```
/*
        Introduction to Formal Proof, Oxford, 2014
*/
INITIALISE foldsequents         false
INITIALISE multiassumptionlines false
INITIALISE foldformulae         false
INITIALISE outermostbox false

INITIALISE outerassumptionword   premiss
INITIALISE outerassumptionplural premisses
INITIALISE innerassumptionword   assumption
INITIALISE innerassumptionplural assumptions
INITIALISE seektipselection false
INITIALISE multihypsel true

USE "IFP_syntax.j"
USE "IFP_rules.j"
USE "IFP_style_menus.j"
USE "IFP_tactics.j"

/* Establish an order for the menus */
MENU Rules IS END
MENU "[Rules]" IS END
MENU "Quantifier Rules" IS END

USE "IFP_intro_conjectures.j"

KEYBOARD     ¬      /* */

MENU "[Rules]" IS
   ENTRY hyp
   ENTRY " elim"

SEPARATOR

   ENTRY " intro"
   ENTRY " intro"
   ENTRY " intro(L)"
   ENTRY " intro(R)"
   ENTRY " intro"

SEPARATOR


   ENTRY " elim"
   ENTRY " elim(L)"
   ENTRY " elim(R)"
   ENTRY " elim"
   ENTRY " elim(L)"
   ENTRY " elim(R)"

SEPARATOR

   ENTRY "¬ intro"
   ENTRY "¬ elim"
   ENTRY "¬¬ elim"

SEPARATOR
SEPARATOR

   ENTRY cut
   ENTRY thin

SEPARATOR
SEPARATOR
  ENTRY " intro"
  ENTRY " elim"

END

MENU "Quantifier Rules" IS

   ENTRY " intro" IS (WITHSELECTIONS   " intro")
   ENTRY " intro" IS (WITHSELECTIONS   " intro")

SEPARATOR
```

```
   ENTRY " elim"  IS (WITHSELECTIONS   " elim")
   ENTRY " elim"  IS (WITHSELECTIONS   " elim")

END

MENU "= Rules" IS
   ENTRY "= intro"
   ENTRY "= elim"
   SEPARATOR
   ENTRY "= elim (select t1=t2) (text-select t1 in (t1) to cut with (t2))" IS
         ForwardEqElim
END


/************************* Double-clicking ********************/
USE "IFP_hits.j"

/************************* Derived Rules **********************/
MENU "Rules" IS
   SEPARATOR
   SEPARATOR
   RULE "RAA (derived)" IS FROM ¬A   INFER A
   RULE "  (derived)" IS FROM A , B  C INFER AB  C
   RULE "¬¬ (derived)" IS FROM B  C INFER ¬¬B  C
END
   /* DERIVED */
   RULE ""(T)  IS FROM P(T)   C INFER x . P(x)  C


TACTIC ResolveOrTheorem (thm) IS (ALT (ApplyOrResolve thm) (TheoremForwardOrBackward thm))
TACTIC ApplyOrResolve   (thm) IS (WITHHYPSEL (ALT thm (RESOLVE thm)))
```

```
CONJECTUREPANEL Conjectures
    THEOREM IS  E, F  EF
    THEOREM IS  EF  E
    THEOREM IS  EF  F
    THEOREM IS  E(FG)  (EF)G
    THEOREM IS  (EF)G  E(FG)

    THEOREM IS  E, EF  F
    THEOREM IS  EF, FG, E  G
    THEOREM IS  E(FG), EF, E  G
    THEOREM IS  EF, FG  EG
    THEOREM IS  EFG  FEG
    THEOREM IS  EFG  (EF)(EG)
    THEOREM IS  E  FE
    THEOREM IS   EFE
    THEOREM IS  EF  (FG)EG
    THEOREM IS  EFGH  GFEH
    THEOREM IS   (EFG)(EF)EG
    THEOREM IS  (EF)G  EFG

    THEOREM IS  EF  EF
    THEOREM IS  (EF)(EG)  E(FG)
    THEOREM IS  E(FG)  (EF)(EG)
    THEOREM IS  EFG  (EF)G
    THEOREM IS  (EF)G  EFG
    THEOREM IS  (EF)G  (EF)G
    THEOREM IS  E(FG)  (EF)G

    THEOREM IS  E  EF
    THEOREM IS  F  EF
    THEOREM IS  EF  FE

    THEOREM IS  FG  (EF)(EG)
    THEOREM IS  EE  E
    THEOREM IS  E  EE
    THEOREM IS   ((EF)E)E

    THEOREM IS  E(FG)  (EF)(EG)
    THEOREM IS  EFEG  E(FG)
    THEOREM IS  EFG  (EF)(EG)
    THEOREM IS  (EF)(EG)  EFG

    THEOREM IS  (EG)(FG)  EFG
    THEOREM IS  EFG  (EG)(FG)

    THEOREM IS  E  E
    THEOREM IS  E E
    THEOREM IS  E  E

    THEOREM IS  E  E
    THEOREM IS  E E
    THEOREM IS  E  E

    THEOREM IS  E  ¬¬E
    THEOREM IS  ¬E  EF
    THEOREM IS  EF  ¬F¬E

    THEOREM IS  EF, ¬F  E
    THEOREM IS  EF, ¬E  F

    THEOREM IS  EF  ¬(¬E¬F)
    THEOREM IS  EF  ¬(¬E¬F)

    THEOREM IS  ¬(EF)  ¬E¬F
    THEOREM IS  ¬E¬F  ¬(EF)
    THEOREM IS  ¬E¬F  ¬(EF)
    THEOREM IS    ¬(E¬E)

    THEOREM IS  E¬E  F

    THEOREM IS  R(j), x.(R(x)S(x))  S(j)
    THEOREM IS  x.(R(x)S(x))  y.R(y)z.S(z)
    THEOREM IS  x.(R(x)S(x)), y.(S(y)T(y))  z.(R(z)T(z))
    THEOREM IS  x.R(x)y.S(y)  z.(R(z)S(z))
    THEOREM IS  x.(R(x)S(x))  y.R(y)z.S(z)
    THEOREM IS  x.(R(x)S(x)), y.R(y)  z.S(z)
```

```
    THEOREM IS  x.(R(x)S(x))  y.R(y)z.S(z)
    THEOREM IS  x.R(x)y.S(y)  z.(R(z)S(z))
    THEOREM IS  x.(R(x)S(x))  y.R(y)z.S(z)

    THEOREM IS x.(x=0x=1), x.R(x), ¬R(1)  R(0)

    THEOREM IS ( x . T(x))  ( x . (P(x)T(x))),
               ( x . Ph(x))  ( x . (T(x)Ph(x)))

               (x . (T(x)  Ph(x)))  ( x . (P(x)Ph(x)))


    BUTTON Apply    IS apply ResolveOrTheorem COMMAND

END
```

```
/* $Id: IFP_derived.j 642 2014-06-03 18:10:20Z sufrin $
   Opportunity to do proofs of derived rules, that we have introduced in the
   regular rules menu as honest-to-god rules.
*/

CONJECTUREPANEL "Derived Rules"
    DERIVED RULE "(RAA)"     IS FROM ¬A   INFER A
    DERIVED RULE "()"      IS FROM A , B  C INFER AB  C
    DERIVED RULE "(¬¬)"     IS FROM B  C INFER ¬¬BC
    DERIVED RULE "()"      IS FROM P(T)   C INFER x . P(x)  C
    DERIVED RULE "(= sym)"    IS FROM T1=T2 INFER T2=T1
    DERIVED RULE "(= trans)"  IS FROM T1=T2 AND T2=T3 INFER T1=T3
    BUTTON Apply IS apply ForbidDerived COMMAND
END


TACTIC ForbidDerived(rulename)
      (Fail ("(All you can do with %s here is to start proving it)\
              \\nThe derived rules are already available (as if proven) on the Rules menu.", rulename))


/** Proofs left unloaded

CONJECTUREPANEL "Derived Rules"
PROOF "()"
FROM A,
     B
      C
INFER AB
      C
FORMULAE
0 AB,
1 A,
2 B,
3 C
IS
SEQ (cut[B,C\1,3]) (" elim(L)"[A,B\1,2]) (hyp[A\0]) (cut[B,C\2,3]) (" elim(R)"[A,B\1,2]) (hyp[A\0]) (GIVEN 0)
END
CONJECTUREPANEL "Derived Rules"
PROOF "(¬¬)"
FROM B
      C
INFER ¬¬B
      C
FORMULAE
0 ¬¬B,
1 B,
2 C
IS
SEQ (cut[B,C\1,2]) ("¬¬ elim"[B\1]) (hyp[A\0]) (GIVEN 0)
END
CONJECTUREPANEL "Derived Rules"
PROOF "()"
FROM P(T)
      C
INFER x.P(x)
      C
FORMULAE
0 x.P(x),
1 x,
2 T,
3 P(x),
4 P(T),
5 C
IS
SEQ (cut[B,C\4,5]) (" elim"[x,T,P\1,2,3]) (hyp[A\0]) (GIVEN 0)
END
CONJECTUREPANEL "Derived Rules"
PROOF "(RAA)"
FROM ¬A

INFER A
FORMULAE
0 ¬A,
1 A
IS
```

```
SEQ ("¬¬ elim"[B\1]) ("¬ intro"[A\0]) (GIVEN 0)
END
**/
```

```
/*
        Mouse-clicks on formulae
*/

HYPHIT  P    P IS hyp
HYPHIT  PQ  R IS ALT  (SEQ " elim(L)" (WITHHYPSEL hyp))
                      (SEQ " elim(R)" (WITHHYPSEL hyp))
                      (ForwardAndElimBoth)

HYPHIT  PQ  P IS (SEQ " elim(L)" (WITHHYPSEL hyp))
HYPHIT  PQ  Q IS (SEQ " elim(R)" (WITHHYPSEL hyp))
HYPHIT  PQ  R IS (SEQ ForwardAndElimL ForwardAndElimR)
HYPHIT  PQ  P IS ForwardIffElimL
HYPHIT  PQ  Q IS ForwardIffElimR
HYPHIT  PQ  R IS ForwardIffElimBoth

HYPHIT  PQ   R    IS ForwardImpElim
HYPHIT  P, PQ   R    IS ForwardImpElim

HYPHIT  PQ   R    IS ForwardUncut 0 " elim"
HYPHIT  ¬¬P   Q    IS NotNotElimHit
HYPHIT  ¬P        IS NotElim
HYPHIT  ¬P,R  Q    IS ForwardNotElim
HYPHIT  x.P  Q           IS ForwardAllElim
HYPHIT  fresh i, x.P  Q    IS ForwardAllElim
HYPHIT  x.P  Q    IS (WITHARGSEL " elim")
HYPHIT      Q    IS " elim"

/************************ = *********************************/

HYPHIT T1=T2  Q IS ForwardEqElim

TACTIC ForwardEqElim IS
       WHEN (LETHYPSUBSTSEL (_P[_x\_S])
                     (LETHYP (_T1=_T2)
                             (ALT (SEQ (UNIFY _S _T1) (cut (_P[_x\_T2])) (SEQ ("= elim") hyp) )
                                  (ForwardEqElimExplain (_T1=_T2))
                             )))
            (LETHYP (_T1=_T2) (ALT (LETCONC _P (SEQ ("= elim") hyp)) (ForwardEqElimExplain (_T1=_T2))))

TACTIC ForwardEqElimExplain(T) IS
       ALERT ("You selected the equation %s; but you either need to select a conclusion\n\
              \or to select an instance of the left-hand of the equation within a formula.", T)
              ("OK", STOP)
              ("Explain = elim", "Explain = elim")

MENU "= Rules" IS
   ENTRY "= intro"
   ENTRY "= elim"
   SEPARATOR
   ENTRY "= elim (select t1=t2) (text-select t1 in (t1) to cut with (t2))" IS
         ForwardEqElim
   SEPARATOR
   RULE "= sym"    IS FROM T1=T2 INFER T2=T1
   RULE "= trans"  IS FROM T1=T2 AND T2=T3 INFER T1=T3
   SEPARATOR
   TACTIC "= sym (forward)" IS
          WHEN (LETHYP  (_T1=_T2) (cut (_T2=_T1)) ("= sym"))
               (Fail "'= sym (forward)' requires you to select an equation")
END

/************************************************************/

CONCHIT QR     IS " intro"
CONCHIT QR     IS (ALT
                     (SEQ " intro(L)" hyp) (SEQ " intro(R)" hyp)
                     (Fail "Neither of the  intro rules is immediately applicable"))
CONCHIT QR     IS " intro"
CONCHIT ¬Q      IS "¬ intro"
CONCHIT x.Q    IS " intro"

CONCHIT fresh i  x.Q IS VarExIntro
CONCHIT H  x.Q IS (SEQ " intro" (ALT (WITHHYPSEL hyp)))
CONCHIT  x.Q   IS (SEQ " intro" )

MENU "Rules" IS
```

```
  ENTRY hyp
  ENTRY "¬ elim" IS NotElim
  ENTRY " elim"
  SEPARATOR
  SEPARATOR
  ENTRY " elim (forward)"    IS ForwardImpElim
  ENTRY "¬ elim (forward)"    IS ForwardNotElim
  ENTRY "¬¬ elim (forward)"   IS ForwardNotNotElim
  SEPARATOR
  ENTRY " elim(L) (forward)" IS ForwardAndElimL
  ENTRY " elim(R) (forward)" IS ForwardAndElimR
  ENTRY " intro (forward)"    IS ForwardAndIntro(" intro")
  ENTRY " intro(R) (forward)" IS ForwardOrIntro(" intro(R)")
  ENTRY " intro(L) (forward)" IS ForwardOrIntro(" intro(L)")
  ENTRY " elim(L) (forward)" IS ForwardIffElimL
  ENTRY " elim(R) (forward)" IS ForwardIffElimR
END


/* Entries test individual tactics
MENU "[Test Tactics]" IS
  ENTRY TwoHypsNotElim
  ENTRY ForwardNotElim
END
*/


TACTIC SelectFail(x) IS (ALERT x ("OK", STOP) ("Show the current selections", SHOWSELECTIONS))

/************************ Negation ************************/
TACTIC ForwardNotNotElim
 WHEN (LETHYP (¬¬_B) (SEQ (cut(_B)) ("¬¬ elim"(_B))))
      (Fail "¬¬ elim (forward) cannot be used here")

TACTIC NotNotElimHit
 WHEN (LETHYP (¬¬_B)
         (ALT (LETGOAL (_G)
                 /* We cannot simply say: (LETGOAL (_B) ...) here.
                    The LET forms introduce NEW pattern variables,
                    so we need to check that the goal is the hyp
                    with a (LETMATCH _B _G ...)
                 */
                 /* (ALERT ("Goal is %s |- %s", _B, _G)) */
                 /* Goal same as hyp: just apply the rule */
                     (LETMATCH _G _B ("¬¬ elim"[B\_B])))
                 /* Goal distinct from hyp: cut with _B, then apply the rule */
                     (SEQ (cut[B\_B]) ("¬¬ elim"[B\_B]) (WITHHYPSEL (hyp)))))
      (Fail "¬¬ elim cannot be used forwards or backwards here")

TACTIC NotElim IS
 WHEN
  (LETGOAL
   (WHEN
    (LETHYP2 (B) (¬ _B)  (TwoHypsNotElim ()))
    (LETHYP (¬ _B)    (SEQ ("¬ elim" _B) (WITHHYPSEL (hyp (¬ _B)))))
    (Fail "¬ elim needs a goal of  and a selected hypothesis of the form ¬P")
    ))
  (Fail "Your goal should be : try -elim or RAA first")

TACTIC TwoHypsNotElim(G) IS
 WHEN
  (LETGOAL
   (WHEN
    (LETHYP2 _B (¬ _B)
        (ALT (SEQ  ("¬ elim" _B) (WITHHYPSEL (hyp (¬ _B))) (WITHHYPSEL (hyp _B)))
             (ALERT ("¬ elim is not applicable with selected hyps %s and %s and goal %s.\n\
                    \(this may have happened because you refused to select a rule-match)", _B, ¬_B, G)
                    ("OK", STOP)
                    ("Explain 'rule-match'", "The rule ... matches in N different ways ...")
                    ))
        )
     (TwoHypsNotElimError(G))))
  (Fail "Your goal should be : try -elim or RAA first")


TACTIC TwoHypsNotElimError(G) IS
```

```
  WHEN (LETHYP2 _A (¬ _B)
             (ALT (SEQ (UNIFY _A _B)
                       (Fail ("You selected goal %s and hypotheses %s and %s\n\
                              \but ¬-elim cannot be applied", G, _A, (¬ _B))))
                  (ALERT ("You selected goal %s and hypotheses %s and %s\n\
                              \but ¬-elim cannot be applied, because the hypotheses cannot be made to match\n\
                              \(if it is not obvious why, then perhaps a proviso was violated)?", G, _A, (¬
_B))
                              ("OK", STOP)
                              ("Explain provisos", "What are provisos?"))))
       (SelectFail "¬ elim needs a goal of  and selected hypotheses of the form  and ¬")

TACTIC ForwardNotElim IS
 WHEN
  (LETGOAL  (TwoHypsNotElim()))
  (LETHYP2 _B (¬ _A) (LETGOAL _G (SEQ (cut ) (TwoHypsNotElim(_G)))))
  (SelectFail "Using ¬ elim forwards this way needs hypotheses of the form  and ¬ to be selected")


/************************* Disjunction ************************/
TACTIC ForwardOrIntro(introtac) IS
 WHEN
   (LETHYP _A
     (WHEN (LETARGSEL (_B)
            (ALT (SEQ (cut (_A_B)) (introtac) (WITHHYPSEL (hyp)))
             (SelectFail ("You text-selected %s and formula selected the hypothesis %s.\
                    \\nBut %s (forward) failed: because %s couldn't be used immediately to prove conclusion
%s",
                    _A, _B,  introtac, introtac, (_A  _B)))
            ))
           (SelectFail ("You formula-selected the hypothesis %s.\nYou also need to text-select  in order to use
%s forward to establish %s",
                  _A, introtac, _A))))
   (LETARGSEL _A (Fail ("You text-selected %s.\nYou also need to formula-select a hypothesis  in order to use
%s forward to establish %s",
                     _A, introtac, _A)))
   (SelectFail ("To use %s forwards to establish  you need to formula-select a hypothesis  and text-select ",
introtac))

/************************* Conjunction ************************/
TACTIC ForwardAndIntro(introtac) IS
 WHEN
   (LETHYP _A
     (WHEN (LETARGSEL (_B)
            (ALT (SEQ (cut (_A_B)) (introtac) (WITHHYPSEL (hyp)))
             (SelectFail ("You text-selected %s and formula selected the hypothesis %s.\
                    \\nBut %s (forward) failed: because %s couldn't be used immediately to prove conclusion
%s",
                    _A, _B,  introtac, introtac, (_A  _B)))
            ))
           (SelectFail ("You formula-selected the hypothesis %s.\nYou also need to text-select  in order to use
%s forward to establish %s",
                  _A, introtac, _A))))
   (LETARGSEL _A (Fail ("You text-selected %s.\nYou also need to formula-select a hypothesis  in order to use
%s forward to establish %s",
                     _A, introtac, _A)))
   (SelectFail ("To use %s forwards to establish  you need to formula-select a hypothesis  and text-select ",
introtac))

TACTIC ForwardAndElimL IS
 WHEN
  (LETHYP (_A  _B) (SEQ (cut (_A)) " elim(L)") (WITHHYPSEL (hyp(_A  _B))))
  (SelectFail "You need to select a conjunctive hypothesis for forward -elim(L) to work")

TACTIC ForwardAndElimR IS
 WHEN
  (LETHYP (_A  _B) (SEQ (cut (_B)) " elim(R)")(WITHHYPSEL (hyp(_A  _B))))
  (SelectFail "You need to select a conjunctive hypothesis for forward -elim(R) to work")

TACTIC ForwardAndElimBoth IS
  WHEN
   (LETHYP (_A  B) (SEQ ForwardAndElimL ForwardAndElimR))
   (SelectFail "You need to select a conjunctive hypothesis for forward -elim to work")

/************************************ iff ****************************************/
TACTIC ForwardIffElimL IS
```

```
 WHEN
  (LETHYP (_A  _B) (SEQ (CUTIN " elim(L)" (WITHHYPSEL (hyp)))))
  (SelectFail "You need to select an  hypothesis for forward -elim(L) to work")

TACTIC ForwardIffElimR IS
 WHEN
  (LETHYP (_A  _B) (SEQ (CUTIN " elim(R)" (WITHHYPSEL (hyp)))))
  (SelectFail "You need to select an  hypothesis for forward -elim(R) to work")

TACTIC ForwardIffElimBoth IS
  WHEN
   (LETHYP (_A  B) (SEQ ForwardIffElimL ForwardIffElimR))
   (SelectFail "You need to select an  hypothesis for forward -elim to work")

/************************ Implication ************************/
TACTIC ForwardImpElim IS
  WHEN   (LETHYP2 _A (_A  _B)
             (ALT
               (SEQ (cut _B)
                    (WITHHYPSEL (" elim"[A,B\_A,_B]))
                    (Fail ("You selected %s and %s%s, but  elim (forward) failed: Tell Bernard", _A, _A, _B))))
         (LETHYP (_A_B)  (SEQ (cut _B) (WITHHYPSEL (" elim"[A,B\_A,_B]))))
       (SelectFail "You need to select a hypothesis of the form  for  elim (forward) (you can also select the
)")

/************************  ***************************************/

TACTIC VarExIntro IS
  WHEN (LETHYP (fresh _i) (SEQ (" intro"[T\_i]) (ALT hyp (SEQ))))
       (Fail "VarExIntro invoked without a fresh declaration selected.")

TACTIC ForwardExIntro IS
  WHEN (LETHYPSUBSTSEL (_P[_x\_S])
                       /* (ALERT ("%s %s %s", _P, _x, _S)) */
                       (CUTIN (" intro"[x,P\_x,_P]) (hyp (_P[_x\_S]))))
       (Fail ("You have invoked  intro (forward).\n\
               \The  intro rule is: FROM () INFER  x(x).\n\
               \Before doing this you should have text-selected a term  from within a hypothesis formula () \
               \so as to make clear what you meant."))


/************************  ***************************************/

CONSTANT

TACTIC ForwardAllElim IS
    WHEN (LETARGSEL (_T)
         (ALT
          (LETHYP ( _x . _P)
                  (CUTIN " elim" [T\_T] (WITHHYPSEL (hyp))))
                   /* Didn't seem to be able to do this with cut/ elim/hyp
                      Maybe I didn't specify them enough and the following might have worked:
                        (SEQ (cut[B\(_P[_x\_T])] (" elim" [T\_T]) (WITHHYPSEL (hyp))))
                      but, as they say, IIABDFI!
                   */
          (ForwardAllElimSelectFail)))
       (LETHYP2 (fresh _T) ( _x . _P) (CUTIN " elim" [T\_T] (WITHHYPSEL (hyp))))
       (LETHYP ( _x . _P)
        (ALERT
             ("You have invoked  elim (forward) by double-clicking or from the Quantifier Rules menu, \
              \with selected hypothesis\n   %s\n\n\
              \For this to work, you need to text-select a term,  (say), to substitute for %s in\n   %s\n\n\
              \The result will be\n   %s\n\n\
              \If you select an assumption of the form (fresh ) then Jape will behave as if you had text-
selected \n\n\
              \If you really want Jape to invent a term proof variable for you, then you can \
              \invoke ' elim (forward, inventing a proof variable)' below, or from a menu.", ( _x . _P), _x,
_P, (_P[_x\]))
              ("OK", STOP)
              (" elim (forward, inventing a proof variable)", ForwardAllElimInvent)))
       (ForwardAllElimSelectFail)


TACTIC ForwardAllElimSelectFail IS
       (SelectFail "If you want to use  elim (forward) at this point, then \
              \you need to select a conclusion, \n\
```

```
             \and a formula of the form  x . (x), \n\
             \and text select a term T, or a 'fresh' declaration.")


TACTIC ForwardAllElimInvent IS
   WHEN (LETHYP ( _x . _P) (CUTIN " elim"  (WITHHYPSEL (hyp))))
        (SelectFail "If you want to use  elim and have it invent a term at this point, then \
             \you need to select a formula of the form  x . (x).")

MENU "Quantifier Rules" IS

SEPARATOR

   ENTRY " intro (forward) (needs a text-selected term in a hypothesis; invents a variable)" IS ForwardExIntro

SEPARATOR

   ENTRY " elim (forward)" IS ForwardAllElim
   ENTRY " elim (forward, inventing a term proof-variable)" IS ForwardAllElimInvent
   ENTRY " (derived)" IS ""
END

/************************ Helpistry *********************************/

TACTIC EXPLAINTHENSTOP(x) IS
    SEQ (ALERT x) STOP

 PATCHALERT "double-click is not defined"
            "Double-clicking has no effect with the current selection(s).\n\
            \You can find out what (IFP Jape) thinks the current selection is from the Help menu.\n\
            \You will also find explanations there of how to make selections."
            ("OK")

 TACTIC SHOWSELECTIONSWITHTEXT(caption, value) IS
             WHEN (LETCONC _G (ALT (LETHYPS _H
                                    (ALERT ("Selected hypothes(is/es)\n   %s\n\n\
                                            \Selected conclusion\n   %s\n\n%s %s", _H, _G, caption, value)
                                           ("OK", STOP)))
                                    (ALERT ("No Selected hypothesis\n\n\
                                            \Selected conclusion\n   %s\n\n%s %s", _G, caption, value)
                                           ("OK", STOP))
                                ))
                   (LETHYPS _H (ALERT ("Selected hypothes(is/es) \n   %s\n\n\
                                        \(no selected conclusion)\n\n%s %s", _H, caption, value)
                                       ("OK", STOP))
                                )
                   (LETCONC _G (ALERT ("Selected conclusion\n   %s\n\n\
                                        \(no selected hypothesis)\n\n%s %s", _G, caption, value)
                                       ("OK", STOP))
                                )
                   (ALERT ("No selected hypothesis or conclusion.\n\n%s %s", caption, value)
                          ("OK", STOP))

 TACTIC SHOWSELECTIONS IS
        WHEN     (LETMULTIARG _T (SHOWSELECTIONSWITHTEXT "Selected texts:" _T))
                 (LETARGSEL _T   (SHOWSELECTIONSWITHTEXT "Selected text:" _T))
                 (SHOWSELECTIONSWITHTEXT "No selected" text)

MENU Help IS
 TACTIC "Show the current formula(e) selection(s)" IS SHOWSELECTIONS
 TACTIC "Explain help" IS
        EXPLAINTHENSTOP "General help for Jape can be found on the Help menu.\n\
                        \The IFP logic is explained in the notes for \
                        \the Oxford University course \"Introduction to Formal Proof\"."
 SEPARATOR
 SEPARATOR
 TACTIC "Explain Text selection (also called subformula selection)"  IS (SHOWHOWTO TextSelect)
 TACTIC "Explain Formula selection" IS (SHOWHOWTO FormulaSelect)
 SEPARATOR
 SEPARATOR
 TACTIC "Explain = elim" IS
   ALERT "To make a forward inference of the form (2) from an hypothesis of the form 1=2 \
          \(when the goal conclusion is already of the form (2)) just double-click \
          \on the hypothesis 1=2.\n\nWhen the goal conclusion is not already of the form (2) \
          \then text-select the 1 in an hypothesis of the form (1) first."
          ("OK", STOP)
```

```
 SEPARATOR
 TACTIC "Hypotheses and Conclusions" IS
   ALERT
        "When you select a hypothesis, you get a downward-pointing selection (a box round the \
        \formula, open at the bottom). You work forward from a hypothesis selection.\n\n\
        \Unproved conclusion formulae are written with three dots above them. \
        \When you select an unproved conclusion, you get \
        \an upward-pointing selection (a box round the formula, open at the top). You can work \
        \backwards from a conclusion selection, or it can be a target for a forward step.\n\n\
        \Some formulae introduced by buttons labelled (forward) can be used as a hypothesis or as an \
        \unproved conclusion. The selection box for these has a dotted horizontal line. \
        \Click in the bottom half of the formula to make an \
        \hypothesis selection, in the top half for a conclusion selection.\n\n\
        \Any formula can be used as a hypothesis if there are relevant unproved conclusions below it \
        \in the proof."
        ("OK", STOP)
        ("Explain hypotheses in scope", "Hypotheses in scope")

 TACTIC "Meaning of ..." IS
   EXPLAINTHENSTOP
        "An ... (ellipsis) in a proof-in-progress marks a place (an open goal) where there is still work to be
done. \
        \The formula just below the ... is an unproved conclusion (notice that it doesn't have a reason beside
it). \
        \Your job is to show that each unproved conclusion follows \
        \from the hypotheses in scope for it.\n\n\
        \When making a proof step you nearly always need have an open goal selected: \
        \IFP Jape is set up to do this automatically after each move; but you can \
        \change the goal by selecting another; and you can change Jape's policy from the Proof Style menu.\n\n\
        \When there are no unproved conclusions left, the proof is finished."

 TACTIC "Hypotheses in scope" IS
   EXPLAINTHENSTOP
        "The hypotheses in scope at a formula are the ones that are not greyed-out when the conclusion \
        \is selected.\n\n\
        \You can also find out what they are by changing the view to Tree display \
        \(though I don't recommend doing this in the middle of a large proof.)"

 TACTIC "The Rules menu and the [Rules] menu"
    ALERT
        "The Rules menu contains frequently-used, and some derived rules. \n\
        \It also has buttons that invoke proof-tactics that appear to have a 'forward' effect: \
        \these operate by using 'cut' to establish a new formula in the context.\n\n\
        \The [Rules] menu has /all/ the rules on it: Jape invokes them without trying to be clever."
        ("OK", STOP)
        ("Explain the cut rule", "The cut rule")

 TACTIC "The cut rule" IS
     EXPLAINTHENSTOP
        "The rule cut() transforms a proof goal of the form    into the two proof sub-goals: \n\n\
        \         \n\
        \       ,   \n\n\
        \In the box form display IFP Jape normally shows only the second subgoal, \
        \thereby giving the appearance that  has been added to the context.\n\n\
        \Tactics labelled (forward) on the Rules menu generally operate by inserting a cut, \
        \then trying to use the rule named on their button to close the    goal. \
        \If that rule fails, then the tactic itself fails, and IFP Jape will try to explain why."

 TACTIC "The rule ... matches in N different ways ..." IS
     EXPLAINTHENSTOP
        "Occasionally a 'forward' tactic will put up a dialogue box that asks you to \
        \choose between different ways of applying a rule. Sometimes it shows \
        \ways that are not actually different[*], but in any case any of them can be safely selected.\n\n\
        \[*]What is happening here is that Jape is mis-reporting the fact that it has found two distinct lines
with the same \
        \hypothesis on."

 TACTIC "What are provisos?" IS
    EXPLAINTHENSTOP
        "A proviso is Jape's representation of one of the side-conditions of a quantifier rule.\n\
        \Provisos usually constrain the way that proof-variables invented by Jape may later be resolved /
bound.\n\
        \The provisos, if any, on proof-variables present in the current context are always visible on a panel
\
        \below the proof display, that starts with the line 'Provided:'.\n\
        \It is possible that the invocation of a forward tactic, or of a quantifier rule with an inappropriate
```

```
\
        \text-selection, would violate a proviso that you cannot yet see; in that case you will usually receive
an explanation of what went wrong."

END
```

```
CONJECTUREPANEL "Introductory Conjectures"
    THEOREM IS  ¬¬E  E

    THEOREM IS E, ¬E  F
    THEOREM IS E, ¬E, G  F
    THEOREM IS EFGH  GFEH

    THEOREM IS  E(FG)  (EF)G
    THEOREM IS  (EF)G  E(FG)

    THEOREM IS  ¬(EF)  ¬E¬F
    THEOREM IS   E¬E

    THEOREM IS  ¬F¬E  EF

    THEOREM IS  ¬E  ¬F  ¬(EF)
    THEOREM IS  ¬E  ¬F  ¬(EF)
    THEOREM IS  E  F  ¬(¬E¬F)
    THEOREM IS  E  F  ¬(¬E¬F)
    THEOREM IS  ¬(¬E¬F)  EF
    THEOREM IS  ¬(¬E¬F)  EF
    THEOREM IS  ¬(EF)  ¬E¬F
    THEOREM IS  ¬(EF)  ¬E¬F
    THEOREM IS  (EF)(FE)

    THEOREM IS  x.R(x)  y.R(y)
    THEOREM IS   x . y . R(x,y)   y . x . R(x,y)

    THEOREM IS  ¬x.R(x)  y.¬R(y)
    THEOREM IS  ¬x.R(x)  y.¬R(y)
    THEOREM IS  ¬x.¬R(x)  y.R(y)
    THEOREM IS  ¬x.¬R(x)  y.R(y)
    THEOREM IS  x.R(x)  ¬y.¬R(y)
    THEOREM IS  x.R(x)  ¬y.¬R(y)
    THEOREM IS  x.¬R(x)  ¬y.R(y)
    THEOREM IS  x.¬R(x)  ¬y.R(y)

    THEOREM IS  R(k)  x.(R(x)R(j)R(k))


    BUTTON Apply    IS apply ResolveOrTheorem COMMAND

END
```

```
/*
        $Id: IFP_rules.j 671 2015-04-16 10:29:30Z sufrin $
*/

/*
        Rules
*/


   RULE hyp(A)   IS INFER A  A
   RULE " elim" IS FROM  INFER B
   RULE " intro" IS INFER
   RULE " elim"  IS FROM C INFER   C


   RULE " intro"          IS FROM A  B INFER AB
   RULE " intro"          IS FROM A AND B INFER A  B
   RULE " intro(L)"(B)    IS FROM A INFER A  B
   RULE " intro(R)"(B)    IS FROM A INFER B  A
   RULE " intro"          IS FROM AB AND BA INFER A  B


   RULE " elim"    IS FROM AB AND A INFER B
   RULE " elim(L)" IS FROM A  B INFER A
   RULE " elim(R)" IS FROM A  B INFER B
   RULE " elim"    IS FROM A  B AND A  C AND B  C INFER C
   RULE " elim(L)"   IS FROM A  B INFER AB
   RULE " elim(R)"   IS FROM A  B INFER BA


   RULE "¬ intro"     IS FROM A   INFER ¬A
   RULE "¬ elim"(B)   IS FROM ¬B AND B INFER
   RULE "¬¬ elim"     IS FROM ¬¬B INFER B

   RULE cut(B)  IS FROM B AND B  C INFER C
   RULE thin(A) IS FROM C INFER A  C

   RULE " intro"(OBJECT i)
       WHERE FRESH i IS
       FROM fresh i  P(i)
       INFER x .P(x)

   RULE " intro"(T) IS FROM P(T) INFER x.P(x)


   RULE " elim"(T)  IS FROM x. P(x) INFER P(T)

   RULE " elim"(OBJECT i)
         WHERE FRESH i  /* AND i NOTIN x.P(x) */
         IS    FROM fresh i, P(i)  C INFER x.P(x)  C

   RULE "= intro" IS T=T

   RULE "= elim"(T1,T2,ABSTRACTION P) IS FROM T1=T2 AND P(T1) INFER P(T2)


/*
        Declare the names of the standard structural rules
*/
IDENTITY        hyp
CUT             cut
WEAKEN          thin
AUTOMATCH       hyp

/*      and the ''formula'' used in the presentation of FRESH */
SCOPEHYP        x IN fresh x

/* ---------------- */
```

```
/*
        $Id: IFP_style_menus.j 631 2014-05-30 13:22:08Z sufrin $
*/

UMENU "Proof Style"
    RADIOBUTTON applyconjectures
        "Apply conjectures as theorems" IS all
    AND "Apply only proven theorems"    IS none
    INITIALLY none
    END

    CHECKBOX tryresolution "Cut, if necessary, when applying theorems"

    CHECKBOX autoselect "Select goal automatically"
END

INITIALISE tryresolution     false
INITIALISE showallproofsteps false
INITIALISE autoselect        true
INITIALISE hidecut           true
INITIALISE hidehyp           true


MENU "View"

    RADIOBUTTON displaystyle IS
        "Box display"   IS box
    AND "Tree display"  IS tree
    INITIALLY box
    END

    CHECKBOX showallprovisos      "Show all provisos"
    CHECKBOX showallproofsteps    "Show compressed steps"
    CHECKBOX hideuselesscuts      "Hide unnecessary forward steps"

    SEPARATOR

    CHECKBOX hidecut              "Hide cuts in box display"
    CHECKBOX hidehyp              "Hide duplicated hyp lines in box display"

    SEPARATOR

    CHECKBOX hidetransitivity     "Hide transitive steps"
    CHECKBOX hidereflexivity      "Hide reflexive steps"

    SEPARATOR

    CHECKBOX foldsequents         "Fold long sequents in tree display"
    CHECKBOX multiassumptionlines "Multiple assumptions per line in box display"
    CHECKBOX foldformulae         "Fold long formulae in box display"

    SEPARATOR
    SEPARATOR

    CHECKBOX tactictracing "Trace tactic applications"

END
```

```
/*
        $Id: IFP_syntax.j 647 2014-09-30 16:51:20Z sufrin $
*/

CLASS VARIABLE  x y z
                i j k
CLASS FORMULA   A B C D
                E F G H
                P Q
                R S T
CLASS BAG FORMULA

CONSTANT

PREFIX 10 fresh term

INFIX   100R
INFIX   100R
INFIX   120L
INFIX   140L

PREFIX  200 ¬

LEFTFIX 200  .
LEFTFIX 200  .

INFIX 500T =

JUXTFIX 3000
SUBSTFIX    4000

BIND x SCOPE A IN x . A
BIND x SCOPE B IN x . B

SEQUENT IS BAG  FORMULA

INITIALISE autoAdditiveLeft     true /* allow rules to be stated without an explicit left context */
INITIALISE interpretpredicates  true /* allow predicate syntax ... */
```

```
/*
        Tactics that support ''do what I probably mean'' interaction
        for the IFP presentation of logic.
*/

TACTIC Fail (x) IS SEQ (ALERT x) STOP

TACTIC ForwardCut (n,rule)
    CUTIN (ForwardUncut n rule)

TACTIC ForwardUncut (n,Rule)
    WHEN
        (LETHYP _Ah
            (LETGOALPATH G (WITHARGSEL Rule) (GOALPATH (SUBGOAL G n)) (WITHHYPSEL hyp) (GOALPATH G) NEXTGOAL))
        /* If LETHYP fails at this point, we had better have a singleton LHS.*/
        (LETLHS _Ah
            (LETGOALPATH G
                (WITHARGSEL Rule)
                (GOALPATH (SUBGOAL G n))
                (LETGOAL _Ag
                    (ALT (UNIFY _Ag _Ah)
                         (Fail ("Error in IFP Jape (can't unify lhs %t with rhs %t in ForwardUncut).",
                                _Ah, _Ag)))
                    (ANY hyp)
                )
                (GOALPATH G)
                NEXTGOAL))
        (Fail "Error in IFP Jape (ForwardUncut falls through).")

TACTIC TheoremForward (thm) IS CUTIN (ALT thm (RESOLVE thm))

TACTIC TheoremForwardOrBackward(thm) IS
    WHEN
        (LETHYP _A
            (ALT    (TheoremForward (WITHHYPSEL (WITHARGSEL thm)))))
        (LETHYPS _As
            (Fail ("At present IFP Jape can't deal with multiple hypothesis selections when applying theorems.
Sorry.\
                   \\nCancel all but one of them and try again.")))
        (LETGOAL _A
            (ALT (WITHARGSEL thm)
                (RESOLVE (WITHARGSEL thm))
                (TheoremForward (WITHARGSEL thm))
                (Fail   "Theorem application failed -- tell Bernard Sufrin")))
        (LETOPENSUBGOAL G _A
            (Fail ("Error in IFP Jape (open subgoal in TheoremForwardOrBackward). Tell Bernard Sufrin.")))
        (LETOPENSUBGOALS _As
            (ALERT  ("There is more than one unproved conclusion in the proof. Please select one – \
                        \or select a hypothesis – to show \
                        \Jape where to apply the theorem.")
                    ("OK", STOP)
                    ))
        (ALERT  "The proof is finished -- there are no unproved conclusions left."
                ("OK", STOP)
                )
```

```
CONJECTUREPANEL Conjecture2c

THEOREM IS
( x . T(x))  ( x . (P(x)  T(x))),
( x . Ph(x))  ( x . (T(x)  Ph(x)))
( x . (T(x)  Ph(x)))  ( x . (P(x)  Ph(x)))

THEOREM IS
(x.R(x))(y.S(y))  z.(R(z)S(z))

END
```

```
CONJECTUREPANEL "Introductory Conjectures"
PROOF "¬x.¬R(x)  y.R(y)"
INFER ¬x.¬R(x)
      y.R(y)
FORMULAE
0 ¬R(i),
1 i,
2 ¬R(x),
3 x,
4 ¬x.¬R(x),
5 x.¬R(x),
6 R(i),
7 R(y),
8 y,
9 fresh i
IS
SEQ (" intro"[i,P,x\1,7,8]) ("RAA (derived)"[A\6]) ("¬ elim"[B\5]) (hyp[A\4]) (" intro"[T,P,x\1,2,3])
(hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "¬x.R(x)  y.¬R(y)"
INFER ¬x.R(x)
      y.¬R(y)
FORMULAE
0 R(i),
1 i,
2 R(x),
3 x,
4 ¬x.R(x),
5 x.R(x),
6 ¬R(y),
7 y,
8 fresh i
IS
SEQ (" intro"[i,P,x\1,6,7]) ("¬ intro"[A\0]) ("¬ elim"[B\5]) (hyp[A\4]) (" intro"[T,P,x\1,2,3]) (hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "¬x.¬R(x)  y.R(y)"
INFER ¬x.¬R(x)
      y.R(y)
FORMULAE
0 R(y),
1 y,
2 x,
3 ¬x.¬R(x),
4 x.¬R(x),
5 y.R(y),
6 ¬y.R(y)
IS
SEQ ("RAA (derived)"[A\5]) ("¬ elim"[B\4]) (hyp[A\3]) ("¬x.R(x)  y.¬R(y)"[R,x,y\0,1,2])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "x.R(x)  ¬y.¬R(y)"
INFER x.R(x)
      ¬y.¬R(y)
FORMULAE
0 R(i),
1 ¬R(i),
2 x.R(x),
3 i,
4 R(x),
5 x,
6 ,
7 ¬R(y),
8 y,
9 y.¬R(y),
10 fresh i
IS
SEQ ("¬ intro"[A\9]) (" elim"[i,C,P,x\3,6,7,8]) (cut[B,C\0,6]) (" elim"[T,P,x\3,4,5]) (hyp[A\2]) ("¬
elim"[B\0]) (hyp[A\1]) (hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "x.R(x)  ¬y.¬R(y)"
INFER x.R(x)
      ¬y.¬R(y)
FORMULAE
```

```
0 R(i),
1 ¬R(i),
2 y.¬R(y),
3 i,
4 ¬R(y),
5 y,
6 ,
7 R(x),
8 x,
9 fresh i,
10 x.R(x)
IS
SEQ ("¬ intro"[A\2]) (" elim"[i,C,P,x\3,6,7,8]) (cut[B,C\1,6]) (" elim"[T,P,x\3,4,5]) (hyp[A\2]) ("¬
elim"[B\0]) (hyp[A\1]) (hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "¬x.R(x)  y.¬R(y)"
INFER ¬x.R(x)
      y.¬R(y)
FORMULAE
0 ¬R(i),
1 i,
2 ¬R(y),
3 y,
4 ¬y.¬R(y),
5 y.¬R(y),
6 R(i),
7 R(x),
8 x,
9 ¬x.R(x),
10 x.R(x),
11 fresh i
IS
SEQ ("RAA (derived)"[A\5]) ("¬ elim"[B\10]) (hyp[A\9]) (" intro"[i,P,x\1,7,8]) ("RAA (derived)"[A\6]) ("¬
elim"[B\5]) (hyp[A\4]) (" intro"[T,P,x\1,2,3]) (hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "x.¬R(x)  ¬y.R(y)"
INFER x.¬R(x)
      ¬y.R(y)
FORMULAE
0 R(i),
1 ¬R(i),
2 y.R(y),
3 i,
4 R(y),
5 y,
6 ,
7 ¬R(x),
8 x,
9 fresh i,
10 x.¬R(x)
IS
SEQ ("¬ intro"[A\2]) (" elim"[i,C,P,x\3,6,7,8]) (cut[B,C\0,6]) (" elim"[T,P,x\3,4,5]) (hyp[A\2]) ("¬
elim"[B\0]) (hyp[A\1]) (hyp[A\0])
END
CONJECTUREPANEL "Introductory Conjectures"
PROOF "x.¬R(x)  ¬y.R(y)"
INFER x.¬R(x)
      ¬y.R(y)
FORMULAE
0 R(i),
1 ¬R(i),
2 x.¬R(x),
3 i,
4 ¬R(x),
5 x,
6 ,
7 R(y),
8 y,
9 y.R(y),
10 fresh i
IS
SEQ ("¬ intro"[A\9]) (" elim"[i,C,P,x\3,6,7,8]) (cut[B,C\1,6]) (" elim"[T,P,x\3,4,5]) (hyp[A\2]) ("¬
elim"[B\0]) (hyp[A\1]) (hyp[A\0])
END
```

```
CONJECTUREPANEL "Introductory Conjectures"
CURRENTPROOF "¬(EF)  ¬E¬F"
INFER ¬(EF)
      ¬E¬F
FORMULAE
0 ¬(¬E¬F),
1 ¬E¬F,
2 E,
3 F,
4 ¬(EF),
5 EF,
6 ¬E,
7 ¬(EF)
IS
SEQ ("RAA (derived)"[A\1]) ("¬ elim"[B\5]) (hyp[A\4]) (" intro"[A,B\2,3]) ("RAA (derived)"[A\2]) ("¬
elim"[B\1]) (hyp[A\0]) (NEXTGOAL) (NEXTGOAL)
END
CONJECTUREPANEL "Introductory Conjectures"
CURRENTPROOF "¬(EF)  ¬E¬F"
INFER ¬(EF)
      ¬E¬F
FORMULAE
0 E,
1 F,
2 ¬(EF),
3 EF,
4 ¬E,
5 ¬F,
6 ¬(EF)
IS
SEQ (" intro"[A,B\4,5]) ("¬ intro"[A\0]) ("¬ elim"[B\3]) (hyp[A\2]) (" intro(L)"[B,A\1,0]) (hyp[A\0])
(NEXTGOAL)
END
CONJECTUREPANEL "Introductory Conjectures"
CURRENTPROOF "¬(¬E¬F)  EF"
INFER ¬(¬E¬F)
      EF
FORMULAE
0 ¬(¬E¬F),
1 ¬E¬F,
2 E,
3 F,
4 ¬(¬E¬F),
5 ¬E
IS
SEQ (" intro"[A,B\2,3]) ("RAA (derived)"[A\2]) ("¬ elim"[B\1]) (hyp[A\0]) (NEXTGOAL) (NEXTGOAL)
END
```