
            <- lecolor C2' (s (s C)) F2
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') C2' NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                    -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
```

```
% Case: F=(neg U2) \/ (pos U1) \/ (neg U1)
%       U1=True & U2=_

conv''c_46B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''46 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2' cgc)) F1)
             <- store_var U1 true H1
             <- store_color V2 C2' CGV2'
             <- store_color V1' (s C1') CGV1'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U2) \/ (neg U1) \/ (pos U1)
%       U1=True & U2=_

conv''c_47A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''47 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/2 (satp H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 true H1
             <- store_color V2 C2' CGV2
             <- store_color V1 (s C1') CGV1
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).

%%
% Case: F=(neg U2) \/ (neg U1) \/ (pos U1)
%       U1=False & U2=_

conv''c_47B :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''47 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (CG c cgc) F2 F1 NE CGV2 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V2 C2' CGV2
             <- store_color V1' (s C1') CGV1'
             <- lecolor C2' (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C1') C2' NE
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%
```

```
% Case: F=(neg U2) \/ (neg U1) \/ (neg U1)
%       U1=False & U2=False

conv''c_48A :{R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''48 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F !=z2 CGV2 cgc) <=z F !=z2 CGV1 cgc)) F)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C1') (s (s C)) F
             <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U2) \/ (neg U1) \/ (neg U1)
%       U1=True & U2=False

conv''c_48B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''48 R2 R1 D ^ VAR2 ^ VAR1) (sat\/1 (satn H2))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) <=z F2 !=z2 CGV2 cgc) F1 F2 NE CGV1 cgc)) F2)
             <- store_var U2 false H2
             <- store_color V1 (s C1') CGV1
             <- store_color V1' z CGV1'
             <- store_color V2 z CGV2
             <- store_color V2' (s C2') CGV2'
             <- lecolor (s C2') (s (s C)) F2
             <- lecolor (s C1') (s (s C)) F1
             <- necolor (s C2') (s C1') NE
             <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s C)) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U2) \/ (neg U1) \/ (neg U1)
%       U1=False & U2=True

conv''c_48C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             conv''_color (s (s C)) (c''48 R2 R1 D ^ VAR2 ^ VAR1) (sat\/2 (sat\/1 (satn H1)))
                       (cgvertex ([c] [cgc] (cgedge (cgedge
                           (CG c cgc) F2 F1 NE CGV2 cgc) <=z F1 !=z2 CGV1 cgc)) F1)
             <- store_var U1 false H1
             <- store_color V1 z CGV1
             <- store_color V1' (s C1') CGV1'
             <- store_color V2 (s C2') CGV2
             <- store_color V2' z CGV2'
             <- lecolor (s C2') (s (s C)) F2
```

```
            <- lecolor (s C1') (s (s C)) F1
            <- necolor (s C1') (s C2') NE
            <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                     -> conv'''_color C (s (s C)) C1' cgc (D c) (CG c cgc)).


%%                                      %%
% Case: F=(pos U1) \/ (pos U2) \/ (pos U3)%
%         U1=True & U2=_ & U3=_           %
%%                                      %%
conv''c_51A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''51 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                  F3 F1 NE3 CGV3' cgc) F2 F1 NE2 CGV2' cgc)
                                             <=z F1 !=z2 CGV1' cgc)) F1)
          <- store_var U1 true H1
          <- store_color V1 (s C1') CGV1
          <- store_color V1' z CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                    -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U2) \/ (pos U3)
%         U1=_ & U2=True & U3=_

conv''c_51B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''51 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satp H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                  F3 F2 NE3 CGV3' cgc) <=z F2 !=z2 CGV2' cgc)
                                            F1 F2 NE1 CGV1' cgc)) F2)
          <- store_var U2 true H2
          <- store_color V1' C1' CGV1'
          <- store_color V2 (s C2') CGV2
          <- store_color V2' z CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
```

```
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U2) \/ (pos U3)
%         U1=_ & U2=_ & U3=True

conv''c_51C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''51 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                    (sat\/2 (sat\/2 (satp H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3' cgc) F2 F3 NE2 CGV2' cgc)
                                              F1 F3 NE1 CGV1' cgc)) F3)
          <- store_var U3 true H3
          <- store_color V1' C1' CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3 (s C3') CGV3
          <- store_color V3' z CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                       -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (pos U2) \/ (neg U3)
%         U1=True & U2=_ & U3=_

conv''c_52A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''52 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                  F3 F1 NE3 CGV3 cgc) F2 F1 NE2 CGV2' cgc)
                                            <=z F1 !=z2 CGV1' cgc)) F1)
          <- store_var U1 true H1
          <- store_color V1 (s C1') CGV1
          <- store_color V1' z CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
```

```
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (pos U2) \/ (neg U3)
%         U1=_ & U2=True & U3=_

conv''c_52B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''52 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                    (sat\/2 (sat\/1(satp H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3 cgc) <=z F2 !=z2 CGV2' cgc)
                                                F1 F2 NE1 CGV1' cgc)) F2)
          <- store_var U2 true H2
          <- store_color V1' C1' CGV1'
          <- store_color V2 (s C2') CGV2
          <- store_color V2' z CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                       -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).



% Case: F=(pos U1) \/ (pos U2) \/ (neg U3)
%         U1=_ & U2=_ & U3=False

conv''c_52C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''52 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                    (sat\/2 (sat\/2 (satn H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3 cgc) F2 F3 NE2 CGV2' cgc)
                                                F1 F3 NE1 CGV1' cgc)) F3)
          <- store_var U3 false H3
          <- store_color V1' C1' CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3 z CGV3
          <- store_color V3' (s C3') CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
```

```
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                       -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U2) \/ (pos U3)
%         U1=True & U2=_ & U3=_

conv''c_53A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''53 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F1 NE3 CGV3' cgc) F2 F1 NE2 CGV2 cgc)
                                               <=z F1 !=z2 CGV1' cgc)) F1)
          <- store_var U1 true H1
          <- store_color V1 (s C1') CGV1
          <- store_color V1' z CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3' C3' CGV3'
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                       -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=False & U3=_

conv''c_53B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''53 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                    (sat\/2 (sat\/1 (satn H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3' cgc) <=z F2 !=z2 CGV2 cgc)
                                               F1 F2 NE1 CGV1' cgc)) F2)
          <- store_var U2 false H2
          <- store_color V1' C1' CGV1'
          <- store_color V2 z CGV2
          <- store_color V2' (s C2') CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
```

```
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=_ & U3=True

conv''c_53C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             {CGV3':colorvertex V3' z}
             conv''_color (s (s (s C))) (c''53 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satp H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3' cgc) F2 F3 NE2 CGV2 cgc)
                                              F1 F3 NE1 CGV1' cgc)) F3)
          <- store_var U3 true H3
          <- store_color V1' C1' CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3 (s C3') CGV3
          <- store_color V3' z CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(pos U1) \/ (neg U2) \/ (neg U3)
%         U1=True & U2=_ & U3=_

conv''c_54A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''54 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satp H1))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F1 NE3 CGV3 cgc) F2 F1 NE2 CGV2 cgc)
                                              <=z F1 !=z2 CGV1' cgc)) F1)
          <- store_var U1 true H1
          <- store_color V1 (s C1') CGV1
          <- store_color V1' z CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3 C3' CGV3
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
```

```
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                        -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(pos U1) \/ (neg U2) \/ (neg U3)
%         U1=_ & U2=False & U3=_

conv''c_54B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''54 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satn H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3 cgc) <=z F2 !=z2 CGV2 cgc)
                                              F1 F2 NE1 CGV1' cgc)) F2)
          <- store_var U2 false H2
          <- store_color V1' C1' CGV1'
          <- store_color V2 z CGV2
          <- store_color V2' (s C2') CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).



% Case: F=(pos U1) \/ (neg U2) \/ (neg U3)
%         U1=_ & U2=_ & U3=False

conv''c_54C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''54 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satn H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3 cgc) F2 F3 NE2 CGV2 cgc)
                                              F1 F3 NE1 CGV1' cgc)) F3)
          <- store_var U3 false H3
          <- store_color V1' C1' CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3 z CGV3
          <- store_color V3' (s C3') CGV3'
          <- lecolor C1' (s (s (s C))) F1
```

```
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U2) \/ (pos U3)
%         U1=False & U2=_ & U3=_

conv''c_55A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''55 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F1 NE3 CGV3' cgc) F2 F1 NE2 CGV2' cgc)
                                              <=z F1 !=z2 CGV1 cgc)) F1)
          <- store_var U1 false H1
          <- store_color V1 z CGV1
          <- store_color V1' (s C1') CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                        -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (pos U2) \/ (pos U3)
%         U1=_ & U2=True & U3=_

conv''c_55B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''55 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satp H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3' cgc) <=z F2 !=z2 CGV2' cgc)
                                              F1 F2 NE1 CGV1 cgc)) F2)
          <- store_var U2 true H2
          <- store_color V1 C1' CGV1
          <- store_color V2 (s C2') CGV2
          <- store_color V2' z CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
```

```
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (pos U2) \/ (pos U3)
%         U1=_ & U2=_ & U3=True

conv''c_55C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''55 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satp H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3' cgc) F2 F3 NE2 CGV2' cgc)
                                              F1 F3 NE1 CGV1 cgc)) F3)
          <- store_var U3 true H3
          <- store_color V1 C1' CGV1
          <- store_color V2' C2' CGV2'
          <- store_color V3 (s C3') CGV3
          <- store_color V3' z CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (pos U2) \/ (neg U3)
%         U1=False & U2=_ & U3=_

conv''c_56A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''56 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                  F3 F1 NE3 CGV3 cgc) F2 F1 NE2 CGV2' cgc)
                                            <=z F1 !=z2 CGV1 cgc)) F1)
          <- store_var U1 false H1
          <- store_color V1 z CGV1
          <- store_color V1' (s C1') CGV1'
          <- store_color V2' C2' CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
```

```
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                        -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (pos U2) \/ (neg U3)
%         U1=_ & U2=True & U3=_

conv''c_56B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''56 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satp H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3 cgc) <=z F2 !=z2 CGV2' cgc)
                                              F1 F2 NE1 CGV1 cgc)) F2)
          <- store_var U2 true H2
          <- store_color V1 C1' CGV1
          <- store_color V2 (s C2') CGV2
          <- store_color V2' z CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C2') C3' NE3
     <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).



% Case: F=(neg U1) \/ (pos U2) \/ (neg U3)
%         U1=_ & U2=_ & U3=False

conv''c_56C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''56 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satn H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3 cgc) F2 F3 NE2 CGV2' cgc)
                                              F1 F3 NE1 CGV1 cgc)) F3)
          <- store_var U3 false H3
          <- store_color V1 C1' CGV1
          <- store_color V2' C2' CGV2'
          <- store_color V3 z CGV3
          <- store_color V3' (s C3') CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
```

```
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
  <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (neg U2) \/ (pos U3)
%         U1=False & U2=_ & U3=_

conv''c_57A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''57 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F1 NE3 CGV3' cgc) F2 F1 NE2 CGV2 cgc)
                                              <=z F1 !=z2 CGV1 cgc)) F1)
          <- store_var U1 false H1
          <- store_color V1 z CGV1
          <- store_color V1' (s C1') CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3' C3' CGV3'
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                        -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).

% Case: F=(neg U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=False & U3=_

conv''c_57B: {R1:relate U1 V1 V1' _ }{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''57 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satn H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3' cgc) <=z F2 !=z2 CGV2 cgc)
                                              F1 F2 NE1 CGV1 cgc)) F2)
          <- store_var U2 false H2
          <- store_color V1 C1' CGV1
          <- store_color V2 z CGV2
          <- store_color V2' (s C2') CGV2'
          <- store_color V3' C3' CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
```

```
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).



% Case: F=(neg U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=_ & U3=True

conv''c_57C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''57 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satp H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3' cgc) F2 F3 NE2 CGV2 cgc)
                                              F1 F3 NE1 CGV1 cgc)) F3)
          <- store_var U3 true H3
          <- store_color V1 C1' CGV1
          <- store_color V2 C2' CGV2
          <- store_color V3 (s C3') CGV3
          <- store_color V3' z CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).


%%
% Case: F=(neg U1) \/ (neg U2) \/ (neg U3)
%         U1=False & U2=_ & U3=_

conv''c_58A: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''58 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1) (sat\/1 (satn H1))
                       (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                  F3 F1 NE3 CGV3 cgc) F2 F1 NE2 CGV2 cgc)
                                            <=z F1 !=z2 CGV1 cgc)) F1)
          <- store_var U1 false H1
          <- store_color V1 z CGV1
          <- store_color V1' (s C1') CGV1'
          <- store_color V2 C2' CGV2
          <- store_color V3 C3' CGV3
          <- lecolor (s C1') (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
          <- necolor (s C1') C2' NE2
```

```
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C1') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C1')} store_color c (s C1') cgc
                        -> conv'''_color C (s (s (s C))) C1' cgc (D c) (CG c cgc)).


% Case: F=(neg U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=False & U3=_

conv''c_58B: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''58 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/1 (satn H2)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    F3 F2 NE3 CGV3 cgc) <=z F2 !=z2 CGV2 cgc)
                                              F1 F2 NE1 CGV1 cgc)) F2)
          <- store_var U2 false H2
          <- store_color V1 C1' CGV1
          <- store_color V2 z CGV2
          <- store_color V2' (s C2') CGV2'
          <- store_color V3 C3' CGV3
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor (s C2') (s (s (s C))) F2
          <- necolor (s C2') C1' NE1
          <- lecolor C3' (s (s (s C))) F3
          <- necolor (s C2') C3' NE3
          <- ({c:vertex}{cgc:colorvertex c (s C2')} store_color c (s C2') cgc
                        -> conv'''_color C (s (s (s C))) C2' cgc (D c) (CG c cgc)).



% Case: F=(neg U1) \/ (neg U2) \/ (pos U3)
%         U1=_ & U2=_ & U3=False

conv''c_58C: {R1:relate U1 V1 V1' _}{R2:relate U2 V2 V2' _}
             {R3:relate U3 V3 V3' _}
             conv''_color (s (s (s C))) (c''58 R3 R2 R1 D ^ VAR3 ^ VAR2 ^ VAR1)
                   (sat\/2 (sat\/2 (satn H3)))
                        (cgvertex ([c] [cgc] (cgedge (cgedge (cgedge (CG c cgc)
                                    <=z F3 !=z2 CGV3 cgc) F2 F3 NE2 CGV2 cgc)
                                              F1 F3 NE1 CGV1 cgc)) F3)
          <- store_var U3 false H3
          <- store_color V1 C1' CGV1
          <- store_color V2 C2' CGV2
          <- store_color V3 z CGV3
          <- store_color V3' (s C3') CGV3'
          <- lecolor C1' (s (s (s C))) F1
          <- lecolor C2' (s (s (s C))) F2
```

```
          <- necolor (s C3') C1' NE1
          <- lecolor (s C3') (s (s (s C))) F3
          <- necolor (s C3') C2' NE2
          <- ({c:vertex}{cgc:colorvertex c (s C3')} store_color c (s C3') cgc
                        -> conv'''_color C (s (s (s C))) C3' cgc (D c) (CG c cgc)).

% Correctness of coloring of all graphs corresponding to all clauses
% on the continuation
conv'_color : {C:nat} conv' K G -> satK K -> coloring C G -> type.
%mode conv'_color +C +CV +CK -CG.
conv'c_base : conv'_color C (c'_* ^ ()) satK* cg#.
conv'c_cont : conv'_color C (c'_; (D1 , D2)) (satK; E2 E1) (cgunion CG1 CG2)
               <- conv'_color C D1 E1 CG1
               <- conv''_color C D2 E2 CG2.

% Proof of the Reduction - from 3-SAT to Chromatic Number
main_theorem : {C:nat} conv F C C' K G -> sat F -> satK K -> coloring C' G -> type.
%mode main_theorem +C +CV +CF +CK -CG.
case1t      : main_theorem C (c_new D) (satnewt E) F
                    (cgvertex ([v] [cgv] (cgvertex ([v'][cgv']
                    (cgvertex ([x][cgx] (cgedge (CG v v' x cgv cgv' cgx)
                              <=z H !=z2 cgv' cgv)) H)) <=z)) H)
                <- ({u:v} {hypt: hyp u true}{v:vertex}{v':vertex}{x:vertex}
                    {r:relate u v v' x}{var: var u}{cgv :colorvertex v (s C)}
                    {cgv':colorvertex v' z}{cgx :colorvertex x (s C)}
                    store_color v (s C) cgv -> store_color v' z cgv' ->
                    store_color x (s C) cgx -> store_var u true hypt ->
                    main_theorem (s C) (D u v v' x r ^ var) (E u hypt) F
                                                (CG v v' x cgv cgv' cgx))
                <- ({u:v}{v:vertex}{v':vertex}{x:vertex}
                    {r:relate u v v' x}{var: var u}
                       conv_invariant (D u v v' x r ^ var) H).

case1f      : main_theorem C (c_new D) (satnewf E) F
                    (cgvertex ([v] [cgv] (cgvertex ([v'][cgv']
                    (cgvertex ([x][cgx] (cgedge (CG v v' x cgv cgv' cgx)
                              H <=z !=z1 cgv' cgv)) H)) H)) <=z)
                <- ({u:v} {hypf: hyp u false}{v:vertex}{v':vertex}{x:vertex}
                    {r:relate u v v' x}{var: var u}{cgv :colorvertex v z}
                    {cgv':colorvertex v' (s C)}{cgx :colorvertex x (s C)}
                    store_color v z cgv -> store_color v' (s C) cgv' ->
                    store_color x (s C) cgx -> store_var u false hypf ->
                    main_theorem (s C) (D u v v' x r ^ var) (E u hypf) F
                                                 (CG v v' x cgv cgv' cgx))
                <-({u:v}{v:vertex}{v':vertex}{x:vertex}
                   {r:relate u v v' x}{var: var u}
                     conv_invariant (D u v v' x r ^ var) H).

case2       : main_theorem C (c_/\ D) (sat/\ E1 E2) F CG
```

```
                <- main_theorem C D E2 (satK; E1 F) CG.

case3       : main_theorem C (c_\/ (D1 , D2 , D3)) E F (cgunion CG1 (cgunion CG2 CG3))
                  <- conv'_color C D1 (satK; E F) CG1
                  <- clique_color C D2 CG2
                  <- vars2clique_color C D3 CG3.
```