PUZ axioms

PUZ001-0.ax Mars and Venus axioms
from_mars(x) or from_venus(x) cnf(from_mars_or_venus, axiom)
from_mars(x) $\Rightarrow$ $\neg$ from_venus(x) cnf(not_from_mars_and_venus, axiom)
male(x) or female(x) cnf(male_or_female, axiom)
male(x) $\Rightarrow$ $\neg$ female(x) cnf(not_male_and_female, axiom)
truthteller(x) or liar(x) cnf(truthteller_or_liar, axiom)
truthteller(x) $\Rightarrow$ $\neg$ liar(x) cnf(not_truthteller_and_liar, axiom)
(says(x, y) and a_truth(y)) $\Rightarrow$ a_truth(y) cnf(statements_are_true_or_not, axiom)
says(x, y) $\Rightarrow$ y = statement_by(x) cnf(people_say_their_statements, axiom)
a_truth(statement_by(x)) $\Rightarrow$ truthteller(x) cnf(true_statements_made_by_truthtellers, axiom)
a_truth(statement_by(x)) or liar(x) cnf(false_statements_made_by_liars, axiom)
(from_venus(x) and female(x)) $\Rightarrow$ truthteller(x) cnf(venusian_female_are_truthtellers, axiom)
(from_venus(x) and male(x)) $\Rightarrow$ liar(x) cnf(venusian_males_are_liars, axiom)
(from_mars(x) and male(x)) $\Rightarrow$ truthteller(x) cnf(marsian_males_are_truthtellers, axiom)
(from_mars(x) and female(x)) $\Rightarrow$ liar(x) cnf(marsian_females_are_liars, axiom)
(truthteller(x) and says(x, y)) $\Rightarrow$ a_truth(y) cnf(truthtellers_make_true_statements, axiom)
(liar(x) and says(x, y)) $\Rightarrow$ $\neg$ a_truth(y) cnf(liars_make_false_statements, axiom)

PUZ002-0.ax Truthtellers and Liars axioms for two types of people
Axioms for two types of people; truthtellers and liars.
a_truth(truthteller(x)) or a_truth(liar(x)) cnf(truthteller_or_liar, axiom)
a_truth(truthteller(x)) $\Rightarrow$ $\neg$ a_truth(liar(x)) cnf(not_both, axiom)
(a_truth(truthteller(truthteller)) and a_truth(says(truthteller, statement))) $\Rightarrow$ a_truth(statement) cnf(truthtellers_tell_tr
(a_truth(liar(liar)) and a_truth(says(liar, statement))) $\Rightarrow$ $\neg$ a_truth(statement) cnf(liars_lie, axiom)
(a_truth(statement) and a_truth(says(truthteller, statement))) $\Rightarrow$ a_truth(truthteller(truthteller)) cnf(truths_are_told_by
a_truth(says(liar, statement)) $\Rightarrow$ (a_truth(statement) or a_truth(liar(liar))) cnf(liars_are_told_by_liars, axiom)

PUZ003-0.ax Truthtellers and Liars axioms for three types of people
Axioms for three types of people; truthtellers, liars and normal people.
a_truth(truthteller(x)) or a_truth(liar(x)) or a_truth(normal(x)) cnf(person_is_one_type, axiom)
a_truth(truthteller(x)) $\Rightarrow$ $\neg$ a_truth(normal(x)) cnf(not_truthteller_and_normal, axiom)
a_truth(truthteller(x)) $\Rightarrow$ $\neg$ a_truth(liar(x)) cnf(not_truthteller_and_liar, axiom)
a_truth(liar(x)) $\Rightarrow$ $\neg$ a_truth(normal(x)) cnf(not_liar_and_normal, axiom)
(a_truth(truthteller(x)) and a_truth(says(x, y))) $\Rightarrow$ a_truth(y) cnf(truthtellers_tell_truth, axiom)
(a_truth(liar(x)) and a_truth(says(x, y))) $\Rightarrow$ $\neg$ a_truth(y) cnf(liars_lie, axiom)
(a_truth(x) and a_truth(says(y, x))) $\Rightarrow$ (a_truth(truthteller(y)) or a_truth(normal(y))) cnf(truthtellers_and_normal_tell_t
a_truth(says(y, x)) $\Rightarrow$ (a_truth(x) or a_truth(liar(y)) or a_truth(normal(y))) cnf(liars_and_normal_lie, axiom)

PUZ problems

PUZ001+1.p Dreadbury Mansion
Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Agatha is not the butler. Therefore : Agatha killed herself.
$\exists x$: (lives(x) and killed(x, agatha)) fof(pel55$_1$, axiom)
lives(agatha) fof(pel55_2$_1$, axiom)
lives(butler) fof(pel55_2$_2$, axiom)
lives(charles) fof(pel55_2$_3$, axiom)
$\forall x$: (lives(x) $\Rightarrow$ (x = agatha or x = butler or x = charles)) fof(pel55$_3$, axiom)
$\forall x, y$: (killed(x, y) $\Rightarrow$ hates(x, y)) fof(pel55$_4$, axiom)
$\forall x, y$: (killed(x, y) $\Rightarrow$ $\neg$ richer(x, y)) fof(pel55$_5$, axiom)
$\forall x$: (hates(agatha, x) $\Rightarrow$ $\neg$ hates(charles, x)) fof(pel55$_6$, axiom)
$\forall x$: ($x \neq$ butler $\Rightarrow$ hates(agatha, x)) fof(pel55$_7$, axiom)
$\forall x$: ($\neg$ richer(x, agatha) $\Rightarrow$ hates(butler, x)) fof(pel55$_8$, axiom)
$\forall x$: (hates(agatha, x) $\Rightarrow$ hates(butler, x)) fof(pel55$_9$, axiom)
$\forall x$: $\exists y$: $\neg$ hates(x, y) fof(pel55$_{10}$, axiom)

agatha $\neq$ butler fof(pel55_{11}, axiom)
killed(agatha, agatha) fof(pel_{55}, conjecture)

PUZ001+2.p Dreadbury Mansion
Someone who lives in DreadburyMansion kills AuntAgatha. If somebody X lives in DreadburyMansion then X is AuntAgatha or X is the Butler or X is Charles. Everyone hates everyone that he kills. Noone is richer than someone that he kills. Charles hates noone who is hated by AuntAgatha. AuntAgatha does not hate the Butler. Everyone that is not the Butler is hated by AuntAgatha. The Butler hates everyone who is not richer than AuntAgatha. The Butler hates everyone who is hated by AuntAgatha. Noone hates everyone. AuntAgatha is not the Butler. Therefore, AuntAgatha kills AuntAgatha.
$\exists a, b, c$: ($true and $\text{predicate}_1(b, \text{live}, a)$ and modifier_pp(b, in, 'DreadburyMansion') and $\text{predicate}_2(c, \text{kill}, a,$ 'AuntAgatha') a
($d =$ 'AuntAgatha' or $d =$ 'Butler' or $d =$ 'Charles')) and $\forall f$: ($true $\Rightarrow$ $\forall g, h$: (($true and $\text{predicate}_2(h, \text{kill}, f, g)$) $\Rightarrow$
$\exists i$: $\text{predicate}_2(i, \text{hate}, f, g)$))) and $\forall j$: ($true $\Rightarrow$ $\neg\, \exists k, l, m$: ($true and $\text{predicate}_2(l, \text{kill}, j, k)$ and $\text{property}_2(m, \text{rich}, \text{comp\_than}$
m)) and $\forall n, o$: (($true and $\text{predicate}_2(o, \text{hate}, \text{'AuntAgatha'}, n)$) $\Rightarrow$ $\neg\, \exists p$: $\text{predicate}_2(p, \text{hate}, \text{'Charles'}, n)$) and $\neg\, \exists q$: predicat
'Butler') $\Rightarrow$ $\exists s$: $\text{predicate}_2(s, \text{hate}, \text{'AuntAgatha'}, r)$) and $\forall t$: (($true and $\neg\, \exists u$: ($\text{property}_2(u, \text{rich}, \text{comp\_than}, \text{'AuntAgatha'})$
u)) $\Rightarrow$ $\exists v$: $\text{predicate}_2(v, \text{hate}, \text{'Butler'}, t)$) and $\forall w, x$: (($true and $\text{predicate}_2(x, \text{hate}, \text{'AuntAgatha'}, w)$) $\Rightarrow$ $\exists y$: $\text{predicate}_2(y,$ l
$\neg\, \forall a_1$: ($true $\Rightarrow$ $\exists b_1$: $\text{predicate}_2(b_1, \text{hate}, z, a_1)$)) and 'AuntAgatha' $\neq$ 'Butler') fof(background, axiom)
$\exists a$: $\text{predicate}_2(a, \text{kill}, \text{'AuntAgatha'}, \text{'AuntAgatha'})$ fof(prove, conjecture)

PUZ001-1.p Dreadbury Mansion
Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Agatha is not the butler. Therefore : Agatha killed herself.
lives(agatha) cnf(agatha, hypothesis)
lives(butler) cnf(butler, hypothesis)
lives(charles) cnf(charles, hypothesis)
killed(x, y) $\Rightarrow$ $\neg$ richer(x, y) cnf(poorer_killer, hypothesis)
hates(agatha, x) $\Rightarrow$ $\neg$ hates(charles, x) cnf(different_hates, hypothesis)
(hates(x, agatha) and hates(x, butler)) $\Rightarrow$ $\neg$ hates(x, charles) cnf(no_one_hates_everyone, hypothesis)
hates(agatha, agatha) cnf(agatha_hates_agatha, hypothesis)
hates(agatha, charles) cnf(agatha_hates_charles, hypothesis)
killed(x, y) $\Rightarrow$ hates(x, y) cnf(killer_hates_victim, hypothesis)
hates(agatha, x) $\Rightarrow$ hates(butler, x) cnf(same_hates, hypothesis)
lives(x) $\Rightarrow$ (richer(x, agatha) or hates(butler, x)) cnf(butler_hates_poor, hypothesis)
killed(butler, agatha) or killed(charles, agatha) cnf(prove_neither_charles_nor_butler_did_it, negated_conjecture)

PUZ001-2.p Dreadbury Mansion
Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Agatha is not the butler. Therefore : Agatha killed herself.
lives_at_dreadsbury(someone) cnf(someone_in_mansion, axiom)
killed(someone, aunt_agatha) cnf(someone_killed_agatha, axiom)
lives_at_dreadsbury(aunt_agatha) cnf(agatha_lives_at_mansion, axiom)
lives_at_dreadsbury(butler) cnf(butler_lives_at_mansion, axiom)
lives_at_dreadsbury(charles) cnf(charles_lives_at_mansion, axiom)
lives_at_dreadsbury(person) $\Rightarrow$ (person = aunt_agatha or person = butler or person = charles) cnf(noone_else_lives_at_m
killed(killer, victim) $\Rightarrow$ hates(killer, victim) cnf(killer_hates_victim, axiom)
killed(killer, victim) $\Rightarrow$ $\neg$ richer(killer, victim) cnf(killer_poorer_than_victim, axiom)
hates(aunt_agatha, person) $\Rightarrow$ $\neg$ hates(charles, person) cnf(charles_and_agatha_hate_different_people, axiom)
person = butler or hates(aunt_agatha, person) cnf(agatha_likes_only_butler, axiom)
richer(person, aunt_agatha) or hates(butler, person) cnf(butler_hates_poor_people, axiom)
hates(aunt_agatha, person) $\Rightarrow$ hates(butler, person) cnf(butler_and_agatha_hate_the_same_people, axiom)
$\neg$ hates(person, every_one_but(person)) cnf(noone_hates_everyone, axiom)
aunt_agatha $\neq$ butler cnf(agatha_is_not_the_butler, axiom)
$\neg$ killed(aunt_agatha, aunt_agatha) cnf(prove_agatha_killed_herself, negated_conjecture)

PUZ001-3.p Dreadbury Mansion

Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Agatha is not the butler. Therefore : Agatha killed herself.
lives(agatha) cnf(agatha, axiom)
lives(butler) cnf(butler, axiom)
lives(charles) cnf(charles, axiom)
killed(x, y) $\Rightarrow$ $\neg$ richer(x, y) cnf(poorer_killer, axiom)
hates(agatha, x) $\Rightarrow$ $\neg$ hates(charles, x) cnf(different_hates, axiom)
(hates(x, agatha) and hates(x, butler)) $\Rightarrow$ $\neg$ hates(x, charles) cnf(no_one_hates_everyone, axiom)
hates(agatha, agatha) cnf(agatha_hates_agatha, axiom)
hates(agatha, charles) cnf(agatha_hates_charles, axiom)
killed(x, y) $\Rightarrow$ hates(x, y) cnf(killer_hates_victim, axiom)
hates(agatha, x) $\Rightarrow$ hates(butler, x) cnf(same_hates, axiom)
lives(x) $\Rightarrow$ (richer(x, agatha) or hates(butler, x)) cnf(butler_hates_poor, axiom)
killed(agatha, agatha) or killed(butler, agatha) or killed(charles, agatha) cnf(somebody_did_it, negated_conjecture)

PUZ002-1.p The Animals Puzzle
1) The only animals in this house are cats. 2) Every animal is suitable for a pet, that loves to gaze at the moon. 3) When I detest an animal, I avoid it. 4) No animals are carnivorous, unless they prowl at night. 5) No cat fails to kill mice. 6) No animals ever take to me, except what are in this house. 7) Kangaroos are not suitable for pets. 8) None but carnivora kill mice. 9) I detest animals that do not take to me. 10) Animals that prowl at night always love to gaze at the moon. The problem is to prove that "I always avoid a kangaroo".
in_house(cat) $\Rightarrow$ cat(cat) cnf(only_cats_in_house, axiom)
gazer(gazer) $\Rightarrow$ suitable_pet(gazer) cnf(gazers_are_suitable_pets, axiom)
detested(detested) $\Rightarrow$ avoided(detested) cnf(avoid_detested, axiom)
carnivore(carnivore) $\Rightarrow$ prowler(carnivore) cnf(carnivores_are_prowlers, axiom)
cat(cat) $\Rightarrow$ mouse_killer(cat) cnf(cats_are_mice_killers, axiom)
takes_to_me(taken_animal) $\Rightarrow$ in_house(taken_animal) cnf(in_house_if_takes_to_me, axiom)
kangaroo(kangaroo) $\Rightarrow$ $\neg$ suitable_pet(kangaroo) cnf(kangaroos_are_not_pets, axiom)
mouse_killer(killer) $\Rightarrow$ carnivore(killer) cnf(mouse_killers_are_carnivores, axiom)
takes_to_me(animal) or detested(animal) cnf(takes_to_me_or_detested, axiom)
prowler(prowler) $\Rightarrow$ gazer(prowler) cnf(prowlers_are_gazers, axiom)
kangaroo(the_kangaroo) cnf(kangaroo_is_a_kangaroo, axiom)
$\neg$ avoided(the_kangaroo) cnf(avoid_kangaroo, negated_conjecture)

PUZ003-1.p The Barber Puzzle
There is a barbers' club that obeys the following three conditions: (1) If any member A has shaved any other member B - whether himself or another - then all members have shaved A, though not necessarily at the same time. (2) Four of the members are named Guido, Lorenzo, Petrucio, and Cesare. (3) Guido has shaved Cesare. Prove Petrucio has shaved Lorenzo
(member(x) and member(y) and shaved(x, y)) $\Rightarrow$ shaved(members, x) cnf(one_shaved_then_all_shaved, axiom)
(shaved(members, x) and member(y)) $\Rightarrow$ shaved(y, x) cnf(all_shaved_then_one_shaved, axiom)
member(guido) cnf(guido, hypothesis)
member(lorenzo) cnf(lorenzo, hypothesis)
member(petruchio) cnf(petruchio, hypothesis)
member(cesare) cnf(cesare, hypothesis)
shaved(guido, cesare) cnf(guido_has_shaved_cesare, hypothesis)
$\neg$ shaved(petruchio, lorenzo) cnf(prove_petruchio_has_shaved_lorenzo, negated_conjecture)

PUZ004-1.p The Letters Puzzle
(1) All the dated letters in this room are written on blue paper. (2) None of them are in black ink except those that are written in the third person. (3) I have not filed any of them that I can read. (4) None of them that are written on one sheet are undated. (5) All of them that are not crossed are in black ink. : (6) All of them written by Brown begin with "Dear Sir" : (7) All of them written on blue paper are filed. : (8) None of them written on more than one sheet are crossed. (9) None of them that begin with "Dear Sir" are written in third person. Prove that letters by Brown cannot be read.
dated $\Rightarrow$ on_blue_paper cnf(dated_on_blue_paper, axiom)
in_third_person $\Rightarrow$ in_black_ink cnf(third_person_in_black_ink, axiom)
in_black_ink $\Rightarrow$ in_third_person cnf(black_ink_in_third_person, axiom)

can_be_read $\Rightarrow$ $\neg$ filed cnf(not_filed_if_read, axiom)
on_one_sheet $\Rightarrow$ dated cnf(dated_if_on_one_sheet, axiom)
crossed or in_black_ink cnf(not_crossed_in_black_ink, axiom)
by_brown $\Rightarrow$ begins_with_dear_sir cnf(brown_starts_with_sir, axiom)
on_blue_paper $\Rightarrow$ filed cnf(filed_if_on_blue_paper, axiom)
crossed $\Rightarrow$ on_one_sheet cnf(on_one_sheet_if_crossed, axiom)
begins_with_dear_sir $\Rightarrow$ $\neg$ in_third_person cnf(no_dear_sirs_in_third_person, axiom)
by_brown cnf(letter_by_brown, hypothesis)
can_be_read cnf(prove_it_cannot_be_read, negated_conjecture)

PUZ006-1.p Determine sex and race on Mars and Venus
Here's the situation: human observers in this exclusive club on Ganymede can't distinguish Martians from Venusians, males from females, except for the fact that Venusian women and Martian men always tell the truth and Venusian men and Martian women always lie. Ork says "Bog is from Venus." Bog says "Ork is from Mars." Ork says "Bog is male." Bog says "Ork is female." Who's what? (sex & race).
include('Axioms/PUZ001-0.ax')
says(ork, bog_is_from_venus) cnf(ork_says_bog_is_from_venus, hypothesis)
says(bog, ork_is_from_mars) cnf(bog_says_ork_is_from_mar, hypothesis)
says(ork, bog_is_male) cnf(ork_says_bog_is_male, hypothesis)
says(bog, ork_is_female) cnf(bog_says_ork_is_female, hypothesis)
a_truth(bog_is_from_venus) $\Rightarrow$ from_venus(bog) cnf(bog_is_from_venus$_1$, hypothesis)
a_truth(ork_is_from_mars) $\Rightarrow$ from_mars(ork) cnf(ork_is_from_mars$_1$, hypothesis)
a_truth(bog_is_male) $\Rightarrow$ male(bog) cnf(bog_is_male$_1$, hypothesis)
a_truth(ork_is_female) $\Rightarrow$ female(ork) cnf(ork_is_female$_1$, hypothesis)
from_venus(bog) $\Rightarrow$ a_truth(bog_is_from_venus) cnf(bog_is_from_venus$_2$, hypothesis)
from_mars(ork) $\Rightarrow$ a_truth(ork_is_from_mars) cnf(ork_is_from_mars$_2$, hypothesis)
male(bog) $\Rightarrow$ a_truth(bog_is_male) cnf(bog_is_male$_2$, hypothesis)
female(ork) $\Rightarrow$ a_truth(ork_is_female) cnf(ork_is_female$_2$, hypothesis)
$\neg$ female(bog) cnf(prove_bog_is_female, negated_conjecture)

PUZ007-1.p Mixed couples on Mars and Venus
Here's the situation: human observers in this exclusive club on Ganymede can't distinguish Martians from Venusians, males from females, except for the fact that Venusian women and Martian men always tell the truth and Venusian men and Martian women always lie. A says "I'm from Mars" B exclaims "That's not true!" A and B are married; are they a mixed couple?
include('Axioms/PUZ001-0.ax')
says(a, a_from_mars) cnf(a_says_hes_from_mars, hypothesis)
says(b, a_has_lied) cnf(b_says_a_lies, hypothesis)
from_mars(a) $\Rightarrow$ a_truth(a_from_mars) cnf(a_from_mars$_1$, hypothesis)
a_truth(a_from_mars) $\Rightarrow$ from_mars(a) cnf(a_from_mars$_2$, hypothesis)
a_truth(a_from_mars) $\Rightarrow$ $\neg$ a_truth(statement_by(b)) cnf(a_from_mars$_3$, hypothesis)
statement_by(a) = a_from_mars cnf(a_states, hypothesis)
statement_by(b) = a_has_lied cnf(b_states, hypothesis)
a_truth(a_from_mars) or a_truth(statement_by(b)) cnf(truth_of_bs_statement, hypothesis)
female(a) $\Rightarrow$ male(b) cnf(different_sex$_1$, hypothesis)
male(a) $\Rightarrow$ female(b) cnf(different_sex$_2$, hypothesis)
from_mars(b) or from_mars(a) cnf(one_from_mars, negated_conjecture)
from_venus(a) or from_venus(b) cnf(one_from_venus, negated_conjecture)

PUZ008-1.p Missionaries and Cannibals
There are 3 missionaries, 3 cannibals, and a boat on the west bank of a river. All wish to cross, but the boat holds at most 2 people. If the cannibals ever outnumber the missionaries on either bank of the river the outnumbered missionaries will be eaten. Can they all safely cross the river? If so, how? (The boat cannot cross empty.)
achievable(west($m(x), c(s(y))$), boatonwest, east($m(z), c(w)$)) $\Rightarrow$ achievable(west($m(x), c(y)$), boatoneast, east($m(z), c(s(w))$)
achievable(west($m(x), c(y)$), boatoneast, east($m(z), c(s(w))$)) $\Rightarrow$ achievable(west($m(x), c(s(y))$), boatonwest, east($m(z), c(w)$)
achievable(west($m(x), c(s(s(y)))$), boatonwest, east($m(z), c(w)$)) $\Rightarrow$ achievable(west($m(x), c(y)$), boatoneast, east($m(z), c(s(s$
achievable(west($m(x), c(y)$), boatoneast, east($m(z), c(s(s(w)))$)) $\Rightarrow$ achievable(west($m(x), c(s(s(y)))$), boatonwest, east($m(z)$,
achievable(west($m(s(x)), c(y)$), boatonwest, east($m(z), c(w)$)) $\Rightarrow$ achievable(west($m(x), c(y)$), boatoneast, east($m(s(z)), c(w)$)
achievable(west($m(x), c(y)$), boatoneast, east($m(s(z)), c(w)$)) $\Rightarrow$ achievable(west($m(s(x)), c(y)$), boatonwest, east($m(z), c(w)$)
achievable(west($m(s(s(x))), c(y)$), boatonwest, east($m(z), c(w)$)) $\Rightarrow$ achievable(west($m(x), c(y)$), boatoneast, east($m(s(s(z)))$,
achievable(west($m(x), c(y)$), boatoneast, east($m(s(s(z))), c(w)$)) $\Rightarrow$ achievable(west($m(s(s(x))), c(y)$), boatonwest, east($m(z)$,

achievable(west($m(s(x)), c(s(y))$), boatonwest, east($m(z), c(w)$)) ⇒ achievable(west($m(x), c(y)$), boatoneast, east($m(s(z)), c($
achievable(west($m(x), c(y)$), boatoneast, east($m(s(z)), c(s(w))$)) ⇒ achievable(west($m(s(x)), c(s(y))$), boatonwest, east($m(z)$,
achievable(west($m(s(x)), c(s(s(x))))$, y, east(z, w)) cnf(extra_cannibal_meal_on_west_bank, axiom)
achievable(west($m(s(x)), c(s(s(s(x)))))$, y, east(z, w)) cnf(two_extra_cannibals_meal_on_west_bank, axiom)
achievable(west(x, y), z, east($m(s(w)), c(s(s(w))))$) cnf(extra_cannibal_meal_on_east_bank, axiom)
achievable(west(x, y), z, east($m(s(w)), c(s(s(s(w)))))$) cnf(two_extra_cannibals_meal_on_east_bank, axiom)
achievable(west($m(s(s(s(n_0)))), c(s(s(s(n_0)))))$, boatonwest, east($m(n_0), c(n_0)$)) cnf(start_on_west_bank, hypothesis)
¬ achievable(west($m(n_0), c(n_0)$), x, east($m(s(s(s(n_0)))), c(s(s(s(n_0)))))$) cnf(prove_can_get_to_east_bank, negated_conjectu

PUZ009-1.p Looking for Oona

In another curious incident, when the husband arrived on an island looking for Oona, he met 5 natives A,B,C,D,E who all guessed his purpose and grinned at meeting him. They said: A: Oona is on this island. B: Oona is not on this island. C: Oona was here yesterday. D: Oona is not here today, and she was not here yesterday. E: Either D is a knave or C is a knight. The logician thought for a while, but could get nowhere. 'Won't one of you please make another statement?' the logician pleaded. At this point A said: Either E is a knave or C is a knight. Is Oona on the island?"

a_is_a_knight ⇒ ¬ b_is_a_knight cnf(c_1, axiom)
a_is_a_knight ⇒ ¬ d_is_a_knight cnf(c_2, axiom)
b_is_a_knight ⇒ ¬ a_is_a_knight cnf(c_3, axiom)
b_is_a_knight or a_is_a_knight cnf(c_4, axiom)
b_is_a_knight ⇒ (c_is_a_knight or d_is_a_knight) cnf(c_5, axiom)
c_is_a_knight ⇒ ¬ d_is_a_knight cnf(c_6, axiom)
d_is_a_knight ⇒ ¬ a_is_a_knight cnf(c_7, axiom)
d_is_a_knight ⇒ b_is_a_knight cnf(c_8, axiom)
d_is_a_knight ⇒ ¬ c_is_a_knight cnf(c_9, axiom)
(e_is_a_knight and d_is_a_knight) ⇒ c_is_a_knight cnf(c_{10}, axiom)
e_is_a_knight or d_is_a_knight cnf(c_{11}, axiom)
c_is_a_knight ⇒ e_is_a_knight cnf(c_{12}, axiom)
d_is_a_knight or a_is_a_knight or c_is_a_knight cnf(c_{13}, axiom)
b_is_a_knight ⇒ (d_is_a_knight or c_is_a_knight) cnf(c_{14}, axiom)
(a_is_a_knight and e_is_a_knight) ⇒ c_is_a_knight cnf(c_{15}, axiom)
a_is_a_knight or e_is_a_knight cnf(c_{16}, axiom)
c_is_a_knight ⇒ a_is_a_knight cnf(c_{17}, axiom)
b_is_a_knight cnf(c_{18}, negated_conjecture)

PUZ011-1.p An ocean that borders on an African and an Asian country

There is a database of assertions about various countries and oceans and their relationships. Find which ocean borders on African and Asian countries.

ocean(atlantic) cnf(atlantic, hypothesis)
ocean(indian) cnf(indian, hypothesis)
borders(atlantic, brazil) cnf(atlantic_brazil, hypothesis)
borders(atlantic, uruguay) cnf(atlantic_uruguay, hypothesis)
borders(atlantic, venesuela) cnf(atlantic_venesuela, hypothesis)
borders(atlantic, zaire) cnf(atlantic_zaire, hypothesis)
borders(atlantic, nigeria) cnf(atlantic_nigeria, hypothesis)
borders(atlantic, angola) cnf(atlantic_angola, hypothesis)
borders(indian, india) cnf(indian_india, hypothesis)
borders(indian, pakistan) cnf(indian_pakistan, hypothesis)
borders(indian, iran) cnf(indian_iran, hypothesis)
borders(indian, somalia) cnf(indian_somalia, hypothesis)
borders(indian, kenya) cnf(indian_kenya, hypothesis)
borders(indian, tanzania) cnf(indian_tanzania, hypothesis)
south_american(brazil) cnf(brazil, hypothesis)
south_american(uruguay) cnf(uruguay, hypothesis)
south_american(venesuela) cnf(venezuela, hypothesis)
african(zaire) cnf(zaire, hypothesis)
african(nigeria) cnf(nigeria, hypothesis)
african(angola) cnf(angola, hypothesis)
african(somalia) cnf(somalia, hypothesis)
african(kenya) cnf(kenya, hypothesis)

african(tanzania) cnf(tanzania, hypothesis)
asian(india) cnf(india, hypothesis)
asian(pakistan) cnf(pakistan, hypothesis)
asian(iran) cnf(iran, hypothesis)
(ocean(ocean) and borders(ocean, african) and african(african) and borders(ocean, asian)) $\Rightarrow$ $\neg$ asian(asian) cnf(prove_th

PUZ012-1.p The Mislabeled Boxes
There are three boxes a, b, and c on a table. Each box contains apples or bananas or oranges. No two boxes contain the same thing. Each box has a label that says it contains apples or says it contains bananas or says it contains oranges. No box contains what it says on its label. The label on box a says "apples". The label on box b says "oranges". The label on box c says "bananas". You pick up box b and it contains apples. What do the other two boxes contain?
equal_fruits(x, x) cnf(reflexivity_for_fruits, axiom)
equal_boxes(x, x) cnf(reflexivity_for_boxes, axiom)
label(x, y) $\Rightarrow$ $\neg$ contains(x, y) cnf(label_is_wrong, axiom)
contains(boxa, x) or contains(boxb, x) or contains(boxc, x) cnf(each_thing_is_in_a_box, axiom)
contains(x, apples) or contains(x, bananas) or contains(x, oranges) cnf(each_box_contains_something, axiom)
(contains(x, y) and contains(x, z)) $\Rightarrow$ equal_fruits(y, z) cnf(contains_is_well_defined$_1$, axiom)
(contains(x, y) and contains(z, y)) $\Rightarrow$ equal_boxes(x, z) cnf(contains_is_well_defined$_2$, axiom)
$\neg$ equal_boxes(boxa, boxb) cnf(boxa_not_boxb, axiom)
$\neg$ equal_boxes(boxb, boxc) cnf(boxb_not_boxc, axiom)
$\neg$ equal_boxes(boxa, boxc) cnf(boxa_not_boxc, axiom)
$\neg$ equal_fruits(apples, bananas) cnf(apples_not_bananas, axiom)
$\neg$ equal_fruits(bananas, oranges) cnf(bananas_not_oranges, axiom)
$\neg$ equal_fruits(apples, oranges) cnf(apples_not_oranges, axiom)
label(boxa, apples) cnf(boxa_labelled_apples, hypothesis)
label(boxb, oranges) cnf(boxb_labelled_oranges, hypothesis)
label(boxc, bananas) cnf(boxc_labelled_bananas, hypothesis)
contains(boxb, apples) cnf(boxb_contains_apples, hypothesis)
contains(boxa, bananas) $\Rightarrow$ $\neg$ contains(boxc, oranges) cnf(prove_boxa_contains_bananas_and_boxc_oranges, negated_conje

PUZ012_1.p The Mislabeled Boxes
There are three boxes a, b, and c on a table. Each box contains apples or bananas or oranges. No two boxes contain the same thing. Each box has a label that says it contains apples or says it contains bananas or says it contains oranges. No box contains what it says on its label. The label on box a says "apples". The label on box b says "oranges". The label on box c says "bananas". You pick up box b and it contains apples. What do the other two boxes contain?
box: $tType tff(box_type, type)
fruit: $tType tff(fruit_type, type)
boxa: box tff(boxa_type, type)
boxb: box tff(boxb_type, type)
boxc: box tff(boxc_type, type)
apples: fruit tff(apples_type, type)
bananas: fruit tff(bananas_type, type)
oranges: fruit tff(oranges_type, type)
equal_fruits: (fruit $\times$ fruit) $\rightarrow$ $o tff(equal_fruits_type, type)
equal_boxes: (box $\times$ box) $\rightarrow$ $o tff(equal_boxes_type, type)
contains: (box $\times$ fruit) $\rightarrow$ $o tff(contains_type, type)
label: (box $\times$ fruit) $\rightarrow$ $o tff(label_type, type)
$\forall x$: fruit: equal_fruits(x, x) tff(reflexivity_for_fruits, axiom)
$\forall x$: box: equal_boxes(x, x) tff(reflexivity_for_boxes, axiom)
$\forall x$: box, y: fruit: $\neg$ label(x, y) and contains(x, y) tff(label_is_wrong, axiom)
$\forall x$: fruit: (contains(boxa, x) or contains(boxb, x) or contains(boxc, x)) tff(each_thing_is_in_a_box, axiom)
$\forall x$: box: (contains(x, apples) or contains(x, bananas) or contains(x, oranges)) tff(each_box_contains_something, axiom)
$\forall x$: box, y: fruit, z: fruit: ((contains(x, y) and contains(x, z)) $\Rightarrow$ equal_fruits(y, z)) tff(contains_is_well_defined$_1$, axiom)
$\forall x$: box, y: fruit, z: box: ((contains(x, y) and contains(z, y)) $\Rightarrow$ equal_boxes(x, z)) tff(contains_is_well_defined$_2$, axiom)
$\neg$ equal_boxes(boxa, boxb) tff(boxa_not_boxb, axiom)
$\neg$ equal_boxes(boxb, boxc) tff(boxb_not_boxc, axiom)
$\neg$ equal_boxes(boxa, boxc) tff(boxa_not_boxc, axiom)
$\neg$ equal_fruits(apples, bananas) tff(apples_not_bananas, axiom)

$\neg$ equal_fruits(bananas, oranges) tff(bananas_not_oranges, axiom)
$\neg$ equal_fruits(apples, oranges) tff(apples_not_oranges, axiom)
label(boxa, apples) tff(boxa_labelled_apples, hypothesis)
label(boxb, oranges) tff(boxb_labelled_oranges, hypothesis)
label(boxc, bananas) tff(boxc_labelled_bananas, hypothesis)
contains(boxb, apples) tff(boxb_contains_apples, hypothesis)
contains(boxa, bananas) and contains(boxc, oranges) tff(prove_boxa_contains_bananas_and_boxc_oranges, conjecture)

PUZ015-1.p Checkerboard and Dominoes : Opposing corners removed
There is a checker board whose upper left and lower right squares have been removed. There is a box of dominoes that are one square by two squares in size. Can you exactly cover the checker board with dominoes?
achievable(row(x), squares(not_covered, not_covered, $y_3, y_4, y_5, y_6, y_7, y_8$)) $\Rightarrow$ achievable(row(x), squares(covered, covered, y_3, y
achievable(row(x), squares(y_1, not_covered, not_covered, y_4, y_5, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, covered, covered, y
achievable(row(x), squares(y_1, y_2, not_covered, not_covered, y_5, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, covered, covere
achievable(row(x), squares(y_1, y_2, y_3, not_covered, not_covered, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, covered, cov
achievable(row(x), squares(y_1, y_2, y_3, y_4, not_covered, not_covered, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, y_4, covered,
achievable(row(x), squares(y_1, y_2, y_3, y_4, y_5, not_covered, not_covered, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, y_4, y_5, cover
achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6$, not_covered, not_covered)) $\Rightarrow$ achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6$, co
achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8$)) $\Rightarrow$ achievable(row(successor(x)), squares($y'_1, y'_2, y'_3, y'_4, y'_5, y'_6, y'_7, y'_8$))
successor(n_1) = n_2 cnf(successor_of_1_is$_2$, axiom)
successor(n_2) = n_3 cnf(successor_of_2_is$_3$, axiom)
successor(n_3) = n_4 cnf(successor_of_3_is$_4$, axiom)
successor(n_4) = n_5 cnf(successor_of_4_is$_5$, axiom)
successor(n_5) = n_6 cnf(successor_of_5_is$_6$, axiom)
successor(n_6) = n_7 cnf(successor_of_6_is$_7$, axiom)
successor(n_7) = n_8 cnf(successor_of_7_is$_8$, axiom)
successor(n_8) = n_9 cnf(successor_of_8_is$_9$, axiom)
covered$'$ = not_covered cnf(complement_of_covered_is_not_covered, axiom)
not_covered$'$ = covered cnf(complement_of_not_covered_is_covered, axiom)
removed$'$ = not_covered cnf(complement_of_r_is_not_covered, axiom)
achievable(row(n_1), squares(removed, not_covered, not_covered, not_covered, not_covered, not_covered, not_covered, not_covered
$\neg$ achievable(row(n_8), squares(covered, covered, covered, covered, covered, covered, covered, not_covered)) cnf(try_prove_row

PUZ016-1.p Checkerboard and Dominoes : Row 1, columns 2 and 3 removed
There is a checker board whose second and third squares from the first row have been removed. There is a box of dominoes that are one square by two squares in size. Can you exactly cover the checker board with dominoes?
achievable(row(x), squares(not_covered, not_covered, $y_3, y_4, y_5, y_6, y_7, y_8$)) $\Rightarrow$ achievable(row(x), squares(covered, covered, y_3, y
achievable(row(x), squares(y_1, not_covered, not_covered, y_4, y_5, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, covered, covered, y
achievable(row(x), squares(y_1, y_2, not_covered, not_covered, y_5, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, covered, covere
achievable(row(x), squares(y_1, y_2, y_3, not_covered, not_covered, y_6, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, covered, cov
achievable(row(x), squares(y_1, y_2, y_3, y_4, not_covered, not_covered, y_7, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, y_4, covered,
achievable(row(x), squares(y_1, y_2, y_3, y_4, y_5, not_covered, not_covered, y_8)) $\Rightarrow$ achievable(row(x), squares(y_1, y_2, y_3, y_4, y_5, cover
achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6$, not_covered, not_covered)) $\Rightarrow$ achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6$, co
achievable(row(x), squares($y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8$)) $\Rightarrow$ achievable(row(successor(x)), squares($y'_1, y'_2, y'_3, y'_4, y'_5, y'_6, y'_7, y'_8$))
successor(n_1) = n_2 cnf(successor_of_1_is$_2$, axiom)
successor(n_2) = n_3 cnf(successor_of_2_is$_3$, axiom)
successor(n_3) = n_4 cnf(successor_of_3_is$_4$, axiom)
successor(n_4) = n_5 cnf(successor_of_4_is$_5$, axiom)
successor(n_5) = n_6 cnf(successor_of_5_is$_6$, axiom)
successor(n_6) = n_7 cnf(successor_of_6_is$_7$, axiom)
successor(n_7) = n_8 cnf(successor_of_7_is$_8$, axiom)
successor(n_8) = n_9 cnf(successor_of_8_is$_9$, axiom)
covered$'$ = not_covered cnf(complement_of_covered_is_not_covered, axiom)
not_covered$'$ = covered cnf(complement_of_not_covered_is_covered, axiom)
removed$'$ = not_covered cnf(complement_of_removed_is_not_covered, axiom)
achievable(row(n_1), squares(not_covered, removed, removed, not_covered, not_covered, not_covered, not_covered, not_covered))
$\neg$ achievable(row(n_8), squares(covered, covered, covered, covered, covered, covered, covered, covered)) cnf(prove_row_8_can_

PUZ020-1.p A knights & knaves problem, if he's a knight, so is she
person(x) $\Rightarrow$ (knight(x) or knave(x)) cnf(everyone_a_knight_or_knave, axiom)
(person(x) and knight(x)) $\Rightarrow$ $\neg$ knave(x) cnf(not_both_a_knight_and_knave, axiom)

(says(x, y) and a_truth(y)) $\Rightarrow$ a_truth(y) cnf(statements_are_true_or_false, axiom)
says(x, y) $\Rightarrow$ $x \neq y$ cnf(people_do_not_equal_their_statements$_1$, axiom)
says(x, y) $\Rightarrow$ $y =$ statement_by(x) cnf(peoples_statements, axiom)
person(x) $\Rightarrow$ $x \neq$ statement_by(y) cnf(people_do_not_equal_their_statement$_2$, axiom)
(person(x) and a_truth(statement_by(x))) $\Rightarrow$ knight(x) cnf(knights_make_true_statements, axiom)
person(x) $\Rightarrow$ (a_truth(statement_by(x)) or knave(x)) cnf(knaves_make_false_statements, axiom)
(knight(x) and says(x, y)) $\Rightarrow$ a_truth(y) cnf(knights_say_the_truth, axiom)
(knave(x) and says(x, y)) $\Rightarrow$ $\neg$ a_truth(y) cnf(knaves_do_not_say_the_truth, axiom)
person(husband) cnf(husband, hypothesis)
person(wife) cnf(wife, hypothesis)
husband $\neq$ wife cnf(husband_not_wife, hypothesis)
says(husband, statement_by(husband)) cnf(husband_makes_statements, hypothesis)
(a_truth(statement_by(husband)) and knight(husband)) $\Rightarrow$ knight(wife) cnf(truthful_knight_husband_means_knight_wife
knight(husband) $\Rightarrow$ a_truth(statement_by(husband)) cnf(knight_husband_makes_true_statements, hypothesis)
a_truth(statement_by(husband)) or knight(wife) cnf(knight_wife_or_truthful_husband, hypothesis)
knight(wife) $\Rightarrow$ a_truth(statement_by(husband)) cnf(knight_wife_means_truthful_husband, hypothesis)
$\neg$ knight(husband) cnf(prove_knight_husband, negated_conjecture)

PUZ021-1.p How to Win a Bride
Suppose you are an inhabitant of the island of 'knights' and 'knaves'. The knights always tell the truth and the knaves always lie. You fall in love with a girl there and wish to marry her. However, this girl has strange tastes; for some odd reason she does not wish to marry a knight; she wants to marry only a knave. But she wants a rich knave, not a poor one. (We assume for convenience that everyone is classified as either rich or poor.) Suppose, in fact, that you are a rich knave. You are allowed to make only one statement, can you convince her that you are a rich knave?
a_truth(knight$(x), y)$ $\Rightarrow$ $\neg$ a_truth(knave$(x), y)$ cnf(not_knight_and_knave, axiom)
a_truth(knight$(x), y)$ or a_truth(knave$(x), y)$ cnf(knight_or_knave, axiom)
a_truth(rich$(x), y)$ $\Rightarrow$ $\neg$ a_truth(poor$(x), y)$ cnf(not_rich_and_poor, axiom)
a_truth(rich$(x), y)$ or a_truth(poor$(x), y)$ cnf(rich_or_poor, axiom)
(a_truth(knight$(x), z)$ and says(x, y)) $\Rightarrow$ a_truth(y, z) cnf(knights_tell_truth$_1$, axiom)
(a_truth(knight$(x), z)$ and a_truth(y, z)) $\Rightarrow$ says(x, y) cnf(knights_tell_truth$_2$, axiom)
(a_truth(knave$(x), z)$ and says(x, y)) $\Rightarrow$ $\neg$ a_truth(y, z) cnf(knaves_lie$_1$, axiom)
a_truth(knave$(x), z)$ $\Rightarrow$ (says(x, y) or a_truth(y, z)) cnf(knaves_lie$_2$, axiom)
a_truth(and$(x, y), z)$ $\Rightarrow$ a_truth(x, z) cnf(conjunction$_1$, axiom)
a_truth(and$(x, y), z)$ $\Rightarrow$ a_truth(y, z) cnf(conjunction$_2$, axiom)
(a_truth(x, z) and a_truth(y, z)) $\Rightarrow$ a_truth(and$(x, y), z)$ cnf(conjunction$_3$, axiom)
says(me, x) $\Rightarrow$ $\neg$ a_truth(and(knave(me), rich(me)), x) cnf(prove_statement_exists$_1$, negated_conjecture)
says(me, x) or a_truth(and(knave(me), rich(me)), x) cnf(prove_statement_exists$_2$, negated_conjecture)

PUZ022-1.p An ocean that borders on two adjacent Australian states
There is a database of assertions about Australian states and oceans and their relationships. Find which ocean borders on two adjacent Australian states.
borders(y, x) $\Rightarrow$ borders(x, y) cnf(symmetry_of_borders, axiom)
ocean(atlantic) cnf(atlantic, negated_conjecture)
ocean(indian) cnf(indian, negated_conjecture)
ocean(pacific) cnf(pacific, negated_conjecture)
ocean(southern) cnf(southern, negated_conjecture)
state(western_australia) cnf(western_australia, negated_conjecture)
state(northern_territory) cnf(northern_territory, negated_conjecture)
state(queensland) cnf(queensland, negated_conjecture)
state(south_australia) cnf(south_australia, negated_conjecture)
state(new_south_wales) cnf(new_south_wales, negated_conjecture)
state(victoria) cnf(victoria, negated_conjecture)
state(tasmania) cnf(tasmania, negated_conjecture)
borders(western_australia, northern_territory) cnf(wa_nt, negated_conjecture)
borders(western_australia, south_australia) cnf(wa_sa, negated_conjecture)
borders(south_australia, northern_territory) cnf(sa_nt, negated_conjecture)
borders(south_australia, queensland) cnf(sa_qld, negated_conjecture)
borders(south_australia, new_south_wales) cnf(sa_nsw, negated_conjecture)
borders(south_australia, victoria) cnf(sa_vic, negated_conjecture)
borders(northern_territory, queensland) cnf(nt_qld, negated_conjecture)

borders(queensland, new_south_wales) cnf(qld_nsw, negated_conjecture)
borders(new_south_wales, victoria) cnf(nsw_vic, negated_conjecture)
borders(indian, western_australia) cnf(indian_wa, negated_conjecture)
borders(indian, northern_territory) cnf(indian_nt, negated_conjecture)
borders(indian, queensland) cnf(indian_qld, negated_conjecture)
borders(southern, western_australia) cnf(southern_wa, negated_conjecture)
borders(southern, south_australia) cnf(southern_sa, negated_conjecture)
borders(southern, victoria) cnf(southern_vic, negated_conjecture)
borders(southern, tasmania) cnf(southern_tas, negated_conjecture)
borders(pacific, queensland) cnf(pacific_qld, negated_conjecture)
borders(pacific, new_south_wales) cnf(pacific_nsw, negated_conjecture)
borders(pacific, victoria) cnf(pacific_vic, negated_conjecture)
borders(pacific, tasmania) cnf(pacific_tas, negated_conjecture)
(state(state$_1$) and state(state$_2$) and borders(state$_1$, state$_2$) and borders(state$_1$, ocean) and borders(state$_2$, ocean)) $\Rightarrow$ $\neg$ ocean(ocean) cnf(prove_borders, negated_conjecture)

PUZ023-1.p Knights and Knaves #27
There is an island with exactly two types of people : truthtellers who always tell the truth and liars who always lie. There are a group of three people, A, B, and C on the island. A stranger passes by and asks A, "How many truthtellers are among you ?" A answers indistinctly. So the stranger asks B, "what did A say?". B replies "A said that there is exactly one truthteller among us." Then C says, "Don't believe B; he is lying!" What are B and C. Answer: B is a liar and C is a truth-teller.
include('Axioms/PUZ002-0.ax')
(people(x, y, z) and a_truth(one_truthteller)) $\Rightarrow$ (a_truth(truthteller(x)) or a_truth(truthteller(y)) or a_truth(truthteller(z)))
(people(x, y, z) and a_truth(truthteller(x)) and a_truth(truthteller(y))) $\Rightarrow$ $\neg$ a_truth(one_truthteller) cnf(two_truthtellers
(people(x, y, z) and a_truth(truthteller(x)) and a_truth(truthteller(z))) $\Rightarrow$ $\neg$ a_truth(one_truthteller) cnf(two_truthtellers
(people(x, y, z) and a_truth(truthteller(y)) and a_truth(truthteller(z))) $\Rightarrow$ $\neg$ a_truth(one_truthteller) cnf(two_truthtellers
(people(x, y, z) and a_truth(truthteller(x))) $\Rightarrow$ (a_truth(one_truthteller) or a_truth(truthteller(y)) or a_truth(truthteller(z)))
(people(x, y, z) and a_truth(truthteller(y))) $\Rightarrow$ (a_truth(one_truthteller) or a_truth(truthteller(x)) or a_truth(truthteller(z)))
(people(x, y, z) and a_truth(truthteller(z))) $\Rightarrow$ (a_truth(one_truthteller) or a_truth(truthteller(y)) or a_truth(truthteller(x)))
people(a, b, c) cnf(a_b_and_c_are_people, hypothesis)
a_truth(says(a, garbage)) cnf(a_says_garbage, hypothesis)
a_truth(says(b, says(a, one_truthteller))) cnf(b_says_a_says_one_truthteller, hypothesis)
a_truth(says(c, liar(b))) cnf(c_says_b_lies, hypothesis)
(a_truth(liar(b)) and a_truth(liar(c))) $\Rightarrow$ an_answer(b_and_c_liars) cnf(b_and_c_liars, hypothesis)
(a_truth(liar(b)) and a_truth(truthteller(c))) $\Rightarrow$ an_answer(b_liar_and_c_truthteller) cnf(b_liar_and_c_truthteller, hypothes
(a_truth(truthteller(b)) and a_truth(liar(c))) $\Rightarrow$ an_answer(b_truthteller_and_c_liar) cnf(b_truthteller_and_c_liar, hypothes
(a_truth(truthteller(b)) and a_truth(truthteller(c))) $\Rightarrow$ an_answer(b_and_c_truthtellers) cnf(b_and_c_truthtellers, hypothe
$\neg$ an_answer(x) cnf(prove_there_is_an_answer, negated_conjecture)

PUZ024-1.p Knights and Knaves #31
There is an island with exactly two types of people - truthtellers who always tell the truth and liars who always lie. There are a group of three people, A, B, and C on the island. A and B make the following statements. A: All of us are liars; B: Exactly one of us is a truthteller. What are A, B, and C? Answer: A is a liar, B is a truthteller, and C is a liar.
include('Axioms/PUZ002-0.ax')
(a_truth(says(x, one_truthteller)) and people(x, y, z) and a_truth(truthteller(x)) and a_truth(truthteller(y))) $\Rightarrow$ $\neg$ a_truth(tru
(a_truth(one_truthteller) and people(x, y, z)) $\Rightarrow$ (a_truth(truthteller(x)) or a_truth(truthteller(y)) or a_truth(truthteller(z)))
(a_truth(one_truthteller) and people(x, y, z) and a_truth(truthteller(y)) and a_truth(truthteller(x))) $\Rightarrow$ $\neg$ a_truth(truthteller
(a_truth(one_truthteller) and people(x, y, z) and a_truth(liar(x)) and a_truth(truthteller(y))) $\Rightarrow$ a_truth(liar(z)) cnf(one
(people(x, y, z) and a_truth(liar(x)) and a_truth(truthteller(y))) $\Rightarrow$ (a_truth(one_truthteller) or a_truth(truthteller(z))) c
(people(x, y, z) and a_truth(liar(x)) and a_truth(liar(y))) $\Rightarrow$ (a_truth(one_truthteller) or a_truth(liar(z))) cnf(three_liars
(a_truth(says(x, all_are_liars)) and people(x, y, z)) $\Rightarrow$ $\neg$ a_truth(truthteller(x)) cnf(speaker_is_lying, axiom)
people(x, y, z) $\Rightarrow$ (a_truth(all_are_liars) or a_truth(truthteller(x)) or a_truth(truthteller(y)) or a_truth(truthteller(z))) cn
people(b, c, a) cnf(b_c_a_people, hypothesis)
people(a, c, b) cnf(a_c_b_people, hypothesis)
people(c, b, a) cnf(c_b_a_people, hypothesis)
a_truth(says(a, all_are_liars)) cnf(a_says_all_are_liars, hypothesis)
a_truth(says(b, one_truthteller)) cnf(b_says_one_truthteller, hypothesis)
$\neg$ a_truth(truthteller(truthteller)) cnf(prove_there_is_a_truthteller, negated_conjecture)

PUZ025-1.p Knights and Knaves #35
There is an island with exactly two types of people - truthtellers who always tell the truth and liars who always lie. There are a group of three people, A, B, and C on the island. A and B make the following statements. A: "B and C are the same type". Someone asks "Are A and B of the same type ? " What does C answer? Answer: "yes"
include('Axioms/PUZ002-0.ax')
(people(x, y, z) and a_truth(liar(x)) and a_truth(liar(y))) $\Rightarrow$ a_truth(equal_type(x, y)) cnf(two_liars_are_equal, axiom)
(people(x, y, z) and a_truth(truthteller(x)) and a_truth(truthteller(y))) $\Rightarrow$ a_truth(equal_type(x, y)) cnf(two_truthtellers
(a_truth(equal_type(x, y)) and a_truth(truthteller(x))) $\Rightarrow$ a_truth(truthteller(y)) cnf(truthteller_equals_truthtller, axiom)
(a_truth(equal_type(x, y)) and a_truth(liar(x))) $\Rightarrow$ a_truth(liar(y)) cnf(liar_equals_liar, axiom)
a_truth(truthteller(x)) $\Rightarrow$ (a_truth(equal_type(x, y)) or a_truth(liar(y))) cnf(truthteller_not_equal_liar, axiom)
a_truth(liar(x)) $\Rightarrow$ (a_truth(equal_type(x, y)) or a_truth(truthteller(y))) cnf(liar_not_equal_truthteller, axiom)
a_truth(equal_type(x, y)) $\Rightarrow$ a_truth(equal_type(y, x)) cnf(symmetry_of_equal_type, axiom)
(ask_1_if$_2$(x, y) and a_truth(truthteller(x)) and a_truth(y)) $\Rightarrow$ answer(yes) cnf(truthteller_identifies_truths, axiom)
(ask_1_if$_2$(x, y) and a_truth(truthteller(x))) $\Rightarrow$ (a_truth(y) or answer(no)) cnf(truthteller_denies_lies, axiom)
(ask_1_if$_2$(x, y) and a_truth(liar(x)) and a_truth(y)) $\Rightarrow$ answer(no) cnf(liar_denies_truths, axiom)
(ask_1_if$_2$(x, y) and a_truth(liar(x))) $\Rightarrow$ (a_truth(y) or answer(yes)) cnf(liar_supports_lies, axiom)
people(b, c, a) cnf(b_c_a_are_people, hypothesis)
people(a, b, a) cnf(a_b_c_are_people, hypothesis)
people(a, c, b) cnf(a_c_b_are_people, hypothesis)
people(c, b, a) cnf(c_b_a_are_people, hypothesis)
a_truth(says(a, equal_type(b, c))) cnf(a_says_b_and_c_equal, hypothesis)
ask_1_if$_2$(c, equal_type(a, b)) cnf(ask_c_if_a_and_b_equal, hypothesis)
$\neg$ answer(answer) cnf(prove_there_is_an_answer, negated_conjecture)

PUZ026-1.p Knights and Knaves #39
There is an island with exactly three types of people - truthtellers who always tell the truth, and liars who always lie, and normals who sometimes tell the truth and sometimes lie. We are given three people, A, B, C, one of whom is a truthteller, one a liar, and one a normal (but not neccesarily in that order). They make the following statements. A: I am normal; B: That is true. C: I am not normal. What are A,B, and C? Answer: A is a liar, B is a normal, and C is a truthteller.
include('Axioms/PUZ003-0.ax')
a_truth(not_normal(x)) $\Rightarrow$ $\neg$ a_truth(normal(x)) cnf(not_normal_and_not_normal, axiom)
a_truth(not_normal(x)) or a_truth(normal(x)) cnf(normal_or_not_normal, axiom)
(people(x, y, z) and a_truth(truthteller(x))) $\Rightarrow$ $\neg$ a_truth(truthteller(y)) cnf(not_two_truthtellers$_1$, axiom)
(people(x, y, z) and a_truth(truthteller(x))) $\Rightarrow$ $\neg$ a_truth(truthteller(z)) cnf(not_two_truthtellers$_2$, axiom)
(people(x, y, z) and a_truth(liar(x))) $\Rightarrow$ $\neg$ a_truth(liar(y)) cnf(not_two_liars$_1$, axiom)
(people(x, y, z) and a_truth(liar(x))) $\Rightarrow$ $\neg$ a_truth(liar(z)) cnf(not_two_liars$_2$, axiom)
(people(x, y, z) and a_truth(normal(x))) $\Rightarrow$ $\neg$ a_truth(normal(y)) cnf(not_two_normal$_1$, axiom)
(people(x, y, z) and a_truth(normal(x))) $\Rightarrow$ $\neg$ a_truth(normal(z)) cnf(not_two_normal$_2$, axiom)
people(a, b, c) cnf(a_b_c_are_people, hypothesis)
people(b, c, a) cnf(b_c_a_are_people, hypothesis)
people(c, b, a) cnf(c_b_a_are_people, hypothesis)
a_truth(says(a, normal(a))) cnf(a_says_a_normal, hypothesis)
a_truth(says(b, normal(a))) cnf(b_says_a_normal, hypothesis)
a_truth(says(c, not_normal(c))) cnf(c_says_c_not_normal, hypothesis)
(a_truth(liar(liar)) and a_truth(normal(normal))) $\Rightarrow$ $\neg$ a_truth(truthteller(truthteller)) cnf(prove_one_of_each, negated_co

PUZ027-1.p Knights and Knaves #42
There is an island with exactly three types of people - truthtellers who always tell the truth, and liars who always lie, and normals who sometimes tell the truth and sometimes lie. Liars are said to be of the lowest rank, normals are middle rank, and truthtellers of the highest rank. Two people A and B on the island make the following statements. A: I am of lower rank than B. B: That's not true! What are the ranks of A and B, and which of the two statements are true? Answer: A is a normal and B is a normal.
include('Axioms/PUZ003-0.ax')
a_truth(not_lower(x, x)) cnf(not_lower_is_irreflexive, axiom)
a_truth(not_lower(x, y)) $\Rightarrow$ $\neg$ a_truth(lower(x, y)) cnf(not_not_lower_and_lower, axiom)
a_truth(not_lower(x, y)) or a_truth(lower(x, y)) cnf(not_lower_or_lower, axiom)
(a_truth(lower(x, y)) and a_truth(liar(x))) $\Rightarrow$ (a_truth(normal(y)) or a_truth(truthteller(y))) cnf(liars_lowest, axiom)
(a_truth(lower(x, y)) and a_truth(normal(x))) $\Rightarrow$ a_truth(truthteller(y)) cnf(truthtellers_highest, axiom)
a_truth(lower(x, y)) $\Rightarrow$ $\neg$ a_truth(truthteller(x)) cnf(truthtellers_lower_than_no_one, axiom)

(a_truth(lower(x, y)) and a_truth(truthteller(y))) ⇒ (a_truth(normal(x)) or a_truth(liar(x))) cnf(normal_and_liars_lower
(a_truth(lower(x, y)) and a_truth(normal(y))) ⇒ a_truth(liar(x)) cnf(liars_lower_than_normal, axiom)
a_truth(lower(x, y)) ⇒ ¬ a_truth(liar(y)) cnf(no_one_lower_than_liars, axiom)
(a_truth(not_lower(x, y)) and a_truth(truthteller(x))) ⇒ (a_truth(truthteller(y)) or a_truth(lower(y, x))) cnf(not_lower_t
(a_truth(not_lower(x, y)) and a_truth(liar(x))) ⇒ (a_truth(liar(y)) or a_truth(lower(y, x))) cnf(not_lower_than_liar, axion
(a_truth(not_lower(x, y)) and a_truth(normal(x))) ⇒ (a_truth(normal(y)) or a_truth(lower(y, x))) cnf(not_lower_than_no
a_truth(says(a, lower(a, b))) cnf(a_says_a_lower_than_b, hypothesis)
a_truth(says(b, not_lower(a, b))) cnf(b_says_a_not_lower_than_b, hypothesis)
(a_truth(truthteller(a)) and a_truth(truthteller(b))) ⇒ answer(a_and_b_truthteller) cnf(a_and_b_truthteller, hypothesis)
(a_truth(truthteller(a)) and a_truth(normal(b))) ⇒ answer(a_truthteller_b_normal) cnf(a_truthteller_b_normal, hypothesi
(a_truth(truthteller(a)) and a_truth(liar(b))) ⇒ answer(a_truthteller_b_liar) cnf(a_truthteller_b_liar, hypothesis)
(a_truth(normal(a)) and a_truth(truthteller(b))) ⇒ answer(a_normal_b_truthteller) cnf(a_normal_b_truthteller, hypothesi
(a_truth(normal(a)) and a_truth(normal(b))) ⇒ answer(a_and_b_normal) cnf(a_and_b_normal, hypothesis)
(a_truth(normal(a)) and a_truth(liar(b))) ⇒ answer(a_normal_b_liar) cnf(a_normal_b_liar, hypothesis)
(a_truth(liar(a)) and a_truth(truthteller(b))) ⇒ answer(a_liar_b_truthteller) cnf(a_liar_b_truthteller, hypothesis)
(a_truth(liar(a)) and a_truth(normal(b))) ⇒ answer(a_liar_b_normal) cnf(a_liar_b_normal, hypothesis)
(a_truth(liar(a)) and a_truth(liar(b))) ⇒ answer(a_and_b_liar) cnf(a_and_b_liar, hypothesis)
¬ answer(answer) cnf(prove_there_is_an_answer, negated_conjecture)

PUZ028-1.p People at a party
We can always choose 3 persons who are either familiar with each other or not familiar with each other, from 6 persons who meet at a party.
person(1) cnf(person$_1$, axiom)
person(two) cnf(person$_2$, axiom)
person(three) cnf(person$_3$, axiom)
person(four) cnf(person$_4$, axiom)
person(five) cnf(person$_5$, axiom)
person(six) cnf(person$_6$, axiom)
after(1, two) cnf(order$_1$, axiom)
after(two, three) cnf(order$_2$, axiom)
after(three, four) cnf(order$_3$, axiom)
after(four, five) cnf(order$_4$, axiom)
after(five, six) cnf(order$_5$, axiom)
(after(large, medium) and after(medium, small)) ⇒ after(large, small) cnf(transitivity_of_order, axiom)
(person(x) and person(y) and after(x, y)) ⇒ (familiar(x, y) or not_familiar(x, y)) cnf(familiar_or_not, axiom)
(familiar(x_1, x_2) and familiar(x_2, x_3)) ⇒ ¬ familiar(x_3, x_1) cnf(three_familiar, negated_conjecture)
(not_familiar(x_1, x_2) and not_familiar(x_2, x_3)) ⇒ ¬ not_familiar(x_3, x_1) cnf(three_not_familiar, negated_conjecture)

PUZ028-2.p People at a party
We can always choose 3 persons who are either familiar with each other or not familiar with each other, from 6 persons who meet at a party.
person(n_1) cnf(person$_1$, axiom)
person(n_2) cnf(person$_2$, axiom)
person(n_3) cnf(person$_3$, axiom)
person(n_4) cnf(person$_4$, axiom)
person(n_5) cnf(person$_5$, axiom)
person(n_6) cnf(person$_6$, axiom)
not_equal(n_1, n_2) cnf(not_equal_1$_2$, axiom)
not_equal(n_1, n_3) cnf(not_equal_1$_3$, axiom)
not_equal(n_1, n_4) cnf(not_equal_1$_4$, axiom)
not_equal(n_1, n_5) cnf(not_equal_1$_5$, axiom)
not_equal(n_1, n_6) cnf(not_equal_1$_6$, axiom)
not_equal(n_2, n_1) cnf(not_equal_2$_1$, axiom)
not_equal(n_2, n_3) cnf(not_equal_2$_3$, axiom)
not_equal(n_2, n_4) cnf(not_equal_2$_4$, axiom)
not_equal(n_2, n_5) cnf(not_equal_2$_5$, axiom)
not_equal(n_2, n_6) cnf(not_equal_2$_6$, axiom)
not_equal(n_3, n_1) cnf(not_equal_3$_1$, axiom)
not_equal(n_3, n_2) cnf(not_equal_3$_2$, axiom)
not_equal(n_3, n_4) cnf(not_equal_3$_4$, axiom)

$\text{not\_equal}(n_3, n_5)$ $\text{cnf}(\text{not\_equal\_3}_5, \text{axiom})$
$\text{not\_equal}(n_3, n_6)$ $\text{cnf}(\text{not\_equal\_3}_6, \text{axiom})$
$\text{not\_equal}(n_4, n_1)$ $\text{cnf}(\text{not\_equal\_4}_1, \text{axiom})$
$\text{not\_equal}(n_4, n_2)$ $\text{cnf}(\text{not\_equal\_4}_2, \text{axiom})$
$\text{not\_equal}(n_4, n_3)$ $\text{cnf}(\text{not\_equal\_4}_3, \text{axiom})$
$\text{not\_equal}(n_4, n_5)$ $\text{cnf}(\text{not\_equal\_4}_5, \text{axiom})$
$\text{not\_equal}(n_4, n_6)$ $\text{cnf}(\text{not\_equal\_4}_6, \text{axiom})$
$\text{not\_equal}(n_5, n_1)$ $\text{cnf}(\text{not\_equal\_5}_1, \text{axiom})$
$\text{not\_equal}(n_5, n_2)$ $\text{cnf}(\text{not\_equal\_5}_2, \text{axiom})$
$\text{not\_equal}(n_5, n_3)$ $\text{cnf}(\text{not\_equal\_5}_3, \text{axiom})$
$\text{not\_equal}(n_5, n_4)$ $\text{cnf}(\text{not\_equal\_5}_4, \text{axiom})$
$\text{not\_equal}(n_5, n_6)$ $\text{cnf}(\text{not\_equal\_5}_6, \text{axiom})$
$\text{not\_equal}(n_6, n_1)$ $\text{cnf}(\text{not\_equal\_6}_1, \text{axiom})$
$\text{not\_equal}(n_6, n_2)$ $\text{cnf}(\text{not\_equal\_6}_2, \text{axiom})$
$\text{not\_equal}(n_6, n_3)$ $\text{cnf}(\text{not\_equal\_6}_3, \text{axiom})$
$\text{not\_equal}(n_6, n_4)$ $\text{cnf}(\text{not\_equal\_6}_4, \text{axiom})$
$\text{not\_equal}(n_6, n_5)$ $\text{cnf}(\text{not\_equal\_6}_5, \text{axiom})$
$(\text{person}(x) \text{ and } \text{person}(y) \text{ and } \text{not\_equal}(x, y)) \Rightarrow (\text{familiar}(x, y) \text{ or } \text{not\_familiar}(x, y))$ $\text{cnf}(\text{familiar\_or\_not}, \text{axiom})$
$(\text{familiar}(x_1, x_2) \text{ and } \text{familiar}(x_2, x_3)) \Rightarrow \neg\, \text{familiar}(x_3, x_1)$ $\text{cnf}(\text{three\_familiar}, \text{negated\_conjecture})$
$(\text{not\_familiar}(x_1, x_2) \text{ and } \text{not\_familiar}(x_2, x_3)) \Rightarrow \neg\, \text{not\_familiar}(x_3, x_1)$ $\text{cnf}(\text{three\_not\_familiar}, \text{negated\_conjecture})$

PUZ028-5.p People at a party
We can always choose 3 persons who are either familiar with each other or not familiar with each other, from 6 persons who meet at a party.
$\text{person}(1)$ $\text{cnf}(\text{person}_1, \text{axiom})$
$\text{person}(\text{two})$ $\text{cnf}(\text{person}_2, \text{axiom})$
$\text{person}(\text{three})$ $\text{cnf}(\text{person}_3, \text{axiom})$
$\text{person}(\text{four})$ $\text{cnf}(\text{person}_4, \text{axiom})$
$\text{person}(\text{five})$ $\text{cnf}(\text{person}_5, \text{axiom})$
$\text{person}(\text{six})$ $\text{cnf}(\text{person}_6, \text{axiom})$
$\text{after}(1, \text{two})$ $\text{cnf}(\text{order}_1, \text{axiom})$
$\text{after}(\text{two}, \text{three})$ $\text{cnf}(\text{order}_2, \text{axiom})$
$\text{after}(\text{three}, \text{four})$ $\text{cnf}(\text{order}_3, \text{axiom})$
$\text{after}(\text{four}, \text{five})$ $\text{cnf}(\text{order}_4, \text{axiom})$
$\text{after}(\text{five}, \text{six})$ $\text{cnf}(\text{order}_5, \text{axiom})$
$(\text{after}(\text{large}, \text{medium}) \text{ and } \text{after}(\text{medium}, \text{small})) \Rightarrow \text{after}(\text{large}, \text{small})$ $\text{cnf}(\text{transitivity\_of\_order}, \text{axiom})$
$(\text{person}(x) \text{ and } \text{person}(y) \text{ and } \text{after}(x, y)) \Rightarrow (\text{familiar}(x, y) \text{ or } \text{not\_familiar}(x, y))$ $\text{cnf}(\text{familiar\_or\_not}, \text{axiom})$
$(\text{familiar}(x_1, x_2) \text{ and } \text{familiar}(x_2, x_3)) \Rightarrow \neg\, \text{familiar}(x_1, x_3)$ $\text{cnf}(\text{three\_familiar}, \text{negated\_conjecture})$
$(\text{not\_familiar}(x_1, x_2) \text{ and } \text{not\_familiar}(x_2, x_3)) \Rightarrow \neg\, \text{not\_familiar}(x_1, x_3)$ $\text{cnf}(\text{three\_not\_familiar}, \text{negated\_conjecture})$

PUZ028-6.p People at a party
We can always choose 3 persons who are either familiar with each other or not familiar with each other, from 6 persons who meet at a party.
$\text{person}(n_1)$ $\text{cnf}(\text{person}_1, \text{axiom})$
$\text{person}(n_2)$ $\text{cnf}(\text{person}_2, \text{axiom})$
$\text{person}(n_3)$ $\text{cnf}(\text{person}_3, \text{axiom})$
$\text{person}(n_4)$ $\text{cnf}(\text{person}_4, \text{axiom})$
$\text{person}(n_5)$ $\text{cnf}(\text{person}_5, \text{axiom})$
$\text{person}(n_6)$ $\text{cnf}(\text{person}_6, \text{axiom})$
$\text{not\_equal}(n_1, n_2)$ $\text{cnf}(\text{not\_equal\_1}_2, \text{axiom})$
$\text{not\_equal}(n_1, n_3)$ $\text{cnf}(\text{not\_equal\_1}_3, \text{axiom})$
$\text{not\_equal}(n_1, n_4)$ $\text{cnf}(\text{not\_equal\_1}_4, \text{axiom})$
$\text{not\_equal}(n_1, n_5)$ $\text{cnf}(\text{not\_equal\_1}_5, \text{axiom})$
$\text{not\_equal}(n_1, n_6)$ $\text{cnf}(\text{not\_equal\_1}_6, \text{axiom})$
$\text{not\_equal}(n_2, n_1)$ $\text{cnf}(\text{not\_equal\_2}_1, \text{axiom})$
$\text{not\_equal}(n_2, n_3)$ $\text{cnf}(\text{not\_equal\_2}_3, \text{axiom})$
$\text{not\_equal}(n_2, n_4)$ $\text{cnf}(\text{not\_equal\_2}_4, \text{axiom})$
$\text{not\_equal}(n_2, n_5)$ $\text{cnf}(\text{not\_equal\_2}_5, \text{axiom})$
$\text{not\_equal}(n_2, n_6)$ $\text{cnf}(\text{not\_equal\_2}_6, \text{axiom})$
$\text{not\_equal}(n_3, n_1)$ $\text{cnf}(\text{not\_equal\_3}_1, \text{axiom})$

not_equal(n_3, n_2) cnf(not_equal_3_2, axiom)
not_equal(n_3, n_4) cnf(not_equal_3_4, axiom)
not_equal(n_3, n_5) cnf(not_equal_3_5, axiom)
not_equal(n_3, n_6) cnf(not_equal_3_6, axiom)
not_equal(n_4, n_1) cnf(not_equal_4_1, axiom)
not_equal(n_4, n_2) cnf(not_equal_4_2, axiom)
not_equal(n_4, n_3) cnf(not_equal_4_3, axiom)
not_equal(n_4, n_5) cnf(not_equal_4_5, axiom)
not_equal(n_4, n_6) cnf(not_equal_4_6, axiom)
not_equal(n_5, n_1) cnf(not_equal_5_1, axiom)
not_equal(n_5, n_2) cnf(not_equal_5_2, axiom)
not_equal(n_5, n_3) cnf(not_equal_5_3, axiom)
not_equal(n_5, n_4) cnf(not_equal_5_4, axiom)
not_equal(n_5, n_6) cnf(not_equal_5_6, axiom)
not_equal(n_6, n_1) cnf(not_equal_6_1, axiom)
not_equal(n_6, n_2) cnf(not_equal_6_2, axiom)
not_equal(n_6, n_3) cnf(not_equal_6_3, axiom)
not_equal(n_6, n_4) cnf(not_equal_6_4, axiom)
not_equal(n_6, n_5) cnf(not_equal_6_5, axiom)
(person(x) and person(y) and not_equal(x, y)) $\Rightarrow$ (familiar(x, y) or not_familiar(x, y)) cnf(familiar_or_not, axiom)
familiar(x_1, x_2) $\Rightarrow$ familiar(x_2, x_1) cnf(symmetry_of_familiar, axiom)
not_familiar(x_1, x_2) $\Rightarrow$ not_familiar(x_2, x_1) cnf(symmetry_of_not_familiar, axiom)
(familiar(x_1, x_2) and familiar(x_2, x_3)) $\Rightarrow$ $\neg$ familiar(x_3, x_1) cnf(three_familiar, negated_conjecture)
(not_familiar(x_1, x_2) and not_familiar(x_2, x_3)) $\Rightarrow$ $\neg$ not_familiar(x_3, x_1) cnf(three_not_familiar, negated_conjecture)

PUZ029-1.p The pigs and balloons puzzle
1) All, who neither dance on tight ropes nor eat penny-buns, are old. 2) Pigs, that are liable to giddiness, are treated with respect. 3) A wise balloonist takes an umbrella with him. 4) No one ought to lunch in public who looks ridiculous and eats penny-buns. 5) Young creatures, who go up in balloons, are liable to giddiness. 6) Fat creatures, who look ridiculous, may lunch in public, provided that they do not dance on tight ropes. 7) No wise creatures dance on tight ropes, if liable to giddiness. 8) A pig looks ridiculous, carrying an umbrella. 9) All, who do not dance on tight ropes, and who are treated with respect are fat. Show that no wise young pigs go up in balloons.
dances_on_tightropes(x) or eats_pennybuns(x) or old(x) cnf(boring_old_people, axiom)
(pig(x) and liable_to_giddiness(x)) $\Rightarrow$ treated_with_respect(x) cnf(giddy_pigs_reated_with_respect, axiom)
(wise(x) and balloonist(x)) $\Rightarrow$ has_umbrella(x) cnf(wise_ballonists_have_umbrellas, axiom)
(looks_ridiculous(x) and eats_pennybuns(x)) $\Rightarrow$ $\neg$ eats_lunch_in_public(x) cnf(dont_look_ridiculous_eating_buns_in_public
(balloonist(x) and young(x)) $\Rightarrow$ liable_to_giddiness(x) cnf(young_balloonists_get_giddy, axiom)
(fat(x) and looks_ridiculous(x)) $\Rightarrow$ (dances_on_tightropes(x) or eats_lunch_in_public(x)) cnf(fat_ridiculous_off_tightrope_
(liable_to_giddiness(x) and wise(x)) $\Rightarrow$ $\neg$ dances_on_tightropes(x) cnf(wise_giddy_dont_dance_on_tightrope, axiom)
(pig(x) and has_umbrella(x)) $\Rightarrow$ looks_ridiculous(x) cnf(pigs_look_ridiculous_with_umbrellas, axiom)
treated_with_respect(x) $\Rightarrow$ (dances_on_tightropes(x) or fat(x)) cnf(non_dancers_who_are_respected_are_fat, axiom)
young(x) or old(x) cnf(young_or_old, axiom)
young(x) $\Rightarrow$ $\neg$ old(x) cnf(not_young_and_old, axiom)
wise(piggy) cnf(piggy_is_wise, hypothesis)
young(piggy) cnf(piggy_is_young, hypothesis)
pig(piggy) cnf(piggy_is_a_pig, hypothesis)
balloonist(piggy) cnf(prove_piggy_is_no_balloonist, negated_conjecture)

PUZ030-1.p Salt and Mustard Problem
both(x) $\Rightarrow$ salt(x) cnf(both$_1$, axiom)
both(x) $\Rightarrow$ mustard(x) cnf(both$_2$, axiom)
(salt(x) and mustard(x)) $\Rightarrow$ both(x) cnf(both$_3$, axiom)
oneof(x) $\Rightarrow$ (salt(x) or mustard(x)) cnf(oneof$_1$, axiom)
oneof(x) $\Rightarrow$ $\neg$ both(x) cnf(oneof$_2$, axiom)
oneof(x) $\Rightarrow$ $\neg$ neither(x) cnf(oneof$_3$, axiom)
both(x) or neither(x) or oneof(x) cnf(one_condition_holds$_1$, axiom)
(oneof(x) and salt(x)) $\Rightarrow$ $\neg$ mustard(x) cnf(oneof$_4$, axiom)
both(x) $\Rightarrow$ $\neg$ neither(x) cnf(neither$_1$, axiom)
neither(x) $\Rightarrow$ $\neg$ salt(x) cnf(neither$_2$, axiom)
neither(x) $\Rightarrow$ $\neg$ mustard(x) cnf(neither$_3$, axiom)

salt(x) or mustard(x) or neither(x) cnf(neither$_4$, axiom)
salt(barry) ⇒ (oneof(cole) or oneof(lang)) cnf(rule1$_1$, hypothesis)
oneof(cole) ⇒ salt(barry) cnf(rule1$_2$, hypothesis)
oneof(lang) ⇒ salt(barry) cnf(rule1$_3$, hypothesis)
mustard(barry) ⇒ (neither(dix) or both(mill)) cnf(rule2$_1$, hypothesis)
neither(dix) ⇒ mustard(barry) cnf(rule2$_2$, hypothesis)
both(mill) ⇒ mustard(barry) cnf(rule2$_3$, hypothesis)
salt(cole) ⇒ (oneof(barry) or neither(mill)) cnf(rule3$_1$, hypothesis)
oneof(barry) ⇒ salt(cole) cnf(rule3$_2$, hypothesis)
neither(mill) ⇒ salt(cole) cnf(rule3$_3$, hypothesis)
mustard(cole) ⇒ (both(dix) or both(lang)) cnf(rule4$_1$, hypothesis)
both(dix) ⇒ mustard(cole) cnf(rule4$_2$, hypothesis)
both(lang) ⇒ mustard(cole) cnf(rule4$_3$, hypothesis)
salt(dix) ⇒ (neither(barry) or both(cole)) cnf(rule5$_1$, hypothesis)
neither(barry) ⇒ salt(dix) cnf(rule5$_2$, hypothesis)
both(cole) ⇒ salt(dix) cnf(rule5$_3$, hypothesis)
mustard(dix) ⇒ (neither(lang) or neither(mill)) cnf(rule6$_1$, hypothesis)
neither(lang) ⇒ mustard(dix) cnf(rule6$_2$, hypothesis)
neither(mill) ⇒ mustard(dix) cnf(rule6$_3$, hypothesis)
salt(lang) ⇒ (oneof(barry) or oneof(dix)) cnf(rule7$_1$, hypothesis)
oneof(barry) ⇒ salt(lang) cnf(rule7$_2$, hypothesis)
oneof(dix) ⇒ salt(lang) cnf(rule7$_3$, hypothesis)
mustard(lang) ⇒ (neither(cole) or neither(mill)) cnf(rule8$_1$, hypothesis)
neither(cole) ⇒ mustard(lang) cnf(rule8$_2$, hypothesis)
neither(mill) ⇒ mustard(lang) cnf(rule8$_3$, hypothesis)
salt(mill) ⇒ (both(barry) or both(lang)) cnf(rule9$_1$, hypothesis)
both(barry) ⇒ salt(mill) cnf(rule9$_2$, hypothesis)
both(lang) ⇒ mustard(mill) cnf(rule9$_3$, hypothesis)
mustard(mill) ⇒ (oneof(cole) or oneof(dix)) cnf(rule10$_1$, hypothesis)
oneof(cole) ⇒ mustard(mill) cnf(rule10$_2$, hypothesis)
oneof(dix) ⇒ mustard(mill) cnf(rule10$_3$, hypothesis)
(neither(cole) and neither(dix) and both(barry) and oneof(lang) and salt(mill) and mustard(lang)) ⇒ ¬ oneof(mill) cnf(

PUZ031+1.p Schubert's Steamroller
Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain eating animal.
∀x: (wolf(x) ⇒ animal(x)) fof(pel47_1$_1$, axiom)
∃x_1: wolf(x_1) fof(pel47_1$_2$, axiom)
∀x: (fox(x) ⇒ animal(x)) fof(pel47_2$_1$, axiom)
∃x_1: fox(x_1) fof(pel47_2$_2$, axiom)
∀x: (bird(x) ⇒ animal(x)) fof(pel47_3$_1$, axiom)
∃x_1: bird(x_1) fof(pel47_3$_2$, axiom)
∀x: (caterpillar(x) ⇒ animal(x)) fof(pel47_4$_1$, axiom)
∃x_1: caterpillar(x_1) fof(pel47_4$_2$, axiom)
∀x: (snail(x) ⇒ animal(x)) fof(pel47_5$_1$, axiom)
∃x_1: snail(x_1) fof(pel47_5$_2$, axiom)
∃x: grain(x) fof(pel47_6$_1$, axiom)
∀x_1: (grain(x_1) ⇒ plant(x_1)) fof(pel47_6$_2$, axiom)
∀x: (animal(x) ⇒ (∀y: (plant(y) ⇒ eats(x, y)) or ∀y_1: ((animal(y_1) and much_smaller(y_1, x) and ∃z: (plant(z) and eats(y_1,
eats(x, y_1)))) fof(pel47$_7$, axiom)
∀x, y: ((bird(y) and (snail(x) or caterpillar(x))) ⇒ much_smaller(x, y)) fof(pel47$_8$, axiom)
∀x, y: ((bird(x) and fox(y)) ⇒ much_smaller(x, y)) fof(pel47$_9$, axiom)
∀x, y: ((fox(x) and wolf(y)) ⇒ much_smaller(x, y)) fof(pel47$_{10}$, axiom)
∀x, y: ((wolf(x) and (fox(y) or grain(y))) ⇒ ¬ eats(x, y)) fof(pel47$_{11}$, axiom)
∀x, y: ((bird(x) and caterpillar(y)) ⇒ eats(x, y)) fof(pel47$_{12}$, axiom)
∀x, y: ((bird(x) and snail(y)) ⇒ ¬ eats(x, y)) fof(pel47$_{13}$, axiom)

$\forall x$: ((caterpillar(x) or snail(x)) $\Rightarrow$ $\exists y$: (plant(y) and eats(x, y))) fof(pel47$_{14}$, axiom)
$\exists x, y$: (animal(x) and animal(y) and $\exists z$: (grain(z) and eats(y, z) and eats(x, y))) fof(pel$_{47}$, conjecture)

PUZ031+2.p Schubert's Steamroller
Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain eating animal.
$\exists a$: wolf(a) fof(wolf_type, axiom)
$\exists a$: fox(a) fof(fox_type, axiom)
$\exists a$: bird(a) fof(bird_type, axiom)
$\exists a$: caterpillar(a) fof(caterpillar_type, axiom)
$\exists a$: snail(a) fof(snail_type, axiom)
$\forall x$: (wolf(x) $\Rightarrow$ animal(x)) fof(pel47_1$_1$, axiom)
$\forall x$: (fox(x) $\Rightarrow$ animal(x)) fof(pel47_2$_1$, axiom)
$\forall x$: (bird(x) $\Rightarrow$ animal(x)) fof(pel47_3$_1$, axiom)
$\forall x$: (caterpillar(x) $\Rightarrow$ animal(x)) fof(pel47_4$_1$, axiom)
$\forall x$: (snail(x) $\Rightarrow$ animal(x)) fof(pel47_4$_2$, axiom)
$\exists a$: grain(a) fof(grain_type, axiom)
$\forall x$: (grain(x) $\Rightarrow$ plant(x)) fof(pel47_6$_2$, axiom)
$\forall x$: (animal(x) $\Rightarrow$ ($\forall y$: (plant(y) $\Rightarrow$ eats(x, y)) or $\forall y_1$: ((animal(y_1) and much_smaller(y_1, x) and $\exists z$: (plant(z) and eats(y_1, eats(x, y_1)))) fof(pel47$_7$, axiom)
$\forall y, x$: ((bird(y) and snail(x)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_8$, axiom)
$\forall y, x$: ((bird(y) and caterpillar(x)) $\Rightarrow$ much_smaller(x, y)) fof(pel47_8a, axiom)
$\forall x, y$: ((bird(x) and fox(y)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_9$, axiom)
$\forall x, y$: ((fox(x) and wolf(y)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_{10}$, axiom)
$\forall x, y$: ((wolf(x) and fox(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47$_{11}$, axiom)
$\forall x, y$: ((wolf(x) and grain(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47_11a, axiom)
$\forall x, y$: ((bird(x) and caterpillar(y)) $\Rightarrow$ eats(x, y)) fof(pel47$_{12}$, axiom)
$\forall x, y$: ((bird(x) and snail(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47$_{13}$, axiom)
$\forall x$: (caterpillar(x) $\Rightarrow$ $\exists y$: (plant(y) and eats(x, y))) fof(pel47$_{14}$, axiom)
$\forall x$: (snail(x) $\Rightarrow$ $\exists y$: (plant(y) and eats(x, y))) fof(pel47_14a, axiom)
$\exists x, y$: (animal(x) and animal(y) and $\exists z$: (grain(z) and eats(y, z) and eats(x, y))) fof(pel$_{47}$, conjecture)

PUZ031+3.p Schubert's Steamroller
Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain eating animal.
$\exists a$: edible(a) fof(edible_type, axiom)
$\exists a$: animal(a) fof(animal_type, axiom)
$\forall a$: (animal(a) $\Rightarrow$ edible(a)) fof(animal_is_edible, axiom)
$\exists a$: wolf(a) fof(wolf_type, axiom)
$\forall a$: (wolf(a) $\Rightarrow$ animal(a)) fof(wolf_is_animal, axiom)
$\exists a$: fox(a) fof(fox_type, axiom)
$\forall a$: (fox(a) $\Rightarrow$ animal(a)) fof(fox_is_animal, axiom)
$\exists a$: bird(a) fof(bird_type, axiom)
$\forall a$: (bird(a) $\Rightarrow$ animal(a)) fof(bird_is_animal, axiom)
$\exists a$: caterpillar(a) fof(caterpillar_type, axiom)
$\forall a$: (caterpillar(a) $\Rightarrow$ animal(a)) fof(caterpillar_is_animal, axiom)
$\exists a$: snail(a) fof(snail_type, axiom)
$\forall a$: (snail(a) $\Rightarrow$ animal(a)) fof(snail_is_animal, axiom)
$\exists a$: plant(a) fof(plant_type, axiom)
$\forall a$: (plant(a) $\Rightarrow$ edible(a)) fof(plant_is_edible, axiom)
$\exists a$: grain(a) fof(grain_type, axiom)
$\forall a$: (grain(a) $\Rightarrow$ plant(a)) fof(grain_is_plant, axiom)

$\forall x$: (animal(x) $\Rightarrow$ ($\forall y$: (plant(y) $\Rightarrow$ eats(x, y)) or $\forall y_1$: (animal(y_1) $\Rightarrow$ ((much_smaller(y_1, x) and $\exists z$: (plant(z) and eats(y_1, eats(x, y_1))))) fof(pel47$_7$, axiom)
$\forall y, x$: ((bird(y) and snail(x)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_8$, axiom)
$\forall y, x$: ((bird(y) and caterpillar(x)) $\Rightarrow$ much_smaller(x, y)) fof(pel47_8a, axiom)
$\forall x, y$: ((bird(x) and fox(y)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_9$, axiom)
$\forall x, y$: ((fox(x) and wolf(y)) $\Rightarrow$ much_smaller(x, y)) fof(pel47$_{10}$, axiom)
$\forall x, y$: ((wolf(x) and fox(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47$_{11}$, axiom)
$\forall x, y$: ((wolf(x) and grain(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47_11a, axiom)
$\forall x, y$: ((bird(x) and caterpillar(y)) $\Rightarrow$ eats(x, y)) fof(pel47$_{12}$, axiom)
$\forall x, y$: ((bird(x) and snail(y)) $\Rightarrow$ $\neg$ eats(x, y)) fof(pel47$_{13}$, axiom)
$\forall x$: (caterpillar(x) $\Rightarrow$ $\exists y$: (plant(y) and eats(x, y))) fof(pel47$_{14}$, axiom)
$\forall x$: (snail(x) $\Rightarrow$ $\exists y$: (plant(y) and eats(x, y))) fof(pel47_14a, axiom)
$\exists x, y, z$: (animal(x) and animal(y) and grain(z) and eats(y, z) and eats(x, y)) fof(pel$_{47}$, conjecture)

PUZ031-1.p Schubert's Steamroller
Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain eating animal.
wolf(x) $\Rightarrow$ animal(x) cnf(wolf_is_an_animal, axiom)
fox(x) $\Rightarrow$ animal(x) cnf(fox_is_an_animal, axiom)
bird(x) $\Rightarrow$ animal(x) cnf(bird_is_an_animal, axiom)
caterpillar(x) $\Rightarrow$ animal(x) cnf(caterpillar_is_an_animal, axiom)
snail(x) $\Rightarrow$ animal(x) cnf(snail_is_an_animal, axiom)
wolf(a_wolf) cnf(there_is_a_wolf, axiom)
fox(a_fox) cnf(there_is_a_fox, axiom)
bird(a_bird) cnf(there_is_a_bird, axiom)
caterpillar(a_caterpillar) cnf(there_is_a_caterpillar, axiom)
snail(a_snail) cnf(there_is_a_snail, axiom)
grain(a_grain) cnf(there_is_a_grain, axiom)
grain(x) $\Rightarrow$ plant(x) cnf(grain_is_a_plant, axiom)
(animal(animal) and plant(plant) and animal(small_animal) and plant(other_plant) and much_smaller(small_animal, animal)
(eats(animal, plant) or eats(animal, small_animal)) cnf(eating_habits, axiom)
(caterpillar(catapillar) and bird(bird)) $\Rightarrow$ much_smaller(catapillar, bird) cnf(caterpillar_smaller_than_bird, axiom)
(snail(snail) and bird(bird)) $\Rightarrow$ much_smaller(snail, bird) cnf(snail_smaller_than_bird, axiom)
(bird(bird) and fox(fox)) $\Rightarrow$ much_smaller(bird, fox) cnf(bird_smaller_than_fox, axiom)
(fox(fox) and wolf(wolf)) $\Rightarrow$ much_smaller(fox, wolf) cnf(fox_smaller_than_wolf, axiom)
(wolf(wolf) and fox(fox)) $\Rightarrow$ $\neg$ eats(wolf, fox) cnf(wolf_dont_eat_fox, axiom)
(wolf(wolf) and grain(grain)) $\Rightarrow$ $\neg$ eats(wolf, grain) cnf(wolf_dont_eat_grain, axiom)
(bird(bird) and caterpillar(catapillar)) $\Rightarrow$ eats(bird, catapillar) cnf(bird_eats_caterpillar, axiom)
(bird(bird) and snail(snail)) $\Rightarrow$ $\neg$ eats(bird, snail) cnf(bird_dont_eat_snail, axiom)
caterpillar(catapillar) $\Rightarrow$ plant(caterpillar_food_of(catapillar)) cnf(caterpillar_food_is_a_plant, axiom)
caterpillar(catapillar) $\Rightarrow$ eats(catapillar, caterpillar_food_of(catapillar)) cnf(caterpillar_eats_caterpillar_food, axiom)
snail(snail) $\Rightarrow$ plant(snail_food_of(snail)) cnf(snail_food_is_a_plant, axiom)
snail(snail) $\Rightarrow$ eats(snail, snail_food_of(snail)) cnf(snail_eats_snail_food, axiom)
(animal(animal) and animal(grain_eater) and grain(grain) and eats(animal, grain_eater)) $\Rightarrow$ $\neg$ eats(grain_eater, grain) cn

PUZ031^5.p TPS problem from BASIC-FO-THMS
eats: $i $\rightarrow$ $i $\rightarrow$ $o thf(eats, type)
grain: $i $\rightarrow$ $o thf(grain, type)
animal: $i $\rightarrow$ $o thf(animal, type)
snail: $i $\rightarrow$ $o thf(snail, type)
sf: $i $\rightarrow$ $i thf(sf, type)
plant: $i $\rightarrow$ $o thf(plant, type)
caterpillar: $i $\rightarrow$ $o thf(caterpillar, type)
cf: $i $\rightarrow$ $i thf(cf, type)
bird: $i $\rightarrow$ $o thf(bird, type)
wolf: $i $\rightarrow$ $o thf(wolf, type)

fox: $i → $o thf(fox, type)
msmaller: $i → $i → $o thf(msmaller, type)
a_grain: $i thf(a_grain, type)
a_snail: $i thf(a_snail, type)
a_caterpillar: $i thf(a_caterpillar, type)
a_bird: $i thf(a_bird, type)
a_fox: $i thf(a_fox, type)
a_wolf: $i thf(a_wolf, type)
¬ ∀x: $i: (animal@$x$ or ¬ wolf@x) and ∀x: $i: (animal@$x$ or ¬ fox@x) and ∀x: $i: (animal@$x$ or ¬ bird@x) and ∀x: $i: (animal

PUZ031_1.p Schubert's Steamroller
Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain eating animal.
animal: $tType tff(animal_type, type)
wolf: $tType tff(wolf_type, type)
wolf_to_animal: wolf → animal tff(wolf_is_animal, type)
fox: $tType tff(fox_type, type)
fox_to_animal: fox → animal tff(fox_is_animal, type)
bird: $tType tff(bird_type, type)
bird_to_animal: bird → animal tff(bird_is_animal, type)
caterpillar: $tType tff(caterpillar_type, type)
caterpillar_to_animal: caterpillar → animal tff(caterpillar_is_animal, type)
snail: $tType tff(snail_type, type)
snail_to_animal: snail → animal tff(snail_is_animal, type)
plant: $tType tff(plant_type, type)
grain: $tType tff(grain_type, type)
grain_to_plant: grain → plant tff(grain_is_plant, type)
edible: $tType tff(edible_type, type)
animal_to_edible: animal → edible tff(animal_is_edible, type)
plant_to_edible: plant → edible tff(plant_is_edible, type)
eats: (animal × edible) → $o tff(eats_type, type)
much_smaller: (animal × animal) → $o tff(much_smaller_type, type)
∀x: animal: (∀y: plant: eats(x, plant_to_edible(y)) or ∀y_1: animal: ((much_smaller(y_1, x) and ∃z: plant: eats(y_1, plant_to_edible
eats(x, animal_to_edible(y_1)))) tff(pel47$_7$, axiom)
∀x: snail, y: bird: much_smaller(snail_to_animal(x), bird_to_animal(y)) tff(pel47$_8$, axiom)
∀x: caterpillar, y: bird: much_smaller(caterpillar_to_animal(x), bird_to_animal(y)) tff(pel47_8a, axiom)
∀x: bird, y: fox: much_smaller(bird_to_animal(x), fox_to_animal(y)) tff(pel47$_9$, axiom)
∀x: fox, y: wolf: much_smaller(fox_to_animal(x), wolf_to_animal(y)) tff(pel47$_{10}$, axiom)
∀x: wolf, y: fox: ¬ eats(wolf_to_animal(x), animal_to_edible(fox_to_animal(y))) tff(pel47$_{11}$, axiom)
∀x: wolf, y: grain: ¬ eats(wolf_to_animal(x), plant_to_edible(grain_to_plant(y))) tff(pel47_11a, axiom)
∀x: bird, y: caterpillar: eats(bird_to_animal(x), animal_to_edible(caterpillar_to_animal(y))) tff(pel47$_{12}$, axiom)
∀x: bird, y: snail: ¬ eats(bird_to_animal(x), animal_to_edible(snail_to_animal(y))) tff(pel47$_{13}$, axiom)
∀x: caterpillar: ∃y: plant: eats(caterpillar_to_animal(x), plant_to_edible(y)) tff(pel47$_{14}$, axiom)
∀x: snail: ∃y: plant: eats(snail_to_animal(x), plant_to_edible(y)) tff(pel47_14a, axiom)
∃x: animal, y: animal, z: grain: (eats(y, plant_to_edible(grain_to_plant(z))) and eats(x, animal_to_edible(y))) tff(pel$_{47}$, conj

PUZ032-1.p Knights and Knaves #26
On a certain island the inhabitants are partitioned into those who always tell the truth and those who always lie. I landed on the island and met three inhabitants A, B, and C. I asked A, 'Are you a truthteller or a liar?' He mumbled something which I couldn't make out. I asked B what A had said. B replied, 'A said he was a liar'. C then volunteered, 'Don't believe B, he's lying!' Prove C is a truthteller.
include('Axioms/PUZ002-0.ax')
a_truth(says(a, mumble)) cnf(a_mumbles, hypothesis)
a_truth(says(b, says(a, liar(a)))) cnf(b_says_a_says_hes_a_liar, hypothesis)
a_truth(says(c, liar(b))) cnf(c_says_b_is_a_liar, hypothesis)
¬ a_truth(truthteller(c)) cnf(prove_c_is_a_truthteller, negated_conjecture)

PUZ033-1.p The Winds and the Windows Puzzle
(1) There is always sunshine when the wind is in the East. (2) When it is cold and foggy, my neighbor practices the flute. (3) When my fire smokes, I set the door open. (4) When it is cold and I feel rheumatic, I light my fire. (5) When the wind is in the East and comes in gusts, my fire smokes. (6) When I keep the door open, I am free from headache. (7) Even when the sun is shining and it is not cold, I keep my window shut if it is foggy. (8) When the wind does not come in gusts, and when I have a fire and keep the door shut, I do not feel rheumatic. (9) Sunshine always brings on fog. (10) When my neighbor practices the flute, I shut the door, even if I have no headache. (11) When there is a fog and the wind is in the East, I feel rheumatic. Show that when the wind is in the East, I keep my windows shut.
wind_in_east $\Rightarrow$ sunshine cnf(c_1, axiom)
(cold and foggy) $\Rightarrow$ neighbor_practices_flute cnf(c_2, axiom)
fire_smokes $\Rightarrow$ door_is_open cnf(c_3, axiom)
(cold and i_feel_rheumatic) $\Rightarrow$ fire_is_lit cnf(c_4, axiom)
(wind_in_east and wind_in_gusts) $\Rightarrow$ fire_smokes cnf(c_5, axiom)
door_is_open $\Rightarrow$ $\neg$ headache cnf(c_6, axiom)
(sunshine and foggy) $\Rightarrow$ (cold or window_is_shut) cnf(c_7, axiom)
(fire_is_lit and i_feel_rheumatic) $\Rightarrow$ (wind_in_gusts or door_is_open) cnf(c_8, axiom)
sunshine $\Rightarrow$ foggy cnf(c_9, axiom)
neighbor_practices_flute $\Rightarrow$ $\neg$ door_is_open cnf(c_{10}, axiom)
(foggy and wind_in_east) $\Rightarrow$ i_feel_rheumatic cnf(c_{11}, axiom)
wind_in_east cnf(c_{12}, hypothesis)
$\neg$ window_is_shut cnf(prove_window_is_shut, negated_conjecture)

PUZ034-1.003.p N queens problem
The problem is to place 3 queens on an 3x3 chess board, so that no queen can attack another.
(less(low, high) and low+$s(n_0)$=newLow and range(newLow, high, restOfNumbers)) $\Rightarrow$ range(low, high, cons(low, restOfNum
range(same, same, cons(same, empty_list)) cnf(make_list_of_numbers$_2$, axiom)
less(n_0, $s(x)$) cnf(less$_1$, axiom)
less(x, y) $\Rightarrow$ less($s(x)$, $s(y)$) cnf(less$_2$, axiom)
$x + n_0$=x cnf(add$_0$, axiom)
$x + y$=z $\Rightarrow$ $x + s(y)$=$s(z)$ cnf(add, axiom)
select(head, cons(head, tail), tail) cnf(select$_1$, axiom)
select(element, tail, newTail) $\Rightarrow$ select(element, cons(head, tail), cons(head, newTail)) cnf(select$_2$, axiom)
$\neg$ same($s(x)$, n_0) cnf(same_definition$_1$, axiom)
$\neg$ same(n_0, $s(x)$) cnf(same_definition$_2$, axiom)
same($s(x)$, $s(y)$) $\Rightarrow$ same(x, y) cnf(same_definition$_3$, axiom)
attack(queen, placedQueens) $\Rightarrow$ diagonal_attack(queen, $s(n_0)$, placedQueens) cnf(attack, axiom)
(diagonal_attack(queen, queenNumber, cons(placedQueen, otherPlacedQueens)) and diagonal$_1$+queenNumber=placedQueen
queenNumber=diagonal$_2$ and queenNumber+$s(n_0)$=nextQueenNumber) $\Rightarrow$ (same(diagonal$_1$, queen) or same(diagonal$_2$, que
$\neg$ diagonal_attack(queen, lastQueen, empty_list) cnf(check_diagonals$_2$, axiom)
(select(aQueen, unplacedQueens, restOfUnplacedQueens) and do_queens(restOfUnplacedQueens, cons(aQueen, safeQueens), p
(do_queens(unplacedQueens, safeQueens, placement) or attack(aQueen, safeQueens)) cnf(place_a_queen$_1$, axiom)
do_queens(empty_list, placement, placement) cnf(place_a_queen$_2$, axiom)
(numberOfQueens+$s(n_0)$=low and numberOfQueens+numberOfQueens=high and range(low, high, positions) and do_queens
queens(numberOfQueens, placement) cnf(set_up_queens, axiom)
$\neg$ queens($s(s(s(n_0)))$, placement) cnf(place_queens, negated_conjecture)

PUZ034-1.004.p N queens problem
The problem is to place 4 queens on an 4x4 chess board, so that no queen can attack another.
(less(low, high) and low+$s(n_0)$=newLow and range(newLow, high, restOfNumbers)) $\Rightarrow$ range(low, high, cons(low, restOfNum
range(same, same, cons(same, empty_list)) cnf(make_list_of_numbers$_2$, axiom)
less(n_0, $s(x)$) cnf(less$_1$, axiom)
less(x, y) $\Rightarrow$ less($s(x)$, $s(y)$) cnf(less$_2$, axiom)
$x + n_0$=x cnf(add$_0$, axiom)
$x + y$=z $\Rightarrow$ $x + s(y)$=$s(z)$ cnf(add, axiom)
select(head, cons(head, tail), tail) cnf(select$_1$, axiom)
select(element, tail, newTail) $\Rightarrow$ select(element, cons(head, tail), cons(head, newTail)) cnf(select$_2$, axiom)
$\neg$ same($s(x)$, n_0) cnf(same_definition$_1$, axiom)
$\neg$ same(n_0, $s(x)$) cnf(same_definition$_2$, axiom)
same($s(x)$, $s(y)$) $\Rightarrow$ same(x, y) cnf(same_definition$_3$, axiom)

attack(queen, placedQueens) ⇒ diagonal_attack(queen, $s(n_0)$, placedQueens) cnf(attack, axiom)
(diagonal_attack(queen, queenNumber, cons(placedQueen, otherPlacedQueens)) and diagonal$_1$+queenNumber=placedQueen
queenNumber=diagonal$_2$ and queenNumber+$s(n_0)$=nextQueenNumber) ⇒ (same(diagonal$_1$, queen) or same(diagonal$_2$, que
¬ diagonal_attack(queen, lastQueen, empty_list) cnf(check_diagonals$_2$, axiom)
(select(aQueen, unplacedQueens, restOfUnplacedQueens) and do_queens(restOfUnplacedQueens, cons(aQueen, safeQueens), p
(do_queens(unplacedQueens, safeQueens, placement) or attack(aQueen, safeQueens)) cnf(place_a_queen$_1$, axiom)
do_queens(empty_list, placement, placement) cnf(place_a_queen$_2$, axiom)
(numberOfQueens+$s(n_0)$=low and numberOfQueens+numberOfQueens=high and range(low, high, positions) and do_queens
queens(numberOfQueens, placement) cnf(set_up_queens, axiom)
¬ queens($s(s(s(s(n_0))))$, placement) cnf(place_queens, negated_conjecture)

PUZ035-1.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
person(x) ⇒ (isa(x, knight) or isa(x, knave)) cnf(everyone_a_knight_or_knave, axiom)
isa(x, knight) ⇒ ¬ isa(x, knave) cnf(not_both_a_knight_and_knave, axiom)
isa(x, knight) ⇒ tell_the_truth(x) cnf(knights_make_true_statements, axiom)
isa(x, knave) ⇒ lies(x) cnf(knaves_make_false_statements, axiom)
tell_the_truth(x) ⇒ ¬ lies(x) cnf(statements_are_true_or_false, axiom)
(response(yes) and tell_the_truth(asked)) ⇒ (isa(asked, knight) or isa(other, knight)) cnf(true_one_is_a_knight, axiom)
(response(no) and lies(asked)) ⇒ (isa(asked, knight) or isa(other, knight)) cnf(lie_one_is_a_knight, axiom)
(response(yes) and lies(asked)) ⇒ ¬ isa(anyone, knight) cnf(knight_answers$_1$, axiom)
(response(no) and tell_the_truth(asked)) ⇒ ¬ isa(anyone, knight) cnf(knight_answers$_2$, axiom)
(tell_the_truth(asked) and isa(anyone, knight)) ⇒ response(yes) cnf(knight_answers$_3$, axiom)
tell_the_truth(asked) ⇒ (response(no) or isa(asked, knight) or isa(other, knight)) cnf(truthful_answer, axiom)
response(yes) or response(no) cnf(two_answers, axiom)
person(asked) cnf(asked_person, axiom)
person(other) cnf(other_person, axiom)
(response(a) and isa(asked, x)) ⇒ ¬ isa(other, y) cnf(prove_answer, negated_conjecture)

PUZ035-2.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
person(x) ⇒ (isa(x, knight) or isa(x, knave)) cnf(everyone_a_knight_or_knave, axiom)
isa(x, knight) ⇒ ¬ isa(x, knave) cnf(not_both_a_knight_and_knave, axiom)
isa(x, knight) ⇒ tell_the_truth(x) cnf(knights_make_true_statements, axiom)
isa(x, knave) ⇒ lies(x) cnf(knaves_make_false_statements, axiom)
tell_the_truth(x) ⇒ ¬ lies(x) cnf(statements_are_true_or_false, axiom)
person(x) ⇒ (tell_the_truth(x) or lies(x)) cnf(statements_are_true_or_false$_2$, axiom)
(response(yes) and tell_the_truth(asked)) ⇒ (isa(asked, knight) or isa(other, knight)) cnf(true_one_is_a_knight, axiom)
(response(no) and lies(asked)) ⇒ (isa(asked, knight) or isa(other, knight)) cnf(lie_one_is_a_knight, axiom)
(response(yes) and lies(asked)) ⇒ ¬ isa(anyone, knight) cnf(knight_answers$_1$, axiom)
(response(no) and tell_the_truth(asked)) ⇒ ¬ isa(anyone, knight) cnf(knight_answers$_2$, axiom)
(tell_the_truth(asked) and isa(anyone, knight)) ⇒ response(yes) cnf(knight_answers$_3$, axiom)
tell_the_truth(asked) ⇒ (response(no) or isa(asked, knight) or isa(other, knight)) cnf(truthful_answer, axiom)
response(yes) or response(no) cnf(two_answers, axiom)
person(asked) cnf(asked_person, axiom)
person(other) cnf(other_person, axiom)
(response(a) and isa(asked, x)) ⇒ ¬ isa(other, y) cnf(prove_answer, negated_conjecture)

PUZ035-3.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
person(p) ⇒ (truth(isa(p, knight)) or truth(isa(p, knave))) cnf(everyone_a_knight_or_knave, axiom)
truth(isa(p, knight)) ⇒ ¬ truth(isa(p, knave)) cnf(not_both_a_knight_and_knave, axiom)
(says(p, s) and truth(isa(p, knight))) ⇒ truth(s) cnf(knights_make_true_statements$_1$, axiom)
(says(p, s) and truth(s)) ⇒ truth(isa(p, knight)) cnf(knights_make_true_statements$_2$, axiom)
(says(p, s) and truth(isa(p, knave))) ⇒ truth(not(s)) cnf(knaves_make_false_statements$_1$, axiom)

(says(p, s) and truth(not(s))) $\Rightarrow$ truth(isa(p, knave)) cnf(knaves_make_false_statements$_2$, axiom)
says(asked, or(isa(asked, knight), isa(other, knight))) or says(asked, not(or(isa(asked, knight), isa(other, knight)))) cnf(one
says(anyone, s) $\Rightarrow$ (truth(s) or truth(not(s))) cnf(statements_are_true_or_false, axiom)
says(anyone, not(s)) $\Rightarrow$ (truth(s) or truth(not(s))) cnf(statements_are_true_or_false$_2$, axiom)
truth(s) $\Rightarrow$ $\neg$ truth(not(s)) cnf(statements_are_true_or_false$_3$, axiom)
truth(or(s_1, s_2)) $\Rightarrow$ (truth(s_1) or truth(s_2)) cnf(or_def, axiom)
truth(s_1) $\Rightarrow$ $\neg$ truth(not(or(s_1, anything))) cnf(not$_1$, axiom)
truth(s_2) $\Rightarrow$ $\neg$ truth(not(or(anything, s_2))) cnf(not$_2$, axiom)
person(asked) cnf(asked_person, axiom)
person(other) cnf(other_person, axiom)
(truth(isa(asked, p_1)) and truth(isa(other, p_2))) $\Rightarrow$ $\neg$ says(asked, s) cnf(prove_answer, negated_conjecture)

PUZ035-4.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
person(p) $\Rightarrow$ (truth(isa(p, knight)) or truth(isa(p, knave))) cnf(everyone_a_knight_or_knave, axiom)
truth(isa(p, knight)) $\Rightarrow$ $\neg$ truth(isa(p, knave)) cnf(not_both_a_knight_and_knave, axiom)
(says(p, s) and truth(isa(p, knight))) $\Rightarrow$ truth(s) cnf(knights_make_true_statements$_1$, axiom)
(says(p, s) and truth(s)) $\Rightarrow$ truth(isa(p, knight)) cnf(knights_make_true_statements$_2$, axiom)
says(asked, or(isa(asked, knight), isa(other, knight))) or says(asked, not(or(isa(asked, knight), isa(other, knight)))) cnf(one
says(anyone, s) $\Rightarrow$ (truth(s) or truth(not(s))) cnf(statements_are_true_or_false, axiom)
says(anyone, not(s)) $\Rightarrow$ (truth(s) or truth(not(s))) cnf(statements_are_true_or_false$_2$, axiom)
truth(s) $\Rightarrow$ $\neg$ truth(not(s)) cnf(statements_are_true_or_false$_3$, axiom)
truth(or(s_1, s_2)) $\Rightarrow$ (truth(s_1) or truth(s_2)) cnf(or_def, axiom)
truth(s_1) $\Rightarrow$ $\neg$ truth(not(or(s_1, anything))) cnf(not$_1$, axiom)
truth(s_2) $\Rightarrow$ $\neg$ truth(not(or(anything, s_2))) cnf(not$_2$, axiom)
person(asked) cnf(asked_person, axiom)
person(other) cnf(other_person, axiom)
(truth(isa(asked, p_1)) and truth(isa(other, p_2))) $\Rightarrow$ $\neg$ says(asked, s) cnf(prove_answer, negated_conjecture)

PUZ035-5.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
truth(isa(p, knight)) or truth(isa(p, knave)) cnf(everyone_a_knight_or_knave, axiom)
truth(isa(p, knight)) $\Rightarrow$ $\neg$ truth(isa(p, knave)) cnf(not_both_a_knight_and_knave, axiom)
(truth(isa(p, knight)) and says(p, s)) $\Rightarrow$ truth(s) cnf(knights_make_true_statements$_1$, axiom)
(truth(s) and says(p, s)) $\Rightarrow$ truth(isa(p, knight)) cnf(knights_make_true_statements$_2$, axiom)
truth(or(a, b)) $\Rightarrow$ (truth(a) or truth(b)) cnf(or$_1$, axiom)
truth(a) $\Rightarrow$ truth(or(a, b)) cnf(or$_2$, axiom)
truth(b) $\Rightarrow$ truth(or(a, b)) cnf(or$_3$, axiom)
says(asked, or(isa(asked, knight), isa(other, knight))) cnf(says_yes, axiom)
truth(isa(asked, x)) $\Rightarrow$ $\neg$ truth(isa(other, y)) cnf(query, negated_conjecture)

PUZ035-6.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
truth(isa(p, knight)) or truth(isa(p, knave)) cnf(everyone_a_knight_or_knave, axiom)
truth(isa(p, knight)) $\Rightarrow$ $\neg$ truth(isa(p, knave)) cnf(not_both_a_knight_and_knave, axiom)
(truth(isa(p, knight)) and says(p, s)) $\Rightarrow$ truth(s) cnf(knights_make_true_statements$_1$, axiom)
(truth(s) and says(p, s)) $\Rightarrow$ truth(isa(p, knight)) cnf(knights_make_true_statements$_2$, axiom)
truth(or(a, b)) $\Rightarrow$ (truth(a) or truth(b)) cnf(or$_1$, axiom)
truth(a) $\Rightarrow$ truth(or(a, b)) cnf(or$_2$, axiom)
truth(b) $\Rightarrow$ truth(or(a, b)) cnf(or$_3$, axiom)
truth(c) or truth(not(c)) cnf(not$_1$, axiom)
truth(c) $\Rightarrow$ $\neg$ truth(not(c)) cnf(not$_2$, axiom)
says(asked, not(or(isa(asked, knight), isa(other, knight)))) cnf(says_yes, axiom)
truth(isa(asked, x)) $\Rightarrow$ $\neg$ truth(isa(other, y)) cnf(query, negated_conjecture)

PUZ035-7.p Knights and Knaves #36
On an island, there live exactly two types of people: knights and knaves. Knights always tell the truth and knaves always lie. I landed on the island, met two inhabitants, asked one of them: "Is one of you a knight?" and he answered me. What can be said about the types of the asked and the other person depending on the answer I get?
$\text{truth}(\text{isa}(p, \text{knight}))$ or $\text{truth}(\text{isa}(p, \text{knave}))$ cnf(everyone_a_knight_or_knave, axiom)
$\text{truth}(\text{isa}(p, \text{knight})) \Rightarrow \neg\, \text{truth}(\text{isa}(p, \text{knave}))$ cnf(not_both_a_knight_and_knave, axiom)
$(\text{truth}(\text{isa}(p, \text{knight}))$ and $\text{reply}(p, q, \text{yes})) \Rightarrow \text{truth}(q)$ $\text{cnf}(\text{knights\_make\_true\_statements}_1, \text{axiom})$
$(\text{truth}(q)$ and $\text{reply}(p, q, \text{yes})) \Rightarrow \text{truth}(\text{isa}(p, \text{knight}))$ $\text{cnf}(\text{knights\_make\_true\_statements}_2, \text{axiom})$
$(\text{truth}(q)$ and $\text{truth}(\text{isa}(p, \text{knight}))) \Rightarrow \text{reply}(p, q, \text{yes})$ $\text{cnf}(\text{knights\_make\_true\_statements}_3, \text{axiom})$
$\text{truth}(q)$ or $\text{truth}(\text{isa}(p, \text{knight}))$ or $\text{reply}(p, q, \text{yes})$ $\text{cnf}(\text{knights\_make\_true\_statements}_4, \text{axiom})$
$\text{reply}(p, q, \text{yes})$ or $\text{reply}(p, q, \text{no})$ cnf(yes_or_no, axiom)
$\text{reply}(p, q, \text{yes}) \Rightarrow \neg\, \text{reply}(p, q, \text{no})$ cnf(not_yes_and_no, axiom)
$\text{truth}(c)$ or $\text{truth}(\text{not}(c))$ $\text{cnf}(\text{true}_1, \text{axiom})$
$\text{truth}(c) \Rightarrow \neg\, \text{truth}(\text{not}(c))$ $\text{cnf}(\text{true}_2, \text{axiom})$
$\text{truth}(\text{or}(a, b)) \Rightarrow (\text{truth}(a)$ or $\text{truth}(b))$ $\text{cnf}(\text{or}_1, \text{axiom})$
$\text{truth}(a) \Rightarrow \text{truth}(\text{or}(a, b))$ $\text{cnf}(\text{or}_2, \text{axiom})$
$\text{truth}(b) \Rightarrow \text{truth}(\text{or}(a, b))$ $\text{cnf}(\text{or}_3, \text{axiom})$
$(\text{reply}(\text{asked}, \text{or}(\text{isa}(\text{asked}, \text{knight}), \text{isa}(\text{other}, \text{knight})), r)$ and $\text{truth}(\text{isa}(\text{asked}, x))) \Rightarrow \neg\, \text{truth}(\text{isa}(\text{other}, y))$ cnf(prove_ans

PUZ036-1.005.p TopSpin
TopSpin consists of a circular track with 20 pieces numbered 1..20 placed in the track, with a turnstile in the track that always holds four consecutive pieces. There are three legal moves in TopSpin: slide all the pieces round the track in either direction, or flip the turnstile. Given any initial board with scrambled pieces on the track, the problem is to find a sequence of moves that unscrambles the pieces.
$\neg\, \text{state}(p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{19}, p_{20})$ cnf(make_like_this, negated_conjecture)
$\text{state}(p_5, p_4, p_3, p_2, p_1, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{19}, p_{20})$ cnf(initial_configuration, hypothesis)
$\text{state}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}) \Rightarrow \text{state}(x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10},$
$\text{state}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}) \Rightarrow \text{state}(x_{20}, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8,$
$\text{state}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}) \Rightarrow \text{state}(x_4, x_3, x_2, x_1, x_5, x_6, x_7, x_8, x_9, x$

PUZ038-1.p Quo vadis 1
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is a simple goal for testing.
include('Axioms/PUZ004-0.ax')
$\text{state}(\text{bb}(o, s(o)), v_1(o, o), v_2(o, s(s(s(o)))), v_3(s(s(o)), o), v_4(s(s(o)), s(s(s(o)))), h(s(s(o)), s(o)), s_1(s(s(s(o))), s(o)), s_2(s(s(s(o)$
$\neg\, \text{state}(b, v_1, v_2, v_3(s(s(s(o))), o), v_4, h, s_1, s_2, s_3, s_4, e_1, e_2)$ cnf(goal_state, negated_conjecture)

PUZ039-1.p Quo vadis 2
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is an intermediate goal for testing.
include('Axioms/PUZ004-0.ax')
$\text{state}(\text{bb}(o, s(o)), v_1(o, o), v_2(o, s(s(s(o)))), v_3(s(s(o)), o), v_4(s(s(o)), s(s(s(o)))), h(s(s(o)), s(o)), s_1(s(s(s(o))), s(o)), s_2(s(s(s(o)$
$\neg\, \text{state}(b, v_1, v_2, v_3, v_4(s(s(o)), s(s(o))), h, s_1, s_2, s_3, s_4, e_1, e_2)$ cnf(goal_state, negated_conjecture)

PUZ040-1.p Quo vadis 3
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is an intermediate goal for testing.
include('Axioms/PUZ004-0.ax')
$\text{state}(\text{bb}(o, s(o)), v_1(o, o), v_2(o, s(s(s(o)))), v_3(s(s(o)), o), v_4(s(s(o)), s(s(s(o)))), h(s(s(o)), s(o)), s_1(s(s(s(o))), s(o)), s_2(s(s(s(o)$
$\neg\, \text{state}(\text{bb}(o, o), v_1, v_2, v_3, v_4, h, s_1, s_2, s_3, s_4, e_1, e_2)$ cnf(goal_state, negated_conjecture)

PUZ041-1.p Quo vadis 4
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is the true goal from the puzzle.
include('Axioms/PUZ004-0.ax')
$\text{state}(\text{bb}(o, s(o)), v_1(o, o), v_2(o, s(s(s(o)))), v_3(s(s(o)), o), v_4(s(s(o)), s(s(s(o)))), h(s(s(o)), s(o)), s_1(s(s(s(o))), s(o)), s_2(s(s(s(o)$
$\neg\, \text{state}(\text{bb}(s(s(s(o))), s(o)), v_1, v_2, v_3, v_4, h, s_1, s_2, s_3, s_4, e_1, e_2)$ cnf(goal_state, negated_conjecture)

PUZ042-1.p Quo vadis 5

bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is a medium goal for testing.
include('Axioms/PUZ004-0.ax')
state(bb(o, $s(o)$), $v_1(o, o)$, $v_2(o, s(s(o)))$, $v_3(s(s(o)), o)$, $v_4(s(s(o)), s(s(o)))$, $h(s(s(o)), s(o))$, $s_1(s(s(s(o))), s(o))$, $s_2(s(s(o)$
$\neg$ state($b, v_1, v_2, v_3, v_4, h(s(s(s(o))), s(o)), s_1, s_2, s_3, s_4, e_1, e_2$) cnf(goal_state, negated_conjecture)

PUZ043-1.p Mars and Venus axioms
include('Axioms/PUZ001-0.ax')

PUZ044-1.p Truthtellers and Liars axioms for two types of people
Axioms for two types of people; truthtellers and liars.
include('Axioms/PUZ002-0.ax')

PUZ045-1.p Truthtellers and Liars axioms for three types of people
Axioms for three types of people; truthtellers, liars and normal people.
include('Axioms/PUZ003-0.ax')

PUZ046-1.p Quo vadis axioms
include('Axioms/PUZ004-0.ax')

PUZ047+1.p Taking the wolf, goat, and cabbage across river
(p(south, south, south, south, start) and $\forall t$: (p(south, north, south, north, t) $\Rightarrow$ p(north, north, south, north, go_alone(t))) and
p(south, north, south, north, go_alone(t_1))) and $\forall t_2$: (p(south, south, north, south, t_2) $\Rightarrow$ p(north, south, north, south, go_alone
p(south, south, north, south, go_alone(t_3))) and $\forall t_4$: (p(south, south, south, north, t_4) $\Rightarrow$ p(north, north, south, north, take_wol
p(south, south, south, north, take_wolf(t_5))) and $\forall t_6$: (p(south, south, north, south, t_6) $\Rightarrow$ p(north, north, north, south, take_wo
p(south, south, north, south, take_wolf(t_7))) and $\forall x, y, u$: (p(south, x, south, y, u) $\Rightarrow$ p(north, x, north, y, take_goat(u))) and $\forall z$
p(south, x_1, south, y_1, take_goat(v))) and $\forall t_8$: (p(south, north, south, south, t_8) $\Rightarrow$ p(north, north, south, north, take_cabbage(t
p(south, north, south, south, take_cabbage(t_9))) and $\forall u_1$: (p(south, south, north, south, u_1) $\Rightarrow$ p(north, south, north, north, tak
p(south, south, north, south, take_cabbage(v_1)))) $\Rightarrow$ $\exists z$: p(north, north, north, north, z) fof(thm_{100}, conjecture)

PUZ047-1.p Taking the wolf, goat, and cabbage across river
p(south, south, south, south, start) cnf(thm100_1, negated_conjecture)
p(south, north, south, north, a) $\Rightarrow$ p(north, north, south, north, go_alone(a)) cnf(thm100_2, negated_conjecture)
p(north, north, south, north, a) $\Rightarrow$ p(south, north, south, north, go_alone(a)) cnf(thm100_3, negated_conjecture)
p(south, south, north, south, a) $\Rightarrow$ p(north, south, north, south, go_alone(a)) cnf(thm100_4, negated_conjecture)
p(north, south, north, south, a) $\Rightarrow$ p(south, south, north, south, go_alone(a)) cnf(thm100_5, negated_conjecture)
p(south, south, south, north, a) $\Rightarrow$ p(north, north, south, north, take_wolf(a)) cnf(thm100_6, negated_conjecture)
p(north, north, south, north, a) $\Rightarrow$ p(south, south, south, north, take_wolf(a)) cnf(thm100_7, negated_conjecture)
p(south, south, north, south, a) $\Rightarrow$ p(north, north, north, south, take_wolf(a)) cnf(thm100_8, negated_conjecture)
p(north, north, north, south, a) $\Rightarrow$ p(south, south, north, south, take_wolf(a)) cnf(thm100_9, negated_conjecture)
p(south, a, south, b, c) $\Rightarrow$ p(north, a, north, b, take_goat(c)) cnf(thm100_{10}, negated_conjecture)
p(north, a, north, b, c) $\Rightarrow$ p(south, a, south, b, take_goat(c)) cnf(thm100_{11}, negated_conjecture)
p(south, north, south, south, a) $\Rightarrow$ p(north, north, south, north, take_cabbage(a)) cnf(thm100_{12}, negated_conjecture)
p(north, north, south, north, a) $\Rightarrow$ p(south, north, south, south, take_cabbage(a)) cnf(thm100_{13}, negated_conjecture)
p(south, south, north, south, a) $\Rightarrow$ p(north, south, north, north, take_cabbage(a)) cnf(thm100_{14}, negated_conjecture)
p(north, south, north, north, a) $\Rightarrow$ p(south, south, north, south, take_cabbage(a)) cnf(thm100_{15}, negated_conjecture)
$\neg$ p(north, north, north, north, a) cnf(thm100_{16}, negated_conjecture)

PUZ047∧5.p TPS problem THM100A
A naive formalization of the problem of moving man wolf goat cabbage from south to north side of river.
a: $tType thf(a_type, type)
b: $tType thf(b_type, type)
cN: a thf(cN, type)
cP: $a \to a \to a \to a \to b \to$ $o thf(cP, type)
cD: $b \to b$ thf(cD, type)
cS: a thf(cS, type)
cG: $b \to b$ thf(cG, type)
cW: $b \to b$ thf(cW, type)
cL: $b \to b$ thf(cL, type)
cO: b thf(cO, type)
(cP@cS@cS@cS@cS@cO and $\forall t$: b: ((cP@cS@cN@cS@cN@t) $\Rightarrow$ (cP@cN@cN@cS@cN@(cL@t))) and $\forall t_1$: b: ((cP@cN@cN@c
(cP@cS@cN@cS@cN@(cL@t_1))) and $\forall t_2$: b: ((cP@cS@cS@cN@cS@t_2) $\Rightarrow$ (cP@cN@cS@cN@cS@(cL@t_2))) and $\forall t_3$: b: ((cP@

(cP@cS@cS@cN@cS@(cL@t_3))) and $\forall t_4$: b: ((cP@cS@cS@cS@cN@t_4) $\Rightarrow$ (cP@cN@cN@cS@cN@(cW@t_4))) and $\forall t_5$: b: ((cP@
(cP@cS@cS@cS@cN@(cW@t_5))) and $\forall t_6$: b: ((cP@cS@cS@cN@cS@t_6) $\Rightarrow$ (cP@cN@cN@cN@cS@(cW@t_6))) and $\forall t_7$: b: ((cP
(cP@cS@cS@cN@cS@(cW@t_7))) and $\forall x$: a, y: a, u: b: ((cP@cS@x@cS@y@u) $\Rightarrow$ (cP@cN@x@cN@y@(cG@u))) and $\forall x_1$: a, y_1
(cP@cS@x_1@cS@y_1@(cG@v))) and $\forall t_8$: b: ((cP@cS@cN@cS@cS@t_8) $\Rightarrow$ (cP@cN@cN@cS@cN@(cD@t_8))) and $\forall t_9$: b: ((cP@c
(cP@cS@cN@cS@cS@(cD@t_9))) and $\forall u_1$: b: ((cP@cS@cS@cN@cS@u_1) $\Rightarrow$ (cP@cN@cS@cN@cN@(cD@u_1))) and $\forall v_1$: b: ((cP
(cP@cS@cS@cN@cS@(cD@v_1)))) $\Rightarrow$ $\exists z$: b: (cP@cN@cN@cN@cN@z) thf(cTHM100A, conjecture)

PUZ048-1.p Quo vadis 6 - initial to intermediate
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is the true goal from the puzzle.
include('Axioms/PUZ004-0.ax')
state(bb(o, $s(o)$), $v_1(o, o)$, $v_2(o, s(s(s(o))))$, $v_3(s(s(o)), o)$, $v_4(s(s(o)), s(s(s(o))))$, $h(s(s(o)), s(o))$, $s_1(s(s(s(o))), s(o))$, $s_2(s(s(s(o)$
$\neg$ state(bb($s(s(o))$, $s(s(o))$), $v_1(o, o)$, $v_2(o, s(o))$, $v_3(o, s(s(o)))$, $v_4(o, s(s(s(o))))$, $h(s(s(s(s(o)))), o)$, $s_1(s(s(o)), o)$, $s_2(s(s(o)), s(o)$

PUZ049-1.p Quo vadis 6 - intermediate to final
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is the true goal from the puzzle.
include('Axioms/PUZ004-0.ax')
state(bb($s(s(o))$, $s(s(o))$), $v_1(o, o)$, $v_2(o, s(o))$, $v_3(o, s(s(o)))$, $v_4(o, s(s(s(o))))$, $h(s(s(s(s(o)))), o)$, $s_1(s(s(o)), o)$, $s_2(s(s(o)), s(o))$,
$\neg$ state(bb($s(s(s(o)))$, $s(s(o))$, $v_1, v_2, v_3, v_4, h, s_1, s_2, s_3, s_4, e_1, e_2$) cnf(goal_state, negated_conjecture)

PUZ050-1.p Quo vadis 6 - initial to intermediate
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is the true goal from the puzzle.
include('Axioms/PUZ004-0.ax')
state(bb(o, $s(o)$), $v_1(o, o)$, $v_2(o, s(s(s(o))))$, $v_3(s(s(o)), o)$, $v_4(s(s(o)), s(s(s(o))))$, $h(s(s(o)), s(o))$, $s_1(s(s(s(o))), s(o))$, $s_2(s(s(s(o)$
$\neg$ state(bb(o, o), $v_1(s(s(o)), o)$, $v_2(o, s(s(s(o))))$, $v_3(o, s(s(o)))$, $v_4(s(s(s(o))), s(s(o)))$, $h(s(s(s(s(o)))), o)$, $s_1(s(s(o)), s(o))$, $s_2(s(s$

PUZ051-1.p Quo vadis 6 - intermediate to final
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is the true goal from the puzzle.
include('Axioms/PUZ004-0.ax')
state(bb(o, o), $v_1(s(s(o)), o)$, $v_2(o, s(s(s(o))))$, $v_3(o, s(s(o)))$, $v_4(s(s(s(o))), s(s(o)))$, $h(s(s(s(s(o)))), o)$, $s_1(s(s(o)), s(o))$, $s_2(s(s(
$\neg$ state(bb($s(s(s(o)))$, $s(s(o))$, $v_1, v_2, v_3, v_4, h, s_1, s_2, s_3, s_4, e_1, e_2$) cnf(goal_state, negated_conjecture)

PUZ054-1.p Take black and white balls from a bag
Start with a bag with 10 white balls and 9 black balls. Take out two balls: if they have the same color, put a black ball back; if they have a different color, put a white ball back. The last ball left cannot be white.
$p(s(s(s(s(s(s(s(s(s(s(n_0)))))))))), s(s(s(s(s(s(s(s(s(n_0))))))))))$ cnf(initial_state, axiom)
$p(s(s(x)), y) \Rightarrow p(x, s(y))$ cnf(two_whites_out_one_black_in, axiom)
$p(x, s(s(y))) \Rightarrow p(x, s(y))$ cnf(two_blacks_out_one_black_in, axiom)
$p(s(x), s(y)) \Rightarrow p(s(x), y)$ cnf(two_different_balls_out_one_white_in, axiom)
$\neg p(s(n_0), n_0)$ cnf(goal_state, negated_conjecture)

PUZ055-1.p Show that Sam Loyd's fifteen-puzzle is not solvable
The classic fifteen-puzzle with the 14 and 15 interchanged. Show that it is not solvable by showing satisfiability of the following problem.
state($l(n_1, l(n_2, l(n_3, l(n_4, l(\text{end}, l(n_5, l(n_6, l(n_7, l(n_8, l(\text{end}, l(n_9, l(n_{10}, l(n_{11}, l(n_{12}, l(\text{end}, l(n_{13}, l(n_{15}, l(n_{14}, l(x, l(\text{end}, \text{nil})))))))$
$\neg$ state($l(n_1, l(n_2, l(n_3, l(n_4, l(\text{end}, l(n_5, l(n_6, l(n_7, l(n_8, l(\text{end}, l(n_9, l(n_{10}, l(n_{11}, l(n_{12}, l(\text{end}, l(n_{13}, l(n_{14}, l(n_{15}, l(x, l(\text{end}, \text{nil})))))$
$l(x, l(a, r)) = l(a, l(x, r))$ or $a = \text{end}$ cnf(shift_sideways, axiom)
$l(x, l(a, l(b, l(c, l(d, l(e, r)))))) = l(e, l(a, l(b, l(c, l(d, l(x, r))))))$ cnf(shift_up_down, axiom)

PUZ056-1.p Tower of Hanoi
state(nextto(on(small, on(medium, on(large, pin(n_1)))), nextto(pin(n_2), pin(n_3)))) cnf(initial_state, axiom)
$\neg$ state(nextto(pin(n_1), nextto(pin(n_2), on(small, on(medium, on(large, pin(n_3))))))) cnf(final_state, negated_conjecture)
nextto(x, y) = nextto(y, x) cnf(nextto_commutative, axiom)
nextto(x, nextto(y, z)) = nextto(nextto(x, y), z) cnf(nextto_associative, axiom)
(fits(x, b) and fits(x, a)) $\Rightarrow$ nextto(on(x, a), b) = nextto(a, on(x, b)) cnf(a_move, axiom)
fits(small, pin(n)) cnf(fits$_1$, axiom)

$\text{fits}(\text{medium}, \text{pin}(n))$ $\text{cnf}(\text{fits}_2, \text{axiom})$
$\text{fits}(\text{large}, \text{pin}(n))$ $\text{cnf}(\text{fits}_3, \text{axiom})$
$\text{fits}(\text{small}, \text{on}(\text{medium}, a))$ $\text{cnf}(\text{fits}_4, \text{axiom})$
$\text{fits}(\text{medium}, \text{on}(\text{large}, a))$ $\text{cnf}(\text{fits}_5, \text{axiom})$
$\text{fits}(\text{small}, \text{on}(\text{large}, a))$ $\text{cnf}(\text{fits}_6, \text{axiom})$

PUZ056-2.005.p Towers of Hanoi k=05
Each instance encodes Tower of Hanoi with n discs as a reachability problem.
$p(i, t_1, t_2, t_3, t_4) \Rightarrow p(j, t_1, t_2, t_3, t_4)$ $\text{cnf}(\text{rule}_1, \text{axiom})$
$(p(t_0, i, t_2, t_3, t_4)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)) \Rightarrow p(t_0, j, t_2, t_3, t_4)$ $\text{cnf}(\text{rule}_2, \text{axiom})$
$(p(t_0, t_1, i, t_3, t_4)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)) \Rightarrow p(t_0, t_1, j, t_3, t_4)$ $\text{cnf}(\text{rule}_3, \text{axiom})$
$(p(t_0, t_1, t_2, i, t_4)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)) \Rightarrow p(t_0, t_1, t_2, j, t_4)$
$(p(t_0, t_1, t_2, t_3, i)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and $\text{neq}(t_3, i)$ and neq
$p(t_0, t_1, t_2, t_3, j)$ $\text{cnf}(\text{rule}_5, \text{axiom})$
$\neg\, \text{neq}(s_0, s_0)$ $\text{cnf}(\text{neq}_1, \text{axiom})$
$\text{neq}(s_0, s_1)$ $\text{cnf}(\text{neq}_2, \text{axiom})$
$\text{neq}(s_0, s_2)$ $\text{cnf}(\text{neq}_3, \text{axiom})$
$\text{neq}(s_1, s_0)$ $\text{cnf}(\text{neq}_4, \text{axiom})$
$\neg\, \text{neq}(s_1, s_1)$ $\text{cnf}(\text{neq}_5, \text{axiom})$
$\text{neq}(s_1, s_2)$ $\text{cnf}(\text{neq}_6, \text{axiom})$
$\text{neq}(s_2, s_0)$ $\text{cnf}(\text{neq}_7, \text{axiom})$
$\text{neq}(s_2, s_1)$ $\text{cnf}(\text{neq}_8, \text{axiom})$
$\neg\, \text{neq}(s_2, s_2)$ $\text{cnf}(\text{neq}_9, \text{axiom})$
$p(s_0, s_0, s_0, s_0, s_0)$ $\text{cnf}(\text{init}, \text{axiom})$
$\neg\, p(s_2, s_2, s_2, s_2, s_2)$ $\text{cnf}(\text{goal}, \text{negated\_conjecture})$

PUZ056-2.010.p Towers of Hanoi k=10
Each instance encodes Tower of Hanoi with n discs as a reachability problem.
$p(i, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9) \Rightarrow p(j, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_1, \text{axiom})$
$(p(t_0, i, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)) \Rightarrow p(t_0, j, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_2, \text{axiom})$
$(p(t_0, t_1, i, t_3, t_4, t_5, t_6, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)) \Rightarrow p(t_0, t_1, j, t_3, t_4, t_5, t_6, t_7, t_8, t_9)$
$(p(t_0, t_1, t_2, i, t_4, t_5, t_6, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)) \Rightarrow$
$p(t_0, t_1, t_2, j, t_4, t_5, t_6, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_4, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, i, t_5, t_6, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, j, t_5, t_6, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_5, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, t_4, i, t_6, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, t_4, j, t_6, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_6, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, t_4, t_5, i, t_7, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, t_4, t_5, j, t_7, t_8, t_9)$ $\text{cnf}(\text{rule}_7, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, i, t_8, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, j, t_8, t_9)$ $\text{cnf}(\text{rule}_8, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, i, t_9)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, j, t_9)$ $\text{cnf}(\text{rule}_9, \text{axiom})$
$(p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, i)$ and $\text{neq}(t_0, i)$ and $\text{neq}(t_0, j)$ and $\text{neq}(t_1, i)$ and $\text{neq}(t_1, j)$ and $\text{neq}(t_2, i)$ and $\text{neq}(t_2, j)$ and neq
$p(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, j)$ $\text{cnf}(\text{rule}_{10}, \text{axiom})$
$\neg\, \text{neq}(s_0, s_0)$ $\text{cnf}(\text{neq}_1, \text{axiom})$
$\text{neq}(s_0, s_1)$ $\text{cnf}(\text{neq}_2, \text{axiom})$
$\text{neq}(s_0, s_2)$ $\text{cnf}(\text{neq}_3, \text{axiom})$
$\text{neq}(s_1, s_0)$ $\text{cnf}(\text{neq}_4, \text{axiom})$
$\neg\, \text{neq}(s_1, s_1)$ $\text{cnf}(\text{neq}_5, \text{axiom})$
$\text{neq}(s_1, s_2)$ $\text{cnf}(\text{neq}_6, \text{axiom})$
$\text{neq}(s_2, s_0)$ $\text{cnf}(\text{neq}_7, \text{axiom})$
$\text{neq}(s_2, s_1)$ $\text{cnf}(\text{neq}_8, \text{axiom})$
$\neg\, \text{neq}(s_2, s_2)$ $\text{cnf}(\text{neq}_9, \text{axiom})$
$p(s_0, s_0, s_0, s_0, s_0, s_0, s_0, s_0, s_0, s_0)$ $\text{cnf}(\text{init}, \text{axiom})$
$\neg\, p(s_2, s_2, s_2, s_2, s_2, s_2, s_2, s_2, s_2, s_2)$ $\text{cnf}(\text{goal}, \text{negated\_conjecture})$

PUZ057-1.p Show the Hanoi problem is not solvable anymore
Move blocks from pin to pin, until a goal configuration is reached. Some rules are removed: one can only move a block to a pin next to it; one cannot place the small block on the large block.
$\text{state}(\text{nextto}(\text{on}(\text{small}, \text{on}(\text{medium}, \text{on}(\text{large}, \text{pin}(n_1)))), \text{nextto}(\text{pin}(n_2), \text{pin}(n_3))))$ $\text{cnf}(\text{initial\_state}, \text{axiom})$

$\neg$ state(nextto(pin(n_1), nextto(pin(n_2), on(small, on(medium, on(large, pin(n_3))))))) cnf(final_state, negated_conjecture)
nextto(x, nextto(y, z)) = nextto(nextto(x, y), z) cnf(nextto_associative, axiom)
(fits(x, b) and fits(x, a)) $\Rightarrow$ nextto(on(x, a), b) = nextto(a, on(x, b)) cnf(a_move, axiom)
fits(small, pin(n)) cnf(fits$_1$, axiom)
fits(medium, pin(n)) cnf(fits$_2$, axiom)
fits(large, pin(n)) cnf(fits$_3$, axiom)
fits(small, on(medium, a)) cnf(fits$_4$, axiom)
fits(medium, on(large, a)) cnf(fits$_5$, axiom)

PUZ058-1.p Show the Hanoi problem is not solvable anymore
Move blocks from pin to pin, until a goal configuration is reached. Some rules are removed: one cannot move the medium block onto the second pin directly.
state(nextto(on(small, on(medium, on(large, pin(n_1)))), nextto(pin(n_2), pin(n_3)))) cnf(initial_state, axiom)
$\neg$ state(nextto(pin(n_1), nextto(pin(n_2), on(small, on(medium, on(large, pin(n_3))))))) cnf(final_state, negated_conjecture)
nextto(x, y) = nextto(y, x) cnf(nextto_commutative, axiom)
nextto(x, nextto(y, z)) = nextto(nextto(x, y), z) cnf(nextto_associative, axiom)
(fits(x, b) and fits(x, a)) $\Rightarrow$ nextto(on(x, a), b) = nextto(a, on(x, b)) cnf(a_move, axiom)
fits(small, pin(n)) cnf(fits$_1$, axiom)
fits(medium, pin(n)) or $n = n_2$ cnf(fits$_2$, axiom)
fits(large, pin(n)) cnf(fits$_3$, axiom)
fits(small, on(medium, a)) cnf(fits$_4$, axiom)
fits(medium, on(large, a)) cnf(fits$_5$, axiom)
fits(small, on(large, a)) cnf(fits$_6$, axiom)

PUZ059-1.p Quo vadis 7 - an unreachable state
bb is big block (square, size=4 tiles). s1-s4 : 4 small square blocks, size=1 tile v1-v4: 4 vertical blocks, size= 2 tiles b1: horizontal block, size= 2 tiles e1, e2 are the 2 blank tiles It's a 5x4 playing field to move from the start state to the goal state. This is a simple goal for testing.
include('Axioms/PUZ004-0.ax')
state(bb(o, $s(o)$), $v_1(o, o)$, $v_2(o, s(s(s(o))))$, $v_3(s(s(o)), o)$, $v_4(s(s(o)), s(s(s(o))))$, $h(s(s(o)), s(o))$, $s_1(s(s(s(o))), s(o))$, $s_2(s(s(o)$
$\neg$ state(bb(o, $s(o)$), $v_1(o, o)$, $v_2(s(s(s(o))), o)$, $v_3(s(o), s(s(s(o))))$, $v_4(s(s(s(s(o)))), s(s(s(o))))$, $h(s(s(s(s(o)))), s(s(o)))$, $s_1(s(s(o$

PUZ060+1.p Food problems
$\forall$peanuts, john, bill, sue, apples, chicken: (($\forall x$: (food(x) $\Rightarrow$ likes(john, x)) and food(apples) and food(chicken) and $\forall x$: ($\exists y$: (ea
food(x)) and eats(bill, peanuts) and alive(bill) and $\forall x$: (eats(bill, x) $\Rightarrow$ eats(sue, x)) and $\forall y$: (alive(y) $\Rightarrow$ $\forall x$: not_killed_by(y
likes(john, peanuts)) fof(prove_this, conjecture)

PUZ061+1.p Food problems
$\forall$peanuts, john, bill, sue, apples, chicken: (($\forall x, y$: ((alive(y) and eats(y, chicken)) $\Rightarrow$ likes(y, x)) and $\forall x$: (food(x) $\Rightarrow$
likes(john, x)) and food(apples) and food(chicken) and $\forall x$: ($\exists y$: (eats(y, x) and not_killed_by(y, x)) $\Rightarrow$ food(x)) and eats(bill
eats(sue, x)) and $\forall y$: (alive(y) $\Rightarrow$ $\forall x$: not_killed_by(y, x))) $\Rightarrow$ likes(john, peanuts)) fof(prove_this, conjecture)

PUZ062-2.p Problem about mutilated chessboard problem
c_in(v_A, c_Finite__Set_OFinites, tc_set(t_a)) $\Rightarrow$ (c_in(v_x, v_A, t_a) or c_Finite__Set_Ocard(c_insert(v_x, v_A, t_a), t_a) =
c_Suc(c_Finite__Set_Ocard(v_A, t_a))) cnf(cls_Finite__Set_Ocard__insert__disjoint$_0$, axiom)
c_in(v_G, c_Finite__Set_OFinites, tc_set(t_a)) $\Rightarrow$ c_in(c_inter(v_F, v_G, t_a), c_Finite__Set_OFinites, tc_set(t_a)) cnf(cls_Fini
c_in(v_t, c_Mutil_Otiling(c_Mutil_Odomino, tc_prod(tc_nat, tc_nat)), tc_set(tc_prod(tc_nat, tc_nat))) $\Rightarrow$ c_in(v_t, c_Finite__Set_
c_in(v_c, c_inter(v_A, v_B, t_a), t_a) $\Rightarrow$ c_in(v_c, v_B, t_a) cnf(cls_Set_OInt__iff$_1$, axiom)
(c_in(v_c, v_B, t_a) and c_in(v_c, v_A, t_a)) $\Rightarrow$ c_in(v_c, c_inter(v_A, v_B, t_a), t_a) cnf(cls_Set_OInt__iff$_2$, axiom)
c_in(v_x, c_insert(v_x, v_A, t_a), t_a) cnf(cls_Set_Oinsert__iff$_1$, axiom)
$\neg$ c_in(v_c, c_emptyset, t_a) cnf(cls_Set_Oempty__iff$_0$, axiom)
c_in(v_t, c_Mutil_Otiling(c_Mutil_Odomino, tc_prod(tc_nat, tc_nat)), tc_set(tc_prod(tc_nat, tc_nat))) cnf(cls_conjecture$_0$, ne
c_Finite__Set_Ocard(c_inter(c_Mutil_Ocoloured(c_0), v_t, tc_prod(tc_nat, tc_nat)), tc_prod(tc_nat, tc_nat)) = c_Finite__Set_Ocard
c_inter(v_a, v_t, tc_prod(tc_nat, tc_nat)) = c_emptyset cnf(cls_conjecture$_2$, negated_conjecture)
c_inter(c_Mutil_Ocoloured(c_0), v_a, tc_prod(tc_nat, tc_nat)) = c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_emptyset, tc_prod(tc_nat
c_inter(c_Mutil_Ocoloured(c_Suc(c_0)), v_a, tc_prod(tc_nat, tc_nat)) = c_insert(c_Pair(v_m, v_n, tc_nat, tc_nat), c_emptyset, tc_pr
c_Finite__Set_Ocard(c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_inter(c_Mutil_Ocoloured(c_0), v_t, tc_prod(tc_nat, tc_nat)), tc_prod
c_Finite__Set_Ocard(c_insert(c_Pair(v_m, v_n, tc_nat, tc_nat), c_inter(c_Mutil_Ocoloured(c_Suc(c_0)), v_t, tc_prod(tc_nat, tc_nat)

PUZ063-1.p Problem about mutilated chessboard problem
include('Axioms/MSC001-1.ax')
include('Axioms/MSC001-0.ax')
c_in(c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_insert(c_Pair(v_i, c_Suc(v_j), tc_nat, tc_nat), c_emptyset, tc_prod(tc_nat, tc_nat)), t

c_in(c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_insert(c_Pair(c_Suc(v_i), v_j, tc_nat, tc_nat), c_emptyset, tc_prod(tc_nat, tc_nat)), t
(c_in(v_a, v_A, tc_set(t_a)) and c_in(v_t, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) and c_inter(v_a, v_t, t_a) = c_emptyset) ⇒
c_in(c_union(v_a, v_t, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_Mutil_Otiling_OUn$_0$, axiom)
c_in(c_emptyset, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_Mutil_Otiling_Oempty$_0$, axiom)
c_union(c_union(v_A, v_B, t_a), v_C, t_a) = c_union(v_A, c_union(v_B, v_C, t_a), t_a) cnf(cls_Set_OUn__assoc$_0$, axiom)
c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_0$, negated_conjecture)
c_inter(c_emptyset, v_u, t_a) = c_emptyset cnf(cls_conjecture$_1$, negated_conjecture)
¬ c_in(c_union(c_emptyset, v_u, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_2$, negated_conjecture)

PUZ063-2.p Problem about mutilated chessboard problem
c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_0$, negated_conjecture)
¬ c_in(c_union(c_emptyset, v_u, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_2$, negated_conjecture)
c_union(c_emptyset, v_y, t_a) = v_y cnf(cls_Set_OUn__empty__left$_0$, axiom)

PUZ064-1.p Problem about mutilated chessboard problem
include('Axioms/MSC001-1.ax')
include('Axioms/MSC001-0.ax')
c_in(c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_insert(c_Pair(v_i, c_Suc(v_j), tc_nat, tc_nat), c_emptyset, tc_prod(tc_nat, tc_nat)), t
c_in(c_insert(c_Pair(v_i, v_j, tc_nat, tc_nat), c_insert(c_Pair(c_Suc(v_i), v_j, tc_nat, tc_nat), c_emptyset, tc_prod(tc_nat, tc_nat)), t
(c_in(v_a, v_A, tc_set(t_a)) and c_in(v_t, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) and c_inter(v_a, v_t, t_a) = c_emptyset) ⇒
c_in(c_union(v_a, v_t, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_Mutil_Otiling_OUn$_0$, axiom)
c_in(c_emptyset, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_Mutil_Otiling_Oempty$_0$, axiom)
c_union(c_union(v_A, v_B, t_a), v_C, t_a) = c_union(v_A, c_union(v_B, v_C, t_a), t_a) cnf(cls_Set_OUn__assoc$_0$, axiom)
c_in(v_a, v_A, tc_set(t_a)) cnf(cls_conjecture$_0$, negated_conjecture)
c_in(v_t, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_1$, negated_conjecture)
c_inter(v_a, v_t, t_a) = c_emptyset cnf(cls_conjecture$_2$, negated_conjecture)
c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_3$, negated_conjecture)
c_inter(c_union(v_a, v_t, t_a), v_u, t_a) = c_emptyset cnf(cls_conjecture$_4$, negated_conjecture)
¬ c_in(c_union(c_union(v_a, v_t, t_a), v_u, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_5$, negated_conjecture
(c_inter(v_t, v_u, t_a) = c_emptyset and c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a))) ⇒ c_in(c_union(v_t, v_u, t_a), c_Mutil

PUZ064-2.p Problem about mutilated chessboard problem
c_in(v_a, v_A, tc_set(t_a)) cnf(cls_conjecture$_0$, negated_conjecture)
c_inter(v_a, v_t, t_a) = c_emptyset cnf(cls_conjecture$_2$, negated_conjecture)
c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_3$, negated_conjecture)
c_inter(c_union(v_a, v_t, t_a), v_u, t_a) = c_emptyset cnf(cls_conjecture$_4$, negated_conjecture)
¬ c_in(c_union(c_union(v_a, v_t, t_a), v_u, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_conjecture$_5$, negated_conjecture
(c_inter(v_t, v_u, t_a) = c_emptyset and c_in(v_u, c_Mutil_Otiling(v_A, t_a), tc_set(t_a))) ⇒ c_in(c_union(v_t, v_u, t_a), c_Mutil
(c_in(v_a, v_A, tc_set(t_a)) and c_in(v_t, c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) and c_inter(v_a, v_t, t_a) = c_emptyset) ⇒
c_in(c_union(v_a, v_t, t_a), c_Mutil_Otiling(v_A, t_a), tc_set(t_a)) cnf(cls_Mutil_Otiling_OUn$_0$, axiom)
c_Relation_OImage(c_Relation_Odiag(v_A, t_a), v_B, t_a, t_a) = c_inter(v_A, v_B, t_a) cnf(cls_Relation_OImage__diag$_0$, axio
c_uminus(c_minus(v_A, v_B, tc_set(t_a)), tc_set(t_a)) = c_union(c_uminus(v_A, tc_set(t_a)), v_B, t_a) cnf(cls_Set_OCompl__I
c_uminus(c_UNIV, tc_set(t_a)) = c_emptyset cnf(cls_Set_OCompl__UNIV__eq$_0$, axiom)
c_minus(v_A, c_uminus(v_B, tc_set(t_a)), tc_set(t_a)) = c_inter(v_A, v_B, t_a) cnf(cls_Set_ODiff__Compl$_0$, axiom)
c_inter(c_UNIV, v_y, t_a) = v_y cnf(cls_Set_OInt__UNIV__left$_0$, axiom)
c_in(v_c, c_inter(v_A, v_B, t_a), t_a) ⇒ c_in(v_c, v_B, t_a) cnf(cls_Set_OInt__iff$_1$, axiom)
c_union(c_minus(v_B, v_A, tc_set(t_a)), v_A, t_a) = c_union(v_B, v_A, t_a) cnf(cls_Set_OUn__Diff__cancel2$_0$, axiom)
c_union(v_A, c_minus(v_B, v_A, tc_set(t_a)), t_a) = c_union(v_A, v_B, t_a) cnf(cls_Set_OUn__Diff__cancel$_0$, axiom)
c_union(c_UNIV, v_B, t_a) = c_UNIV cnf(cls_Set_OUn__UNIV__left$_0$, axiom)
c_union(v_y, v_y, t_a) = v_y cnf(cls_Set_OUn__absorb$_0$, axiom)
c_union(c_union(v_A, v_B, t_a), v_C, t_a) = c_union(v_A, c_union(v_B, v_C, t_a), t_a) cnf(cls_Set_OUn__assoc$_0$, axiom)
c_union(c_emptyset, v_y, t_a) = v_y cnf(cls_Set_OUn__empty__left$_0$, axiom)
c_union(v_y, c_emptyset, t_a) = v_y cnf(cls_Set_OUn__empty__right$_0$, axiom)
c_lessequals(c_union(v_A, v_B, t_a), v_C, tc_set(t_a)) ⇒ c_lessequals(v_A, v_C, tc_set(t_a)) cnf(cls_Set_OUn__subset__iff$_0$, ax
c_lessequals(c_union(v_A, v_B, t_a), v_C, tc_set(t_a)) ⇒ c_lessequals(v_B, v_C, tc_set(t_a)) cnf(cls_Set_OUn__subset__iff$_1$, ax
(c_lessequals(v_B, v_C, tc_set(t_a)) and c_lessequals(v_A, v_C, tc_set(t_a))) ⇒ c_lessequals(c_union(v_A, v_B, t_a), v_C, tc_set(t
c_uminus(c_uminus(v_y, tc_set(t_a)), tc_set(t_a)) = v_y cnf(cls_Set_Odouble__complement$_0$, axiom)
¬ c_in(v_c, c_emptyset, t_a) cnf(cls_Set_Oempty__iff$_0$, axiom)
c_lessequals(c_emptyset, v_A, tc_set(t_a)) cnf(cls_Set_Oempty__subsetI$_0$, axiom)
c_in(c_Main_OsubsetI_1(v_A, v_B, t_a), v_A, t_a) or c_lessequals(v_A, v_B, tc_set(t_a)) cnf(cls_Set_OsubsetI$_0$, axiom)
(c_lessequals(v_B, v_A, tc_set(t_a)) and c_lessequals(v_A, v_B, tc_set(t_a))) ⇒ v_A = v_B cnf(cls_Set_Osubset__antisym$_0$, a

c_lessequals(v_A, v_A, tc_set(t_a)) cnf(cls_Set_Osubset__refl$_0$, axiom)

PUZ065+1.p Sudoku 13273
include('Axioms/PUZ005+0.ax')
ssA$(n_1, n_2) = n_1$ fof(ax$_{353}$, axiom)
ssA$(n_1, n_9) = n_9$ fof(ax$_{354}$, axiom)
ssA$(n_2, n_4) = n_3$ fof(ax$_{355}$, axiom)
ssA$(n_2, n_7) = n_8$ fof(ax$_{356}$, axiom)
ssA$(n_3, n_7) = n_6$ fof(ax$_{357}$, axiom)
ssA$(n_4, n_5) = n_1$ fof(ax$_{358}$, axiom)
ssA$(n_4, n_6) = n_2$ fof(ax$_{359}$, axiom)
ssA$(n_4, n_7) = n_4$ fof(ax$_{360}$, axiom)
ssA$(n_5, n_1) = n_7$ fof(ax$_{361}$, axiom)
ssA$(n_5, n_3) = n_3$ fof(ax$_{362}$, axiom)
ssA$(n_6, n_1) = n_5$ fof(ax$_{363}$, axiom)
ssA$(n_7, n_1) = n_8$ fof(ax$_{364}$, axiom)
ssA$(n_7, n_4) = n_6$ fof(ax$_{365}$, axiom)
ssA$(n_8, n_5) = n_4$ fof(ax$_{366}$, axiom)
ssA$(n_8, n_8) = n_2$ fof(ax$_{367}$, axiom)
ssA$(n_9, n_4) = n_7$ fof(ax$_{368}$, axiom)
ssA$(n_9, n_8) = n_5$ fof(ax$_{369}$, axiom)

PUZ066+1.p Sudoku 19262
include('Axioms/PUZ005+0.ax')
ssA$(n_1, n_2) = n_6$ fof(ax$_{353}$, axiom)
ssA$(n_1, n_4) = n_2$ fof(ax$_{354}$, axiom)
ssA$(n_1, n_7) = n_5$ fof(ax$_{355}$, axiom)
ssA$(n_2, n_2) = n_1$ fof(ax$_{356}$, axiom)
ssA$(n_2, n_3) = n_3$ fof(ax$_{357}$, axiom)
ssA$(n_2, n_5) = n_6$ fof(ax$_{358}$, axiom)
ssA$(n_4, n_2) = n_7$ fof(ax$_{359}$, axiom)
ssA$(n_4, n_8) = n_3$ fof(ax$_{360}$, axiom)
ssA$(n_4, n_9) = n_1$ fof(ax$_{361}$, axiom)
ssA$(n_5, n_1) = n_8$ fof(ax$_{362}$, axiom)
ssA$(n_5, n_4) = n_5$ fof(ax$_{363}$, axiom)
ssA$(n_6, n_4) = n_4$ fof(ax$_{364}$, axiom)
ssA$(n_7, n_5) = n_1$ fof(ax$_{365}$, axiom)
ssA$(n_7, n_8) = n_6$ fof(ax$_{366}$, axiom)
ssA$(n_8, n_1) = n_9$ fof(ax$_{367}$, axiom)
ssA$(n_8, n_7) = n_2$ fof(ax$_{368}$, axiom)
ssA$(n_9, n_1) = n_4$ fof(ax$_{369}$, axiom)

PUZ067+1.p Sudoku 8618
include('Axioms/PUZ005+0.ax')
ssA$(n_1, n_4) = n_4$ fof(ax$_{353}$, axiom)
ssA$(n_1, n_5) = n_7$ fof(ax$_{354}$, axiom)
ssA$(n_1, n_7) = n_1$ fof(ax$_{355}$, axiom)
ssA$(n_2, n_2) = n_5$ fof(ax$_{356}$, axiom)
ssA$(n_2, n_4) = n_2$ fof(ax$_{357}$, axiom)
ssA$(n_3, n_2) = n_3$ fof(ax$_{358}$, axiom)
ssA$(n_3, n_9) = n_8$ fof(ax$_{359}$, axiom)
ssA$(n_4, n_1) = n_6$ fof(ax$_{360}$, axiom)
ssA$(n_4, n_5) = n_1$ fof(ax$_{361}$, axiom)
ssA$(n_4, n_7) = n_7$ fof(ax$_{362}$, axiom)
ssA$(n_5, n_3) = n_4$ fof(ax$_{363}$, axiom)
ssA$(n_5, n_6) = n_3$ fof(ax$_{364}$, axiom)
ssA$(n_6, n_8) = n_5$ fof(ax$_{365}$, axiom)
ssA$(n_7, n_4) = n_7$ fof(ax$_{366}$, axiom)
ssA$(n_7, n_8) = n_3$ fof(ax$_{367}$, axiom)
ssA$(n_8, n_1) = n_1$ fof(ax$_{368}$, axiom)
ssA$(n_9, n_9) = n_4$ fof(ax$_{369}$, axiom)

PUZ068+1.p Monday's Sudoku
47 1 5 2 3 8 6 1 7 4 3 5 7 3 1 4
include('Axioms/PUZ005+0.ax')
$\mathrm{ssA}(n_1, n_4) = n_4$ $\mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_5) = n_7$ $\mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_7) = n_1$ $\mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_2) = n_5$ $\mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_4) = n_2$ $\mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_2) = n_3$ $\mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_9) = n_8$ $\mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_1) = n_6$ $\mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_5) = n_1$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_7) = n_7$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_3) = n_4$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_6) = n_3$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$\mathrm{ssA}(n_6, n_8) = n_5$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_4) = n_7$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_8) = n_3$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_1) = n_1$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$\mathrm{ssA}(n_9, n_9) = n_4$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ068+2.p Monday's Sudoku
47 1 5 2 3 8 6 1 7 4 3 5 7 3 1 4
include('Axioms/PUZ006+0.ax')
$p(n_1, n_4, n_4)$ $\mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$p(n_1, n_5, n_7)$ $\mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$p(n_1, n_7, n_1)$ $\mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$p(n_2, n_2, n_5)$ $\mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$p(n_2, n_4, n_2)$ $\mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$p(n_3, n_2, n_3)$ $\mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$p(n_3, n_9, n_8)$ $\mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$p(n_4, n_1, n_6)$ $\mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$p(n_4, n_5, n_1)$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$p(n_4, n_7, n_7)$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$p(n_5, n_3, n_4)$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$p(n_5, n_6, n_3)$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$p(n_6, n_8, n_5)$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$p(n_7, n_4, n_7)$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$p(n_7, n_8, n_3)$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$p(n_8, n_1, n_1)$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$p(n_9, n_9, n_4)$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ068-1.p Monday's Sudoku
47 1 5 2 3 8 6 1 7 4 3 5 7 3 1 4
include('Axioms/PUZ005-0.ax')
$\mathrm{el}(s(n_0), s(s(s(s(n_0)))), s(s(s(s(n_0))))$ $\mathrm{cnf}(c_{01}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(n_0))))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{02}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(s(s(n_0)))))))), s(n_0))$ $\mathrm{cnf}(c_{03}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(n_0)), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{04}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(n_0)))), s(s(n_0)))$ $\mathrm{cnf}(c_{05}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(n_0)), s(s(s(n_0))))$ $\mathrm{cnf}(c_{06}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(s(s(s(s(s(s(s(n_0)))))))))), s(s(s(s(s(s(s(s(s(n_0))))))))))$ $\mathrm{cnf}(c_{07}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(n_0), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{08}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(n_0))))), s(n_0))$ $\mathrm{cnf}(c_{09}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{10}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(n_0))), s(s(s(s(n_0)))))$ $\mathrm{cnf}(c_{11}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(s(n_0)))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{12}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(n_0)))))), s(s(s(s(s(s(s(s(n_0))))))))), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{13}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{14}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(s(s(s(n_0))))))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{15}, \mathrm{axiom})$

$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(n_0), s(n_0))$ $\mathrm{cnf}(c_{16}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(s(n_0))))))))), s(s(s(s(s(s(s(s(s(n_0))))))))), s(s(s(n_0)))))$ $\mathrm{cnf}(c_{17}, \mathrm{axiom})$

PUZ069+1.p Tuesday's Sudoku
1 76 85 2 3 24 9 6 53 7 1 3 2
include('Axioms/PUZ005+0.ax')
$\mathrm{ssA}(n_1, n_1) = n_1$ $\mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_3) = n_7$ $\mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_4) = n_6$ $\mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_6) = n_8$ $\mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_7) = n_5$ $\mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_2) = n_2$ $\mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_7) = n_3$ $\mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_4) = n_2$ $\mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_5) = n_4$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_1) = n_9$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_8) = n_6$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$\mathrm{ssA}(n_6, n_5) = n_5$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$\mathrm{ssA}(n_6, n_6) = n_3$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_4) = n_7$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_8) = n_1$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_2) = n_3$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_7) = n_2$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ069+2.p Tuesday's Sudoku
1 76 85 2 3 24 9 6 53 7 1 3 2
include('Axioms/PUZ006+0.ax')
$p(n_1, n_1, n_1)$ $\mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$p(n_1, n_3, n_7)$ $\mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$p(n_1, n_4, n_6)$ $\mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$p(n_2, n_6, n_8)$ $\mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$p(n_2, n_7, n_5)$ $\mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$p(n_3, n_2, n_2)$ $\mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$p(n_3, n_7, n_3)$ $\mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$p(n_4, n_4, n_2)$ $\mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$p(n_4, n_5, n_4)$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$p(n_5, n_1, n_9)$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$p(n_5, n_8, n_6)$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$p(n_6, n_5, n_5)$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$p(n_6, n_6, n_3)$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$p(n_7, n_4, n_7)$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$p(n_7, n_8, n_1)$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$p(n_8, n_2, n_3)$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$p(n_8, n_7, n_2)$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ069-1.p Tuesday's Sudoku
1 76 85 2 3 24 9 6 53 7 1 3 2
include('Axioms/PUZ005-0.ax')
$\mathrm{el}(s(n_0), s(n_0), s(n_0))$ $\mathrm{cnf}(c_{01}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(n_0))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{02}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{03}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(n_0)))))), s(s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{04}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{05}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(n_0)), s(s(n_0)))$ $\mathrm{cnf}(c_{06}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(s(s(s(s(s(n_0))))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{07}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(n_0)))), s(s(n_0)))$ $\mathrm{cnf}(c_{08}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(n_0))))), s(s(s(s(n_0)))))$ $\mathrm{cnf}(c_{09}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(n_0), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{10}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{11}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(n_0))))), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{12}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(n_0))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{13}, \mathrm{axiom})$

$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(n_0)))), s(s(s(s(s(s(s(s(n_0)))))))) \quad \mathrm{cnf}(c_{14}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(s(s(s(n_0)))))))), s(n_0)) \quad \mathrm{cnf}(c_{15}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(s(n_0)), s(s(s(n_0)))) \quad \mathrm{cnf}(c_{16}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(s(n_0))))))), s(s(n_0))) \quad \mathrm{cnf}(c_{17}, \mathrm{axiom})$

PUZ070+1.p Wednesday's Sudoku
7 59 6 1 4 6 83 2 95
82 4 1 5
include('Axioms/PUZ005+0.ax')
$\mathrm{ssA}(n_1, n_2) = n_7 \quad \mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_5) = n_5 \quad \mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_6) = n_9 \quad \mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_7) = n_6 \quad \mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_9) = n_1 \quad \mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_1) = n_4 \quad \mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_5) = n_6 \quad \mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_4) = n_8 \quad \mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_5) = n_3 \quad \mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_8) = n_2 \quad \mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_1) = n_9 \quad \mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_2) = n_5 \quad \mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_3) = n_8 \quad \mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_4) = n_2 \quad \mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_8) = n_4 \quad \mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_4) = n_1 \quad \mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$\mathrm{ssA}(n_9, n_7) = n_5 \quad \mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ070-1.p Wednesday's Sudoku
7 59 6 1 4 6 83 2 95
82 4 1 5
include('Axioms/PUZ005-0.ax')
$\mathrm{el}(s(n_0), s(s(n_0)), s(s(s(s(s(s(s(n_0))))))))) \quad \mathrm{cnf}(c_{01}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(n_0)))))), s(s(s(s(s(n_0)))))) \quad \mathrm{cnf}(c_{02}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(s(n_0)))))), s(s(s(s(s(s(s(s(s(s(n_0)))))))))) \quad \mathrm{cnf}(c_{03}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(s(s(n_0))))))) \quad \mathrm{cnf}(c_{04}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(s(s(s(n_0)))))))))), s(n_0)) \quad \mathrm{cnf}(c_{05}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(n_0), s(s(s(s(n_0))))) \quad \mathrm{cnf}(c_{06}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(s(s(s(n_0))))), s(s(s(s(s(s(s(n_0))))))) \quad \mathrm{cnf}(c_{07}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(n_0)))), s(s(s(s(s(s(s(s(s(n_0)))))))))) \quad \mathrm{cnf}(c_{08}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(n_0))))), s(s(s(n_0)))) \quad \mathrm{cnf}(c_{09}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(s(s(s(n_0)))))))), s(s(n_0))) \quad \mathrm{cnf}(c_{10}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(n_0), s(s(s(s(s(s(s(s(s(n_0)))))))))) \quad \mathrm{cnf}(c_{11}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(n_0)), s(s(s(s(s(n_0)))))) \quad \mathrm{cnf}(c_{12}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(n_0))), s(s(s(s(s(s(s(s(s(n_0)))))))))) \quad \mathrm{cnf}(c_{13}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(n_0)))), s(s(n_0))) \quad \mathrm{cnf}(c_{14}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(n_0))))) \quad \mathrm{cnf}(c_{15}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(n_0)))), s(n_0)) \quad \mathrm{cnf}(c_{16}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(s(n_0)))))))))), s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(n_0)))))) \quad \mathrm{cnf}(c_{17}, \mathrm{axiom})$

PUZ071+1.p Thursday's Sudoku
45 8 71 2 16 3 5 4 2
2 8 1 3
include('Axioms/PUZ005+0.ax')
$\mathrm{ssA}(n_1, n_1) = n_4 \quad \mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_2) = n_5 \quad \mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_5) = n_8 \quad \mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_8) = n_7 \quad \mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_9) = n_1 \quad \mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_2) = n_2 \quad \mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_3) = n_1 \quad \mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_4) = n_6 \quad \mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$

$\mathrm{ssA}(n_4, n_8) = n_3$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_1) = n_5$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_5) = n_4$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_7) = n_2$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_1) = n_2$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_7) = n_8$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_4) = n_1$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_6) = n_3$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$\mathrm{ssA}(n_9, n_4) = n_7$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ071-1.p Thursday's Sudoku
45 8 71 2 16 3 5 4 2
2 8 1 3
include('Axioms/PUZ005-0.ax')
$\mathrm{el}(s(n_0), s(n_0), s(s(s(s(n_0)))))$ $\mathrm{cnf}(c_{01}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(n_0)), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{02}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(n_0))))), s(s(s(s(s(s(s(n_0)))))))))$ $\mathrm{cnf}(c_{03}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{04}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(s(s(n_0)))))))))), s(n_0))$ $\mathrm{cnf}(c_{05}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(n_0)), s(s(n_0)))$ $\mathrm{cnf}(c_{06}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(n_0))), s(n_0))$ $\mathrm{cnf}(c_{07}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(n_0)))), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{08}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0)))))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{09}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(n_0), s(s(s(s(s(n_0))))))$ $\mathrm{cnf}(c_{10}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(n_0))))), s(s(s(s(n_0)))))$ $\mathrm{cnf}(c_{11}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(s(s(s(s(n_0))))))), s(s(n_0)))$ $\mathrm{cnf}(c_{12}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(n_0), s(s(n_0)))$ $\mathrm{cnf}(c_{13}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(s(s(n_0)))))))))$ $\mathrm{cnf}(c_{14}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0)))))))), s(s(s(n_0))), s(n_0))$ $\mathrm{cnf}(c_{15}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(s(s(s(s(n_0))))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{16}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(n_0)))), s(s(s(s(s(s(n_0)))))))$ $\mathrm{cnf}(c_{17}, \mathrm{axiom})$

PUZ072+1.p Friday's Sudoku
1 9 3 8 6 124 7 3 5 8 6 4 2 7 5
include('Axioms/PUZ005+0.ax')
$\mathrm{ssA}(n_1, n_2) = n_1$ $\mathrm{fof}(\mathrm{ax}_{353}, \mathrm{axiom})$
$\mathrm{ssA}(n_1, n_9) = n_9$ $\mathrm{fof}(\mathrm{ax}_{354}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_4) = n_3$ $\mathrm{fof}(\mathrm{ax}_{355}, \mathrm{axiom})$
$\mathrm{ssA}(n_2, n_7) = n_8$ $\mathrm{fof}(\mathrm{ax}_{356}, \mathrm{axiom})$
$\mathrm{ssA}(n_3, n_7) = n_6$ $\mathrm{fof}(\mathrm{ax}_{357}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_5) = n_1$ $\mathrm{fof}(\mathrm{ax}_{358}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_6) = n_2$ $\mathrm{fof}(\mathrm{ax}_{359}, \mathrm{axiom})$
$\mathrm{ssA}(n_4, n_7) = n_4$ $\mathrm{fof}(\mathrm{ax}_{360}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_1) = n_7$ $\mathrm{fof}(\mathrm{ax}_{361}, \mathrm{axiom})$
$\mathrm{ssA}(n_5, n_3) = n_3$ $\mathrm{fof}(\mathrm{ax}_{362}, \mathrm{axiom})$
$\mathrm{ssA}(n_6, n_1) = n_5$ $\mathrm{fof}(\mathrm{ax}_{363}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_1) = n_8$ $\mathrm{fof}(\mathrm{ax}_{364}, \mathrm{axiom})$
$\mathrm{ssA}(n_7, n_4) = n_6$ $\mathrm{fof}(\mathrm{ax}_{365}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_5) = n_4$ $\mathrm{fof}(\mathrm{ax}_{366}, \mathrm{axiom})$
$\mathrm{ssA}(n_8, n_8) = n_2$ $\mathrm{fof}(\mathrm{ax}_{367}, \mathrm{axiom})$
$\mathrm{ssA}(n_9, n_4) = n_7$ $\mathrm{fof}(\mathrm{ax}_{368}, \mathrm{axiom})$
$\mathrm{ssA}(n_9, n_8) = n_5$ $\mathrm{fof}(\mathrm{ax}_{369}, \mathrm{axiom})$

PUZ072-1.p Friday's Sudoku
1 9 3 8 6 124 7 3 5 8 6 4 2 7 5
include('Axioms/PUZ005-0.ax')
$\mathrm{el}(s(n_0), s(s(n_0)), s(n_0))$ $\mathrm{cnf}(c_{01}, \mathrm{axiom})$
$\mathrm{el}(s(n_0), s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(s(s(n_0))))))))))$ $\mathrm{cnf}(c_{02}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(n_0)))), s(s(s(n_0))))$ $\mathrm{cnf}(c_{03}, \mathrm{axiom})$
$\mathrm{el}(s(s(n_0)), s(s(s(s(s(s(n_0)))))), s(s(s(s(s(s(s(n_0))))))))$ $\mathrm{cnf}(c_{04}, \mathrm{axiom})$
$\mathrm{el}(s(s(s(n_0))), s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(s(n_0)))))))$ $\mathrm{cnf}(c_{05}, \mathrm{axiom})$

$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(n_0))))), s(n_0))$ cnf(c_{06}, axiom)
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(s(n_0)))))), s(s(n_0)))$ cnf(c_{07}, axiom)
$\mathrm{el}(s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))), s(s(s(s(n_0)))))$ cnf(c_{08}, axiom)
$\mathrm{el}(s(s(s(s(s(n_0))))), s(n_0), s(s(s(s(s(s(n_0)))))))$ cnf(c_{09}, axiom)
$\mathrm{el}(s(s(s(s(s(n_0))))), s(s(s(n_0))), s(s(s(n_0))))$ cnf(c_{10}, axiom)
$\mathrm{el}(s(s(s(s(s(s(n_0)))))), s(n_0), s(s(s(s(s(n_0))))))$ cnf(c_{11}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(n_0), s(s(s(s(s(s(s(n_0)))))))))$ cnf(c_{12}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(n_0))))))), s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))))$ cnf(c_{13}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(n_0))))), s(s(s(s(n_0)))))$ cnf(c_{14}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(s(n_0))))))))), s(s(s(s(s(s(s(n_0)))))))), s(s(n_0)))$ cnf(c_{15}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(s(s(n_0)))))))))), s(s(s(s(n_0)))), s(s(s(s(s(s(s(n_0))))))))$ cnf(c_{16}, axiom)
$\mathrm{el}(s(s(s(s(s(s(s(s(s(n_0))))))))))), s(s(s(s(s(s(s(n_0)))))))), s(s(s(s(s(n_0))))))$ cnf(c_{17}, axiom)

PUZ078+1.p Leo the Liar
Leo the Liar is a strange liar. On six days of the week, he lies. On the seventh day he always tells the truth. He made the following statements on three consecutive days: Day 1. I lie on Monday and Tuesday. Day 2. Today it is Thursday, Saturday, or Sunday. Day 3. I lie on Wednesday and Friday. Prove that Leo tells the truth on Tuesday.
$\forall x$: (x = monday or x = tuesday or x = wednesday or x = thursday or x = friday or x = saturday or x = sunday) fof(days_of_week, axiom)
monday ≠ tuesday and monday ≠ wednesday and monday ≠ thursday and monday ≠ friday and monday ≠ saturday and m
sunday and tuesday ≠ wednesday and tuesday ≠ thursday and tuesday ≠ friday and tuesday ≠ saturday and tuesday ≠
sunday and wednesday ≠ thursday and wednesday ≠ friday and wednesday ≠ saturday and wednesday ≠ sunday and thursd
friday and thursday ≠ saturday and thursday ≠ sunday and friday ≠ saturday and friday ≠ sunday and saturday ≠
sunday fof(uniquenames, axiom)
$\forall x, y$: ((truthday(x) and truthday(y)) $\Rightarrow$ $x = y$) fof(truth_unique, axiom)
$\forall x, y$: (($\mathrm{day}_1(x)$ and $\mathrm{day}_1(y)$) $\Rightarrow$ $x = y$) fof(day1_unique, axiom)
$\forall x, y$: (($\mathrm{day}_2(x)$ and $\mathrm{day}_2(y)$) $\Rightarrow$ $x = y$) fof(day2_unique, axiom)
$\forall x, y$: (($\mathrm{day}_3(x)$ and $\mathrm{day}_3(y)$) $\Rightarrow$ $x = y$) fof(day3_unique, axiom)
$\exists x$: truthday(x) fof(truth_once, axiom)
$\exists x$: $\mathrm{day}_1(x)$ fof(day1_once, axiom)
$\exists x$: $\mathrm{day}_2(x)$ fof(day2_once, axiom)
$\exists x$: $\mathrm{day}_3(x)$ fof(day3_once, axiom)
consecutive(sunday) = monday fof(monday_follows_sunday, axiom)
consecutive(monday) = tuesday fof(tuesday_follows_monday, axiom)
consecutive(tuesday) = wednesday fof(wednesday_follows_tuesday, axiom)
consecutive(wednesday) = thursday fof(thursday_follows_wednesday, axiom)
consecutive(thursday) = friday fof(friday_follows_thursday, axiom)
consecutive(friday) = saturday fof(saturday_follows_friday, axiom)
consecutive(saturday) = sunday fof(sunday_follows_saturday, axiom)
$\forall x$: ($\mathrm{day}_1(x)$ $\iff$ day_2(consecutive(x))) fof(day2_follows_day_1, axiom)
$\forall x$: ($\mathrm{day}_2(x)$ $\iff$ day_3(consecutive(x))) fof(day3_follows_day_2, axiom)
$\forall x$: ($\mathrm{day}_1(x)$ $\Rightarrow$ (truthday(x) $\iff$ ($\neg$ truthday(monday) and $\neg$ truthday(tuesday)))) fof($\mathrm{statement}_1$, axiom)
$\forall x$: ($\mathrm{day}_2(x)$ $\Rightarrow$ (truthday(x) $\iff$ (day_2(thursday) or day_2(saturday) or day_2(sunday)))) fof($\mathrm{statement}_2$, axiom)
$\forall x$: ($\mathrm{day}_3(x)$ $\Rightarrow$ (truthday(x) $\iff$ ($\neg$ truthday(wednesday) and $\neg$ truthday(friday)))) fof($\mathrm{statement}_3$, axiom)
truthday(tuesday) fof(tuesday_leo_tells_truths, conjecture)

PUZ079+2.p Another Sudoku
1 6 4 9 5 7 4 9 5 47 53 3 1 91 68 7 5 2 9 3 2 1 6 4
include('Axioms/PUZ006+0.ax')
$p(n_1, n_1, n_1)$ fof(ax_{353}, axiom)
$p(n_1, n_4, n_6)$ fof(ax_{354}, axiom)
$p(n_1, n_6, n_4)$ fof(ax_{355}, axiom)
$p(n_1, n_9, n_9)$ fof(ax_{356}, axiom)
$p(n_2, n_4, n_5)$ fof(ax_{357}, axiom)
$p(n_2, n_6, n_7)$ fof(ax_{358}, axiom)
$p(n_3, n_3, n_4)$ fof(ax_{359}, axiom)
$p(n_3, n_5, n_9)$ fof(ax_{360}, axiom)
$p(n_3, n_7, n_5)$ fof(ax_{369}, axiom)
$p(n_4, n_1, n_4)$ fof(ax_{361}, axiom)
$p(n_4, n_2, n_7)$ fof(ax_{362}, axiom)

$p(n_4, n_8, n_5)$ fof(ax_{363}, axiom)
$p(n_4, n_9, n_3)$ fof(ax_{364}, axiom)
$p(n_5, n_3, n_3)$ fof(ax_{365}, axiom)
$p(n_5, n_7, n_1)$ fof(ax_{366}, axiom)
$p(n_6, n_1, n_9)$ fof(ax_{367}, axiom)
$p(n_6, n_2, n_1)$ fof(ax_{368}, axiom)
$p(n_6, n_8, n_6)$ fof(ax369_1, axiom)
$p(n_6, n_9, n_8)$ fof(ax369_2, axiom)
$p(n_7, n_3, n_7)$ fof(ax369_3, axiom)
$p(n_7, n_5, n_5)$ fof(ax369_4, axiom)
$p(n_7, n_7, n_2)$ fof(ax369_5, axiom)
$p(n_8, n_4, n_9)$ fof(ax369_6, axiom)
$p(n_8, n_6, n_3)$ fof(ax369_7, axiom)
$p(n_9, n_1, n_2)$ fof(ax369_8, axiom)
$p(n_9, n_4, n_1)$ fof(ax369_9, axiom)
$p(n_9, n_6, n_6)$ fof(ax369_{10}, axiom)
$p(n_9, n_9, n_4)$ fof(ax369_{11}, axiom)

PUZ080+2.p Another Sudoku
3 1 7 9 6 9 4 4 8 6 2 4 7 1 67 21 3 1 9 7 8 3
include('Axioms/PUZ006+0.ax')
$p(n_1, n_4, n_3)$ fof(ax_{143}, axiom)
$p(n_1, n_6, n_1)$ fof(ax_{161}, axiom)
$p(n_2, n_2, n_7)$ fof(ax_{227}, axiom)
$p(n_2, n_5, n_9)$ fof(ax_{259}, axiom)
$p(n_2, n_8, n_6)$ fof(ax_{286}, axiom)
$p(n_3, n_1, n_9)$ fof(ax_{319}, axiom)
$p(n_3, n_9, n_4)$ fof(ax_{394}, axiom)
$p(n_4, n_1, n_4)$ fof(ax_{414}, axiom)
$p(n_4, n_4, n_8)$ fof(ax_{448}, axiom)
$p(n_4, n_6, n_6)$ fof(ax_{466}, axiom)
$p(n_4, n_9, n_2)$ fof(ax_{492}, axiom)
$p(n_5, n_4, n_4)$ fof(ax_{544}, axiom)
$p(n_5, n_6, n_7)$ fof(ax_{567}, axiom)
$p(n_6, n_5, n_1)$ fof(ax_{651}, axiom)
$p(n_7, n_3, n_6)$ fof(ax_{736}, axiom)
$p(n_7, n_4, n_7)$ fof(ax_{747}, axiom)
$p(n_7, n_6, n_2)$ fof(ax_{762}, axiom)
$p(n_7, n_7, n_1)$ fof(ax_{771}, axiom)
$p(n_8, n_1, n_3)$ fof(ax_{813}, axiom)
$p(n_8, n_3, n_1)$ fof(ax_{831}, axiom)
$p(n_8, n_7, n_9)$ fof(ax_{879}, axiom)
$p(n_8, n_9, n_7)$ fof(ax_{897}, axiom)
$p(n_9, n_3, n_8)$ fof(ax_{938}, axiom)
$p(n_9, n_7, n_3)$ fof(ax_{973}, axiom)

PUZ081∧1.p 1 of http://philosophy.hku.hk/think/logic/knight.php
A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet two inhabitants: Zoey and Mel. Zoey tells you that Mel is a knave. Mel says, 'Neither Zoey nor I are knaves'. Who is a knight and who is a knave?
islander: $i thf(islander, type)
knight: $i thf(knight, type)
knave: $i thf(knave, type)
says: $i → $o → $o thf(says, type)
zoey: $i thf(zoey, type)
mel: $i thf(mel, type)
is_a: $i → $i → $o thf(is_a, type)
$\forall x$: $i: ((is_a@$x$@islander) $\Rightarrow$ (is_a@x@knight or is_a@x@knave)) thf(kk_6_1, axiom)
$\forall x$: $i: ((is_a@$x$@knight) $\Rightarrow$ $\forall a$: $o: ((says@$x$@$a$) $\Rightarrow$ a)) thf(kk_6_2, axiom)
$\forall x$: $i: ((is_a@$x$@knave) $\Rightarrow$ $\forall a$: $o: ((says@$x$@$a$) $\Rightarrow$ $\neg\, a$)) thf(kk_6_3, axiom)

is_a@zoey@islander and is_a@mel@islander thf(kk_6$_4$, axiom)
says@zoey@(is_a@mel@knave) thf(kk_6$_5$, axiom)
says@mel@¬ is_a@zoey@knave or is_a@mel@knave thf(kk_6$_6$, axiom)
∃y: $i, z: $i: (($y$ = knight < > y = knave) and (z = knight < > z = knave) and is_a@mel@y and is_a@zoey@z) thf(query,

PUZ081∧2.p 1 of http://philosophy.hku.hk/think/logic/knight.php
A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet two inhabitants: Zoey and Mel. Zoey tells you that Mel is a knave. Mel says, 'Neither Zoey nor I are knaves'. Who is a knight and who is a knave?
mel: $i thf(mel_type, type)
zoey: $i thf(zoey_type, type)
knight: $i → $o thf(knight_type, type)
knave: $i → $o thf(knave_type, type)
says: $i → $o → $o thf(says_type, type)
∀p: $i: (knight@$p$) < > (knave@$p$) thf(knights_xor_knaves, axiom)
∀p: $i, s: $o: ((knight@$p$ and says@p@s) ⇒ s) thf(knights_tell_truth, axiom)
∀p: $i, s: $o: ((knave@$p$ and says@p@s) ⇒ ¬ s) thf(knaves_lie, axiom)
says@zoey@(knave@mel) thf(zoey_speaks, axiom)
says@mel@(¬ knave@zoey and ¬ knave@mel) thf(mel_speaks, axiom)
∃tZ: $i → $o, tM: $i → $o: (tZ@zoey and tM@mel) thf(what_are_zoey_and_mel, conjecture)

PUZ081∧3.p 1 of http://philosophy.hku.hk/think/logic/knight.php
A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet two inhabitants: Zoey and Mel. Zoey tells you that Mel is a knave. Mel says, 'Neither Zoey nor I are knaves'. Who is a knight and who is a knave?
mel: $i thf(mel_type, type)
zoey: $i thf(zoey_type, type)
knight: $i → $o thf(knight_type, type)
knave: $i → $o thf(knave_type, type)
says: $i → $o → $o thf(says_type, type)
∀p: $i: (knight@$p$) < > (knave@$p$) thf(knights_xor_knaves, axiom)
∀p: $i, s: $o: ((knight@$p$ and says@p@s) ⇒ s) thf(knights_tell_truth, axiom)
∀p: $i, s: $o: ((knave@$p$ and says@p@s) ⇒ ¬ s) thf(knaves_lie, axiom)
says@zoey@(knave@mel) thf(zoey_speaks, axiom)
says@mel@(¬ knave@zoey and ¬ knave@mel) thf(mel_speaks, axiom)
∃knight: $i, knave: $i: (knight@knight and knave@knave) thf(who_is_knight_and_knave, conjecture)

PUZ082∧1.p Peter the liar
Peter says that everything he says is false. Show that not everything Peter says is false.
peter: $i thf(peter, type)
says: $i → $o → $o thf(says, type)
says@peter@∀x: $o: ((says@peter@$x$) ⇒ ¬ x) thf(ax$_1$, axiom)
¬ ∀x: $o: ((says@peter@$x$) ⇒ ¬ x) thf(thm, conjecture)

PUZ083∧1.p Peter the untruthful
Peter says that everything he says is false. Show that not everything Peter says is true.
peter: $i thf(peter, type)
says: $i → $o → $o thf(says, type)
says@peter@∀x: $o: ((says@peter@$x$) ⇒ ¬ x) thf(ax$_1$, axiom)
¬ ∀x: $o: ((says@peter@$x$) ⇒ x) thf(thm, conjecture)

PUZ084∧1.p The friends puzzle - reflexivity for Peter's wife
(i) Peter is a friend of John, so if Peter knows that John knows something then John knows that Peter knows the same thing. (ii) Peter is married, so if Peter's wife knows something, then Peter knows the same thing. John and Peter have an appointment, let us consider the following situation: (a) Peter knows the time of their appointment. (b) Peter also knows that John knows the place of their appointment. Moreover, (c) Peter's wife knows that if Peter knows the time of their appointment, then John knows that too (since John and Peter are friends). Finally, (d) Peter knows that if John knows the place and the time of their appointment, then John knows that he has an appointment. From this situation we want to prove (e) that each of the two friends knows that the other one knows that he has an appointment.
include('Axioms/LCL013^0.ax')
peter: $i → $i → $o thf(peter, type)

john: $i → $i → $o thf(john, type)
wife: ($i → $i → $o) → $i → $i → $o thf(wife, type)
mreflexive@peter thf(refl_peter, axiom)
mreflexive@john thf(refl_john, axiom)
mreflexive@(wife@peter) thf(refl_wife_peter, axiom)
mtransitive@peter thf(trans_peter, axiom)
mtransitive@john thf(trans_john, axiom)
mtransitive@(wife@peter) thf(trans_wife_peter, axiom)
mvalid@(mforall_prop@λ$a: $i → $o: (mimplies@(mbox@(wife@peter)@$a)@$a)) thf(conj, conjecture)

PUZ085^1.p The friends puzzle - transitivity for Peter's wife
(i) Peter is a friend of John, so if Peter knows that John knows something then John knows that Peter knows the same thing. (ii) Peter is married, so if Peter's wife knows something, then Peter knows the same thing. John and Peter have an appointment, let us consider the following situation: (a) Peter knows the time of their appointment. (b) Peter also knows that John knows the place of their appointment. Moreover, (c) Peter's wife knows that if Peter knows the time of their appointment, then John knows that too (since John and Peter are friends). Finally, (d) Peter knows that if John knows the place and the time of their appointment, then John knows that he has an appointment. From this situation we want to prove (e) that each of the two friends knows that the other one knows that he has an appointment.
include('Axioms/LCL013^0.ax')
peter: $i → $i → $o thf(peter, type)
john: $i → $i → $o thf(john, type)
wife: ($i → $i → $o) → $i → $i → $o thf(wife, type)
mreflexive@peter thf(refl_peter, axiom)
mreflexive@john thf(refl_john, axiom)
mreflexive@(wife@peter) thf(refl_wife_peter, axiom)
mtransitive@peter thf(trans_peter, axiom)
mtransitive@john thf(trans_john, axiom)
mtransitive@(wife@peter) thf(trans_wife_peter, axiom)
mvalid@(mforall_prop@λ$a: $i → $o: (mimplies@(mbox@(wife@peter)@$a)@(mbox@(wife@peter)@(mbox@(wife@peter)@$a))))

PUZ086^1.p The friends puzzle - they both know
(i) Peter is a friend of John, so if Peter knows that John knows something then John knows that Peter knows the same thing. (ii) Peter is married, so if Peter's wife knows something, then Peter knows the same thing. John and Peter have an appointment, let us consider the following situation: (a) Peter knows the time of their appointment. (b) Peter also knows that John knows the place of their appointment. Moreover, (c) Peter's wife knows that if Peter knows the time of their appointment, then John knows that too (since John and Peter are friends). Finally, (d) Peter knows that if John knows the place and the time of their appointment, then John knows that he has an appointment. From this situation we want to prove (e) that each of the two friends knows that the other one knows that he has an appointment.
include('Axioms/LCL013^0.ax')
peter: $i → $i → $o thf(peter, type)
john: $i → $i → $o thf(john, type)
wife: ($i → $i → $o) → $i → $i → $o thf(wife, type)
mreflexive@peter thf(refl_peter, axiom)
mreflexive@john thf(refl_john, axiom)
mreflexive@(wife@peter) thf(refl_wife_peter, axiom)
mtransitive@peter thf(trans_peter, axiom)
mtransitive@john thf(trans_john, axiom)
mtransitive@(wife@peter) thf(trans_wife_peter, axiom)
mvalid@(mforall_prop@λ$a: $i → $o: (mimplies@(mbox@peter@(mbox@john@$a))@(mbox@john@(mbox@peter@$a)))) thf(
mvalid@(mforall_prop@λ$a: $i → $o: (mimplies@(mbox@(wife@peter)@$a)@(mbox@peter@$a))) thf(ax_ii, axiom)
time: $i → $o thf(time, type)
place: $i → $o thf(place, type)
appointment: $i → $o thf(appointment, type)
mvalid@(mbox@peter@time) thf(ax_a, axiom)
mvalid@(mbox@peter@(mbox@john@place)) thf(ax_b, axiom)
mvalid@(mbox@(wife@peter)@(mimplies@(mbox@peter@time)@(mbox@john@time))) thf(ax_c, axiom)
mvalid@(mbox@peter@(mbox@john@(mimplies@(mand@place@time)@appointment))) thf(ax_d, axiom)
mvalid@(mand@(mbox@peter@(mbox@john@appointment))@(mbox@john@(mbox@peter@appointment))) thf(conj, conj

PUZ088∧5.p TPS problem THM68
If everyone likes Bruce and Lyle likes everyone who likes someone then someone likes everyone.
cLIKES: $i → $i → $o thf(cLIKES, type)
cLYLE: $i thf(cLYLE, type)
cBRUCE: $i thf(cBRUCE, type)
(∀x: $i: (cLIKES@$x$@cBRUCE) and ∀$y$: $i: (∃$z$: $i: (cLIKES@$y$@$z$) ⇒ (cLIKES@cLYLE@$y$))) ⇒ ∃$u$: $i: ∀$v$: $i: (cLIKES@$u$

PUZ090∧5.p TPS problem THM210
Lewis Carroll's problem of the winds and windows; from the Ninth Paper on Logic, November 1892.
cOPEN_WINDOW: $o thf(cOPEN_WINDOW, type)
cEAST: $o thf(cEAST, type)
cRHEUMATIC: $o thf(cRHEUMATIC, type)
cFOGGY: $o thf(cFOGGY, type)
cOPEN_DOOR: $o thf(cOPEN_DOOR, type)
cFLUTE: $o thf(cFLUTE, type)
cSUNSHINE: $o thf(cSUNSHINE, type)
cFIRE: $o thf(cFIRE, type)
cGUSTY: $o thf(cGUSTY, type)
cHEADACHE: $o thf(cHEADACHE, type)
cSMOKES: $o thf(cSMOKES, type)
cCOLD: $o thf(cCOLD, type)
cSMOKE: $o thf(cSMOKE, type)
((cEAST ⇒ cSUNSHINE) and ((cCOLD and cFOGGY) ⇒ cFLUTE) and ((cFIRE and cSMOKE) ⇒ cOPEN_DOOR) a
cFIRE) and ((cEAST and cGUSTY) ⇒ cSMOKES) and (cOPEN_DOOR ⇒ ¬ cHEADACHE) and (cFOGGY ⇒
¬ cOPEN_WINDOW) and ((¬ cGUSTY and cFIRE and ¬ cOPEN_DOOR) ⇒ ¬ cRHEUMATIC) and (cSUNSHINE ⇒
cFOGGY) and (cFLUTE ⇒ ¬ cOPEN_DOOR) and ((cFOGGY and cEAST) ⇒ cRHEUMATIC)) ⇒ (cEAST ⇒
¬ cOPEN_WINDOW) thf($cTHM_{210}$, conjecture)

PUZ091∧5.p TPS problem from BASIC-FO-THMS
cIN_BED_BEFORE_FOUR: $o thf(cIN_BED_BEFORE_FOUR, type)
cRISES_AT_FIVE: $o thf(cRISES_AT_FIVE, type)
cLOGICIAN: $o thf(cLOGICIAN, type)
cEARNEST: $o thf(cEARNEST, type)
cGAMBLER: $o thf(cGAMBLER, type)
cLIKELY_LOSE_MONEY: $o thf(cLIKELY_LOSE_MONEY, type)
cLIVELY: $o thf(cLIVELY, type)
cRAVENOUS: $o thf(cRAVENOUS, type)
cBETTER_TAKE_TO_CAB_DRIVING: $o thf(cBETTER_TAKE_TO_CAB_DRIVING, type)
cEATS_PORKCHOPS: $o thf(cEATS_PORKCHOPS, type)
cHAS_LOST_MONEY: $o thf(cHAS_LOST_MONEY, type)
(((cLOGICIAN and cEATS_PORKCHOPS) ⇒ cLIKELY_LOSE_MONEY) and ((cGAMBLER and ¬ cRAVENOUS) ⇒
cLIKELY_LOSE_MONEY) and ((¬ cLIVELY and cHAS_LOST_MONEY and cLIKELY_LOSE_MONEY) ⇒ cRISES_AT_FI
cRAVENOUS) and ((cLIVELY and cIN_BED_BEFORE_FOUR) ⇒ cBETTER_TAKE_TO_CAB_DRIVING) and ((cRAVE
cEATS_PORKCHOPS) and ((cLOGICIAN and cLIKELY_LOSE_MONEY) ⇒ cBETTER_TAKE_TO_CAB_DRIVING) and
¬ cLIKELY_LOSE_MONEY) and ((¬ cGAMBLER and ¬ cRAVENOUS) ⇒ cLIVELY) and ((cLIVELY and cLOGICIAN a
¬ cLIKELY_LOSE_MONEY) and ((cRAVENOUS and cEARNEST) ⇒ ¬ cBETTER_TAKE_TO_CAB_DRIVING) and ((cG
¬ cIN_BED_BEFORE_FOUR) and ((cHAS_LOST_MONEY and ¬ cEATS_PORKCHOPS and ¬ cRISES_AT_FIVE) ⇒
cBETTER_TAKE_TO_CAB_DRIVING) and ((cGAMBLER and cIN_BED_BEFORE_FOUR and ¬ cRAVENOUS) ⇒
¬ cBETTER_TAKE_TO_CAB_DRIVING) and ((cRAVENOUS and ¬ cLIVELY and ¬ cLIKELY_LOSE_MONEY) ⇒
cGAMBLER)) ⇒ ((cEARNEST and cLOGICIAN) ⇒ (cRISES_AT_FIVE and ¬ cIN_BED_BEFORE_FOUR)) thf(cPO

PUZ092∧5.p TPS problem from BASIC-HO-EQ-THMS
∀aCRES: $i, mRSACRES: $i, bARRY: $i, mRSBARRY: $i, cOLE: $i, mRSCOLE: $i, dIX: $i, mRSDIX: $i, eDEN: $i, mRSEDE
mRSACRES and bARRY = mRSBARRY and eDEN = mRSHALL) ⇒ cOLE = mRSDIX) and ((aCRES =
mRSACRES and hALL = mRSHALL and bARRY = mRSCOLE) ⇒ dIX ≠ mRSEDEN) and ((cOLE = mRSCOLE and dI
mRSDIX and dIX = cOLE and aCRES ≠ mRSBARRY) ⇒ eDEN ≠ mRSHALL) and ((aCRES = mRSACRES and dIX =
mRSDIX and bARRY ≠ mRSCOLE) ⇒ eDEN = mRSHALL) and ((eDEN = mRSEDEN and hALL = mRSHALL and cO
mRSDIX) ⇒ aCRES ≠ mRSBARRY) and ((bARRY = mRSBARRY and cOLE = mRSCOLE and cOLE =
bARRY and eDEN ≠ mRSHALL) ⇒ dIX = mRSEDEN)) ⇒ (aCRES ≠ mRSACRES or bARRY ≠ mRSBARRY or cOLE
mRSCOLE or dIX ≠ mRSDIX or eDEN ≠ mRSEDEN or hALL ≠ mRSHALL)) thf(cSIXFRIENDS_EASIER, conjecture

PUZ093^5.p TPS problem from BASIC-HO-EQ-THMS
$\exists$xa: $i, xaa: $i, xb: $i, xbb: $i, xc: $i, xcc: $i, xd: $i, xdd: $i, xe: $i, xee: $i, xh: $i, xhh: $i: $\forall p$: $i $\to$ $i: (((((p@xa) = (p@xaa) and (p@xb) = (p@xbb) and (p@xe) = (p@xhh)) $\Rightarrow$ (p@xc) = (p@xdd)) and (((p@xa) = (p@xaa) and (p@xh) = (p@xhh) and (p@xb) = (p@xcc)) $\Rightarrow$ (p@xd) $\neq$ (p@xee)) and (((p@xc) = (p@xcc) and (p@xcc) = (p@xd) and (p@xd) = (p@xdd) and (p@xa) $\neq$ (p@xbb)) $\Rightarrow$ (p@xe) $\neq$ (p@xhh)) and (((p@xa) = (p@xaa) and (p@xd) = (p@xdd) and (p@xb) $\neq$ (p@xcc)) $\Rightarrow$ (p@xe) = (p@xhh)) and (((p@xe) = (p@xee) and (p@xh) = (p@xhh) and (p@xc) = (p@xdd)) $\Rightarrow$ (p@xa) $\neq$ (p@xbb)) and (((p@xb) = (p@xbb) and (p@xbb) = (p@xc) and (p@xc) = (p@xcc) and (p@xe) $\neq$ (p@xhh)) $\Rightarrow$ (p@xd) = (p@xee))) $\Rightarrow$ ((p@xa) $\neq$ (p@xaa) or (p@xb) $\neq$ (p@xbb) or (p@xc) $\neq$ (p@xcc) or (p@xd) $\neq$ (p@xdd) or (p@xe) $\neq$ (p@xee) or (p@xh) $\neq$ (p@xhh))) thf(cSIXFRIENDS_PTH, conjecture)

PUZ094^5.p TPS problem from BASIC-FO-THMS
Reduced version of Schubert's Steamroller.
eats: $i $\to$ $i $\to$ $o thf(eats, type)
grain: $i $\to$ $o thf(grain, type)
animal: $i $\to$ $o thf(animal, type)
snail: $i $\to$ $o thf(snail, type)
sf: $i $\to$ $i thf(sf, type)
plant: $i $\to$ $o thf(plant, type)
bird: $i $\to$ $o thf(bird, type)
wolf: $i $\to$ $o thf(wolf, type)
fox: $i $\to$ $o thf(fox, type)
msmaller: $i $\to$ $i $\to$ $o thf(msmaller, type)
a_grain: $i thf(a_grain, type)
a_snail: $i thf(a_snail, type)
a_bird: $i thf(a_bird, type)
a_fox: $i thf(a_fox, type)
a_wolf: $i thf(a_wolf, type)
$\neg\forall x$: $i: (animal@$x$ or $\neg$ wolf@x) and $\forall x$: $i: (animal@$x$ or $\neg$ fox@x) and $\forall x$: $i: (animal@$x$ or $\neg$ bird@x) and $\forall x$: $i: (animal

PUZ095^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i $\to$ $i thf(s_type, type)
cCKB6_BLACK: $i $\to$ $i $\to$ $o thf(cCKB6_BLACK_type, type)
cCKB_INF: ($i $\to$ $i $\to$ $o) $\to$ $o thf(cCKB_INF_type, type)
cCKB_INJ: ($i $\to$ $i $\to$ $i $\to$ $i $\to$ $o) $\to$ $o thf(cCKB_INJ_type, type)
cCKB_XPL: ($i $\to$ $i $\to$ $i $\to$ $i $\to$ $o) $\to$ ($i $\to$ $i $\to$ $o) $\to$ $i $\to$ $i $\to$ $o thf(cCKB_XPL_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: $\forall$xw: $i $\to$ $i $\to$ $o: ((xw@$c_1$@$c_1$ and $\forall$xj: $i, xk: $i: ((xw@xj@xk) $\Rightarrow$ (xw@(s@(s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB_INJ = (λxh: $i $\to$ $i $\to$ $i $\to$ $i $\to$ $o: $\forall$xx$_1$: i, xy_1$: i, xx_2$: i, xy_2$: $i, xu: $i, xv: $i: ((xh@xx$_1$@xy$_1$@xu@xv and xh@xx
(xx$_1$ = xx$_2$ and xy$_1$ = xy$_2$))) thf(cCKB_INJ_def, definition)
cCKB_XPL = (λxh: $i $\to$ $i $\to$ $i $\to$ $i $\to$ $o, xk: $i $\to$ $i $\to$ $o, xm: $i, xn: $i: (xk@xm@xn and $\forall$xx: $i, xy: $i: ((xk@xx@xy) =
$\exists$xu: $i, xv: $i: (xh@xx@xy@xu@xv and xk@xu@xv and $\neg$ xu = xm and xv = xn)))) thf(cCKB_XPL_def, definition)
cCKB_INF = (λxk: $i $\to$ $i $\to$ $o: $\exists$xh: $i $\to$ $i $\to$ $i $\to$ $i $\to$ $o, xm: $i, xn: $i: (cCKB_INJ@xh and cCKB_XPL@xh@xk@xm@
cCKB_INF@cCKB6_BLACK thf(cCKB6_L$_{50000}$, conjecture)

PUZ096^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i $\to$ $i $\to$ $i thf(g_type, type)
s: $i $\to$ $i thf(s_type, type)
cCKB6_BLACK: $i $\to$ $i $\to$ $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i $\to$ $i $\to$ $i $\to$ $i $\to$ $o thf(cCKB6_H_type, type)
cCKB_INJ: ($i $\to$ $i $\to$ $i $\to$ $i $\to$ $o) $\to$ $o thf(cCKB_INJ_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: $\forall$xw: $i $\to$ $i $\to$ $o: ((xw@$c_1$@$c_1$ and $\forall$xj: $i, xk: $i: ((xw@xj@xk) $\Rightarrow$ (xw@(s@(s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@(s@(s@xx))@(s@xy)) = c_1 and xu = (s@(s@(s@xx))) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_2 and xu = (s@(s@xx)) and xv = (s@(s@xy))) or ((g@(
c_3 and xu = (s@xx) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_4 and xu = (s@(s@xx)) and xv = xy)))) thf(cCK

cCKB_INJ = (λxh: $i → $i → $i → $i → $o: ∀xx$_1$: i, xy_1$: i, xx_2$: i, xy_2$: $i, xu: $i, xv: $i: ((xh@xx$_1$@xy$_1$@xu@xv and xh@xx
(xx$_1$ = xx$_2$ and xy$_1$ = xy$_2$))) thf(cCKB_INJ_def, definition)
cCKB_INJ@cCKB6_H thf(cCKB6_L$_{25000}$, conjecture)

PUZ097^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB_BLACK: $i → $i → $o thf(cCKB_BLACK_type, type)
cCKB_EVEN: $i → $o thf(cCKB_EVEN_type, type)
cCKB_H: $i → $i → $i → $i → $o thf(cCKB_H_type, type)
cCKB_INJ: ($i → $i → $i → $i → $o) → $o thf(cCKB_INJ_type, type)
cCKB_ODD: $i → $o thf(cCKB_ODD_type, type)
cCKB_INJ = (λxh: $i → $i → $i → $i → $o: ∀xx$_1$: i, xy_1$: i, xx_2$: i, xy_2$: $i, xu: $i, xv: $i: ((xh@xx$_1$@xy$_1$@xu@xv and xh@xx
(xx$_1$ = xx$_2$ and xy$_1$ = xy$_2$))) thf(cCKB_INJ_def, definition)
cCKB_EVEN = (λxx: $i: (xx = ($s@c_1) or xx = ($s@($s@($s@$c_1$))) or xx = ($s@($s@($s@($s@($s@c_1))))) or xx = ($s@($s@($s@($s@
cCKB_ODD = (λxx: $i: (xx = c_1 or xx = ($s@($s@c_1)) or xx = ($s@($s@($s@($s@c_1)))) or xx = ($s@($s@($s@($s@($s@($s@c_1))))))))
cCKB_BLACK = (λxu: $i, xv: $i: ((cCKB_ODD@xu and cCKB_ODD@xv) or (cCKB_EVEN@xu and cCKB_EVEN@xv)))
cCKB_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB_BLACK@xx@xy and (((g@($s@($s@xx))@($s@xy)) = c_1 and xu =
($s@($s@($s@xx))) and xv = ($s@xy)) or ((g@($s@($s@xx))@($s@xy)) = c_2 and xu = ($s@($s@xx)) and xv = ($s@($s@xy))) or ((g@(
c_3 and xu = ($s@xx) and xv = ($s@xy)) or ((g@($s@($s@xx))@($s@xy)) = c_4 and xu = ($s@($s@xx)) and xv = xy)))) thf(cCK
cCKB_INJ@cCKB_H thf(cL$_{2500}$, conjecture)

PUZ098^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB_BLACK: $i → $i → $o thf(cCKB_BLACK_type, type)
cCKB_EVEN: $i → $o thf(cCKB_EVEN_type, type)
cCKB_FIN: ($i → $i → $o) → $o thf(cCKB_FIN_type, type)
cCKB_INF: ($i → $i → $o) → $o thf(cCKB_INF_type, type)
cCKB_INJ: ($i → $i → $i → $i → $o) → $o thf(cCKB_INJ_type, type)
cCKB_ODD: $i → $o thf(cCKB_ODD_type, type)
cCKB_XPL: ($i → $i → $i → $i → $o) → ($i → $i → $o) → $i → $i → $o thf(cCKB_XPL_type, type)
cCKB_INJ = (λxh: $i → $i → $i → $i → $o: ∀xx$_1$: i, xy_1$: i, xx_2$: i, xy_2$: $i, xu: $i, xv: $i: ((xh@xx$_1$@xy$_1$@xu@xv and xh@xx
(xx$_1$ = xx$_2$ and xy$_1$ = xy$_2$))) thf(cCKB_INJ_def, definition)
cCKB_XPL = (λxh: $i → $i → $i → $i → $o, xk: $i → $i → $o, xm: $i, xn: $i: (xk@xm@xn and ∀xx: $i, xy: $i: ((xk@xx@xy) =
∃xu: $i, xv: $i: (xh@xx@xy@xu@xv and xk@xu@xv and ¬ xu = xm and xv = xn)))) thf(cCKB_XPL_def, definition)
cCKB_INF = (λxk: $i → $i → $o: ∃xh: $i → $i → $i → $i → $o, xm: $i, xn: $i: (cCKB_INJ@xh and cCKB_XPL@xh@xk@xm@
cCKB_FIN = (λxk: $i → $i → $o: ¬ cCKB_INF@xk) thf(cCKB_FIN_def, definition)
cCKB_ODD = (λxx: $i: (xx = c_1 or xx = ($s@($s@c_1)) or xx = ($s@($s@($s@($s@c_1)))) or xx = ($s@($s@($s@($s@($s@c_1)))))))
cCKB_EVEN = (λxx: $i: (xx = ($s@c_1) or xx = ($s@($s@($s@$c_1$))) or xx = ($s@($s@($s@($s@($s@c_1))))) or xx = ($s@($s@($s@($s@
cCKB_BLACK = (λxu: $i, xv: $i: ((cCKB_ODD@xu and cCKB_ODD@xv) or (cCKB_EVEN@xu and cCKB_EVEN@xv)))
cCKB_FIN@cCKB_BLACK thf(cL$_{7000}$, conjecture)

PUZ099^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
s: $i → $i thf(s_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s@($s@xu))))) ⇒ (xp@xy))) thf(cCK
¬ cCKB_E$_2$@c_1@c_2 thf(cCKB_L$_{38000}$, conjecture)

PUZ100^5.p TPS problem from CHECKERBOARD-THMS
s: $i → $i thf(s_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s@($s@xu))))) ⇒ (xp@xy))) thf(cCK
∀xx: $i: (cCKB_E$_2$@xx@xx) thf(cCKB_L$_{32000}$, conjecture)

PUZ101^5.p TPS problem from CHECKERBOARD-THMS

c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i → $i → $i → $i → $o thf(cCKB6_H_type, type)
cCKB_XPL: ($i → $i → $i → $i → $o) → ($i → $i → $o) → $i → $i → $o thf(cCKB_XPL_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@(s@(s@xx))@(s@xy)) = c_1 and xu =
(s@(s@(s@xx))) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_2 and xu = (s@(s@xx)) and xv = (s@(s@xy))) or ((g@(
c_3 and xu = (s@xx) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_4 and xu = (s@(s@xx)) and xv = xy)))) thf(cCK
cCKB_XPL = (λxh: $i → $i → $i → $i → $o, xk: $i → $i → $o, xm: $i, xn: $i: (xk@xm@xn and ∀xx: $i, xy: $i: ((xk@xx@xy) =
∃xu: $i, xv: $i: (xh@xx@xy@xu@xv and xk@xu@xv and ¬ xu = xm and xv = xn)))) thf(cCKB_XPL_def, definition)
cCKB_XPL@cCKB6_H@cCKB6_BLACK@c_1@c_1 thf(cCKB6_L$_{40000}$, conjecture)

PUZ102∧5.p TPS problem from CHECKERBOARD-THMS
s: $i → $i thf(s_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s$@($s$@xu))))) ⇒ (xp@xy))) thf(cCK
∀xx: $i: (cCKB_E$_2$@xx@($s$@($s$@xx))) thf(cCKB_L$_{33000}$, conjecture)

PUZ103∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i: ((cCKB6_NUM@xx) ⇒ (cCKB6_NUM@(s@xx))) thf(cCKB6_L$_{3000}$, conjecture)

PUZ104∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i: ((cCKB6_NUM@xx) ⇒ (cCKB6_NUM@(s@(s@xx)))) thf(cCKB6_L$_{4000}$, conjecture)

PUZ105∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s$@($s$@xu))))) ⇒ (xp@xy))) thf(cCK
∀xx: $i: ((cCKB6_NUM@xx) ⇒ (cCKB_E$_2$@(s@(s@xx))@xx)) thf(cCKB_L$_{36000}$, conjecture)

PUZ106∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s$@($s$@xu))))) ⇒ (xp@xy))) thf(cCK
∀xx: $i, xy: $i: ((cCKB6_BLACK@xx@xy) ⇒ (cCKB_E$_2$@xx@xy)) thf(cCKB_L$_{37000}$, conjecture)

PUZ107∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
cCKB_FIN: ($i → $i → $o) → $o thf(cCKB_FIN_type, type)
cCKB_INF: ($i → $i → $o) → $o thf(cCKB_INF_type, type)
cCKB_INJ: ($i → $i → $i → $i → $o) → $o thf(cCKB_INJ_type, type)
cCKB_XPL: ($i → $i → $i → $i → $o) → ($i → $i → $o) → $i → $i → $o thf(cCKB_XPL_type, type)

cCKB_INJ = (λxh: $i → $i → $i → $i → $o: ∀xx$_1$: i, xy_1$: i, xx_2$: i, xy_2$: $i, xu: $i, xv: $i: ((xh@xx$_1$@xy$_1$@xu@xv and xh@xx
(xx$_1$ = xx$_2$ and xy$_1$ = xy$_2$))) thf(cCKB_INJ_def, definition)
cCKB_XPL = (λxh: $i → $i → $i → $i → $o, xk: $i → $i → $o, xm: $i, xn: $i: (xk@xm@xn and ∀xx: $i, xy: $i: ((xk@xx@xy) =
∃xu: $i, xv: $i: (xh@xx@xy@xu@xv and xk@xu@xv and ¬ xu = xm and xv = xn)))) thf(cCKB_XPL_def, definition)
cCKB_INF = (λxk: $i → $i → $o: ∃xh: $i → $i → $i → $i → $o, xm: $i, xn: $i: (cCKB_INJ@xh and cCKB_XPL@xh@xk@xm@
cCKB_FIN = (λxk: $i → $i → $o: ¬ cCKB_INF@xk) thf(cCKB_FIN_def, definition)
cCKB_FIN@λxu: $i, xv: $i: (xu = c_1 and xv = c_1) thf(cL$_{3000}$, conjecture)

PUZ108^5.p TPS problem from CHECKERBOARD-THMS
s: $i → $i thf(s_type, type)
cCKB_E$_2$: $i → $i → $o thf(cCKB_E2_type, type)
cCKB_E$_2$ = (λxx: $i, xy: $i: ∀xp: $i → $o: ((xp@xx and ∀xu: $i: ((xp@xu) ⇒ (xp@($s$@($s$@xu))))) ⇒ (xp@xy))) thf(cCK
∀xx: $i, xy: $i: ((cCKB_E$_2$@xx@xy) ⇒ (cCKB_E$_2$@($s$@xx)@($s$@xy))) thf(cCKB_L$_{35000}$, conjecture)

PUZ109^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xj: $i, xk: $i: ((cCKB6_BLACK@xj@xk) ⇒ (cCKB6_BLACK@(s@xj)@(s@xk))) thf(cCKB6_L$_{9000}$, conjecture)

PUZ110^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xj: $i, xk: $i: ((cCKB6_BLACK@xj@xk) ⇒ (cCKB6_BLACK@(s@(s@xj))@xk)) thf(cCKB6_L$_{8000}$, conjecture)

PUZ111^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
cS: $i → $i thf(cS_type, type)
cX: $i thf(cX_type, type)
s: $i → $i thf(s_type, type)
cCKB_EVEN: $i → $o thf(cCKB_EVEN_type, type)
cCKB_ODD: $i → $o thf(cCKB_ODD_type, type)
cCKB_EVEN = (λxx: $i: (xx = ($s$@$c_1$) or xx = ($s$@($s$@($s$@$c_1$))) or xx = ($s$@($s$@($s$@($s$@$c_1$))))) or xx = ($s$@($s$@($s$@($s$@(s
cCKB_ODD = (λxx: $i: (xx = c_1 or xx = (s@(s@c_1)) or xx = (s@(s@(s@(s@c_1)))) or xx = (s@(s@(s@(s@(s@(s@c_1))))))))
∀xx: $i: ((cCKB_EVEN@xx) ⇒ (cCKB_EVEN@(s@(s@xx)) and cCKB_ODD@(cS@cX))) thf(cCKB_L$_{10000}$, conjecture)

PUZ112^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
cS: $i → $i thf(cS_type, type)
cX: $i thf(cX_type, type)
s: $i → $i thf(s_type, type)
cCKB_EVEN: $i → $o thf(cCKB_EVEN_type, type)
cCKB_ODD: $i → $o thf(cCKB_ODD_type, type)
cCKB_EVEN = (λxx: $i: (xx = ($s$@$c_1$) or xx = ($s$@($s$@($s$@$c_1$))) or xx = ($s$@($s$@($s$@($s$@$c_1$))))) or xx = ($s$@($s$@($s$@($s$@(s
cCKB_ODD = (λxx: $i: (xx = c_1 or xx = (s@(s@c_1)) or xx = (s@(s@(s@(s@c_1)))) or xx = (s@(s@(s@(s@(s@(s@c_1))))))))
∀xx: $i: ((cCKB_ODD@xx) ⇒ (cCKB_ODD@(s@(s@xx)) and cCKB_EVEN@(cS@cX))) thf(cCKB_L$_{9000}$, conjecture)

PUZ113^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xj: $i, xk: $i: ((cCKB6_BLACK@xj@xk) ⇒ (cCKB6_BLACK@(s@(s@xj))@(s@(s@xk)))) thf(cCKB6_L$_{11000}$, conjecture

PUZ114^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)

cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xj: $i, xk: $i: ((cCKB6_BLACK@xj@xk) ⇒ (cCKB6_BLACK@(s@(s@(s@xj)))@(s@xk))) thf(cCKB6_L$_{10000}$, conjecture

PUZ115∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i: ((cCKB6_NUM@xx) ⇒ ($s$@($s$@($s$@($s$@($s$@($s$@($s$@xx)))))))) = xx) thf(cCKB6_L$_{1000}$, conjecture)

PUZ116∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_5: $i thf(c5_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xx: $i, xy: $i: ((cCKB6_BLACK@xx@xy) ⇒ (g@(s@(s@xx))@(s@xy)) ≠ c_5) thf(cCKB_L$_{39100}$, conjecture)

PUZ117∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i, xy: $i: ((cCKB6_NUM@xx and cCKB6_NUM@xy and (s@xx) = (s@xy)) ⇒ xx = xy) thf(cCKB6_L$_{2000}$, conjectu

PUZ118∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
c_5: $i thf(c5_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i → $i → $i → $i → $o thf(cCKB6_H_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s$@($s$@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@(s@(s@xx))@(s@xy)) = c_1 and xu =
(s@(s@(s@xx))) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_2 and xu = (s@(s@xx)) and xv = (s@(s@xy))) or ((g@(
c_3 and xu = (s@xx) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_4 and xu = (s@(s@xx)) and xv = xy)))) thf(cCK
∀xx: $i, xy: $i, xu: $i, xv: $i: ((cCKB6_H@xx@xy@xu@xv) ⇒ (g@xu@xv) ≠ c_5) thf(cCKB6_L$_{31000}$, conjecture)

PUZ119∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i, xy: $i: ((cCKB6_NUM@xx and cCKB6_NUM@xy and (s@(s@xx)) = (s@(s@xy))) ⇒ xx = xy) thf(cCKB6_L250

PUZ120∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
s: $i → $i thf(s_type, type)
cCKB6_NUM: $i → $o thf(cCKB6_NUM_type, type)
cCKB6_NUM = (λxx: $i: ∀xp: $i → $o: ((xp@$c_1$ and ∀xw: $i: ((xp@xw) ⇒ (xp@($s$@xw)))) ⇒ (xp@xx))) thf(cCKB6_NU
∀xx: $i, xy: $i: ((cCKB6_NUM@xx and cCKB6_NUM@xy and (s@(s@(s@xx))) = (s@(s@(s@xy)))) ⇒ xx =
xy) thf(cCKB6_L$_{26000}$, conjecture)

PUZ121∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)

s: $i → $i thf(s_type, type)
cCKB_BLACK: $i → $i → $o thf(cCKB_BLACK_type, type)
cCKB_EVEN: $i → $o thf(cCKB_EVEN_type, type)
cCKB_ODD: $i → $o thf(cCKB_ODD_type, type)
cCKB_EVEN = (λxx: $i: (xx = ($s@c_1) or xx = ($s@($s@($s@$c_1$))) or xx = ($s@($s@($s@($s@($s@c_1)))))) or xx = ($s@($s@($s@($s@(s@
cCKB_ODD = (λxx: $i: (xx = c_1 or xx = ($s@($s@c_1)) or xx = ($s@($s@($s@($s@c_1)))) or xx = ($s@($s@($s@($s@($s@($s@c_1))))))))
cCKB_BLACK = (λxu: $i, xv: $i: ((cCKB_ODD@xu and cCKB_ODD@xv) or (cCKB_EVEN@xu and cCKB_EVEN@xv)))
∀xu: $i, xv: $i: ((cCKB_BLACK@xu@xv) ⇒ (cCKB_BLACK@($s@xu)@($s@xv) and cCKB_BLACK@xu@($s@($s@xv)) and cC

PUZ122^5.p TPS problem from CHECKERBOARD-THMS
cCKB_FIN: ($i → $i → $o) → $o thf(cCKB_FIN_type, type)
cCKB_INF: ($i → $i → $o) → $o thf(cCKB_INF_type, type)
cCKB_INJ: ($i → $i → $i → $i → $o) → $o thf(cCKB_INJ_type, type)
cCKB_XPL: ($i → $i → $i → $i → $o) → ($i → $i → $o) → $i → $i → $o thf(cCKB_XPL_type, type)
cCKB_INJ = (λxh: $i → $i → $i → $i → $o: ∀$xx_1$: $i, xy_1: $i, xx_2: $i, xy_2: $i, xu: $i, xv: $i: ((xh@$xx_1$@$xy_1$@xu@xv and xh@xx
(xx_1 = xx_2 and xy_1 = xy_2))) thf(cCKB_INJ_def, definition)
cCKB_XPL = (λxh: $i → $i → $i → $i → $o, xk: $i → $i → $o, xm: $i, xn: $i: (xk@xm@xn and ∀xx: $i, xy: $i: ((xk@xx@xy) =
∃xu: $i, xv: $i: (xh@xx@xy@xu@xv and xk@xu@xv and ¬ xu = xm and xv = xn)))) thf(cCKB_XPL_def, definition)
cCKB_INF = (λxk: $i → $i → $o: ∃xh: $i → $i → $i → $i → $o, xm: $i, xn: $i: (cCKB_INJ@xh and cCKB_XPL@xh@xk@xm@
cCKB_FIN = (λxk: $i → $i → $o: ¬ cCKB_INF@xk) thf(cCKB_FIN_def, definition)
∀xk: $i → $i → $o, xz: $i, xw: $i: ((cCKB_FIN@xk) ⇒ (cCKB_FIN@λxu: $i, xv: $i: (xk@xu@xv or (xu =
xz and xv = xw)))) thf(cL2000_pme, conjecture)

PUZ123^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_8: $i thf(c8_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s@($s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
∀xx: $i, xy: $i: ((cCKB6_BLACK@xx@xy) ⇒ ¬ (($s@($s@xx)) = c_1 and ($s@xy) = c_1) or (($s@($s@xx)) = c_8 and ($s@xy) =
c_8)) thf(cCKB_L_{39000}, conjecture)

PUZ124^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i → $i → $i → $i → $o thf(cCKB6_H_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s@($s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@($s@($s@xx))@($s@xy)) = c_1 and xu =
($s@($s@($s@xx))) and xv = ($s@xy)) or ((g@($s@($s@xx))@($s@xy)) = c_2 and xu = ($s@($s@xx)) and xv = ($s@($s@xy))) or ((g@(
c_3 and xu = ($s@xx) and xv = ($s@xy)) or ((g@($s@($s@xx))@($s@xy)) = c_4 and xu = ($s@($s@xx)) and xv = xy)))) thf(cCK
∀xx: $i, xy: $i, xu: $i, xv: $i: ((cCKB6_H@xx@xy@xu@xv and (g@xu@xv) = c_3) ⇒ (xu = ($s@($s@($s@xx))) and xv =
($s@xy))) thf(cCKB_$L_{43000}$, conjecture)

PUZ125^5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i → $i → $i → $i → $o thf(cCKB6_H_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@($s@($s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)

cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@(s@(s@xx))@(s@xy)) = c_1 and xu = (s@(s@(s@xx))) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_2 and xu = (s@(s@xx)) and xv = (s@(s@xy))) or ((g@(
c_3 and xu = (s@xx) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_4 and xu = (s@(s@xx)) and xv = xy)))) thf(cCK
∀xx: $i, xy: $i, xu: $i, xv: $i: ((cCKB6_H@xx@xy@xu@xv) ⇒ ((g@xu@xv) = c_1 or (g@xu@xv) = c_2 or (g@xu@xv) = c_3 or (g@xu@xv) = c_4)) thf(cCKB6_L$_{30000}$, conjecture)

PUZ126∧5.p TPS problem from CHECKERBOARD-THMS
c_1: $i thf(c1_type, type)
c_2: $i thf(c2_type, type)
c_3: $i thf(c3_type, type)
c_4: $i thf(c4_type, type)
g: $i → $i → $i thf(g_type, type)
s: $i → $i thf(s_type, type)
cCKB6_BLACK: $i → $i → $o thf(cCKB6_BLACK_type, type)
cCKB6_H: $i → $i → $i → $i → $o thf(cCKB6_H_type, type)
cCKB6_BLACK = (λxu: $i, xv: $i: ∀xw: $i → $i → $o: ((xw@$c_1$@$c_1$ and ∀xj: $i, xk: $i: ((xw@xj@xk) ⇒ (xw@(s@(s@xj))@xk
(xw@xu@xv))) thf(cCKB6_BLACK_def, definition)
cCKB6_H = (λxx: $i, xy: $i, xu: $i, xv: $i: (cCKB6_BLACK@xx@xy and (((g@(s@(s@xx))@(s@xy)) = c_1 and xu = (s@(s@(s@xx))) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_2 and xu = (s@(s@xx)) and xv = (s@(s@xy))) or ((g@(
c_3 and xu = (s@xx) and xv = (s@xy)) or ((g@(s@(s@xx))@(s@xy)) = c_4 and xu = (s@(s@xx)) and xv = xy)))) thf(cCK
∀xx: $i, xy: $i, xu: $i, xv: $i: ((cCKB6_H@xx@xy@xu@xv and (g@xu@xv) = c_3) ⇒ ((s@(s@(s@(s@(s@xu))))) = xx and (s@(s@(s@(s@(s@(s@xv)))))) = xy)) thf(cCKB_L$_{43100}$, conjecture)

PUZ127∧5.p TPS problem from CHECKERBOARD-THMS
c_5: $i thf($c_5$, type)
g: $i → $i → $i thf($g$, type)
c_4: $i thf($c_4$, type)
c_3: $i thf($c_3$, type)
c_2: $i thf($c_2$, type)
c_1: $i thf($c_1$, type)
s: $i → $i thf($s$, type)
c_8: $i thf($c_8$, type)
¬ (s@(s@(s@(s@(s@(s@(s@(s@c_8)))))))) = c_8 and ∀xx: $i: (s@(s@(s@(s@xx)))) ≠ xx and ∀xx: $i, xy: $i: ((g@xx@xy) = c_5 ⟺ ((xx = c_8 and xy = c_8) or (xx = c_1 and xy = c_1))) and ∀xx: $i, xy: $i: ((g@xx@xy) = c_1 ⟺ (g@(s@xx)@xy) = c_3) and ∀xx: $i, xy: $i: ((g@xx@xy) = c_2 ⟺ (g@xx@(s@xy)) = c_4) and ∀xx: $i, xy: $i: ((g@$c_1$@xy) ≠ c_3 and (g@c_8@xy) ≠ c_1 and (g@xx@c_1) ≠ c_4 and (g@xx@c_8) ≠ c_2) and c_1 = (s@c_8) and c_2 = (s@c_1) and c_3 = (s@c_2) and c_4 = (s@c_3) and c_5 = (s@c_4) and ∀xx: $i, xy: $i: ((g@xx@xy) = c_1 or (g@xx@xy) = c_2 or (g@xx@xy) = c_3 or (g@xx@xy) = c_4 or (g@xx@xy) = c_5) thf(cTOUGHNUT$_2$, conjecture)

PUZ128+1.p Iokaste patricide triangle
Iokaste is a parent of Oedipus. Iokaste is a parent of Polyneikes. Oedipus is a parent of Polyneikes. Polyneikes is a parent of Thersandros. Oedipus is a patricide. Thersandros is not a patricide. Therefore, Iokaste is a parent of a patricide who is a parent of somebody who is not a patricide.
parent_of(iokaste, oedipus) fof(iokaste_oedipus, axiom)
parent_of(iokaste, polyneikes) fof(iokaste_polyneikes, axiom)
parent_of(oedipus, polyneikes) fof(oedipus_polyneikes, axiom)
parent_of(polyneikes, thersandros) fof(polyneikes_thersandros, axiom)
patricide(oedipus) fof(oedipus_patricidal, axiom)
¬ patricide(thersandros) fof(thersandros_not_patricidal, axiom)
∃p, nP: (parent_of(iokaste, p) and patricide(p) and parent_of(p, nP) and ¬ patricide(nP)) fof(iokaste_parent_particide_pare

PUZ128+2.p Iokaste patricide triangle
Iokaste is a parent of Oedipus. Iokaste is a parent of Polyneikes. Oedipus is a parent of Polyneikes. Polyneikes is a parent of Thersandros. Oedipus is a patricide. Thersandros is not a patricide. Therefore, Iokaste is a parent of a patricide that is a parent of somebody who is not a patricide.
∃a, b, c, d, e: (parent(a) and relation(a, of, 'Oedipus') and 'Iokaste' = a and parent(b) and relation(b, of, 'Polyneikes') and 'Iok
b and parent(c) and relation(c, of, 'Polyneikes') and 'Oedipus' = c and parent(d) and relation(d, of, 'Thersandros') and 'Polyn
d and patricide(e) and 'Oedipus' = e and ¬ ∃f: (patricide(f) and 'Thersandros' = f)) fof(background, axiom)
∃a, b, c, d: (parent(a) and patricide(b) and parent(c) and $true and ¬ ∃$e$: (patricide($e$) and $d = e$) and relation(c, of, d) and b
c and relation(a, of, b) and 'Iokaste' = a) fof(prove, conjecture)

PUZ129+2.p The grocer is not a cyclist

If every honest and industrious person is healthy, and no grocer is healthy, and every industrious grocer is honest, and every cyclist is industrious, and every unhealthy cyclist is dishonest, and no healthy person is unhealthy, and no honest person is dishonest, and every grocer is a person, and every cyclist is a person then no grocer is a cyclist.
($\forall a$: ((person(a) and property$_1$(a, honest, pos) and property$_1$(a, industrious, pos)) $\Rightarrow$ $\exists b$: (property$_1$(b, healthy, pos) and $a =$ b)) and $\forall c$: (grocer(c) $\Rightarrow$ $\neg\,\exists d$: (property$_1$(d, healthy, pos) and $c = d$)) and $\forall e$: ((grocer(e) and property$_1$(e, industrious, pos)) $\exists f$: (property$_1$(f, honest, pos) and $e = f$)) and $\forall g$: (cyclist(g) $\Rightarrow$ $\exists h$: (property$_1$(h, industrious, pos) and $g =$ h)) and $\forall i$: ((cyclist(i) and property$_1$(i, unhealthy, pos)) $\Rightarrow$ $\exists j$: (property$_1$(j, dishonest, pos) and $i = j$)) and $\forall k$: ((person(k) $\neg\,\exists l$: (property$_1$(l, unhealthy, pos) and $k = l$)) and $\forall m$: ((person(m) and property$_1$(m, honest, pos)) $\Rightarrow$ $\neg\,\exists n$: (property$_1$(n, d n)) and $\forall o$: (grocer(o) $\Rightarrow$ $\exists p$: (person(p) and $o = p$)) and $\forall q$: (cyclist(q) $\Rightarrow$ $\exists r$: (person(r) and $q = r$))) $\Rightarrow$ $\forall s$: (grocer(s) $\Rightarrow$ $\neg\,\exists t$: (cyclist(t) and $s = t$)) fof(prove, conjecture)

PUZ130+1.p Garfield and Odie
Garfield is a cat and Odie is a dog. Cats and dogs are pets. John is a human. Every pet has a human owner. Jon owns Garfield and Odie, and they are the only cat and dog he owns. If a dog chases a cat, then the cat's owner hates the dog's owner. Odie has chased Garfield. Therefore, Jon hates himself.
$\exists a$: cat(a) fof(cat_type, axiom)
cat(garfield) fof(garfield_type, axiom)
$\exists a$: dog(a) fof(dog_type, axiom)
dog(odie) fof(odie_type, axiom)
$\exists a$: pet(a) fof(pet_type, axiom)
$\forall a$: (dog(a) $\Rightarrow$ pet(a)) fof(dog_pet_type, axiom)
$\forall a$: (cat(a) $\Rightarrow$ pet(a)) fof(cat_pet_type, axiom)
$\exists a$: human(a) fof(human_type, axiom)
human(jon) fof(jon_type, axiom)
$\forall a$: (pet(a) $\Rightarrow$ human(owner_of(a))) fof(owner_of_type, axiom)
$\forall x$: (pet(x) $\Rightarrow$ $\exists y$: (human(y) and owner(x, y))) fof(pet_owner_axiom, axiom)
owner(jon, odie) fof(jon_o_owner_axiom, axiom)
owner(jon, garfield) fof(jon_g_owner_axiom, axiom)
$\forall x$: ($\forall x$: (cat(x) $\Rightarrow$ owner(jon, x)) $\Rightarrow$ $x =$ garfield) fof(jon_only_g_owner_axiom, axiom)
$\forall x$: ($\forall x$: (dog(x) $\Rightarrow$ owner(jon, x)) $\Rightarrow$ $x =$ odie) fof(jon_only_o_owner_axiom, axiom)
$\forall x, y$: ((cat(x) and dog(y)) $\Rightarrow$ (chased(y, x) $\Rightarrow$ hates(owner_of(x), owner_of(y)))) fof(cat_chase_axiom, axiom)
$\forall x, y$: ((human(x) and pet(y)) $\Rightarrow$ (owner(x, y) $\Longleftrightarrow$ $x =$ owner_of(y))) fof(owner_def, axiom)
chased(odie, garfield) fof(odie_chase_axiom, axiom)
hates(jon, jon) fof(jon_conjecture, conjecture)

PUZ130_1.p Garfield and Odie
Garfield is a cat and Odie is a dog. Cats and dogs are pets. John is a human. Every pet has a human owner. Jon owns Garfield and Odie, and they are the only cat and dog he owns. If a dog chases a cat, then the cat's owner hates the dog's owner. Odie has chased Garfield. Therefore, Jon hates himself.
animal: $tType tff(animal_type, type)
cat: $tType tff(cat_type, type)
dog: $tType tff(dog_type, type)
human: $tType tff(human_type, type)
cat_to_animal: cat $\rightarrow$ animal tff(cat_to_animal_type, type)
dog_to_animal: dog $\rightarrow$ animal tff(dog_to_animal_type, type)
garfield: cat tff(garfield_type, type)
odie: dog tff(odie_type, type)
jon: human tff(jon_type, type)
owner_of: animal $\rightarrow$ human tff(owner_of_type, type)
chased: (dog $\times$ cat) $\rightarrow$ $o tff(chased_type, type)
hates: (human $\times$ human) $\rightarrow$ $o tff(hates_type, type)
$\forall a$: animal: $\exists h$: human: $h =$ owner_of(a) tff(human_owner, axiom)
jon = owner_of(cat_to_animal(garfield)) tff(jon_owns_garfield, axiom)
jon = owner_of(dog_to_animal(odie)) tff(jon_owns_odie, axiom)
$\forall a$: animal: (jon = owner_of(a) $\Rightarrow$ (a = cat_to_animal(garfield) or a = dog_to_animal(odie))) tff(jon_owns_only, axiom)
$\forall c$: cat, d: dog: (chased(d, c) $\Rightarrow$ hates(owner_of(cat_to_animal(c)), owner_of(dog_to_animal(d)))) tff(dog_chase_cat, axiom)
chased(odie, garfield) tff(odie_chased_garfield, axiom)
hates(jon, jon) tff(jon_hates_jon, conjecture)

PUZ131+1.p Victor teaches Michael

Every student is enrolled in at least one course. Every professor teaches at least one course. Every course has at least one student enrolled. Every course has at least one professor teaching. The coordinator of a course teaches the course. If a student is enroled in a course then the student is taught by every professor who teaches the course. Michael is enrolled in CSC410. Victor is the coordinator of CSC410. Therefore, Michael is taught by Victor.
$\exists a$: student(a) fof(student_type, axiom)
$\exists a$: professor(a) fof(professor_type, axiom)
$\exists a$: course(a) fof(course_type, axiom)
student(michael) fof(michael_type, axiom)
professor(victor) fof(victor_type, axiom)
course(csc_{410}) fof(csc410_type, axiom)
$\forall a$: (course(a) $\Rightarrow$ professor(coordinatorof(a))) fof(coordinator_of_type, axiom)
$\forall x$: (student(x) $\Rightarrow$ $\exists y$: (course(y) and enrolled(x, y))) fof(student_enrolled_axiom, axiom)
$\forall x$: (professor(x) $\Rightarrow$ $\exists y$: (course(y) and teaches(x, y))) fof(professor_teaches, axiom)
$\forall x$: (course(x) $\Rightarrow$ $\exists y$: (student(y) and enrolled(y, x))) fof(course_enrolled, axiom)
$\forall x$: (course(x) $\Rightarrow$ $\exists y$: (professor(y) and teaches(y, x))) fof(course_teaches, axiom)
$\forall x$: (course(x) $\Rightarrow$ teaches(coordinatorof$(x), x$)) fof(coordinator_teaches, axiom)
$\forall x, y$: ((student(x) and course(y)) $\Rightarrow$ (enrolled(x, y) $\Rightarrow$ $\forall z$: (professor(z) $\Rightarrow$ (teaches(z, y) $\Rightarrow$ taughtby(x, z)))))) fof(stu
enrolled(michael, csc_{410}) fof(michael_enrolled_csc410_axiom, axiom)
coordinatorof(csc_{410}) = victor fof(victor_coordinator_csc410_axiom, axiom)
taughtby(michael, victor) fof(teaching_conjecture, conjecture)

PUZ131_1.p Victor teaches Michael
Every student is enrolled in at least one course. Every professor teaches at least one course. Every course has at least one student enrolled. Every course has at least one professor teaching. The coordinator of a course teaches the course. If a student is enroled in a course then the student is taught by every professor who teaches the course. Michael is enrolled in CSC410. Victor is the coordinator of CSC410. Therefore, Michael is taught by Victor.
student: $tType tff(student_type, type)
professor: $tType tff(professor_type, type)
course: $tType tff(course_type, type)
michael: student tff(michael_type, type)
victor: professor tff(victor_type, type)
csc_{410}: course tff(csc410_type, type)
enrolled: (student $\times$ course) $\rightarrow$ $o tff(enrolled_type, type)
teaches: (professor $\times$ course) $\rightarrow$ $o tff(teaches_type, type)
taughtby: (student $\times$ professor) $\rightarrow$ $o tff(taught_by_type, type)
coordinatorof: course $\rightarrow$ professor tff(coordinator_of_type, type)
$\forall x$: student: $\exists y$: course: enrolled(x, y) tff(student_enrolled_axiom, axiom)
$\forall x$: professor: $\exists y$: course: teaches(x, y) tff(professor_teaches, axiom)
$\forall x$: course: $\exists y$: student: enrolled(y, x) tff(course_enrolled, axiom)
$\forall x$: course: $\exists y$: professor: teaches(y, x) tff(course_teaches, axiom)
$\forall x$: course: teaches(coordinatorof$(x), x$) tff(coordinator_teaches, axiom)
$\forall x$: student, y: course: (enrolled(x, y) $\Rightarrow$ $\forall z$: professor: (teaches(z, y) $\Rightarrow$ taughtby(x, z))) tff(student_enrolled_taught, axi
enrolled(michael, csc_{410}) tff(michael_enrolled_csc410_axiom, axiom)
coordinatorof(csc_{410}) = victor tff(victor_coordinator_csc410_axiom, axiom)
taughtby(michael, victor) tff(teaching_conjecture, conjecture)

PUZ132+1.p Crime in beautiful Washington
A capital is a city. USA is a country. Every city has crime. Washington is the capital of the USA. Every country has a beautiful capital. Therefore, Washington is beautiful but has crime.
$\exists a$: capital(a) fof(capital_type, axiom)
$\exists a$: city(a) fof(city_type, axiom)
$\forall a$: (capital(a) $\Rightarrow$ city(a)) fof(capital_city_type, axiom)
$\exists a$: country(a) fof(country_type, axiom)
capital(washington) fof(washington_type, axiom)
country(usa) fof(usa_type, axiom)
$\forall a$: (country(a) $\Rightarrow$ capital(capital_city(a))) fof(country_capital_type, axiom)
$\forall x$: (city(x) $\Rightarrow$ has_crime(x)) fof(crime_axiom, axiom)
capital_city(usa) = washington fof(usa_capital_axiom, axiom)
$\forall x$: (country(x) $\Rightarrow$ beautiful(capital_city(x))) fof(beautiful_capital_axiom, axiom)
beautiful(washington) and has_crime(washington) fof(washington_conjecture, conjecture)

PUZ133+1.p N queens problem has the variable symmetry property
queens_p $\Rightarrow$ $\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n)) $\Rightarrow$ ($p(i) \neq p(j)$ and $p(i) + i \neq p(j) + j$ and $-p(i) \neq -p(j)$))) fof(queens_p, axiom)
$\forall i$: perm(i) $= -s(n)$ fof(permutation, axiom)
$\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n) and (le($s(i), j$) $\iff$ le(s(perm(j)), perm(i)))) $\Rightarrow$ ($q(i) \neq q(j)$ and $q(i) + i \neq q(j) + j$ and $-q(i) \neq -q(j)$))) $\Rightarrow$ queens_q fof(queens_q, axiom)
(queens_p and $\forall i$: $q(i) = p$(perm(i))) $\Rightarrow$ queens_q fof(queens_sym, conjecture)
$\forall i$: ((le($s(n_0), i$) and le(i, n)) $\Rightarrow$ (le($s(n_0)$, perm(i)) and le(perm(i), n))) fof(permutation_range, axiom)
$\forall j, i$: $-i = -$perm(j) fof(permutation_another_one, axiom)
$\forall x, y, z$: ((le(x, y) and le(y, z)) $\Rightarrow$ le(x, z)) fof(le_trans, axiom)
$\forall x$: le($x, s(x)$) fof(succ_le, axiom)
$\forall i, j, k, l$: ($i + j = k + l \iff -i = -l$) fof(plus$_1$, axiom)
$\forall i, j, k, l$: ($-i = -k \iff -i = -j$) fof(minus$_1$, axiom)

PUZ133+2.p N queens problem has the variable symmetry property
queens_p $\Rightarrow$ $\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n)) $\Rightarrow$ ($p(i) \neq p(j)$ and $p(i) + i \neq p(j) + j$ and $-p(i) \neq -p(j)$))) fof(queens_p, axiom)
$\forall i$: ((le($s(n_0), i$) and le(i, n)) $\Rightarrow$ perm(i) $= -s(n)$) fof(permutation, axiom)
$\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n) and (le($s(i), j$) $\iff$ le(s(perm(j)), perm(i)))) $\Rightarrow$ ($q(i) \neq q(j)$ and $q(i) + i \neq q(j) + j$ and $-q(i) \neq -q(j)$))) $\Rightarrow$ queens_q fof(queens_q, axiom)
(queens_p and $\forall i$: $q(i) = p$(perm(i))) $\Rightarrow$ queens_q fof(queens_sym, conjecture)
$\forall i$: ((le($s(n_0), i$) and le(i, n)) $\Rightarrow$ (le($s(n_0)$, perm(i)) and le(perm(i), n))) fof(permutation_range, axiom)
$\forall j, i$: $-i = -$perm(j) fof(permutation_another_one, axiom)
$\forall x, y, z$: ((le(x, y) and le(y, z)) $\Rightarrow$ le(x, z)) fof(le_trans, axiom)
$\forall x$: le($x, s(x)$) fof(succ_le, axiom)
$\forall i, j, k, l$: ($i + j = k + l \iff -i = -l$) fof(plus$_1$, axiom)
$\forall i, j, k, l$: ($-i = -k \iff -i = -j$) fof(minus$_1$, axiom)

PUZ133+3.p N queens problem has the variable symmetry property
queens_p $\Rightarrow$ $\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n)) $\Rightarrow$ ($p(i) \neq p(j)$ and $p(i) + i \neq p(j) + j$ and $-p(i) \neq -p(j)$))) fof(queens_p, axiom)
$\forall i$: perm(i) $= -s(n)$ fof(permutation, axiom)
$\forall i, j$: ((le($s(n_0), i$) and le(i, n) and le($s(i), j$) and le(j, n)) $\Rightarrow$ ($q(i) \neq q(j)$ and $q(i) + i \neq q(j) + j$ and $-q(i) \neq -q(j)$))) $\Rightarrow$ queens_q fof(queens_q, axiom)
(queens_p and $\forall i$: $q(i) = p$(perm(i))) $\Rightarrow$ queens_q fof(queens_sym, conjecture)
$\forall i$: ((le($s(n_0), i$) and le(i, n)) $\Rightarrow$ (le($s(n_0)$, perm(i)) and le(perm(i), n))) fof(permutation_range, axiom)
$\forall i, j$: (lt(i, j) $\iff$ lt(perm(j), perm(i))) fof(permutation_anti_monotone, axiom)
$\forall j, i$: $-i = -$perm(j) fof(permutation_another_one, axiom)
$\forall x, y, z$: ((le(x, y) and le(y, z)) $\Rightarrow$ le(x, z)) fof(le_trans, axiom)
$\forall x$: le($x, s(x)$) fof(succ_le, axiom)
$\forall i, j, k, l$: ($i + j = k + l \iff -i = -l$) fof(plus$_1$, axiom)
$\forall i, j, k, l$: ($-i = -k \iff -i = -j$) fof(minus$_1$, axiom)

PUZ133=2.p N queens problem has the variable symmetry property
queens_q: \$o tff(queens_q_type, type)
queens_p: \$o tff(queens_p_type, type)
q: \$int $\to$ \$int tff(q_type, type)
p: \$int $\to$ \$int tff(p_type, type)
n: \$int tff(n_type, type)
perm: \$int $\to$ \$int tff(perm_type, type)
queens_p $\Rightarrow$ $\forall i$: \$int, j: \$int: ((\$lesseq(1, i) and \$lesseq(i, n) and \$lesseq(\$sum(1, i), j) and \$lesseq(j, n)) $\Rightarrow$ ($p(i) \neq p(j)$ and \$sum($p(i), i$) $\neq$ \$sum($p(j), j$) and \$difference($p(i), i$) $\neq$ \$difference($p(j), j$))) tff(queens_p, axiom)
$\forall i$: \$int: perm(i) $=$ \$difference(\$sum(1, n), i) tff(permutation, axiom)
$\forall i$: \$int, j: \$int: ((\$lesseq(1, i) and \$lesseq(i, n) and \$lesseq(\$sum(1, i), j) and \$lesseq(j, n) and (\$lesseq(\$sum(1, i), j) $\iff$ \$lesseq(\$sum(1, perm(j)), perm(i)))) $\Rightarrow$ ($q(i) \neq q(j)$ and \$sum($q(i), i$) $\neq$ \$sum($q(j), j$) and \$difference($q(i), i$) $\neq$ \$difference($q(j), j$))) $\Rightarrow$ queens_q tff(queens_q, axiom)
(queens_p and $\forall i$: \$int: $q(i) = p$(perm(i))) $\Rightarrow$ queens_q tff(queens_sym, conjecture)
$\forall i$: \$int: ((\$lesseq(1, i) and \$lesseq(i, n)) $\Rightarrow$ (\$lesseq(1, perm(i)) and \$lesseq(perm(i), n))) tff(permutation_range, axiom)
$\forall j$: \$int, i: \$int: \$difference(i, j) $=$ \$difference(perm(j), perm(i)) tff(permutation_another_one, axiom)

PUZ136∧1.p Under two assumptions there are at least two individuals.

parent: $i → $i → $o thf(parent, type)
kronus: $i thf(kronus, type)
zeus: $i thf(zeus, type)
parent@kronus@zeus thf(ax_1, axiom)
sutekh: $i thf(sutekh, type)
horus: $i thf(horus, type)
¬ parent@sutekh@horus thf(ax_2, axiom)
$\exists p$: $i → $o, x: $i, y: $i: ($p@x$ and ¬ $p@y$) thf(hotwo, conjecture)

PUZ137∧1.p Peter the liar says everything
Peter says that everything he says is false. Show that Peter says everything.
peter: $i thf(peter, type)
says: $i → $o → $o thf(says, type)
says@peter@$\forall x$: $o: ((says@peter@$x$) ⇒ ¬ x) thf(ax_1, axiom)
$\forall x$: $o: (says@peter@$x$) thf(thm, conjecture)

PUZ138+2.p Platinum Blonde
+—+—+—+ — — — 12— — — — 3— — 2—3 —4 — +—+—+—+ — 1—8 — 5— — 6 — 7 —8 — — — 9— — +—+—+—+ — 8—5 — — —9 — 4 —5 — —47 — 6— — +—+—+—+
include('Axioms/PUZ006+0.ax')
$p(n_1, n_8, n_1)$ fof(ax_{181}, axiom)
$p(n_1, n_9, n_2)$ fof(ax_{192}, axiom)
$p(n_2, n_9, n_3)$ fof(ax_{293}, axiom)
$p(n_3, n_3, n_2)$ fof(ax_{332}, axiom)
$p(n_3, n_4, n_3)$ fof(ax_{343}, axiom)
$p(n_3, n_7, n_4)$ fof(ax_{374}, axiom)
$p(n_4, n_3, n_1)$ fof(ax_{431}, axiom)
$p(n_4, n_4, n_8)$ fof(ax_{448}, axiom)
$p(n_4, n_9, n_5)$ fof(ax_{495}, axiom)
$p(n_5, n_2, n_6)$ fof(ax_{526}, axiom)
$p(n_5, n_5, n_7)$ fof(ax_{557}, axiom)
$p(n_5, n_7, n_8)$ fof(ax_{578}, axiom)
$p(n_6, n_6, n_9)$ fof(ax_{669}, axiom)
$p(n_7, n_3, n_8)$ fof(ax_{738}, axiom)
$p(n_7, n_4, n_5)$ fof(ax_{745}, axiom)
$p(n_8, n_1, n_9)$ fof(ax_{819}, axiom)
$p(n_8, n_5, n_4)$ fof(ax_{854}, axiom)
$p(n_8, n_7, n_5)$ fof(ax_{875}, axiom)
$p(n_9, n_1, n_4)$ fof(ax_{914}, axiom)
$p(n_9, n_2, n_7)$ fof(ax_{927}, axiom)
$p(n_9, n_6, n_6)$ fof(ax_{966}, axiom)

PUZ139_1.p Caramel vanilla coffee helps people stay awake
beverage: $tType tff(beverage_type, type)
syrup: $tType tff(syrup_type, type)
coffee: beverage tff(coffee_type, type)
vanilla_syrup: syrup tff(vanilla_syrup_type, type)
caramel_syrup: syrup tff(caramel_syrup_type, type)
mixture: ∀ > beverageOrSyrup: $tType: (beverageOrSyrup × syrup) → beverageOrSyrup tff(mixture_type, type)
help_people_stay_awake: beverage → $o tff(help_people_stay_awake_type, type)
∀s: syrup: help_people_stay_awake(mixture(beverage, coffee, s)) tff(mixture_of_coffee_help_people_stay_awake, axiom)
help_people_stay_awake(mixture(beverage, coffee, mixture(syrup, caramel_syrup, vanilla_syrup))) tff(caramel_vanilla_coffee

PUZ140∧1.p A mixture of coffee and syrup that is hot
syrup: $tType thf(syrup_type, type)
beverage: $tType thf(beverage_type, type)
coffee: beverage thf(coffee_type, type)
heat: beverage → beverage thf(heat_type, type)
hot: beverage → $o thf(hot_type, type)
mix: beverage → syrup → beverage thf(mix_type, type)
hot_mixture: beverage → syrup → beverage thf(hot_mixture_type, type)

cold_mixture: beverage → syrup → beverage thf(cold_mixture_type, type)
hot_mixture = (λb: beverage, s: syrup: (heat@(mix@b@s))) thf(hot_mixture_definition, definition)
cold_mixture = (λb: beverage, s: syrup: (mix@b@s)) thf(cold_mixture_definition, definition)
∀b: beverage: (hot@(heat@b)) thf(its_hot, axiom)
∃mixture: beverage → syrup → beverage: ∀s: syrup: ∃b: beverage: ((mixture@coffee@s) = b and hot@b) thf(hot_coffee, co

PUZ140∧2.p A mixture of coffee and syrup that is hot
syrup: $tType thf(syrup_type, type)
beverage: $tType thf(beverage_type, type)
coffee: beverage thf(coffee_type, type)
mix: beverage → syrup → beverage thf(mix_type, type)
coffee_mixture: syrup → beverage thf(coffee_mixture_type, type)
hot: beverage → $o thf(hot_type, type)
coffee_mixture = (mix@coffee) thf(coffee_mixture_definition, definition)
∀s: syrup: ((coffee_mixture@s) = coffee and hot@(coffee_mixture@s)) thf(coffee_and_syrup_is_hot_coffee, axiom)
∃syrupMixer: syrup → beverage: ∀s: syrup: ∃b: beverage: (b = (syrupMixer@s) and b = coffee and hot@b) thf(there_is_ho

PUZ141∧1.p Labyrinth1
Move 2 to the right.
position: $tType thf(position_type, type)
direction: $tType thf(direction_type, type)
left: direction thf(left_type, type)
right: direction thf(right_type, type)
⊤: direction thf(top_type, type)
bottom: direction thf(bottom_type, type)
next: position → direction → position thf(next_type, type)
∀d_1: direction, d_2: direction, p: position: (next@(next@p@d_1)@d_2) = (next@(next@p@d_2)@d_1) thf(next_comm, axiom)
∀p: position: (next@(next@p@left)@right) = p thf(left_right, axiom)
∀p: position: (next@(next@p@⊤)@bottom) = p thf(top_bottom, axiom)
wall: position → $o thf(wall_type, type)
movelist: $tType thf(movelist_type, type)
nomove: movelist thf(nomove_type, type)
movedir: movelist → direction → movelist thf(movedir_type, type)
playerpos: movelist → position thf(playerpos_type, type)
∀pO: position, m: movelist, d: direction: ((playerpos@m) = pO ⇒ (¬ wall@(next@pO@d) ⇒ (playerpos@(movedir@m@d)) =
(next@pO@d))) thf(player_move_legal, axiom)
∀pO: position, m: movelist, d: direction: ((playerpos@m) = pO ⇒ ((wall@(next@pO@d)) ⇒ (playerpos@(movedir@m@d)) =
pO)) thf(player_move_illegal, axiom)
c_{00}: position thf(c00_type, type)
c_{10}: position thf(c10_type, type)
c_{20}: position thf(c20_type, type)
c_{10} = (next@c_{00}@right) thf(c10_defin, definition)
c_{20} = (next@c_{10}@right) thf(c20_defin, definition)
(wall@c_{00}) = $false thf(c00_axiom, axiom)
(wall@c_{10}) = $false thf(c10_axiom, axiom)
(wall@c_{20}) = $false thf(c20_axiom, axiom)
(playerpos@nomove) = c_{00} thf(start_axiom, axiom)
∃m: movelist: (playerpos@m) = c_{20} thf(exercise, conjecture)

PUZ146∧1.p Peter and Mary have different hobbies
hobby: $tType thf(hobby_type, type)
earthling: $tType thf(earthling_type, type)
peter: earthling thf(peter, type)
mary: earthling thf(mary, type)
beer_drinking: hobby thf(beer_drinking, type)
belly_dancing: hobby thf(belly_dancing, type)
weight_lifting: hobby thf(weight_lifting, type)
has_hobby: earthling → hobby → $o thf(has_hobby, type)
peters_hobbies: hobby → $o thf(peters_hobbies, type)
marys_hobbies: hobby → $o thf(marys_hobbies, type)
peter ≠ mary thf(not_the_same$_1$, axiom)

beer_drinking ≠ belly_dancing and belly_dancing ≠ weight_lifting and beer_drinking ≠ weight_lifting thf(not_the_same$_2$, a
peters_hobbies = (has_hobby@peter) thf(peters_hobbies$_{001}$, definition)
marys_hobbies = (has_hobby@mary) thf(marys_hobbies$_{002}$, definition)
marys_hobbies@belly_dancing thf(mary_does_belly_dancing, axiom)
¬ marys_hobbies@beer_drinking thf(mary_does_not_do_beer_drinking, axiom)
peters_hobbies@beer_drinking thf(peter_does_beer_drinking, axiom)
peters_hobbies@weight_lifting thf(peter_does_weight_lifting, axiom)
peters_hobbies ≠ marys_hobbies thf(peter_and_mary_have_different_hobbies, conjecture)