Chapter 6: Extending FELIX to Multiple Intensional Contexts

This chapter proves the second part of the third hypothesis of this thesis:

> *Third Hypothesis*: This new version of situation theory and the associated theorem prover is appropriate as a knowledge representation and reasoning system for theories of perception and belief.

This proof is a demonstration that an automated reasoning system can be developed for the situation theoretic belief and perception theories presented in chapter 5.

Intensional contexts[1] are the basic organizational tool for implementing reasoning about "modal" relations. Intensional relations are those which take formulae as an argument, such as the supports relation or the belief relation. Every intensional context except the root has a parent intensional context and any number (perhaps none) of child contexts. The root intensional context has no parent intensional context, but may have child contexts. An intensional context has a logic mode, either infon or classical. An intensional context is defined by a "pattern" of a relation where the propositional argument of that relation is "filled" by the formulae of the intensional context and the non-propositional arguments are filled with constants. A pattern is written 'X^Y', where X is a propositional variable and Y is the relation which has X as one of its arguments. The pattern for a belief relation is 'P^bel(A, P)', where A is some term naming an "agent" and P is a free meta-variable. The pattern for the supports relation is 'P ^ (S |= P)', where S is some term naming a situation and P is a free meta-variable. The terms may be free variables. Belief contexts have a classical logic mode and supports contexts have an infon logic mode.

Suppose the formula "*s* |= agent(a)"[2] is in the root intensional context. There is a sub-context, call it context 2, of the root intensional context with pattern X^(*s*|=X). Context 2 is for reasoning about the infons which *s* supports. FELIX infers that

[1] The idea of contexts is similar to the use of *attachments*. Particularly the notion of associating a logic mode with a context. [Frisch&Scherl 1991] discuss a theory of attachments. This is also discussed on p. 212 of [Genesereth&Nilsson 1987]. These discussions of attachment focus on reasoning about beliefs, where the use of contexts is intended for any intensional relation, and is particularly extended to the supports relation as well as the belief relation.

[2] This can be read as "The situation *s* supports the infon that 'a' is an agent."

"agent(a)" can be adopted in this sub-context by virtue of "s |= agent(a)" being present in the parent intensional context. This can be thought of as "projecting" the root intensional context formula onto the sub-context. The formula "s |= bel(a, t |= hand(a, ah))" in the root intensional context can be projected onto context 2 as "bel(a, t |= hand(a, ah))". This can in turn be projected onto a sub-context of context 2 (call it context 3) defined by the pattern X^bel(a,X). The projection in context 3 is "t |= hand(a, ah)". This can be projected onto a sub-context of context 3, call it context 4, defined by pattern X^(t|=X). The projection in context 4 is "hand(a, ah)." The formula "s |= bel(b, t |= hand(b, bh))" can be similarly projected onto contexts, with " bel(b, t |= hand(b, bh))" in context 2, "t |= hand(b, bh)" in context 5 (sub-context of context 2, pattern X^bel(b, X)), and "hand(b, bh)" in context 6 (sub-context of context 5, pattern X^(t|=X)). A picture of these contexts is in Exhibit 6. 1 on page 164.

The logical use of contexts is to reason about logical closure for the formula-type argument of the relation. Since an intensional context has a logic mode which depends on the relation generating the intensional context, the logic of the closure depends on the relation generating the intensional context. Logical closure for 'belief' formula arguments is classical. Logical closure for 'supports' formula arguments is infonic. FELIX could be extended to have more modes - nonmonotonic reasoning systems of various kinds (varying sets of "axioms" within the same basic framework, perhaps) - and the selection of mode for a belief intensional context could be made to depend on the agent argument as well as the formula argument. Thus, this provides a mechanism to tailor a major aspect of reasoning about an agent's beliefs to that agent.

A simple supports relation theorem can be used to show the multiple intensional context reasoning mechanism. The theorem is that if 'p' is universally supported and 'p⇒q' is universally supported, then 'q' is universally supported. This is formalized as: Given '$\forall\, t(\mathrm{sit}(t)\rightarrow t \models \mathrm{p})$' and '$\forall u(\mathrm{sit}(u)\rightarrow u \models(\mathrm{p}\Rightarrow\mathrm{q}))$', prove that '$\forall\, s(\mathrm{sit}(s)\rightarrow s \models \mathrm{q})$', where 'sit($x$)' is the claim that x is a situation.

The proof which FELIX develops is given below. Lemma s1 proves the free variable version of the consequent of the theorem (from which the universally bound version can be inferred), and lemma s2 proves the consequent goal of lemma s1 supposing

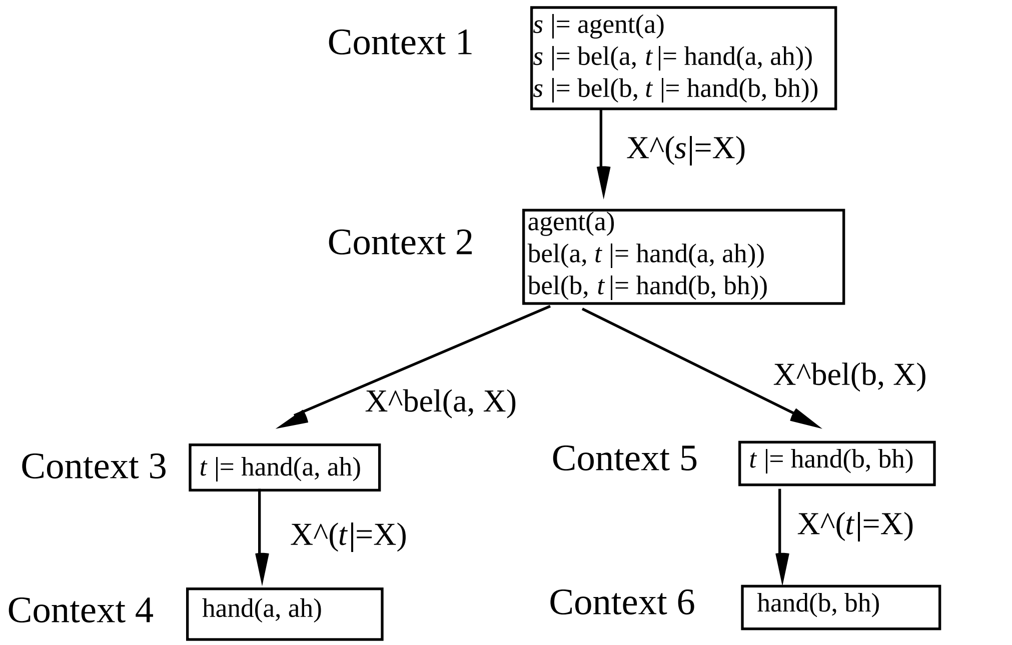

Exhibit 6. 1: An Example Tree of Contexts.

the antecedent of this goal (from which the goal of lemma s1 can be inferred). Lemma s2 reasons in two contexts to achieve its results. In the root intensional context it instantiates the given formulae and applies modus ponens to get steps 4 and 5. Steps 4 and 5 of the root intensional context are projected into context 2 to get steps 1 and 2 of context 2. Reasoning in context 2 using infon logic, FELIX infers that ‘q’ holds in context 2. Expanding ‘q’ into context 2’s parent intensional context, it gets step 6 of context 1. This completes the proof of lemma s2.

PROVE: $\forall s(\mathrm{sit}(s) \rightarrow s \models q)$
GIVEN: $\forall t(\mathrm{sit}(t) \rightarrow t \models p)$
$\forall u(\mathrm{sit}(u) \rightarrow u \models (p \Rightarrow q))$

1 :	$\forall s(\mathrm{sit}(s) \rightarrow s \models q)$	LEMMA s1	universal generalization

----

LEMMA s1:

PROVE: $\mathrm{sit}(s') \rightarrow s' \models q$

1 :	$\mathrm{sit}(s') \rightarrow s' \models q$	LEMMA s2	conditional

----

LEMMA s2:

PROVE: $s' \models q$
SUPPOSE: $\mathrm{sit}(s')$

1 :	$\mathrm{sit}(s')$	given	supposed
2 :	$\forall t(\mathrm{sit}(t) \rightarrow t \models p)$	given	input
3 :	$\forall u(\mathrm{sit}(u) \rightarrow u \models (p \Rightarrow q))$	given	input
4 :	$s' \models p$	[2, 1]	all_detachment
5 :	$s' \models (p \Rightarrow q)$	[3, 1]	all_detachment
6 :	$s' \models q$	CTXT 2: [3]	child

CONTEXT 2 : $s' \models$Formula

1 :	p	CTXT 1: [4]	parent
2 :	$p \Rightarrow q$	CTXT 1: [5]	parent
3 :	q	[2, 1]	modus_ponens

Exhibit 6. 2: Proof of
Universal Conditional Deduction Supports Theorem.

Extending the Implementation of FELIX for Multiple Contexts

The basic framework of FELIX is extended in two ways to handle reasoning in multiple contexts. One set of extensions are independent of the semantics of particular intensional relations, the other extensions are intensional relation specific. The modal-relation-independent extensions are in three areas. One extension is adding rules to the adoption task processing and interest task processing which propagate tasks to the relevant parent and child contexts. When adopting formula F, generate an adoption task of the simple expansion of F for the parent intensional context and if F matches a context-defining pattern then generate an adoption for the projection of F onto its child intensional context. The simple expansion Y’ of a formula F in an intensional context with defining pattern X^Y is found by substituting F for X in Y, where Y' is the substituted version of Y. In the proof in Exhibit 6. 2 on page 165, context 2 of lemma s2 has the defining pattern of X^(*s'* |=X). The simple expansion of 'q' in this intensional context is '*s'* |=q'. There is no projection of 'q' to a subcontext. A formula only has a projection if it is an atomic formula with an intensional relation. The projection is the formula-type argument of the relation and the subcontext pattern is the atomic formula with its formula type argument replaced by a meta-variable. When 'q' is adopted in context 2, a task to adopt '*s'* |=q' in context 1 (the parent intensional context of context 2) is generated. No parent intensional context adoption tasks are generated when a formula is adopted as a result of a projection from the parent intensional context. This prevents some wasted effort. Thus, when 'p' is adopted in context 2, no parent intensional context adoption task is generated.

The agenda mechanism of FELIX is extended to record the intensional context of each task. When a task is selected for processing, the current intensional context of the current supposition is "switched" to the intensional context of the selected task. The ordering of tasks in the agenda relies not only on the priority number of the task (lower priority numbers are processed before higher ones), but also on the intensional context of the task. If two tasks have different priority numbers, then the one with the lower priority number is processed first, as is the case in "single-context" FELIX. If two tasks have the same priority number, then their contexts are used to determine which to process first. The task with an intensional context which is "closer" to a focus intensional context is the one which is processed first. If the two tasks are equally close to a focus intensional context, then the one with the lower intensional context number is (arbitrarily) chosen. This ordering introduces two new concepts, the focus intensional context and the distance between intensional contexts.

A intensional context is deemed a "focus" intensional context if interest is registered in some focusing formula in that intensional context . A "focusing" formula is one which is not an atomic modal formula, i.e. an atomic formula with a intensional relation. In the current version of FELIX, the non-focusing formulae are of the form 'bel(A, P)' or 'S |= I'.

The distance between intensional contexts is measured with respect to the tree of intensional contexts. The distance is the path length in the tree between the two intensional contexts. The focus distance of an intensional context is the shortest path from that intensional context to *any* focus intensional context.

The inference rule implementations are extended to cope with multiple intensional contexts in a fairly simple way. Those rules which create a new supposition (conditional supposition, universal supposition, *reductio* supposition, dilemma supposition) set up the new supposition with the *full expansion* of the formula of interest as the ultimate interest of the new supposition, since linear processing suppositions always starts in context 1 (the root intensional context). Beyond this these rules are essentially unaffected by the intensional context mechanism.

The above describes three areas of FELIX's operation; the projection and expansion of formula adoptions and interests, the agenda task-ordering sensitivity to the intensional context of the task, and supposition-creating inference rules. These three areas of FELIX operate independently of the specific semantics of a particular intensional relation and are not modified in any way when defining new intensional relations, with one exception. The "expand formula" processing must be modified to handle the expansion of formulae which contain existential instantiation terms (skolem terms). This is done for the belief relation.

There is an additional issue relating to the semantics of references "across" intensional contexts. This is particularly interesting with respect to the belief relation. There is no reason to assume that because the a person calls something 'a', that another person about whom the first person has beliefs calls that same thing the same "name". Thus, to say that A believes that B believes 'a' is a car might be represented as: s |= bel(A, t |= bel(B, u |= car(a))). The problem with the representation is that it assumes that whatever A refers to by 'a' is also what B refers to by 'a', and is also what the theorist (the person writing the formula) refers to by 'a'. Konolige introduces a technique for handling this - the "bullet" operator.[3] Constants in the scope of belief relations are given a '•' prefix to indicate that the theorists name for a constant

[3] [Konolige 1986], p. 40-44 introduce the idea of the bullet operator and discuss the formal implications. This is also presented in [Genesereth&Nilsson 1987].

is being used, not that of the "believer". Thus, the previous formula would be: s |= bel(A, t |= bel(B, u |= car(• a))). The belief logic developed in this thesis makes the simplifying assumption that the theorist and all of the believers use the same "names" to refer to the same objects.

The modal-relation-specific modifications which are made to FELIX to define a new intensional relation are in specifying the "pattern" of the relation to give the syntax, specifying the logic mode of the intensional relation and the forward and backward reasons to give the semantics, and specifying the transformation rules for the "detachment normal form". As noted above, there may also be a modification necessary to the "expand formula" procedure to handle existential formulae.

If there are multiple intensional relations defined, then it is generally appropriate to give forward and backward reasons which define the semantics of formulae containing different intensional relations. For instance, in the case of the 'bel' relation and '|=', there is a rule called 'bel_veridicality' which implements the idea that if someone believes something then that thing is true: if 'S |= bel(A, P)' then infer 'P' (all beliefs are assumed to be factual in this simple model). This is present in FELIX as a forward reason.

Belief relation specific extensions

The belief pattern is 'X^bel(A,X)'. This defines an intensional context which contains A's beliefs. The logic mode of belief is *classical*, and thus the formula found in the second argument is "interpreted" as a classical first order logic formula. This is the technical expression of the claim that a person believes *propositions* about the world, things which are (believed to be) true. In the situation theoretic approach, these propositions are commonly of the form 'S |= I', where S is some situation and I is some infon and this proposition is read "The situation S supports the infon I."

The belief relation is an extension of *infon* logic; it is only found in infon formulae. The forward reasons for belief, expressed as natural deduction inference rules, are:

	Introduction	*Elimination*
$\models$ bel	$\dfrac{\Gamma : S \models \mathrm{sees}(A, T) \wedge T \models P}{\Gamma : S \models \mathrm{bel}(A, P)}$	$\dfrac{\Gamma : S \models \mathrm{bel}(A, P)}{\Gamma : P}$
bel $\vee$	$\dfrac{\Gamma : \mathrm{bel}(A, P) \vee \mathrm{bel}(A, Q)}{\Gamma : \mathrm{bel}(A, P \vee Q)}$	

The backward reasons for belief are:

$$\not\models \text{ bel} \qquad \frac{\Gamma : T \not\models P}{\Gamma : S \not\models \mathrm{bel}(A, T \models P)}$$

$$\text{bel } \wedge \qquad \frac{\Gamma : \mathrm{bel}(A, P), \quad \Delta : \mathrm{bel}(A, Q)}{\Gamma, \Delta : \mathrm{bel}(A, P \wedge Q)}$$

The expand-formula procedure is modified so that an existential instantiation term which is created for an existential formula in a belief intensional context is "expanded" by *reconstructing* the existential quantification. For instance, given the formula '$s \models \mathrm{bel}(a, \exists x\ \mathrm{p}(x))$' in the root intensional context, there is the formula '$\mathrm{bel}(a, \exists x\ \mathrm{p}(x))$' projected into context 2 (X^(s |= X)), and the formula '$\exists x\ \mathrm{p}(x)$' projected into context 2's child intensional context 3 (X^bel(a, X)). Using existential instantiation in context 3, FELIX creates a new formula, say 'p(b)' where b is the existential instantiation term for x. This expands to '$\mathrm{bel}(a, \exists x\ \mathrm{p}(x))$', since b was created as an existential instantiation term of a belief existential formula. This preserves the desired meaning of "existential belief", where believing that something exists which satisfies a particular formula does *not* imply that one believes any *particular* instantiation of that formula. For instance, believing "Some day my prince will come" does not imply that one believes that "my prince will come on Monday", even if it is in fact the case that one's prince *is* coming on Monday.

This existential limitation on formula expansions can be easily extended to other intensional relations.

The extensions to the the "detachment normal form" rules are as follows:

bel(A, P $\wedge$ Q)	$\rightarrow\rightarrow$	bel(A, P) $\wedge$ bel(A, Q)
bel(A, (P $\rightarrow$ Q))	$\rightarrow\rightarrow$	bel(A, P) $\Rightarrow$ bel(A, Q)
bel(A, P $\vee$ Q)	$\rightarrow\rightarrow$	~bel(A,~(P $\vee$ Q))

The first rule and second rules are warranted by the logical closure principle of belief. The third rule is warranted by the consistency principle of belief (but *not* by the knowledge principle, as discussed earlier).

Supports (|=) relation specific extensions

The intensional context pattern for the supports relation is X^(S |= X). Support intensional contexts are in infon logic mode.

The extensions to the forward and backward reasons are as follows:

Forward reasons:

	Introduction	*Elimination*
$\models \vee$	$\dfrac{\Gamma : (S \models P) \vee (S \models Q)}{\Gamma : S \models (P \vee Q)}$	$\dfrac{\Gamma : S \models (P \vee Q)}{\Gamma : (S \models P) \vee (S \models Q)}$
$- -$		$\dfrac{\Gamma : S \models - - P}{\Gamma : S \models P}$ [4]
$\models \wedge$		$\dfrac{\Gamma : S \models {}^\wedge P}{\Gamma : S \not\models P}$
sit	$\dfrac{\Gamma : S \models P}{\Gamma : \mathrm{sit}(S)}$	

Backward reasons:

$\models \vee$	$\dfrac{\Gamma : (S \models P) \vee (S \models Q)}{\Gamma : S \models (P \vee Q)}$	
$\models \wedge$	$\dfrac{\Gamma : S \not\models P}{\Gamma : S \models {}^\wedge P}$	
$\models -$		$\dfrac{\Gamma : S \models - P}{\Gamma : S \not\models P}$

No change is necessary to the procedure for expanding formulae from a child intensional context into its parent. That is, $(\exists x\ (\mathrm{s} \models \mathrm{p}(x)))$ iff $\mathrm{s} \models (\exists x\ \mathrm{p}(x))$.

The additional rules for the "detachment normal form" are as follows:

$$\mathrm{S} \models (\mathrm{P} \wedge \mathrm{Q}) \quad \rightarrow\rightarrow \quad (\mathrm{S} \models \mathrm{P}) \wedge (\mathrm{S} \models \mathrm{Q})$$
$$\mathrm{S} \models (\mathrm{P} \vee \mathrm{Q}) \quad \rightarrow\rightarrow \quad (\mathrm{S} \models \mathrm{P}) \vee (\mathrm{S} \models \mathrm{Q})$$

[4] This rule should be superfluous; it is taken care of by infon mode subcontext reasoning.

Applying FELIX to Problems with Beliefs

There are two major example theorem proving problems used to demonstrate FELIX's approach to proofs involving beliefs and the supports relation. One is the poker game which is presented in an earlier chapter. The other the "wise men" puzzle. This puzzle is presented here in a simplified form involving only two wise men, it is usually presented as the "3 wise men", or even the "N wise men," puzzle.

Poker Game

There are two results which FELIX must produce. One of these is Jack's belief that if Pete calls he loses. The other is Zack's belief that if Pete calls he wins. These problems are presented in detail in the previous chapter. What follows here is a discussion of the proofs which FELIX finds for these results. A simplification adopted in both of these proofs is to state the problem as a conditional supposition; prove "Given P, prove Q" rather than "Prove P implies Q". This simplification helps reduce the storage required to solve the problem, which is a difficulty in getting FELIX to run successfully.

The first problem presented is that of Jack's beliefs. The proof is given in several parts. The "given" and "to prove" formulae are in Exhibit 6. 3 on page 173. The proof steps used in the root intensional context (context 1) are in Exhibit 6. 4 on page 174. The proof steps used in context 2 are in Exhibit 6. 5 on page 175. The remainder of the proof, the steps for contexts 3, 4, and 5, is in Exhibit 6. 6 on page 176. The only focus intensional context which FELIX identifies in the course of finding this proof is context 5, and the goal of interest in that intensional context is 'loses(pete)'. Context 5 has the second most "local" adoptions, 4 of them, after the root intensional context (with 6)[5]. A local adoption is one which is not the result of an intensional context expansion or projection. The nonlocal adoptions have as their justification either 'parent' or 'child'. Context 2 is largely a conduit between contexts 1 and 4, and context 4 is entirely a conduit between 2 and

[5] The measure of more interest is what percentage of nonlocal adoption did the intensional context use. The focus intensional context ought to have a high percentage (a high "hit" ratio).

PROVE: sz|=bel(jack, sp|=loses(pete))

GIVEN: sz|=bel(jack, sp|=calls(pete))
sz|=sees(jack, sp)
sz|=knows_poker(jack)
sp|=hand(stone, sh)∧hand(pete, ph)∧players(pete, stone)
better(sh, ph)

$$
\begin{aligned}
&\forall s\ (\mathrm{sit}(s) \\
&\quad \rightarrow s \models \forall a(\mathrm{knows\_poker}(a) \\
&\qquad \Rightarrow \mathrm{bel}(a, \forall t, u(\mathrm{sit}(t) \wedge \mathrm{sit}(u) \\
&\qquad\quad \rightarrow \forall p, px, py\ ((u \models \mathrm{players}(px, py) \vee \mathrm{players}(py, px)) \\
&\qquad\qquad \wedge \exists x, y\ (\mathrm{better}(x, y) \\
&\qquad\qquad\quad \wedge(t \models \mathrm{bel}(p, u \models \mathrm{hand}(px, x) \wedge \mathrm{hand}(py, y)))) \\
&\qquad\quad \rightarrow t \models \mathrm{bel}(p, u \models(\mathrm{calls}(px) \Rightarrow \mathrm{wins}(px))) \\
&\qquad\qquad \wedge \mathrm{bel}(p, u \models(\mathrm{calls}(py) \Rightarrow \mathrm{loses}(py))))))))
\end{aligned}
$$

Exhibit 6. 3: Jack's Proof, part 1:
Problem Statement.

5. The presentation of the proof can be made more succinct by eliminating the explicit presentation of steps which are *only* intermediate steps in the expansion or projection processes. Using this approach, all of the steps of context 4 and most of the steps of context 2 need not be presented explicitly. Thus in context 5, the support step CTXT4:[1] which refers to CTXT2:[1] which refers to CTXT1:[1] can be collapsed to "CTXT1:[1] via 2&4". The collapsed version of the proof is given also. The collapsed version of context 1 is in Exhibit 6. 7 on page 177. The collapsed version of context 2 is in Exhibit 6. 8 on page 178. The collapsed versions of contexts 3, 4, and 5 are in Exhibit 6. 9 on page 179.

The proof of Jack's conditional belief about Pete only uses one of the "poker domain" rules[6]. The proof of Zack's conditional belief is more complex and uses four of the poker domain rules. This proof if presented in several parts. The problem statement is given in Exhibit 6. 10 on page 180. The root intensional context proof is in Exhibit 6. 11 on page 181. The proof steps for context 2 are in Exhibit 6. 12 on page 182. The proof steps for context 3 are in Exhibit 6. 13 on page 183. The proof steps for context 4 are in Exhibit 6. 14 on page 184.

[6] Rule 3: "everybody_who_knows_poker_believes(knowing_better_hand_implies_knowing_results)". This is presented in Exhibit 5.3 of Chapter 5.

<table>
<tr><th>Step :</th><th>Adopted Formula</th><th>Support Steps</th><th>Justification</th></tr>
<tr><td>1 :</td><td>sz|=bel(jack, sp|=calls(pete))</td><td></td><td>given</td></tr>
<tr><td>2 :</td><td>sz|=sees(jack, sp)</td><td></td><td>given</td></tr>
<tr><td>3 :</td><td>sz|=knows_poker(jack)</td><td></td><td>given</td></tr>
<tr><td>4 :</td><td>sp|=hand(stone, sh)∧hand(pete, ph)∧players(pete, stone)</td><td></td><td>given</td></tr>
<tr><td>5 :</td><td>better(sh, ph)</td><td></td><td>given</td></tr>
<tr><td>6 :</td><td>∀ s (sit(s) → s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py) ∨ players(py, px)) ∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧ bel(p, u |= (calls(py) ⇒ loses(py))))))))</td><td></td><td>given</td></tr>
<tr><td>7 :</td><td>sit(sz)</td><td>[1]</td><td>support_sit F</td></tr>
<tr><td>8 :</td><td>sit(sp)</td><td>[4]</td><td>support_sit F</td></tr>
<tr><td>9 :</td><td>sz|=bel(jack, sp|=hand(stone, sh)
∧hand(pete, ph)∧players(pete, stone))</td><td>[2, 4]</td><td>seeing_is_
believing F</td></tr>
<tr><td>10 :</td><td>sp|=players(pete, stone)</td><td>CTXT 3: [3]</td><td>child F</td></tr>
<tr><td>11 :</td><td>sz |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)∨ players(py, px))
∧ ∃ x, y (better(x, y) ∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py)))))))</td><td>[6, 7]</td><td>all_detachment F</td></tr>
<tr><td>12 :</td><td>sz|=bel(jack, sp|=hand(stone, sh))</td><td>CTXT 2: [4]</td><td>child F</td></tr>
<tr><td>13 :</td><td>sz|=bel(jack, sp|=hand(pete, ph))</td><td>CTXT 2: [5]</td><td>child F</td></tr>
<tr><td>14 :</td><td>sz|= bel(jack, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)
∨ players(py, px))∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py))))))</td><td>CTXT 2: [7]</td><td>child F</td></tr>
<tr><td>15 :</td><td>∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)
∨ players(py, px))∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py)))))</td><td>[14]</td><td>bel_veridicality F</td></tr>
<tr><td>16 :</td><td>sz|=bel(jack, sp|=(calls(stone)⇒wins(stone)))
∧bel(jack, sp|=(calls(pete)⇒loses(pete)))</td><td>[15, 8, 7,
10, 13, 12, 5]</td><td>all_detachment F</td></tr>
<tr><td>17 :</td><td>sz|=bel(jack, sp|=loses(pete))</td><td>CTXT 2: [10]</td><td>child F</td></tr>
</table>

Exhibit 6. 4: Jack’s Proof, part 2: Context 1.

This proof is interesting in that FELIX moves back and forth (or up and down, perhaps) between the various intensional contexts. FELIX does considerably more of this shifting between intensional contexts in its search for the proof than is present in

CONTEXT 2 : sz|=Formula

Step :	*Adopted Formula*	*Support Steps*	*Justification*
1 :	bel(jack, sp|=calls(pete))	CTXT 1: [1]	parent F
2 :	knows_poker(jack)	CTXT 1: [3]	parent F
3 :	bel(jack, sp|=hand(stone, sh)/\hand(pete, ph) /\players(pete, stone))	CTXT 1: [9]	parent F
4 :	bel(jack, sp|=hand(stone, sh))	CTXT 4: [3]	child F
5 :	bel(jack, sp|=hand(pete, ph))	CTXT 4: [4]	child F
6 :	∀ *a*(knows_poker(*a*) ⇒ bel(*a*, ∀ *t*, *u*(sit(*t*)∧sit(*u*) → ∀ *p*, *px*, *py* ((*u* |=players(*px*, *py*)∨ players(*py*, *px*)) ∧ ∃ *x*, *y* (better(*x*, *y*) ∧ (*t* |=bel(*p*, *u* |=hand(*px*, *x*)∧hand(*py*, *y*)))) → *t* |= bel(*p*, *u* |=(calls(*px*)⇒wins(*px*))) ∧bel(*p*, *u* |=(calls(*py*)⇒loses(*py*)))))))	CTXT 1: [11]	parent F
7 :	bel(jack, ∀ *t*, *u*(sit(*t*)∧sit(*u*) → ∀ *p*, *px*, *py* ((*u* |=players(*px*, *py*)∨ players(*py*, *px*)) ∧ ∃ *x*, *y* (better(*x*, *y*) ∧ (*t* |=bel(*p*, *u* |=hand(*px*, *x*)∧hand(*py*, *y*)))) → *t* |= bel(*p*, *u* |=(calls(*px*)⇒wins(*px*))) ∧bel(*p*, *u* |=(calls(*py*)⇒loses(*py*))))))	[6, 2]	all_detachment F
8 :	bel(jack, sp|=(calls(stone)->wins(stone))) /\bel(jack, sp|=(calls(pete)->loses(pete)))	CTXT 1: [16]	parent F
9 :	bel(jack, sp|=(calls(pete)->loses(pete)))	[8]	conjunction(1) F
10 :	bel(jack, sp|=loses(pete))	CTXT 4: [6]	child F

Exhibit 6. 5: Jacks’ Proof, part 3: Context 2.

the final proof. However, even in the final proof there is some evidence of this behavior. One might expect to proceed in a fairly direct fashion where some steps are made in context 1, then reasoning “descends” to context 2, then descends again to context 3, then to context 4, then ascends back to context 1 and is done. This is not what happens in this proof.

The proof does start out in this fashion, where in steps 1 through 13 of context 1 the “given” formulae are adopted and some inferences are drawn from these formulae. The final step of context 1, step 14, adopts a formulae which is the final step of context 4. Thus, between steps 13 and 14 of context 1, the proof descends in some fashion to context 4. This descent goes first to context 2. In context 2, various formulae are adopted from context 1 and inferences are drawn from these formulae. Thus, all of the steps of context 2 are completed without reference to contexts 3 or 4. To continue the descent toward context 4, the proof next goes to context 3. This context

CONTEXT 3 : sp|=Formula

Step :Adopted Formula		*Support Steps*	*Justification*
1 :	hand(stone, sh)/\hand(pete, ph) /\players(pete, stone)	CTXT 1: [4]	parent F
2 :	hand(pete, ph)/\players(pete, stone)	[1]	conjunction(1) F
3 :	players(pete, stone)	[2]	conjunction(1) F

CONTEXT 4 : sz|=bel(jack, Formula)

1 :	sp|=calls(pete)	CTXT 2: [1]	parent F
2 :	sp|=hand(stone, sh)/\hand(pete, ph) /\players(pete, stone)	CTXT 2: [3]	parent F
3 :	sp|=hand(stone, sh)	CTXT 5: [4]	child F
4 :	sp|=hand(pete, ph)	CTXT 5: [5]	child F
5 :	sp|=(calls(pete)->loses(pete))	CTXT 2: [9]	parent F
6 :	sp|=loses(pete)	CTXT 5: [7]	child F

CONTEXT 5 : sz|=bel(jack, sp|=Formula)

1 :	calls(pete)	CTXT 4: [1]	parent F
2 :	hand(stone, sh)/\hand(pete, ph) /\players(pete, stone)	CTXT 4: [2]	parent F
3 :	hand(pete, ph)/\players(pete, stone)	[2]	conjunction(1) F
4 :	hand(stone, sh)	[2]	conjunction(2) F
5 :	hand(pete, ph)	[3]	conjunction(2) F
6 :	calls(pete)->loses(pete)	CTXT 4: [5]	parent F
7 :	loses(pete)	[6, 1]	modus_ponens F

Exhibit 6. 6: Jack’s Proof, part 4: Contexts 3, 4, and 5.

shows a more complex motion of the proof through the intensional contexts. Steps 1 through 12 of context 3 consist of adopting formulae from the parent intensional context (2) and making various inferences, in a fashion analogous to the steps of context 2. Step 13 of context 3 is an adoption of a formulae from context 4. Before this step can be made, the appropriate step of context 4 must be derived. So, at this point, before finishing context 3, the proof “thread” descends to context 4.

The proof proceeds through steps 1 through 13 of context 4 in the now familiar pattern of various adoptions from the steps already developed in the parent intensional

<table>
<tr><th>Step :</th><th>Adopted Formula</th><th>Support Steps</th><th>Justification</th></tr>
<tr><td>1 :</td><td>sz|=bel(jack, sp|=calls(pete))</td><td></td><td>given</td></tr>
<tr><td>2 :</td><td>sz|=sees(jack, sp)</td><td></td><td>given</td></tr>
<tr><td>3 :</td><td>sz|=knows_poker(jack)</td><td></td><td>given</td></tr>
<tr><td>4 :</td><td>sp|=hand(stone, sh)/\hand(pete, ph)/\players(pete, stone)</td><td></td><td>given</td></tr>
<tr><td>5 :</td><td>better(sh, ph)</td><td></td><td>given</td></tr>
<tr><td>6 :</td><td>∀ s (sit(s) → s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py) ∨ players(py, px)) ∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧ bel(p, u |= (calls(py) ⇒ loses(py))))))))</td><td></td><td>given</td></tr>
<tr><td>7 :</td><td>sit(sz)</td><td>[1]</td><td>support_sit F</td></tr>
<tr><td>8 :</td><td>sit(sp)</td><td>[4]</td><td>support_sit F</td></tr>
<tr><td>9 :</td><td>sz|=bel(jack, sp|=hand(stone, sh)
/\hand(pete, ph)/\players(pete, stone))</td><td>[2, 4]</td><td>seeing_is_
believing F</td></tr>
<tr><td>10 :</td><td>sp|=players(pete, stone)</td><td>CTXT 3: [3]</td><td>child F</td></tr>
<tr><td>11 :</td><td>sz |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)∨ players(py, px))
∧ ∃ x, y (better(x, y) ∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py)))))))</td><td>[6, 7]</td><td>all_detachment F</td></tr>
<tr><td>12 :</td><td>sz|=bel(jack, sp|=hand(stone, sh))</td><td>CTXT 5: [4] via 2&4, child F</td><td></td></tr>
<tr><td>13 :</td><td>sz|=bel(jack, sp|=hand(pete, ph))</td><td>CTXT 5: [5] via 2&4, child F</td><td></td></tr>
<tr><td>14 :</td><td>sz|= bel(jack, ∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)
∨ players(py, px))∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py))))))</td><td>CTXT 2: [7]</td><td>child F</td></tr>
<tr><td>15 :</td><td>∀ t, u(sit(t)∧sit(u)
→ ∀ p, px, py ((u |=players(px, py)
∨ players(py, px))∧ ∃ x, y (better(x, y)
∧ (t |=bel(p, u |=hand(px, x)∧hand(py, y))))
→ t |= bel(p, u |=(calls(px)⇒wins(px)))
∧bel(p, u |=(calls(py)⇒loses(py)))))</td><td>[14]</td><td>bel_veridicality F</td></tr>
<tr><td>16 :</td><td>sz|=bel(jack, sp|=(calls(stone)->wins(stone)))
/\bel(jack, sp|=(calls(pete)->loses(pete)))</td><td>[15, 8, 7,
10, 13, 12, 5]</td><td>all_detachment F</td></tr>
<tr><td>17 :</td><td>sz|=bel(jack, sp|=loses(pete))</td><td>CTXT 5: [7] via 2&4, child F</td><td></td></tr>
</table>

Exhibit 6. 7: Jack’s Proof, part 2 Collapsed: Context 1.

context (3) and inferences made from those adoptions. Step 14 of context 4 is an adoption of a step in context 3 which has not yet been made, so the thread can’t pro-

CONTEXT 2 : sz|=Formula			
Step :	*Adopted Formula*	*Support Steps*	*Justification*
2 :	knows_poker(jack)	CTXT 1: [3]	parent F
6 :	$\forall$ *a*(knows_poker(*a*) $\Rightarrow$ bel(*a*, $\forall$ *t*, *u*(sit(*t*)$\wedge$sit(*u*) $\rightarrow \forall$ *p*, *px*, *py* ((*u* |=players(*px*, *py*)$\vee$ players(*py*, *px*)) $\wedge \exists$ *x*, *y* (better(*x*, *y*) $\wedge$ (*t* |=bel(*p*, *u* |=hand(*px*, *x*)$\wedge$hand(*py*, *y*)))) $\rightarrow$ *t* |= bel(*p*, *u* |=(calls(*px*)$\Rightarrow$wins(*px*))) $\wedge$bel(*p*, *u* |=(calls(*py*)$\Rightarrow$loses(*py*)))))))	CTXT 1: [11]	parent F
7 :	bel(jack, $\forall$ *t*, *u*(sit(*t*)$\wedge$sit(*u*) $\rightarrow \forall$ *p*, *px*, *py* ((*u* |=players(*px*, *py*)$\vee$ players(*py*, *px*)) $\wedge \exists$ *x*, *y* (better(*x*, *y*) $\wedge$ (*t* |=bel(*p*, *u* |=hand(*px*, *x*)$\wedge$hand(*py*, *y*)))) $\rightarrow$ *t* |= bel(*p*, *u* |=(calls(*px*)$\Rightarrow$wins(*px*))) $\wedge$bel(*p*, *u* |=(calls(*py*)$\Rightarrow$loses(*py*))))))	[6, 2]	all_detachment F
8 :	bel(jack, sp |=(calls(stone)$\Rightarrow$ wins(stone))) $\wedge$bel(jack, sp |= (calls(pete) $\Rightarrow$ loses(pete)))	CTXT 1: [16]	parent F
9 :	bel(jack, sp |= (calls(pete) $\Rightarrow$ loses(pete)))	[8]	conjunction(1) F

Exhibit 6. 8: Jacks' Proof, part 3 collapsed: Context 2.

ceed (yet) in context 4. Context 3 was "suspended" at step 13 waiting to adopt step 13 of context 4, which has now been made, so the proof can proceed in context 3.

The proof proceeds in context 3 by making the adoption from context 4 indicated in step 13, then making the inference of step 14 of context 3, thereby completing context 3. The proof of context 4 was suspended waiting for step 14 of context 3 to be made, so the proof of context 4 can now continue with the adoption from context 3 and the final inference, thereby completing the proof of context 4. This derives the step of context 4 on which context 1 was suspended, allowing the proof in context 1 to make its final adoption, thereby completing the entire proof.

The Wise Men

The proof of the wise men puzzle is quite different from that of the Poker Game. The essential distinction is that for the wise men puzzle it is necessary to use *reductio ad absurdam* reasoning, while this is not the case in the Poker Game problems. Since *reductio ad absurdam* reasoning is only used as a last resort by FELIX, FELIX does

CONTEXT 3 : sp|=Formula

Step :	Adopted Formula	Support Steps	Justification
1 :	hand(stone, sh)/\hand(pete, ph) /\players(pete, stone)	CTXT 1: [4]	parent F
2 :	hand(pete, ph)/\players(pete, stone)	[1]	conjunction(1) F
3 :	players(pete, stone)	[2]	conjunction(1) F

CONTEXT 4 : sz |=bel(jack, Formula)

CONTEXT 5 : sz |=bel(jack, sp |=Formula)

1 :	calls(pete)	CTXT 1: [1] via 4&2, parent F	
2 :	hand(stone, sh)/\hand(pete, ph) /\players(pete, stone)	CTXT 1: [9] via 4&2, parent F	
3 :	hand(pete, ph)/\players(pete, stone)	[2]	conjunction(1) F
4 :	hand(stone, sh)	[2]	conjunction(2) F
5 :	hand(pete, ph)	[3]	conjunction(2) F
6 :	calls(pete)->loses(pete)	CTXT 2: [9] via 4, parent F	
7 :	loses(pete)	[6, 1]	modus_ponens F

Exhibit 6. 9: Jack’s Proof, part 4 collapsed: Contexts 3, 4, and 5.

a lot of fruitless searching in looking for a proof of this puzzle, and the proof it finally generates is much longer than it needs to be. Further, the ‘sit(S)’ facts which FELIX generates via forward reasoning are useless in this puzzle and litter the agenda of tasks. To streamline this process somewhat, the dilemma reasoning is removed for processing this puzzle, as is the ‘sit(S)’ generating forward reason. No other modifications are made to FELIX to achieve the given proof. The statement of the theorem and the proof steps for the theorem are in Exhibit 6. 15 on page 185. The proof of this theorem uses a lemma, “s1”, which is where the *reductio ad absurdam* reasoning is carried out. This lemma is in Exhibit 6. 15 on page 185. The proof of lemma s1 relies on *indirect* contradiction, where “Γ, P |- Q /\ – Q” implies “Γ |- – Q”, for any Γ, P and Q. *Direct* contradiction is “Γ, P |- – P” implies “– P”. In lemma s1, this rule is applied in step 10 of the context 3. Context 3 is the collection of formulae which ‘s’ supports that ‘a’ believes. Context 3 is the *reductio intensional context*, the intensional context in which *reductio ad absurdam* reasoning can be used, since it is the intensional context in which the negated formula is “supposed”. The supposition

PROVE: sj |= bel(zack, sp |= wins(pete))

GIVEN: sj |= bel(zack, sp |= calls(pete))

sj |= bel(zack, sp |= bel(pete, sp |= hand(stone, sh)))

sj |= bel(zack, sp |= players(pete, stone))

sj |= knows_poker(zack)

$\forall$ *s*(sit(*s*)$\rightarrow$ *s* |= $\forall$ *a*(knows_poker(*a*)
$\Rightarrow$ bel(*a*, $\forall$ *t*(sit(*t*)
$\rightarrow$ *t* |= $\forall$ *p1*, *p2*(players(*p1*, *p2*)$\Rightarrow$player(*p1*)$\wedge$player(*p2*))))))

$\forall$ *s*(sit(*s*)$\rightarrow$ *s* |= $\forall$ *a*(knows_poker(*a*)
$\Rightarrow$ bel(*a*, $\forall$ *t*(sit(*t*)$\rightarrow$ *t* |= $\forall$ *p*(player(*p*)$\Rightarrow$ $\exists$ *x* bel(*p*, *t* |=hand(*p*, *x*)))))))

$\forall$ *s*(sit(*s*)$\rightarrow$ *s* |= $\forall$ *a*(knows_poker(*a*)
$\Rightarrow$ bel(*a*, $\forall$ *t*(sit(*t*)
$\rightarrow$ *t* |= $\forall$ *p1*, *p2*(players(*p1*, *p2*)$\vee$players(*p2*, *p1*)
$\Rightarrow$ $\exists$ *x*, *y* bel(*p1*, *t* |= hand(*p1*, *x*)$\wedge$hand(*p2*, *y*))
$\Rightarrow$ bel(*p1*, *t* |=(calls(*p1*)$\Rightarrow$wins(*p1*)))
$\vee$bel(*p1*, *t* |= (calls(*p1*)$\Rightarrow$loses(*p1*))))))))

$\forall$ *s*(sit(*s*)$\rightarrow$ *s* |= $\forall$ *a*(knows_poker(*a*)
$\Rightarrow$ bel(*a*, $\forall$ *t*(sit(*t*)
$\rightarrow$ *t* |= $\forall$ *x*(calls(*x*)$\Rightarrow$ – bel(*x*, *t* |=(calls(*x*)$\Rightarrow$loses(*x*))))))))

Exhibit 6. 10: Zack’s Proof, part 1: Problem Statement.

is ‘s |= bel(a, – (s |= white(a)))’. The negated formula (P) in context 3 is ‘– (s |= white(a))’. The other accepted formulae (Γ) are those that were initially given in the main theorem as they are “projected” into context 3. The contradicted formulae (Q ∧ – Q) are ‘s |= – bel(b, s |= white(b))’ and ‘– (s |= – bel(b, s |= white(b)))’ from steps 5 and 9 of context 3. Thus, the formula derived from the contradiction (– P) is ‘s |= white(a)’, which is step 10 of context 3. This expands in the root intensional context to the “to be proved” formula ‘s |= bel(a, s |= white(a))’.

The *reductio* argument takes place in the intensional context in which one is trying to derive a formula. In this case, that is context 3. It is possible to derive a contradiction in the root intensional context, between ‘s |= – bel(b, s |= white(b))’ and ‘– (s |= – bel(b, s |= white(b)))’. However, this does not allow the derivation of the “proof”

<table>
<tr><th>Step :</th><th>Adopted Formula</th><th>Support Steps</th><th>Justification</th></tr>
<tr><td>1 :</td><td>sj|=bel(zack, sp|=calls(pete))</td><td>given</td><td>input</td></tr>
<tr><td>2 :</td><td>sj|=bel(zack, sp|=bel(pete,
sp|=hand(stone, sh)))</td><td>given</td><td>input</td></tr>
<tr><td>3 :</td><td>sj|=bel(zack,
sp|=players(pete, stone))</td><td>given</td><td>input</td></tr>
<tr><td>4 :</td><td>sj|=knows_poker(zack)</td><td>given</td><td>input</td></tr>
<tr><td>5 :</td><td>∀ s(sit(s)→ s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2)
⇒player(p1)∧player(p2))))))</td><td>given</td><td>input</td></tr>
<tr><td>6 :</td><td>∀ s(sit(s)→ s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)→ t |= ∀ p(player(p)
⇒ ∃ x bel(p, t |=hand(p, x)))))))</td><td>given</td><td>input</td></tr>
<tr><td>7 :</td><td>∀ s(sit(s)→ s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)
→ t |= ∀ p1, p2(players(p1, p2)∨players(p2, p1)
⇒ ∃ x, y bel(p1, t |= hand(p1, x)∧hand(p2, y))
⇒ bel(p1, t |=(calls(p1)⇒wins(p1)))
∨bel(p1, t |= (calls(p1)⇒loses(p1))))))))</td><td>given</td><td>input</td></tr>
<tr><td>8 :</td><td>∀ s(sit(s)→ s |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)→ t |= ∀ x(calls(x)
⇒ – bel(x, t |=(calls(x)⇒loses(x))))))))</td><td>given</td><td>input</td></tr>
<tr><td>9 :</td><td>sit(sj)</td><td>[1]</td><td>support_sit F</td></tr>
<tr><td>10 :</td><td>sj |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2)
⇒player(p1)∧player(p2))))))</td><td>[5, 9]</td><td>all_detachment F</td></tr>
<tr><td>11 :</td><td>sj |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)→ t |= ∀ p(player(p)
⇒ ∃ x bel(p, t |=hand(p, x)))))))</td><td>[6, 9]</td><td>all_detachment F</td></tr>
<tr><td>12 :</td><td>sj |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)
→ t |= ∀ p1, p2(players(p1, p2)∨players(p2, p1)
⇒ ∃ x, y bel(p1, t |= hand(p1, x)∧hand(p2, y))
⇒ bel(p1, t |=(calls(p1)⇒wins(p1)))
∨bel(p1, t |= (calls(p1)⇒loses(p1))))))))</td><td>[7, 9]</td><td>all_detachment F</td></tr>
<tr><td>13 :</td><td>sj |= ∀ a(knows_poker(a)
⇒ bel(a, ∀ t(sit(t)→ t |= ∀ x(calls(x)
⇒ – bel(x, t |=(calls(x)⇒loses(x))))))))</td><td>[8, 9]</td><td>all_detachment F</td></tr>
<tr><td>14 :</td><td>sj |= bel(zack, sp |= wins(pete))</td><td>CTXT 4: [15]</td><td>via 2&3; child F</td></tr>
</table>

Exhibit 6. 11: Zack’s Proof, part 2: Context 1.

goal. All it warrants is the derivation of the negation of the supposition formula *as projected into that context*. Since this is the root intensional context, the projection is the identity operation and the negated formula one can derive is ‘– s |= bel(a, – (s |=

Step :	Adopted Formula	Support Steps	Justification
4 :	knows_poker(zack)	CTXT 1: [4]	parent F
5 :	∀ a(knows_poker(a) ⇒ bel(a, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2) ⇒player(p1)∧player(p2))))))	CTXT 1: [10]	parent F
6 :	bel(zack, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2) ⇒player(p1)∧player(p2))))	[5, 4]	all_detachment F
7 :	∀ a(knows_poker(a) ⇒ bel(a, ∀ t(sit(t)→ t |= ∀ p(player(p) ⇒ ∃ x bel(p, t |=hand(p, x))))))	CTXT 1: [11]	parent F
8 :	bel(zack, ∀ t(sit(t)→ t |= ∀ p(player(p) ⇒ ∃ x bel(p, t |=hand(p, x)))))	[7, 4]	all_detachment F
9 :	∀ a(knows_poker(a) ⇒ bel(a, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2)∨players(p2, p1) ⇒ ∃ x, y bel(p1, t |= hand(p1, x)∧hand(p2, y)) ⇒ bel(p1, t |=(calls(p1)⇒wins(p1))) ∨bel(p1, t |= (calls(p1)⇒loses(p1)))))))	CTXT 1: [12]	parent F
10 :	bel(zack, ∀ t(sit(t) → t |= ∀ p1, p2(players(p1, p2)∨players(p2, p1) ⇒ ∃ x, y bel(p1, t |= hand(p1, x)∧hand(p2, y)) ⇒ bel(p1, t |=(calls(p1)⇒wins(p1))) ∨bel(p1, t |= (calls(p1)⇒loses(p1)))))))	[9, 4]	all_detachment F
11 :	∀ a(knows_poker(a) ⇒ bel(a, ∀ t(sit(t)→ t |= ∀ x(calls(x) ⇒ – bel(x, t |=(calls(x)⇒loses(x))))))	CTXT 1: [13]	parent F
12 :	bel(zack, ∀ t(sit(t)→ t |= ∀ x(calls(x) ⇒ – bel(x, t |=(calls(x)⇒loses(x))))))	[11, 4]	all_detachment F

Exhibit 6. 12: Zack’s Proof, part 3: Context 2.

white(a)))’. This formula does not (directly) warrant the conclusion that ‘s |= bel(a, s |= white(a))’. So, it is necessary to the *reductio* argument in the “interesting” intensional context, which in this case is context 3, where negating the supposition *as projected into that intensional context* yields the desired result.

The second wise men problem involves a theorem that given that *s* supports that B does not believe that *s* supports that B has a white dot, that B knows that either A or B has a white dot, and that B believes that A has a white dot or B believes that A doesn’t have a white dot (since B can see A), prove that B believes A has a white dot. As noted in the previous chapter, proving this theorem involves several belief

Step :	*Adopted Formula*	*Support Steps*	*Justification*
1 :	$sp \models calls(pete)$	CTXT 2: [1]	parent F
2 :	$sit(sp)$	[1]	support_sit F
5 :	$\forall t(sit(t) \rightarrow t \models \forall p1, p2(players(p1, p2)$ $\Rightarrow player(p1) \wedge player(p2))))$	CTXT 2: [6]	parent F
6 :	$sp \models \forall p1, p2(players(p1, p2)$ $\Rightarrow player(p1) \wedge player(p2))))$	[5, 2]	all_detachment F
7 :	$\forall t(sit(t) \rightarrow t \models \forall p(player(p)$ $\Rightarrow \exists x\ bel(p, t \models hand(p, x)))))$	CTXT 2: [8]	parent F
8 :	$sp \models \forall p(player(p)$ $\Rightarrow \exists x\ bel(p, sp \models hand(p, x)))$	[7, 2]	all_detachment F
9 :	$\forall t(sit(t)$ $\rightarrow t \models \forall p1, p2(players(p1, p2) \vee players(p2, p1)$ $\Rightarrow \exists x, y\ bel(p1, t \models hand(p1, x) \wedge hand(p2, y))$ $\Rightarrow bel(p1, t \models (calls(p1) \Rightarrow wins(p1)))$ $\vee bel(p1, t \models (calls(p1) \Rightarrow loses(p1))))$	CTXT 2: [10]	parent F
10 :	$sp \models \forall p1, p2(players(p1, p2) \vee players(p2, p1)$ $\Rightarrow \exists x, y\ bel(p1, sp \models hand(p1, x) \wedge hand(p2, y))$ $\Rightarrow bel(p1, sp \models (calls(p1) \Rightarrow wins(p1)))$ $\vee bel(p1, sp \models (calls(p1) \Rightarrow loses(p1))))$	[9, 2]	all_detachment F
11 :	$\forall t(sit(t) \rightarrow t \models \forall x(calls(x)$ $\Rightarrow - bel(x, t \models (calls(x) \Rightarrow loses(x)))))$	CTXT 2: [12]	parent F
12 :	$sp \models \forall x(calls(x)$ $\Rightarrow - bel(x, sp \models (calls(x) \Rightarrow loses(x))))$	[11, 2]	all_detachment F
13 :	$sp \models bel(pete, sp \models (calls(pete) \Rightarrow wins(pete)))$	CTXT 4: [13]	child F
14 :	$sp \models (calls(pete) \Rightarrow wins(pete))$	[13]	bel_veridicality F

Exhibit 6. 13: Zack's Proof, part 4: Context 3.

principles. This is the only example proof which involves either positive or negative introspection, and it uses them both. The proof is given in Exhibit 6. 16 on page 186.

The positive introspection principle of belief is implicit in the *reductio* step. If B believes P, then B believes that he believes P. Thus, it is contradictory in the intensional context of B's beliefs (context 3) for B to believe P and to believe that he doesn't believe P. The veridicality principle is used in reasoning backwards - this application of the rule being called "bel_anti_veridicality". Logical closure is the principle behind the use of classical mode logic for belief intensional contexts. The negative introspection principle appears directly as a justification in the proof.

<table>
<tr><th>Step :</th><th>Adopted Formula</th><th>Support Steps</th><th>Justification</th></tr>
<tr><td>1 :</td><td>calls(pete)</td><td>CTXT 3: [1]</td><td>parent F</td></tr>
<tr><td>2 :</td><td>bel(pete, sp|=hand(stone, sh))</td><td>CTXT 1: [2] via 2&3;</td><td>parent F</td></tr>
<tr><td>3 :</td><td>players(pete, stone)</td><td>CTXT 1: [3] via 2&3;</td><td>parent F</td></tr>
<tr><td>4 :</td><td>∀ p1, p2(players(p1, p2)
⇒player(p1)∧player(p2))))</td><td>CTXT 3: [6]</td><td>parent F</td></tr>
<tr><td>5 :</td><td>player(pete)</td><td>[4, 3]</td><td>all_detachment F</td></tr>
<tr><td>6 :</td><td>∀ p(player(p) ⇒ ∃ x bel(p, sp |=hand(p, x)))</td><td>CTXT 3: [8]</td><td>parent F</td></tr>
<tr><td>7 :</td><td>– ∀ x – bel(pete, sp |=hand(pete, x)))</td><td>[6, 5]</td><td>all_detachment F</td></tr>
<tr><td>8 :</td><td>∃ x bel(pete, sp |=hand(pete, x)))</td><td>[7]</td><td>negated_universal</td></tr>
<tr><td>9 :</td><td>bel(pete, sp|=hand(pete, xa1@))</td><td>[8]</td><td>existential_ instantiation F</td></tr>
<tr><td>10 :</td><td>∀ p1, p2(players(p1, p2)∨players(p2, p1)
⇒ ∃ x, y bel(p1, sp |= hand(p1, x)∧hand(p2, y))
⇒ bel(p1, sp |=(calls(p1)⇒wins(p1)))
∨bel(p1, sp |= (calls(p1)⇒loses(p1))))</td><td>CTXT 3: [10]</td><td>parent F</td></tr>
<tr><td>11 :</td><td>∀ x(calls(x)
⇒ – bel(x, sp |=(calls(x)⇒loses(x))))</td><td>CTXT 3: [12]</td><td>parent F</td></tr>
<tr><td>12 :</td><td>– bel(pete, sp |=(calls(pete)⇒loses(pete)))</td><td>[11, 1]</td><td>all_detachment F</td></tr>
<tr><td>13 :</td><td>bel(pete, sp|=(calls(pete)⇒wins(pete)))</td><td>[10, 12, 9, 2, 3]</td><td>all_detachment F</td></tr>
<tr><td>14 :</td><td>calls(pete)⇒wins(pete)</td><td>CTXT 3: [14]</td><td>parent F</td></tr>
<tr><td>15 :</td><td>wins(pete)</td><td>[14, 1]</td><td>modus_ponens F</td></tr>
</table>

Exhibit 6. 14: Zack’s Proof, part 5: Context 4.

PROVE: s |= bel(a, s |= white(a))

GIVEN: s |= bel(a, s |= (– white(a) ⇒ bel(b, s |= – white(a))))
s |= bel(a, s |= bel(b, s |= white(a) ∨ white(b)))
s |= bel(a, s |= white(a) ∨ – white(a))
s |= bel(a, s |= – bel(b, s |= white(b)))

Step :	*Adopted Formula*	*Support Steps*	*Justification*
1 :	s |= bel(a, s |= white(a))	CTXT 3: [1] via 2; child B	

CONTEXT 3 : s |= bel(a, Formula)

1 :	s |= white(a)	LEMMA s1	reductio_direct B

LEMMA s1

PROVE: s |= bel(a, s |= white(a))
SUPPOSE: s |= bel(a, – (s |= white(a)))

6 :	s |= bel(a, s |= white(a))	CTXT 3: [10] via 2; child B	

CONTEXT 2 : s |= Formula
CONTEXT 3 : s |= bel(a, Formula)

1 :	– (s |= white(a)	input via CTXT 1&2	
4 :	s |= white(a)∨ – white(a)	input via CTXT 1&2	
5 :	s |= – bel(b, s |= white(b))	input via CTXT 1&2	
6 :	(s |= white(a)) ∨ (s |= – white(a))	[4]	support_ disjunction1 F
7 :	s |= – white(a)	[6, 1]	disjunction_and_ negation F
8 :	s |= bel(b, s |= white(b))	CTXT 4: [5]	child F
9 :	– (s |= – bel(b, s |= white(b)))	[8]	support_strong_ negation B
10 :	s |= white(a)	[5, 9]	reductio_indirect B

CONTEXT 4 : s |= bel(a, s |= Formula)

1 :	– white(a)⇒bel(b, s |= – white(a))	given via CTXT 1, 2&3; parent F	
3 :	– white(a)	CTXT 3: [7]	parent F
4 :	bel(b, s |= – white(a))	[1,3]	modus_ponens F
5 :	bel(b, s |= white(b))	CTXT 6: [3]	child F

CONTEXT 6 : s |= bel(a, s |= bel(b, Formula))
CONTEXT 7 : s |= bel(a, s |= bel(b, s |= Formula))

1 :	white(a)∨white(b)	given via CTXT 1, 2,3;&6; parent F	
2 :	– white(a)	CTXT 4: [4] via 6; parent F	
3 :	white(b)	[1, 2]	disjunction_and_ negation F

Exhibit 6. 15: Two Wise Men Proof

PROVE: s |= bel(b, s |= white(a))
GIVEN: s |= bel(b, s |= white(a) ∨ white(b))
s |= bel(b, s |= white(a)) ∨ bel(b, s |= – white(a))
s |= – bel(b, s |= white(b))

Step :Adopted Formula		*Support Steps*	*Justification*
1 :	s |= bel(b, s |= white(a))	CTXT 3: [1] via 2; child F	

CONTEXT 2 : s |=Formula
CONTEXT 3 : s |=bel(b, Formula)

1 :	s |= white(a)	LEMMA s1	reductio_direct B

LEMMA s1

PROVE: s |= bel(b, s |= white(a))
SUPPOSE: s |= bel(b, –(s |= white(a)))

1 :	s |= bel(b, –(s |= white(a)))	input	
3 :	s |= bel(b, s |= white(a)) ∨ bel(b, s |= – white(a))		input
4 :	s |= – bel(b, s |= white(b))	input	
5 :	– (s |= white(a))	[1]	bel_veridicality F
6 :	(s |= bel(b, s |= white(a))) ∨ (s |= bel(b, s |= – white(a)))	[3]	support_ disjunction1 F
7 :	– (s |= bel(b, s |= white(a)))	[5]	bel_anti_ veridicality B
8 :	(s |= bel(b, s |= – white(a)))	[7, 6]	disjunction_and_ negation B
9 :	s |= bel(b, s |= – bel(b, s |= white(b))	[4]	bel_negative_ introspection B
10 :	s |= bel(b, s |= white(a))	CTXT 3: [6] via 2; child F	

CONTEXT 2 : s |=Formula
CONTEXT 3 : s |=bel(b, Formula)

3 :	s |= white(b)	CTXT 4: [3]	child F
4 :	s |= – bel(b, s |= white(b))	CTXT 1: [9] via 2; parent F	
5 :	– (s |= bel(b, s |= white(b)))	[4]	support_strong_ negation B
6 :	s |= white(a)	[3, 5]	reductio_indirect B

CONTEXT 4 : s |=bel(b, s |=Formula)

1 :	– white(a)	CTXT 1: [8]via 2&3; parent F	
2 :	white(a) ∨ white(b)	input via 1,2&3; parent F	
3 :	white(b)	[2, 1]	disjunction_and_ negation F

Exhibit 6. 16: Wise Men Introspection Proof