Implementation of Zermelo's work of 1908 in Lestrade: Part IV, central impredicative argument for total ordering of $\mathbf{M}$

M. Randall Holmes

March 21, 2020

1 Introduction

This document was originally titled as an essay on the proposition that mathematics is what can be done in Automath (as opposed to what can be done in ZFC, for example). Such an essay is still in in my mind, but this particular document has transformed itself into the large project of implementing Zermelo's two important set theory papers of 1908 in Lestrade, with the further purpose of exploring the actual capabilities of Zermelo's system of 1908 as a mathematical foundation, which we think are perhaps underrated.

This is a new version of this document in modules, designed to make it possible to work more efficiently without repeated execution of slow log files when they do not need to be revisited.

This particular part is monstrously large and slow and needs some fine tuning.

In this section, we prove that $\mathbf{M}$ is totally ordered by inclusion. This involves showing that the collection of elements of $\mathbf{M}$ which either include or are included in each other element of $\mathbf{M}$ is itself a Θ-chain and so actually equal to $\mathbf{M}$. The horrible thing about this is that the proof of the third component of this result contains a proof that a further refinement of this set definition also yields a Θ-chain, with its own four parts.

```
begin Lestrade execution
```

```
      >>> comment  load whatismath3

{function error}

general failure of functionsort line 3030

(paused, type something to continue) >

      {move 2}

      >>> clearcurrent

{move 2}

      >>> declare C obj


      C : obj


      {move 2}

      >>> declare D obj


      D : obj


      {move 2}

      >>> define cuts1 C : (C E Mbold) & Forall \
          [D => (D E Mbold) -> (D <<= C) V (C <<= \
             D)]


      cuts1 : [(C_1 : obj) =>
          ({def} (C_1 E Mbold) & Forall
```

```
      ([(D_3 : obj) =>
         ({def} (D_3 E Mbold) -> (D_3
         <<= C_1) V C_1 <<= D_3 : prop)]) : prop)]


   cuts1 : [(C_1 : obj) => (--- : prop)]


   {move 1}

   >>> save


   {move 2}

   >>> close


{move 1}

>>> declare C666 obj


C666 : obj


{move 1}

>>> define cuts2 Misset, thelawchooses, C666 \
    : cuts1 C666


cuts2 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
       .S_2 <<= .M_1), (inev_2 : that
       Exists ([(x_4 : obj) =>
```

```
         ({def} x_4 E .S_2 : prop)])) =>
      (--- : that .thelaw_1 (.S_2) E .S_2)]), (C666_1
   : obj) =>
   ({def} (C666_1 E Misset_1 Mbold2
   thelawchooses_1) & Forall ([(D_3
      : obj) =>
      ({def} (D_3 E Misset_1 Mbold2
      thelawchooses_1) -> (D_3 <<= C666_1) V C666_1
      <<= D_3 : prop)]) : prop)]


cuts2 : [(.M_1 : obj), (Misset_1
   : that Isset (.M_1)), (.thelaw_1
   : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
   : [(.S_2 : obj), (subsetev_2 : that
      .S_2 <<= .M_1), (inev_2 : that
      Exists ([(x_4 : obj) =>
         ({def} x_4 E .S_2 : prop)])) =>
      (--- : that .thelaw_1 (.S_2) E .S_2)]), (C666_1
   : obj) => (--- : prop)]


{move 0}

>>> open


   {move 2}

   >>> define cuts C : cuts2 Misset, thelawchooses, C


   cuts : [(C_1 : obj) =>
       ({def} cuts2 (Misset, thelawchooses, C_1) : prop)]


   cuts : [(C_1 : obj) => (--- : prop)]
```

```
   {move 1}

   >>> define Cuts1 : Set (Mbold, cuts)


   Cuts1 : Mbold Set cuts


   Cuts1 : obj


   {move 1}

   >>> close


{move 1}

>>> define Cuts3 Misset thelawchooses \
    : Cuts1


Cuts3 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
       .S_2 <<= .M_1), (inev_2 : that
       Exists ([(x_4 : obj) =>
          ({def} x_4 E .S_2 : prop)])) =>
       (--- : that .thelaw_1 (.S_2) E .S_2)]) =>
    ({def} Misset_1 Mbold2 thelawchooses_1
    Set [(C_2 : obj) =>
       ({def} cuts2 (Misset_1, thelawchooses_1, C_2) : prop)] : obj)]


Cuts3 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
```

```
      : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
      : [(.S_2 : obj), (subsetev_2 : that
         .S_2 <<= .M_1), (inev_2 : that
         Exists ([(x_4 : obj) =>
            ({def} x_4 E .S_2 : prop)])) =>
         (--- : that .thelaw_1 (.S_2) E .S_2)]) =>
      (--- : obj)]


   {move 0}

   >>> open


      {move 2}

      >>> define Cuts : Cuts3 Misset, thelawchooses


      Cuts : [
          ({def} Misset Cuts3 thelawchooses
          : obj)]


      Cuts : obj


      {move 1}
end Lestrade execution
```

This defines the predicate "is an element of **M** which either includes or is included in each element of `M`" and the correlated set. These things are packaged so as not to expand. The aim is to show that `Cuts` is a Θ-chain, from which we will be able to show the desired linear ordering result.

```
begin Lestrade execution
```

```
>>> define line1 : Simp1 Mboldtheta

line1 : Simp1 (Mboldtheta)

line1 : that M E Misset Mbold2 thelawchooses

{move 1}
>>> open

   {move 3}
   >>> declare F obj

   F : obj

   {move 3}
   >>> open

      {move 4}
      >>> declare finmbold that F E Mbold

      finmbold : that F E Mbold

      {move 4}
      >>> define line2 finmbold : Iff1 \
```

```
    (Mp finmbold, Ui F Simp1 Simp1 \
    Simp2 Mboldtheta, Ui F Scthm \
    M)


line2 : [(finmbold_1 : that
    F E Mbold) =>
    ({def} finmbold_1 Mp F Ui
    Simp1 (Simp1 (Simp2 (Mboldtheta))) Iff1
    F Ui Scthm (M) : that F <<=
    M)]


line2 : [(finmbold_1 : that
    F E Mbold) => (--- : that
    F <<= M)]


{move 3}

>>> define line3 finmbold : Add1 \
    (M <<= F, line2 finmbold)


line3 : [(finmbold_1 : that
    F E Mbold) =>
    ({def} (M <<= F) Add1 line2
    (finmbold_1) : that (F <<=
    M) V M <<= F)]


line3 : [(finmbold_1 : that
    F E Mbold) => (--- : that
    (F <<= M) V M <<= F)]


{move 3}
```

```
      >>> close


   {move 3}

   >>> define line4 F : Ded line3


   line4 : [(F_1 : obj) =>
       ({def} Ded ([(finmbold_2
          : that F_1 E Mbold) =>
          ({def} (M <<= F_1) Add1
          finmbold_2 Mp F_1 Ui Simp1
          (Simp1 (Simp2 (Mboldtheta))) Iff1
          F_1 Ui Scthm (M) : that
          (F_1 <<= M) V M <<= F_1)]) : that
       (F_1 E Mbold) -> (F_1 <<=
       M) V M <<= F_1)]


   line4 : [(F_1 : obj) => (---
       : that (F_1 E Mbold) -> (F_1
       <<= M) V M <<= F_1)]


   {move 2}

   >>> close


{move 2}

>>> define line5 : Ug line4


line5 : Ug ([(F_2 : obj) =>
    ({def} Ded ([(finmbold_3 : that
       F_2 E Mbold) =>
```

```
        ({def} (M <<= F_2) Add1 finmbold_3
        Mp F_2 Ui Simp1 (Simp1 (Simp2
        (Mboldtheta))) Iff1 F_2 Ui
        Scthm (M) : that (F_2 <<=
        M) V M <<= F_2)]) : that
    (F_2 E Mbold) -> (F_2 <<= M) V M <<=
    F_2)])


line5 : that Forall ([(x'_2 : obj) =>
    ({def} (x'_2 E Mbold) -> (x'_2
    <<= M) V M <<= x'_2 : prop)])


{move 1}

>>> define line6 : Fixform (cuts M, Conj \
    (line1, line5))


line6 : [
    ({def} cuts (M) Fixform line1
    Conj line5 : that cuts (M))]


line6 : that cuts (M)


{move 1}

>>> define line7 : Conj (Simp1 Mboldtheta, line6)


line7 : Simp1 (Mboldtheta) Conj line6


line7 : that (M E Misset Mbold2 thelawchooses) & cuts
 (M)
```

```
      {move 1}

      >>> define line8 : Ui M, Separation \
          (Mbold, cuts)


      line8 : M Ui Mbold Separation cuts


      line8 : that (M E Mbold Set cuts) ==
       (M E Mbold) & cuts (M)


      {move 1}

      >>> define Line9 : Fixform (M E Cuts, Iff2 \
          (line7, line8))


      Line9 : [
          ({def} (M E Cuts) Fixform line7
          Iff2 line8 : that M E Cuts)]


      Line9 : that M E Cuts


      {move 1}
end Lestrade execution
```

This is the first component of the proof that `Cuts` is a Θ-chain.

```
begin Lestrade execution

      >>> define line10 : Fixform (Cuts \
```

```
    <<= (Mbold), Sepsub (Mbold, cuts, Inhabited \
    (Simp1 (Mboldtheta))))


line10 : [
    ({def} (Cuts <<= Mbold) Fixform
    Sepsub (Mbold, cuts, Inhabited
    (Simp1 (Mboldtheta))) : that
    Cuts <<= Mbold)]


line10 : that Cuts <<= Mbold


{move 1}

>>> define line11 : Fixform ((Mbold) <<= \
    Sc M, Sepsub2 (Sc2 M, Refleq (Mbold)))


line11 : [
    ({def} (Mbold <<= Sc (M)) Fixform
    Sc2 (M) Sepsub2 Refleq (Mbold) : that
    Mbold <<= Sc (M))]


line11 : that Mbold <<= Sc (M)


{move 1}

>>> define Line12 : Transsub (line10, line11)


Line12 : [
    ({def} line10 Transsub line11 : that
    Cuts <<= Sc (M))]
```

```
      Line12 : that Cuts <<= Sc (M)


      {move 1}
end Lestrade execution
```

This is the second component of the proof that `Cuts` is a Θ-chain.

```
begin Lestrade execution

      >>> open


         {move 3}

         >>> declare B obj


         B : obj


         {move 3}

         >>> open


            {move 4}

            >>> declare bhyp that B E Cuts


            bhyp : that B E Cuts


            {move 4}
```

```
>>> define line13 bhyp : Iff1 \
    (bhyp, Ui B, Separation (Mbold, cuts))


line13 : [(bhyp_1 : that B E Cuts) =>
    ({def} bhyp_1 Iff1 B Ui Mbold
    Separation cuts : that (B E Mbold) & cuts
    (B))]


line13 : [(bhyp_1 : that B E Cuts) =>
    (--- : that (B E Mbold) & cuts
    (B))]


{move 3}

>>> define line14 bhyp : Simp1 \
    line13 bhyp


line14 : [(bhyp_1 : that B E Cuts) =>
    ({def} Simp1 (line13 (bhyp_1)) : that
    B E Mbold)]


line14 : [(bhyp_1 : that B E Cuts) =>
    (--- : that B E Mbold)]


{move 3}

>>> define linea14 bhyp : Setsinchains \
    Mboldtheta, line14 bhyp


linea14 : [(bhyp_1 : that B E Cuts) =>
    ({def} Mboldtheta Setsinchains
```

```
    line14 (bhyp_1) : that Isset
    (B))]


linea14 : [(bhyp_1 : that B E Cuts) =>
    (--- : that Isset (B))]


{move 3}

>>> define lineb14 bhyp : Iff1 \
    (Mp (line14 bhyp, Ui (B, Simp1 \
    Simp1 Simp2 Mboldtheta)), Ui \
    B, Scthm M)


lineb14 : [(bhyp_1 : that B E Cuts) =>
    ({def} line14 (bhyp_1) Mp
    B Ui Simp1 (Simp1 (Simp2
    (Mboldtheta))) Iff1 B Ui
    Scthm (M) : that B <<= M)]


lineb14 : [(bhyp_1 : that B E Cuts) =>
    (--- : that B <<= M)]


{move 3}

>>> define line15 bhyp : Simp2 \
    Simp2 line13 bhyp


line15 : [(bhyp_1 : that B E Cuts) =>
    ({def} Simp2 (Simp2 (line13
    (bhyp_1))) : that Forall
    ([(D_2 : obj) =>
       ({def} (D_2 E Misset
```

```
        Mbold2 thelawchooses) ->
        (D_2 <<= B) V B <<= D_2
        : prop)]))]


line15 : [(bhyp_1 : that B E Cuts) =>
    (--- : that Forall ([(D_2
        : obj) =>
        ({def} (D_2 E Misset
        Mbold2 thelawchooses) ->
        (D_2 <<= B) V B <<= D_2
        : prop)]))]


{move 3}

>>> open


   {move 5}

   >>> declare F obj


   F : obj


   {move 5}

   >>> declare fhyp that F E (Mbold)


   fhyp : that F E Mbold


   {move 5}

   >>> define line16 fhyp : Fixform \
```

```
    ((prime F) <<= F, Sepsub2 \
    (Setsinchains Mboldtheta, fhyp, Refleq \
    (prime F)))


line16 : [(.F_1 : obj), (fhyp_1
    : that .F_1 E Mbold) =>
    ({def} (prime (.F_1) <<=
    .F_1) Fixform Mboldtheta
    Setsinchains fhyp_1 Sepsub2
    Refleq (prime (.F_1)) : that
    prime (.F_1) <<= .F_1)]


line16 : [(.F_1 : obj), (fhyp_1
    : that .F_1 E Mbold) =>
    (--- : that prime (.F_1) <<=
    .F_1)]


{move 4}

>>> declare Y obj


Y : obj


{move 5}

>>> define cutsa2 Y : (Y <<= \
    prime B) V B <<= Y


cutsa2 : [(Y_1 : obj) =>
    ({def} (Y_1 <<= prime
    (B)) V B <<= Y_1 : prop)]
```

```
   cutsa2 : [(Y_1 : obj) =>
       (--- : prop)]


   {move 4}

   >>> save


   {move 5}

   >>> close


{move 4}

>>> declare Y10 obj


Y10 : obj


{move 4}

>>> define cutsb2 Y10 : cutsa2 \
    Y10


cutsb2 : [(Y10_1 : obj) =>
    ({def} (Y10_1 <<= prime
    (B)) V B <<= Y10_1 : prop)]


cutsb2 : [(Y10_1 : obj) =>
    (--- : prop)]
```

```
   {move 3}

   >>> save


   {move 4}

   >>> close


{move 3}

>>> declare Y11 obj


Y11 : obj


{move 3}

>>> define cutsc2 B Y11 : cutsb2 \
    Y11


cutsc2 : [(B_1 : obj), (Y11_1
    : obj) =>
    ({def} (Y11_1 <<= prime (B_1)) V B_1
    <<= Y11_1 : prop)]


cutsc2 : [(B_1 : obj), (Y11_1
    : obj) => (--- : prop)]


{move 2}

>>> save
```

```
   {move 3}

   >>> close


{move 2}

>>> declare Ba1 obj


Ba1 : obj


{move 2}

>>> declare Y12 obj


Y12 : obj


{move 2}

>>> define cutsd2 Ba1 Y12 : cutsc2 \
    Ba1 Y12


cutsd2 : [(Ba1_1 : obj), (Y12_1
    : obj) =>
    ({def} (Y12_1 <<= prime (Ba1_1)) V Ba1_1
    <<= Y12_1 : prop)]


cutsd2 : [(Ba1_1 : obj), (Y12_1
    : obj) => (--- : prop)]
```

```
   {move 1}

   >>> save


   {move 2}

   >>> close


{move 1}

>>> declare Ba2 obj


Ba2 : obj


{move 1}

>>> declare Y13 obj


Y13 : obj


{move 1}

>>> define cutse2 Misset, thelawchooses, Ba2 \
    Y13 : cutsd2 Ba2 Y13


cutse2 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
       .S_2 <<= .M_1), (inev_2 : that
       Exists ([(x_4 : obj) =>
```

```
        ({def} x_4 E .S_2 : prop)])) =>
      (--- : that .thelaw_1 (.S_2) E .S_2)]), (Ba2_1
    : obj), (Y13_1 : obj) =>
    ({def} (Y13_1 <<= prime2 (.thelaw_1, Ba2_1)) V Ba2_1
    <<= Y13_1 : prop)]


cutse2 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
      .S_2 <<= .M_1), (inev_2 : that
      Exists ([(x_4 : obj) =>
        ({def} x_4 E .S_2 : prop)])) =>
      (--- : that .thelaw_1 (.S_2) E .S_2)]), (Ba2_1
    : obj), (Y13_1 : obj) => (---
    : prop)]


{move 0}

>>> open


   {move 2}

   >>> define cutsf2 Ba1 Y12 : cutse2 \
       Misset, thelawchooses, Ba1 Y12


   cutsf2 : [(Ba1_1 : obj), (Y12_1
       : obj) =>
       ({def} cutse2 (Misset, thelawchooses, Ba1_1, Y12_1) : prop)]


   cutsf2 : [(Ba1_1 : obj), (Y12_1
       : obj) => (--- : prop)]
```

```
{move 1}

>>> open


   {move 3}

   >>> define cutsg2 B Y11 : cutsf2 \
       B Y11


   cutsg2 : [(B_1 : obj), (Y11_1
       : obj) =>
       ({def} B_1 cutsf2 Y11_1 : prop)]


   cutsg2 : [(B_1 : obj), (Y11_1
       : obj) => (--- : prop)]


   {move 2}

   >>> open


      {move 4}

      >>> define cutsh2 Y10 : cutsg2 \
          B Y10


      cutsh2 : [(Y10_1 : obj) =>
          ({def} B cutsg2 Y10_1 : prop)]


      cutsh2 : [(Y10_1 : obj) =>
          (--- : prop)]
```

```
            {move 3}

            >>> open


               {move 5}

               >>> define cutsi2 Y : cutsh2 \
                   Y


               cutsi2 : [(Y_1 : obj) =>
                   ({def} cutsh2 (Y_1) : prop)]


               cutsi2 : [(Y_1 : obj) =>
                   (--- : prop)]


               {move 4}

               >>> define Cuts2 : Set (Mbold, cutsi2)


               Cuts2 : Mbold Set cutsi2


               Cuts2 : obj


               {move 4}
end Lestrade execution
```

We are in the midst of the third component of the proof that `Cuts` is a Θ-chain. We have B which we assume is in `Cuts` and we want to show that `prime(B)` is in `Cuts`. We do this by showing that the set of all elements of

M which are either included in `prime(B)` or include B is a Θ-chain. Thus we have four components of this proof to generate before we get to generating the third component of the proof for `Cuts`.

This is about the time that I defined the `goal` command which is used to generate helpful comments about what we are trying to prove in the rest of the files. I should probably backtrack and insert goal statements earlier!

```
begin Lestrade execution

               >>> goal that thetachain Cuts2


               that thetachain (Cuts2)


               {move 5}

               >>> comment test thetachain

{function error}

general failure of functionsort line 3030

(paused, type something to continue) >

               {move 5}

               >>> goal that M E Cuts2


               that M E Cuts2


               {move 5}

               >>> define line17 : Ui M, Separation4 \
                   Refleq Cuts2
```

```
line17 : M Ui Separation4
 (Refleq (Cuts2))


line17 : that (M E Mbold
 Set cutsi2) == (M E Mbold) & cutsi2
 (M)


{move 4}

>>> define line18 : Conj (Simp1 \
    Mboldtheta, Add2 (M <<= \
    prime B, lineb14 bhyp))


line18 : Simp1 (Mboldtheta) Conj
 (M <<= prime (B)) Add2
 lineb14 (bhyp)


line18 : that (M E Misset
 Mbold2 thelawchooses) & (M <<=
 prime (B)) V B <<= M


{move 4}

>>> define line19 : Fixform \
    (M E Cuts2, Iff2 line18 \
    line17)


line19 : [
    ({def} (M E Cuts2) Fixform
    line18 Iff2 line17 : that
```

```
                    M E Cuts2)]


                line19 : that M E Cuts2


                {move 4}
end Lestrade execution
```

This is the first component of the proof that `Cuts2` is a Θ-chain.

```
begin Lestrade execution

                >>> goal that Cuts2 <<= Sc \
                    M


                that Cuts2 <<= Sc (M)


                {move 5}

                >>> declare D1 obj


                D1 : obj


                {move 5}

                >>> define line20 : Fixform \
                    (Cuts2 <<= Mbold, Sepsub2 \
                    (Separation3 Refleq Mbold, Refleq \
                    Cuts2))


                line20 : [
```

```
    ({def} (Cuts2 <<= Mbold) Fixform
    Separation3 (Refleq (Mbold)) Sepsub2
    Refleq (Cuts2) : that
    Cuts2 <<= Mbold)]


line20 : that Cuts2 <<= Mbold


{move 4}

>>> define line21 : Transsub \
    line20 Simp1 Simp2 Mboldtheta


line21 : [
    ({def} line20 Transsub
    Simp1 (Simp2 (Mboldtheta)) : that
    Cuts2 <<= Sc (M))]


line21 : that Cuts2 <<= Sc
 (M)


{move 4}
```
end Lestrade execution

This is the second component of the proof that `Cuts` is a Θ-chain.

begin Lestrade execution

```
>>> declare F1 obj


F1 : obj
```

```
{move 5}

>>> goal that Forall [D1 \
       => (D1 E Cuts2) -> (prime \
       D1) E Cuts2]


that Forall ([(D1 : obj) =>
    ({def} (D1 E Cuts2) ->
    prime (D1) E Cuts2 : prop)])


{move 5}

>>> open


   {move 6}

   >>> declare D2 obj


   D2 : obj


   {move 6}

   >>> open


      {move 7}

      >>> declare dhyp that \
          D2 E Cuts2


      dhyp : that D2 E Cuts2
```

```
{move 7}

>>> goal that (prime \
    D2) E Cuts2


that prime (D2) E Cuts2


{move 7}

>>> define line22 : Ui \
    prime D2, Separation4 \
    Refleq Cuts2


line22 : prime (D2) Ui
 Separation4 (Refleq
 (Cuts2))


line22 : that (prime
 (D2) E Mbold Set cutsi2) ==
 (prime (D2) E Mbold) & cutsi2
 (prime (D2))


{move 6}

>>> goal that ((prime \
    D2) E Mbold) & ((prime \
    D2) <<= prime B) V (B <<= \
    prime D2)


that (prime (D2) E Mbold) & (prime
```

```
 (D2) <<= prime (B)) V B <<=
 prime (D2)


{move 7}

>>> define line23 dhyp \
    : Iff1 dhyp, Ui D2 \
    Separation4 Refleq Cuts2


line23 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} dhyp_1 Iff1
    D2 Ui Separation4
    (Refleq (Cuts2)) : that
    (D2 E Mbold) & cutsi2
    (D2))]


line23 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that (D2
    E Mbold) & cutsi2
    (D2))]


{move 6}

>>> define line24 dhyp \
    : Simp1 line23 dhyp


line24 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} Simp1 (line23
    (dhyp_1)) : that
    D2 E Mbold)]
```

```
line24 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that D2 E Mbold)]


{move 6}

>>> define line25 dhyp \
    : Simp2 line23 dhyp


line25 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} Simp2 (line23
    (dhyp_1)) : that
    cutsi2 (D2))]


line25 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that cutsi2
    (D2))]


{move 6}

>>> define line26 : Iff1 \
    bhyp, Ui B, Separation4 \
    Refleq Cuts


line26 : [
    ({def} bhyp Iff1
    B Ui Separation4
    (Refleq (Cuts)) : that
    (B E Misset Mbold2
```

```
    thelawchooses) & cuts2
    (Misset, thelawchooses, B))]


line26 : that (B E Misset
 Mbold2 thelawchooses) & cuts2
 (Misset, thelawchooses, B)


{move 6}

>>> define line27 dhyp \
    : Mp line24 dhyp, Ui \
    D2, Simp2 Simp2 line26


line27 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} line24 (dhyp_1) Mp
    D2 Ui Simp2 (Simp2
    (line26)) : that
    (D2 <<= B) V B <<=
    D2)]


line27 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that (D2
    <<= B) V B <<= D2)]


{move 6}

>>> define line28 dhyp \
    : Mp line24 dhyp, Ui \
    D2, Simp1 Simp2 Simp2 \
    Mboldtheta
```

```
line28 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} line24 (dhyp_1) Mp
    D2 Ui Simp1 (Simp2
    (Simp2 (Mboldtheta))) : that
    prime2 ([(S'_3
       : obj) =>
       ({def} thelaw
       (S'_3) : obj)], D2) E Misset
    Mbold2 thelawchooses)]


line28 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that prime2
    ([(S'_3 : obj) =>
       ({def} thelaw
       (S'_3) : obj)], D2) E Misset
    Mbold2 thelawchooses)]


{move 6}

>>> define line29 dhyp \
    : Mp line28 dhyp, Ui \
    prime D2, Simp2 Simp2 \
    line26


line29 : [(dhyp_1
    : that D2 E Cuts2) =>
    ({def} line28 (dhyp_1) Mp
    prime (D2) Ui Simp2
    (Simp2 (line26)) : that
    (prime (D2) <<=
    B) V B <<= prime
    (D2))]
```

```
line29 : [(dhyp_1
    : that D2 E Cuts2) =>
    (--- : that (prime
    (D2) <<= B) V B <<=
    prime (D2))]


{move 6}

>>> goal that ((prime \
    D2) <<= prime B) V (B <<= \
    prime D2)


that (prime (D2) <<=
 prime (B)) V B <<=
 prime (D2)


{move 7}

>>> open


   {move 8}

   >>> declare U obj


   U : obj


   {move 8}

   >>> declare Casehyp1 \
       that B = 0
```

```
that B =0 is not well-formed

(paused, type something to continue) >

                        >>> define linea29 \
                            Casehyp1 : Subs1 \
                            (Eqsymm Casehyp1, Add2 \
                            (prime D2 <<= prime \
                            B, (Zeroissubset \
                            Separation3 Refleq \
                            prime D2)))

Casehyp1 : Subs1 (Eqsymm Casehyp1, Add2 (prime D2 <<= prime B, Zeroissubset Sep

(paused, type something to continue) >

                        >>> declare Casehyp2 \
                            that Exists [U => \
                               U E B]


                        Casehyp2 : that Exists
                         ([(U_2 : obj) =>
                            ({def} U_2 E B : prop)])


                        {move 8}

                        >>> open


                           {move 9}

                           >>> declare casehyp1 \
                               that D2 <<= prime \
                               B
```

```
casehyp1 : that
 D2 <<= prime (B)


{move 9}

>>> declare casehyp2 \
    that B <<= D2


casehyp2 : that
 B <<= D2


{move 9}

>>> define line30 \
    casehyp1 : Transsub \
    (line16 (line24 \
    dhyp), casehyp1)


line30 : [(casehyp1_1
    : that D2 <<=
    prime (B)) =>
    ({def} line16
    (line24 (dhyp)) Transsub
    casehyp1_1
    : that prime
    (D2) <<=
    prime (B))]


line30 : [(casehyp1_1
    : that D2 <<=
    prime (B)) =>
    (--- : that
```

```
    prime (D2) <<=
    prime (B))]


{move 8}

>>> define linea30 \
    casehyp1 : Add1 \
    (B <<= prime \
    D2, line30 casehyp1)


linea30 : [(casehyp1_1
    : that D2 <<=
    prime (B)) =>
    ({def} (B <<=
    prime (D2)) Add1
    line30 (casehyp1_1) : that
    (prime (D2) <<=
    prime (B)) V B <<=
    prime (D2))]


linea30 : [(casehyp1_1
    : that D2 <<=
    prime (B)) =>
    (--- : that
    (prime (D2) <<=
    prime (B)) V B <<=
    prime (D2))]


{move 8}

>>> define line31 \
    : Excmid ((thelaw \
    D2) = thelaw \
    B)
```

```
line31 : [
    ({def} Excmid
    (thelaw (D2) = thelaw
    (B)) : that
    (thelaw (D2) = thelaw
    (B)) V ~ (thelaw
    (D2) = thelaw
    (B)))]


line31 : that
 (thelaw (D2) = thelaw
 (B)) V ~ (thelaw
 (D2) = thelaw
 (B))


{move 8}

>>> define line32 \
    : Separation4 \
    Refleq prime D2


line32 : [
    ({def} Separation4
    (Refleq (prime
    (D2))) : that
    Forall ([(x_2
       : obj) =>
       ({def} (x_2
       E D2 Set
       [(x_5
          : obj) =>
          ({def} ~ (x_5
          E Usc
```

```
         (thelaw
         (D2))) : prop)]) ==
      (x_2 E D2) & ~ (x_2
      E Usc (thelaw
      (D2))) : prop)]))]


line32 : that
 Forall ([(x_2
   : obj) =>
   ({def} (x_2
   E D2 Set [(x_5
      : obj) =>
      ({def} ~ (x_5
      E Usc (thelaw
      (D2))) : prop)]) ==
   (x_2 E D2) & ~ (x_2
   E Usc (thelaw
   (D2))) : prop)])


{move 8}

>>> open


   {move 10}

   >>> declare \
       casehypa1 that \
       (thelaw D2 \
       = thelaw B)


   casehypa1 : that
    thelaw (D2) = thelaw
    (B)
```

```
{move 10}

>>> declare \
    casehypa2 that \
    ~ (thelaw \
    D2 = thelaw \
    B)


casehypa2 : that
 ~ (thelaw
 (D2) = thelaw
 (B))


{move 10}

>>> open


   {move 11}

   >>> declare \
       G obj


   G : obj


   {move 11}

   >>> open


      {move
       12}
```

```
>>> declare \
    onedir \
    that \
    G E prime \
    D2


onedir
 : that
 G E prime
 (D2)


{move
 12}

>>> define \
    line33 \
    onedir \
    : Iff1 \
    onedir, Ui \
    G line32


line33
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} onedir_1
    Iff1
    G Ui
    line32
    : that
    (G E D2) & ~ (G E Usc
    (thelaw
    (D2))))]
```

```
line33
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    (G E D2) & ~ (G E Usc
    (thelaw
    (D2))))]


{move
 11}

>>> define \
    line34 \
    onedir \
    : Simp1 \
    line33 \
    onedir


line34
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} Simp1
    (line33
    (onedir_1)) : that
    G E D2)]


line34
 : [(onedir_1
    : that
```

```
    G E prime
    (D2)) =>
    (---
    : that
    G E D2)]


{move
 11}

>>> define \
    line35 \
    onedir \
    : Simp2 \
    line33 \
    onedir


line35
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} Simp2
    (line33
    (onedir_1)) : that
    ~ (G E Usc
    (thelaw
    (D2))))]


line35
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
```

```
    ~ (G E Usc
    (thelaw
    (D2))))]


{move
 11}

>>> open


   {move
    13}

   >>> \
       declare \
       eqhyp \
       that \
       G = (thelaw \
       D2)


   eqhyp
    : that
    G = thelaw
    (D2)


   {move
    13}

   >>> \
       define \
       line36 \
       eqhyp \
       : Subs1 \
       Eqsymm \
       eqhyp \
```

```
    line35 \
    onedir


line36
 : [(eqhyp_1
    : that
    G = thelaw
    (D2)) =>
    ({def} Eqsymm
    (eqhyp_1) Subs1
    line35
    (onedir) : that
    ~ (G E Usc
    (G)))]


line36
 : [(eqhyp_1
    : that
    G = thelaw
    (D2)) =>
    (---
    : that
    ~ (G E Usc
    (G)))]


{move
 12}

>>> \
    define \
    line37 \
    eqhyp \
    : Mp \
    (Inusc2 \
    G, line36 \
```

```
            eqhyp)


        line37
         : [(eqhyp_1
            : that
            G = thelaw
            (D2)) =>
            ({def} Inusc2
            (G) Mp
            line36
            (eqhyp_1) : that
            ??)]


        line37
         : [(eqhyp_1
            : that
            G = thelaw
            (D2)) =>
            (---
            : that
            ??)]


        {move
         12}

        >>> \
            close


     {move
      12}

     >>> define \
         line38 \
         onedir \
```

```
    : Negintro \
    line37


line38
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} Negintro
    ([(eqhyp_2
       : that
       G = thelaw
       (D2)) =>
       ({def} Inusc2
       (G) Mp
       Eqsymm
       (eqhyp_2) Subs1
       line35
       (onedir_1) : that
       ??)]) : that
    ~ (G = thelaw
    (D2)))]


line38
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    ~ (G = thelaw
    (D2)))]


{move
 11}
```

```
>>> define \
    line39 \
    onedir \
    : Subs1 \
    casehypa1 \
    line38 \
    onedir


line39
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} casehypa1
    Subs1
    line38
    (onedir_1) : that
    ~ (G = thelaw
    (B)))]


line39
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    ~ (G = thelaw
    (B)))]


{move
 11}

>>> define \
```

```
    linea39 \
    onedir \
    : Subs1 \
    casehypa1 \
    line35 \
    onedir


linea39
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} casehypa1
    Subs1
    line35
    (onedir_1) : that
    ~ (G E Usc
    (thelaw
    (B))))]


linea39
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    ~ (G E Usc
    (thelaw
    (B))))]


{move
 11}

>>> open
```

```
{move
 13}

>>> \
    declare \
    casehypb1 \
    that \
    prime \
    D2 \
    <<= \
    B


casehypb1
 : that
 prime
 (D2) <<=
 B


{move
 13}

>>> \
    define \
    line40 \
    casehypb1 \
    : Mp \
    (onedir, Ui \
    G, Simp1 \
    casehypb1)


line40
 : [(casehypb1_1
    : that
```

```
    prime
    (D2) <<=
    B) =>
    ({def} onedir
    Mp
    G Ui
    Simp1
    (casehypb1_1) : that
    G E B)]


line40
 : [(casehypb1_1
    : that
    prime
    (D2) <<=
    B) =>
    (---
    : that
    G E B)]


{move
 12}

>>> \
    declare \
    casehypb2 \
    that \
    B <<= \
    prime \
    D2


casehypb2
 : that
 B <<=
 prime
```

```
 (D2)


{move
 13}

>>> \
    define \
    line41 \
    casehypb2 \
    : Ui \
    thelaw \
    B, Simp1 \
    casehypb2


line41
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} thelaw
    (B) Ui
    Simp1
    (casehypb2_1) : that
    (thelaw
    (B) E B) ->
    thelaw
    (B) E prime
    (D2))]


line41
 : [(casehypb2_1
    : that
    B <<=
    prime
```

```
    (D2)) =>
    (---
    : that
    (thelaw
    (B) E B) ->
    thelaw
    (B) E prime
    (D2))]


{move
 12}

>>> \
    define \
    line42 \
    : thelawchooses \
    (lineb14 \
    bhyp, Casehyp2)


line42
 : lineb14
 (bhyp) thelawchooses
 Casehyp2


line42
 : that
 thelaw
 (B) E B


{move
 12}

>>> \
    define \
```

```
    line43 \
    casehypb2 \
    : Mp \
    (line42, line41 \
    casehypb2)


line43
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} line42
    Mp
    line41
    (casehypb2_1) : that
    thelaw
    (B) E prime
    (D2))]


line43
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    thelaw
    (B) E prime
    (D2))]


{move
 12}
```

```
>>> \
    define \
    line44 \
    casehypb2 \
    : Iff1 \
    (line43 \
    casehypb2, Ui \
    thelaw \
    B, Separation4 \
    Refleq \
    prime \
    D2)


line44
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} line43
    (casehypb2_1) Iff1
    thelaw
    (B) Ui
    Separation4
    (Refleq
    (prime
    (D2))) : that
    (thelaw
    (B) E D2) & ~ (thelaw
    (B) E Usc
    (thelaw
    (D2))))]


line44
 : [(casehypb2_1
    : that
```

```
    B <<=
    prime
    (D2)) =>
    (---
    : that
    (thelaw
    (B) E D2) & ~ (thelaw
    (B) E Usc
    (thelaw
    (D2))))]


{move
 12}

>>> \
    define \
    line45 \
    casehypb2 \
    : Subs1 \
    Eqsymm \
    casehypa1 \
    line44 \
    casehypb2


line45
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} Eqsymm
    (casehypa1) Subs1
    line44
    (casehypb2_1) : that
    (thelaw
    (D2) E D2) & ~ (thelaw
```

```
    (D2) E Usc
    (thelaw
    (D2))))]


line45
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    (thelaw
    (D2) E D2) & ~ (thelaw
    (D2) E Usc
    (thelaw
    (D2))))]


{move
 12}

>>> \
    define \
    line46 \
    casehypb2 \
    : Simp2 \
    line45 \
    casehypb2


line46
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
```

```
    ({def} Simp2
    (line45
    (casehypb2_1)) : that
    ~ (thelaw
    (D2) E Usc
    (thelaw
    (D2))))]


line46
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    ~ (thelaw
    (D2) E Usc
    (thelaw
    (D2))))]


{move
 12}

>>> \
    define \
    line47 \
    casehypb2 \
    : Giveup \
    (G E B, Mp \
    (Inusc2 \
    thelaw \
    D2, line46 \
    casehypb2))
```

```
line47
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} (G E B) Giveup
    Inusc2
    (thelaw
    (D2)) Mp
    line46
    (casehypb2_1) : that
    G E B)]


line47
 : [(casehypb2_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    G E B)]


{move
 12}

>>> \
    close


{move
 12}

>>> define \
    line48 \
```

```
    onedir \
    : Cases \
    (line29 \
    dhyp, line40, line47)


line48
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} Cases
    (line29
    (dhyp), [(casehypb1_2
       : that
       prime
       (D2) <<=
       B) =>
       ({def} onedir_1
       Mp
       G Ui
       Simp1
       (casehypb1_2) : that
       G E B)], [(casehypb2_2
       : that
       B <<=
       prime
       (D2)) =>
       ({def} (G E B) Giveup
       Inusc2
       (thelaw
       (D2)) Mp
       Simp2
       (Eqsymm
       (casehypa1) Subs1
       lineb14
       (bhyp) thelawchooses
       Casehyp2
```

```
        Mp
        thelaw
        (B) Ui
        Simp1
        (casehypb2_2) Iff1
        thelaw
        (B) Ui
        Separation4
        (Refleq
        (prime
        (D2)))) : that
        G E B)]) : that
    G E B)]


line48
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    G E B)]


{move
 11}

>>> define \
    linea48 \
    onedir \
    : Fixform \
    (G E prime \
    (B), Iff2 \
    (Conj \
    (line48 \
    onedir, linea39 \
    onedir), Ui \
```

```
    G, Separation4 \
    Refleq \
    prime \
    B))


linea48
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    ({def} (G E prime
    (B)) Fixform
    line48
    (onedir_1) Conj
    linea39
    (onedir_1) Iff2
    G Ui
    Separation4
    (Refleq
    (prime
    (B))) : that
    G E prime
    (B))]


linea48
 : [(onedir_1
    : that
    G E prime
    (D2)) =>
    (---
    : that
    G E prime
    (B))]


{move
```

```
 11}

>>> declare \
    otherdir \
    that \
    G E B


otherdir
 : that
 G E B


{move
 12}

>>> define \
    line49 \
    otherdir \
    : Mp \
    (otherdir, Ui \
    G Simp1 \
    casehyp2)


line49
 : [(otherdir_1
    : that
    G E B) =>
    ({def} otherdir_1
    Mp
    G Ui
    Simp1
    (casehyp2) : that
    G E D2)]


line49
```

```
 : [(otherdir_1
    : that
    G E B) =>
    (---
    : that
    G E D2)]


{move
 11}

>>> open


   {move
    13}

   >>> \
       declare \
       eqhyp2 \
       that \
       G E Usc \
       thelaw \
       D2


   eqhyp2
    : that
    G E Usc
    (thelaw
    (D2))


   {move
    13}

   >>> \
       define \
```

```
    eqhypa2 \
    eqhyp2 \
    : Oridem \
    (Iff1 \
    (eqhyp2, Ui \
    G, Pair \
    (thelaw \
    D2, thelaw \
    D2)))


eqhypa2
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} Oridem
    (eqhyp2_1
    Iff1
    G Ui
    thelaw
    (D2) Pair
    thelaw
    (D2)) : that
    G = thelaw
    (D2))]


eqhypa2
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    G = thelaw
```

```
    (D2))]


{move
 12}

>>> \
    define \
    line50 \
    eqhyp2 \
    : Subs1 \
    eqhypa2 \
    eqhyp2 \
    otherdir


line50
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} eqhypa2
    (eqhyp2_1) Subs1
    otherdir
    : that
    thelaw
    (D2) E B)]


line50
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
```

```
    thelaw
    (D2) E B)]


{move
 12}

>>> \
    open


   {move
    14}

   >>> \
       declare \
       impossiblesub \
       that \
       B <<= \
       prime \
       D2


   impossiblesub
    : that
    B <<=
    prime
    (D2)


   {move
    14}

   >>> \
       define \
       line51 \
       impossiblesub \
       : Mp \
```

```
    (line50 \
    eqhyp2, Ui \
    (thelaw \
    D2, Simp1 \
    impossiblesub))


line51
 : [(impossiblesub_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} line50
    (eqhyp2) Mp
    thelaw
    (D2) Ui
    Simp1
    (impossiblesub_1) : that
    thelaw
    (D2) E prime
    (D2))]


line51
 : [(impossiblesub_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    thelaw
    (D2) E prime
    (D2))]


{move
```

```
 13}

>>> \
    define \
    line52 \
    impossiblesub \
    : Iff1 \
    (line51 \
    impossiblesub, Ui \
    thelaw \
    D2, Separation4 \
    Refleq \
    prime \
    D2)


line52
 : [(impossiblesub_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} line51
    (impossiblesub_1) Iff1
    thelaw
    (D2) Ui
    Separation4
    (Refleq
    (prime
    (D2))) : that
    (thelaw
    (D2) E D2) & ~ (thelaw
    (D2) E Usc
    (thelaw
    (D2))))]


line52
```

```
 : [(impossiblesub_1
    : that
    B <<=
    prime
    (D2)) =>
    (---
    : that
    (thelaw
    (D2) E D2) & ~ (thelaw
    (D2) E Usc
    (thelaw
    (D2))))]


{move
 13}

>>> \
    define \
    line53 \
    impossiblesub \
    : Mp \
    (Inusc2 \
    thelaw \
    D2, Simp2 \
    line52 \
    impossiblesub)


line53
 : [(impossiblesub_1
    : that
    B <<=
    prime
    (D2)) =>
    ({def} Inusc2
    (thelaw
    (D2)) Mp
```

```
        Simp2
        (line52
        (impossiblesub_1)) : that
        ??)]


    line53
     : [(impossiblesub_1
        : that
        B <<=
        prime
        (D2)) =>
        (---
        : that
        ??)]


    {move
     13}

    >>> \
        close


{move
 13}

>>> \
    define \
    line54 \
    eqhyp2 \
    : Negintro \
    line53


line54
 : [(eqhyp2_1
    : that
```

```
    G E Usc
    (thelaw
    (D2))) =>
    ({def} Negintro
    ([(impossiblesub_2
       : that
       B <<=
       prime
       (D2)) =>
       ({def} Inusc2
       (thelaw
       (D2)) Mp
       Simp2
       (line50
       (eqhyp2_1) Mp
       thelaw
       (D2) Ui
       Simp1
       (impossiblesub_2) Iff1
       thelaw
       (D2) Ui
       Separation4
       (Refleq
       (prime
       (D2)))) : that
       ??)]) : that
    ~ (B <<=
    prime
    (D2)))]


line54
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
```

```
    : that
    ~ (B <<=
    prime
    (D2)))]


{move
 12}

>>> \
    define \
    line55 \
    eqhyp2 \
    : Ds1 \
    line29 \
    dhyp \
    line54 \
    eqhyp2


line55
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} line29
    (dhyp) Ds1
    line54
    (eqhyp2_1) : that
    prime
    (D2) <<=
    B)]


line55
 : [(eqhyp2_1
    : that
```

```
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    prime
    (D2) <<=
    B)]


{move
 12}

>>> \
    open


   {move
    14}

   >>> \
       declare \
       H obj


   H : obj


   {move
    14}

   >>> \
       open


      {move
       15}
```

```
>>> \
    declare \
    hhyp \
    that \
    H E D2


hhyp
 : that
 H E D2


{move
 15}

>>> \
    define \
    line56 \
    : Excmid \
    (H = thelaw \
    D2)


line56
 : [
    ({def} Excmid
    (H = thelaw
    (D2)) : that
    (H = thelaw
    (D2)) V ~ (H = thelaw
    (D2)))]


line56
 : that
 (H = thelaw
 (D2)) V ~ (H = thelaw
 (D2))
```

```
{move
 14}

>>> \
    open


   {move
    16}

   >>> \
       declare \
       casehhyp1 \
       that \
       H = thelaw \
       D2


   casehhyp1
    : that
    H = thelaw
    (D2)


   {move
    16}

   >>> \
       declare \
       casehhyp2 \
       that \
       ~ (H = thelaw \
       D2)


   casehhyp2
```

```
 : that
 ~ (H = thelaw
 (D2))


{move
 16}

>>> \
    define \
    line57 \
    casehhyp1 \
    : Subs1 \
    (Eqsymm \
    casehhyp1, line50 \
    eqhyp2)


line57
 : [(casehhyp1_1
    : that
    H = thelaw
    (D2)) =>
    ({def} Eqsymm
    (casehhyp1_1) Subs1
    line50
    (eqhyp2) : that
    H E B)]


line57
 : [(casehhyp1_1
    : that
    H = thelaw
    (D2)) =>
    (---
    : that
    H E B)]
```

```
{move
 15}

>>> \
    open


   {move
    17}

   >>> \
       declare \
       sillyhyp \
       that \
       H E Usc \
       thelaw \
       D2


   sillyhyp
    : that
    H E Usc
    (thelaw
    (D2))


   {move
    17}

   >>> \
       define \
       line58 \
       sillyhyp \
       : Mp \
       (Oridem \
       (Iff1 \
```

```
    (sillyhyp, Ui \
    H, Pair \
    (thelaw \
    D2, thelaw \
    D2))), casehhyp2)


line58
 : [(sillyhyp_1
    : that
    H E Usc
    (thelaw
    (D2))) =>
    ({def} Oridem
    (sillyhyp_1
    Iff1
    H Ui
    thelaw
    (D2) Pair
    thelaw
    (D2)) Mp
    casehhyp2
    : that
    ??)]


line58
 : [(sillyhyp_1
    : that
    H E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    ??)]


{move
```

```
    16}

   >>> \
       close


{move
 16}

>>> \
    define \
    line59 \
    casehhyp2 \
    : Negintro \
    line58


line59
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    ({def} Negintro
    ([(sillyhyp_2
       : that
       H E Usc
       (thelaw
       (D2))) =>
       ({def} Oridem
       (sillyhyp_2
       Iff1
       H Ui
       thelaw
       (D2) Pair
       thelaw
       (D2)) Mp
       casehhyp2_1
       : that
```

```
      ??)]) : that
    ~ (H E Usc
    (thelaw
    (D2))))]


line59
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    (---
    : that
    ~ (H E Usc
    (thelaw
    (D2))))]


{move
 15}

>>> \
    define \
    line60 \
    casehhyp2 \
    : Fixform \
    (H E prime \
    D2, Iff2 \
    (Conj \
    (hhyp, line59 \
    casehhyp2), Ui \
    H, Separation4 \
    Refleq \
    prime \
    D2))


line60
```

```
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    ({def} (H E prime
    (D2)) Fixform
    hhyp
    Conj
    line59
    (casehhyp2_1) Iff2
    H Ui
    Separation4
    (Refleq
    (prime
    (D2))) : that
    H E prime
    (D2))]


line60
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    (---
    : that
    H E prime
    (D2))]


{move
 15}

>>> \
    define \
    line61 \
    casehhyp2 \
    : Mp \
```

```
    (line60 \
    casehhyp2, Ui \
    H, Simp1 \
    line55 \
    eqhyp2)


line61
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    ({def} line60
    (casehhyp2_1) Mp
    H Ui
    Simp1
    (line55
    (eqhyp2)) : that
    H E B)]


line61
 : [(casehhyp2_1
    : that
    ~ (H = thelaw
    (D2))) =>
    (---
    : that
    H E B)]


{move
 15}

>>> \
    close
```

```
{move
 15}

>>> \
    define \
    line62 \
    hhyp \
    : Cases \
    line56 \
    line57, line61


line62
 : [(hhyp_1
    : that
    H E D2) =>
    ({def} Cases
    (line56, [(casehhyp1_2
       : that
       H = thelaw
       (D2)) =>
       ({def} Eqsymm
       (casehhyp1_2) Subs1
       line50
       (eqhyp2) : that
       H E B)], [(casehhyp2_2
       : that
       ~ (H = thelaw
       (D2))) =>
       ({def} ((H E prime
       (D2)) Fixform
       hhyp_1
       Conj
       Negintro
       ([(sillyhyp_7
          : that
          H E Usc
          (thelaw
```

```
          (D2))) =>
          ({def} Oridem
          (sillyhyp_7
          Iff1
          H Ui
          thelaw
          (D2) Pair
          thelaw
          (D2)) Mp
          casehhyp2_2
          : that
          ??)]) Iff2
       H Ui
       Separation4
       (Refleq
       (prime
       (D2)))) Mp
       H Ui
       Simp1
       (line55
       (eqhyp2)) : that
       H E B)]) : that
    H E B)]


line62
 : [(hhyp_1
    : that
    H E D2) =>
    (---
    : that
    H E B)]


{move
 14}

>>> \
```

```
        close


       {move
        14}

       >>> \
           define \
           line63 \
           H : Ded \
           line62


       line63
        : [(H_1
           : obj) =>
           ({def} Ded
           ([(hhyp_2
              : that
              H_1
              E D2) =>
              ({def} Cases
              (Excmid
              (H_1
              = thelaw
              (D2)), [(casehhyp1_3
                 : that
                 H_1
                 = thelaw
                 (D2)) =>
                 ({def} Eqsymm
                 (casehhyp1_3) Subs1
                 line50
                 (eqhyp2) : that
                 H_1
                 E B)], [(casehhyp2_3
                 : that
                 ~ (H_1
```

```
= thelaw
(D2))) =>
({def} ((H_1
E prime
(D2)) Fixform
hhyp_2
Conj
Negintro
([(sillyhyp_8
   : that
   H_1
   E Usc
   (thelaw
   (D2))) =>
   ({def} Oridem
   (sillyhyp_8
   Iff1
   H_1
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Mp
   casehhyp2_3
   : that
   ??)]) Iff2
H_1
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
H_1
Ui
Simp1
(line55
(eqhyp2)) : that
H_1
```

```
          E B)]) : that
       H_1
       E B)]) : that
    (H_1
    E D2) ->
    H_1 E B)]


line63
 : [(H_1
    : obj) =>
    (---
    : that
    (H_1
    E D2) ->
    H_1
    E B)]


{move
 13}

>>> \
    close


{move
 13}

>>> \
    define \
    line64 \
    eqhyp2 \
    : Ug \
    line63


line64
```

```
: [(eqhyp2_1
   : that
   G E Usc
   (thelaw
   (D2))) =>
   ({def} Ug
   ([(H_2
      : obj) =>
      ({def} Ded
      ([(hhyp_3
         : that
         H_2
         E D2) =>
         ({def} Cases
         (Excmid
         (H_2
         = thelaw
         (D2)), [(casehhyp1_4
            : that
            H_2
            = thelaw
            (D2)) =>
            ({def} Eqsymm
            (casehhyp1_4) Subs1
            line50
            (eqhyp2_1) : that
            H_2
            E B)], [(casehhyp2_4
            : that
            ~ (H_2
            = thelaw
            (D2))) =>
            ({def} ((H_2
            E prime
            (D2)) Fixform
            hhyp_3
            Conj
            Negintro
```

```
                  ([(sillyhyp_9
                     : that
                     H_2
                     E Usc
                     (thelaw
                     (D2))) =>
                     ({def} Oridem
                     (sillyhyp_9
                     Iff1
                     H_2
                     Ui
                     thelaw
                     (D2) Pair
                     thelaw
                     (D2)) Mp
                     casehhyp2_4
                     : that
                     ??)]) Iff2
                  H_2
                  Ui
                  Separation4
                  (Refleq
                  (prime
                  (D2)))) Mp
                  H_2
                  Ui
                  Simp1
                  (line55
                  (eqhyp2_1)) : that
                  H_2
                  E B)]) : that
               H_2
               E B)]) : that
            (H_2
            E D2) ->
            H_2 E B)]) : that
         Forall ([(x'_2
            : obj) =>
```

```
      ({def} (x'_2
      E D2) ->
      x'_2 E B : prop)]))]


line64
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    Forall
    ([(x'_2
       : obj) =>
       ({def} (x'_2
       E D2) ->
       x'_2
       E B : prop)]))]


{move
 12}

>>> \
    define \
    line65 \
    eqhyp2 \
    : Fixform \
    (D2 \
    <<= \
    B, Conj \
    (line64 \
    eqhyp2, Conj \
    (Simp2 \
    Simp2 \
    casehyp2, linea14 \
```

```
    bhyp)))


line65
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} (D2
    <<=
    B) Fixform
    line64
    (eqhyp2_1) Conj
    Simp2
    (Simp2
    (casehyp2)) Conj
    linea14
    (bhyp) : that
    D2
    <<=
    B)]


line65
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    D2
    <<=
    B)]


{move
```

```
 12}

>>> \
    define \
    line66 \
    eqhyp2 \
    : Antisymsub \
    (casehyp2, line65 \
    eqhyp2)


line66
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} casehyp2
    Antisymsub
    line65
    (eqhyp2_1) : that
    B = D2)]


line66
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    B = D2)]


{move
 12}
```

```
>>> \
    define \
    line67 \
    eqhyp2 \
    : Mp \
    (Refleq \
    thelaw \
    D2, Subs1 \
    (line66 \
    eqhyp2, casehypa2))


line67
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    ({def} Refleq
    (thelaw
    (D2)) Mp
    line66
    (eqhyp2_1) Subs1
    casehypa2
    : that
    ??)]


line67
 : [(eqhyp2_1
    : that
    G E Usc
    (thelaw
    (D2))) =>
    (---
    : that
    ??)]
```

```
   {move
    12}

   >>> \
       close


{move
 12}

>>> define \
    line68 \
    otherdir \
    : Fixform \
    (G E prime \
    D2, Iff2 \
    (Conj \
    (line49 \
    otherdir, Negintro \
    line67), Ui \
    G, Separation4 \
    Refleq \
    prime \
    D2))


line68
 : [(otherdir_1
    : that
    G E B) =>
    ({def} (G E prime
    (D2)) Fixform
    line49
    (otherdir_1) Conj
    Negintro
    ([(eqhyp2_5
       : that
```

```
G E Usc
(thelaw
(D2))) =>
({def} Refleq
(thelaw
(D2)) Mp
casehyp2
Antisymsub
(D2
<<=
B) Fixform
Ug
([(H_11
   : obj) =>
   ({def} Ded
   ([(hhyp_12
      : that
      H_11
      E D2) =>
      ({def} Cases
      (Excmid
      (H_11
      = thelaw
      (D2)), [(casehhyp1_13
         : that
         H_11
         = thelaw
         (D2)) =>
         ({def} Eqsymm
         (casehhyp1_13) Subs1
         Oridem
         (eqhyp2_5
         Iff1
         G Ui
         thelaw
         (D2) Pair
         thelaw
         (D2)) Subs1
```

```
otherdir_1
: that
H_11
E B)], [(casehhyp2_13
: that
~ (H_11
= thelaw
(D2))) =>
({def} ((H_11
E prime
(D2)) Fixform
hhyp_12
Conj
Negintro
([(sillyhyp_18
   : that
   H_11
   E Usc
   (thelaw
   (D2))) =>
   ({def} Oridem
   (sillyhyp_18
   Iff1
   H_11
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Mp
   casehhyp2_13
   : that
   ??)]) Iff2
H_11
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
```

```
H_11
Ui
Simp1
(line29
(dhyp) Ds1
Negintro
([(impossiblesub_18
   : that
   B <<=
   prime
   (D2)) =>
   ({def} Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Oridem
   (eqhyp2_5
   Iff1
   G Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Subs1
   otherdir_1
   Mp
   thelaw
   (D2) Ui
   Simp1
   (impossiblesub_18) Iff1
   thelaw
   (D2) Ui
   Separation4
   (Refleq
   (prime
   (D2)))) : that
   ??)])) : that
H_11
E B)]) : that
```

```
                  H_11
                  E B)]) : that
               (H_11
               E D2) ->
               H_11 E B)]) Conj
            Simp2 (Simp2
            (casehyp2)) Conj
            linea14 (bhyp) Subs1
            casehypa2
            : that ??)]) Iff2
         G Ui Separation4
         (Refleq (prime
         (D2))) : that
         G E prime (D2))]


     line68
      : [(otherdir_1
         : that
         G E B) =>
         (---
         : that
         G E prime
         (D2))]


     {move
      11}

     >>> close


  {move 11}

  >>> define \
      line69 G : Ded \
      line68
```

```
line69 : [(G_1
    : obj) =>
    ({def} Ded
    ([(otherdir_2
       : that
       G_1
       E B) =>
       ({def} (G_1
       E prime
       (D2)) Fixform
       otherdir_2
       Mp
       G_1
       Ui
       Simp1
       (casehyp2) Conj
       Negintro
       ([(eqhyp2_6
          : that
          G_1
          E Usc
          (thelaw
          (D2))) =>
          ({def} Refleq
          (thelaw
          (D2)) Mp
          casehyp2
          Antisymsub
          (D2
          <<=
          B) Fixform
          Ug
          ([(H_12
             : obj) =>
             ({def} Ded
             ([(hhyp_13
                : that
```

```
H_12
E D2) =>
({def} Cases
(Excmid
(H_12
= thelaw
(D2)), [(casehhyp1_14
   : that
   H_12
   = thelaw
   (D2)) =>
   ({def} Eqsymm
   (casehhyp1_14) Subs1
   Oridem
   (eqhyp2_6
   Iff1
   G_1
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Subs1
   otherdir_2
   : that
   H_12
   E B)], [(casehhyp2_14
   : that
   ~ (H_12
   = thelaw
   (D2))) =>
   ({def} ((H_12
   E prime
   (D2)) Fixform
   hhyp_13
   Conj
   Negintro
   ([(sillyhyp_19
      : that
```

```
    H_12
    E Usc
    (thelaw
    (D2))) =>
    ({def} Oridem
    (sillyhyp_19
    Iff1
    H_12
    Ui
    thelaw
    (D2) Pair
    thelaw
    (D2)) Mp
    casehhyp2_14
    : that
    ??)]) Iff2
H_12
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
H_12
Ui
Simp1
(line29
(dhyp) Ds1
Negintro
([(impossiblesub_19
    : that
    B <<=
    prime
    (D2)) =>
    ({def} Inusc2
    (thelaw
    (D2)) Mp
    Simp2
    (Oridem
```

```
                              (eqhyp2_6
                              Iff1
                              G_1
                              Ui
                              thelaw
                              (D2) Pair
                              thelaw
                              (D2)) Subs1
                              otherdir_2
                              Mp
                              thelaw
                              (D2) Ui
                              Simp1
                              (impossiblesub_19) Iff1
                              thelaw
                              (D2) Ui
                              Separation4
                              (Refleq
                              (prime
                              (D2)))) : that
                              ??)])) : that
                           H_12
                           E B)]) : that
                        H_12
                        E B)]) : that
                     (H_12
                     E D2) ->
                     H_12 E B)]) Conj
                  Simp2 (Simp2
                  (casehyp2)) Conj
                  linea14 (bhyp) Subs1
                  casehypa2
                  : that ??)]) Iff2
               G_1 Ui Separation4
               (Refleq (prime
               (D2))) : that
               G_1 E prime
               (D2))]) : that
```

```
    (G_1 E B) ->
    G_1 E prime
    (D2))]


line69 : [(G_1
    : obj) =>
    (---
    : that
    (G_1
    E B) ->
    G_1 E prime
    (D2))]


{move 10}

>>> define \
    testline \
    G : Ded \
    linea48


testline
 : [(G_1
    : obj) =>
    ({def} Ded
    ([(onedir_2
       : that
       G_1
       E prime
       (D2)) =>
       ({def} (G_1
       E prime
       (B)) Fixform
       Cases
       (line29
       (dhyp), [(casehypb1_6
```

```
: that
prime
(D2) <<=
B) =>
({def} onedir_2
Mp
G_1
Ui
Simp1
(casehypb1_6) : that
G_1
E B)], [(casehypb2_6
: that
B <<=
prime
(D2)) =>
({def} (G_1
E B) Giveup
Inusc2
(thelaw
(D2)) Mp
Simp2
(Eqsymm
(casehypa1) Subs1
lineb14
(bhyp) thelawchooses
Casehyp2
Mp
thelaw
(B) Ui
Simp1
(casehypb2_6) Iff1
thelaw
(B) Ui
Separation4
(Refleq
(prime
(D2))) : that
```

```
                G_1
                E B)]) Conj
             casehypa1
             Subs1
             Simp2
             (onedir_2
             Iff1
             G_1
             Ui
             line32) Iff2
             G_1
             Ui
             Separation4
             (Refleq
             (prime
             (B))) : that
             G_1
             E prime
             (B))]) : that
          (G_1
          E prime
          (D2)) ->
          G_1 E prime
          (B))]


      testline
       : [(G_1
          : obj) =>
          (---
          : that
          (G_1
          E prime
          (D2)) ->
          G_1 E prime
          (B))]
```

```
   {move 10}

   >>> close


{move 10}

>>> define \
    line70 casehypa2 \
    : Ug line69


line70 : [(casehypa2_1
    : that ~ (thelaw
    (D2) = thelaw
    (B))) =>
    ({def} Ug
    ([(G_2
       : obj) =>
       ({def} Ded
       ([(otherdir_3
          : that
          G_2
          E B) =>
          ({def} (G_2
          E prime
          (D2)) Fixform
          otherdir_3
          Mp
          G_2
          Ui
          Simp1
          (casehyp2) Conj
          Negintro
          ([(eqhyp2_7
             : that
             G_2
             E Usc
```

```
(thelaw
(D2))) =>
({def} Refleq
(thelaw
(D2)) Mp
casehyp2
Antisymsub
(D2
<<=
B) Fixform
Ug
([(H_13
   : obj) =>
   ({def} Ded
   ([(hhyp_14
      : that
      H_13
      E D2) =>
      ({def} Cases
      (Excmid
      (H_13
      = thelaw
      (D2)), [(casehhyp1_15
         : that
         H_13
         = thelaw
         (D2)) =>
         ({def} Eqsymm
         (casehhyp1_15) Subs1
         Oridem
         (eqhyp2_7
         Iff1
         G_2
         Ui
         thelaw
         (D2) Pair
         thelaw
         (D2)) Subs1
```

```
otherdir_3
: that
H_13
E B)], [(casehhyp2_15
: that
~ (H_13
= thelaw
(D2))) =>
({def} ((H_13
E prime
(D2)) Fixform
hhyp_14
Conj
Negintro
([(sillyhyp_20
   : that
   H_13
   E Usc
   (thelaw
   (D2))) =>
   ({def} Oridem
   (sillyhyp_20
   Iff1
   H_13
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Mp
   casehhyp2_15
   : that
   ??)]) Iff2
H_13
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
```

```
H_13
Ui
Simp1
(line29
(dhyp) Ds1
Negintro
([(impossiblesub_20
   : that
   B <<=
   prime
   (D2)) =>
   ({def} Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Oridem
   (eqhyp2_7
   Iff1
   G_2
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Subs1
   otherdir_3
   Mp
   thelaw
   (D2) Ui
   Simp1
   (impossiblesub_20) Iff1
   thelaw
   (D2) Ui
   Separation4
   (Refleq
   (prime
   (D2)))) : that
   ??)])) : that
H_13
```

```
                    E B)]) : that
                 H_13
                 E B)]) : that
              (H_13
              E D2) ->
              H_13 E B)]) Conj
           Simp2 (Simp2
           (casehyp2)) Conj
           linea14 (bhyp) Subs1
           casehypa2_1
           : that ??)]) Iff2
        G_2 Ui Separation4
        (Refleq (prime
        (D2))) : that
        G_2 E prime
        (D2))]) : that
     (G_2 E B) ->
     G_2 E prime
     (D2))]) : that
  Forall ([(x'_2
     : obj) =>
     ({def} (x'_2
     E B) ->
     x'_2
     E prime
     (D2) : prop)]))]


line70 : [(casehypa2_1
    : that ~ (thelaw
    (D2) = thelaw
    (B))) =>
    (--- : that
    Forall ([(x'_2
       : obj) =>
       ({def} (x'_2
       E B) ->
       x'_2
```

```
   E prime
   (D2) : prop)]))]


{move 9}

>>> define \
    line71 casehypa2 \
    : Add2 ((prime \
    D2) <<= prime \
    B, Fixform \
    (B <<= prime \
    D2, Conj (line70 \
    casehypa2, Conj \
    (linea14 bhyp, Separation3 \
    Refleq prime \
    D2))))


line71 : [(casehypa2_1
    : that ~ (thelaw
    (D2) = thelaw
    (B))) =>
    ({def} (prime
    (D2) <<=
    prime (B)) Add2
    (B <<=
    prime (D2)) Fixform
    line70 (casehypa2_1) Conj
    linea14
    (bhyp) Conj
    Separation3
    (Refleq
    (prime
    (D2))) : that
    (prime
    (D2) <<=
    prime (B)) V B <<=
```

```
    prime (D2))]


line71 : [(casehypa2_1
    : that ~ (thelaw
    (D2) = thelaw
    (B))) =>
    (--- : that
    (prime
    (D2) <<=
    prime (B)) V B <<=
    prime (D2))]


{move 9}

>>> define \
    testline2 casehypa1 \
    : Ug testline


testline2 : [(casehypa1_1
    : that thelaw
    (D2) = thelaw
    (B)) =>
    ({def} Ug
    ([(G_2
       : obj) =>
       ({def} Ded
       ([(onedir_3
          : that
          G_2
          E prime
          (D2)) =>
          ({def} (G_2
          E prime
          (B)) Fixform
          Cases
```

```
(line29
(dhyp), [(casehypb1_7
   : that
   prime
   (D2) <<=
   B) =>
   ({def} onedir_3
   Mp
   G_2
   Ui
   Simp1
   (casehypb1_7) : that
   G_2
   E B)], [(casehypb2_7
   : that
   B <<=
   prime
   (D2)) =>
   ({def} (G_2
   E B) Giveup
   Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Eqsymm
   (casehypa1_1) Subs1
   lineb14
   (bhyp) thelawchooses
   Casehyp2
   Mp
   thelaw
   (B) Ui
   Simp1
   (casehypb2_7) Iff1
   thelaw
   (B) Ui
   Separation4
   (Refleq
```

```
            (prime
            (D2)))) : that
            G_2
            E B)]) Conj
         casehypa1_1
         Subs1
         Simp2
         (onedir_3
         Iff1
         G_2
         Ui
         line32) Iff2
         G_2
         Ui
         Separation4
         (Refleq
         (prime
         (B))) : that
         G_2
         E prime
         (B))]) : that
      (G_2
      E prime
      (D2)) ->
      G_2 E prime
      (B))]) : that
   Forall ([(x'_2
      : obj) =>
      ({def} (x'_2
      E prime
      (D2)) ->
      x'_2
      E prime
      (B) : prop)]))]


testline2 : [(casehypa1_1
   : that thelaw
```

```
    (D2) = thelaw
    (B)) =>
    (--- : that
    Forall ([(x'_2
       : obj) =>
       ({def} (x'_2
       E prime
       (D2)) ->
       x'_2
       E prime
       (B) : prop)]))]


{move 9}

>>> define \
    line72 casehypa1 \
    : Add1 (B <<= \
    prime D2, Fixform \
    ((prime D2) <<= \
    prime B, Conj \
    (testline2 \
    casehypa1, Conj \
    (Separation3 \
    Refleq prime \
    D2, Separation3 \
    Refleq prime \
    B))))


line72 : [(casehypa1_1
    : that thelaw
    (D2) = thelaw
    (B)) =>
    ({def} (B <<=
    prime (D2)) Add1
    (prime
    (D2) <<=
```

```
        prime (B)) Fixform
        testline2
        (casehypa1_1) Conj
        Separation3
        (Refleq
        (prime
        (D2))) Conj
        Separation3
        (Refleq
        (prime
        (B))) : that
        (prime
        (D2) <<=
        prime (B)) V B <<=
        prime (D2))]


    line72 : [(casehypa1_1
        : that thelaw
        (D2) = thelaw
        (B)) =>
        (--- : that
        (prime
        (D2) <<=
        prime (B)) V B <<=
        prime (D2))]


    {move 9}

    >>> close


{move 9}

>>> define line73 \
    casehyp2 : Cases \
    line31 line72, line71
```

```
line73 : [(casehyp2_1
    : that B <<=
    D2) =>
    ({def} Cases
    (line31, [(casehypa1_2
       : that thelaw
       (D2) = thelaw
       (B)) =>
       ({def} (B <<=
       prime (D2)) Add1
       (prime
       (D2) <<=
       prime (B)) Fixform
       Ug ([(G_6
          : obj) =>
          ({def} Ded
          ([(onedir_7
             : that
             G_6
             E prime
             (D2)) =>
             ({def} (G_6
             E prime
             (B)) Fixform
             Cases
             (line29
             (dhyp), [(casehypb1_11
                : that
                prime
                (D2) <<=
                B) =>
                ({def} onedir_7
                Mp
                G_6
                Ui
                Simp1
```

```
   (casehypb1_11) : that
   G_6
   E B)], [(casehypb2_11
   : that
   B <<=
   prime
   (D2)) =>
   ({def} (G_6
   E B) Giveup
   Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Eqsymm
   (casehypa1_2) Subs1
   lineb14
   (bhyp) thelawchooses
   Casehyp2
   Mp
   thelaw
   (B) Ui
   Simp1
   (casehypb2_11) Iff1
   thelaw
   (B) Ui
   Separation4
   (Refleq
   (prime
   (D2)))) : that
   G_6
   E B)]) Conj
casehypa1_2
Subs1
Simp2
(onedir_7
Iff1
G_6
Ui
```

```
            line32) Iff2
            G_6
            Ui
            Separation4
            (Refleq
            (prime
            (B))) : that
            G_6
            E prime
            (B))]) : that
         (G_6
         E prime
         (D2)) ->
         G_6 E prime
         (B))]) Conj
      Separation3
      (Refleq
      (prime
      (D2))) Conj
      Separation3
      (Refleq
      (prime
      (B))) : that
      (prime
      (D2) <<=
      prime (B)) V B <<=
      prime (D2))], [(casehypa2_2
      : that ~ (thelaw
      (D2) = thelaw
      (B))) =>
      ({def} (prime
      (D2) <<=
      prime (B)) Add2
      (B <<=
      prime (D2)) Fixform
      Ug ([(G_6
         : obj) =>
         ({def} Ded
```

```
([(otherdir_7
   : that
   G_6
   E B) =>
   ({def} (G_6
   E prime
   (D2)) Fixform
   otherdir_7
   Mp
   G_6
   Ui
   Simp1
   (casehyp2_1) Conj
   Negintro
   ([(eqhyp2_11
      : that
      G_6
      E Usc
      (thelaw
      (D2))) =>
      ({def} Refleq
      (thelaw
      (D2)) Mp
      casehyp2_1
      Antisymsub
      (D2
      <<=
      B) Fixform
      Ug
      ([(H_17
         : obj) =>
         ({def} Ded
         ([(hhyp_18
            : that
            H_17
            E D2) =>
            ({def} Cases
            (Excmid
```

```
(H_17
= thelaw
(D2)), [(casehhyp1_19
   : that
   H_17
   = thelaw
   (D2)) =>
   ({def} Eqsymm
   (casehhyp1_19) Subs1
   Oridem
   (eqhyp2_11
   Iff1
   G_6
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Subs1
   otherdir_7
   : that
   H_17
   E B)], [(casehhyp2_19
   : that
   ~ (H_17
   = thelaw
   (D2))) =>
   ({def} ((H_17
   E prime
   (D2)) Fixform
   hhyp_18
   Conj
   Negintro
   ([(sillyhyp_24
      : that
      H_17
      E Usc
      (thelaw
      (D2))) =>
```

```
   ({def} Oridem
   (sillyhyp_24
   Iff1
   H_17
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Mp
   casehhyp2_19
   : that
   ??)]) Iff2
H_17
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
H_17
Ui
Simp1
(line29
(dhyp) Ds1
Negintro
([(impossiblesub_24
   : that
   B <<=
   prime
   (D2)) =>
   ({def} Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Oridem
   (eqhyp2_11
   Iff1
   G_6
   Ui
```

```
                                        thelaw
                                        (D2) Pair
                                        thelaw
                                        (D2)) Subs1
                                        otherdir_7
                                        Mp
                                        thelaw
                                        (D2) Ui
                                        Simp1
                                        (impossiblesub_24) Iff1
                                        thelaw
                                        (D2) Ui
                                        Separation4
                                        (Refleq
                                        (prime
                                        (D2)))) : that
                                        ??)])) : that
                                     H_17
                                     E B)]) : that
                                  H_17
                                  E B)]) : that
                               (H_17
                               E D2) ->
                               H_17 E B)]) Conj
                            Simp2 (Simp2
                            (casehyp2_1)) Conj
                            linea14 (bhyp) Subs1
                            casehypa2_2
                            : that ??)]) Iff2
                         G_6 Ui Separation4
                         (Refleq (prime
                         (D2))) : that
                         G_6 E prime
                         (D2))]) : that
                      (G_6 E B) ->
                      G_6 E prime
                      (D2))]) Conj
                   linea14
```

```
          (bhyp) Conj
          Separation3
          (Refleq
          (prime
          (D2))) : that
          (prime
          (D2) <<=
          prime (B)) V B <<=
          prime (D2))]) : that
       (prime (D2) <<=
       prime (B)) V B <<=
       prime (D2))]


   line73 : [(casehyp2_1
       : that B <<=
       D2) => (---
       : that (prime
       (D2) <<=
       prime (B)) V B <<=
       prime (D2))]


   {move 8}

   >>> close


{move 8}

>>> define line74 \
    Casehyp2 : Cases \
    (line25 dhyp, linea30, line73)


line74 : [(Casehyp2_1
    : that Exists
    ([(U_3 : obj) =>
```

```
   ({def} U_3
   E B : prop)])) =>
({def} Cases
(line25 (dhyp), [(casehyp1_2
   : that D2 <<=
   prime (B)) =>
   ({def} (B <<=
   prime (D2)) Add1
   line16 (line24
   (dhyp)) Transsub
   casehyp1_2
   : that (prime
   (D2) <<=
   prime (B)) V B <<=
   prime (D2))], [(casehyp2_2
   : that B <<=
   D2) =>
   ({def} Cases
   (Excmid (thelaw
   (D2) = thelaw
   (B)), [(casehypa1_3
      : that thelaw
      (D2) = thelaw
      (B)) =>
      ({def} (B <<=
      prime (D2)) Add1
      (prime
      (D2) <<=
      prime (B)) Fixform
      Ug ([(G_7
         : obj) =>
         ({def} Ded
         ([(onedir_8
            : that
            G_7
            E prime
            (D2)) =>
            ({def} (G_7
```

```
E prime
(B)) Fixform
Cases
(line29
(dhyp), [(casehypb1_12
   : that
   prime
   (D2) <<=
   B) =>
   ({def} onedir_8
   Mp
   G_7
   Ui
   Simp1
   (casehypb1_12) : that
   G_7
   E B)], [(casehypb2_12
   : that
   B <<=
   prime
   (D2)) =>
   ({def} (G_7
   E B) Giveup
   Inusc2
   (thelaw
   (D2)) Mp
   Simp2
   (Eqsymm
   (casehypa1_3) Subs1
   lineb14
   (bhyp) thelawchooses
   Casehyp2_1
   Mp
   thelaw
   (B) Ui
   Simp1
   (casehypb2_12) Iff1
   thelaw
```

```
            (B) Ui
            Separation4
            (Refleq
            (prime
            (D2)))) : that
            G_7
            E B)]) Conj
         casehypa1_3
         Subs1
         Simp2
         (onedir_8
         Iff1
         G_7
         Ui
         Separation4
         (Refleq
         (prime
         (D2)))) Iff2
         G_7
         Ui
         Separation4
         (Refleq
         (prime
         (B))) : that
         G_7
         E prime
         (B))]) : that
      (G_7
      E prime
      (D2)) ->
      G_7 E prime
      (B))]) Conj
   Separation3
   (Refleq
   (prime
   (D2))) Conj
   Separation3
   (Refleq
```

```
(prime
(B))) : that
(prime
(D2) <<=
prime (B)) V B <<=
prime (D2))], [(casehypa2_3
: that ~ (thelaw
(D2) = thelaw
(B))) =>
({def} (prime
(D2) <<=
prime (B)) Add2
(B <<=
prime (D2)) Fixform
Ug ([(G_7
   : obj) =>
   ({def} Ded
   ([(otherdir_8
      : that
      G_7
      E B) =>
      ({def} (G_7
      E prime
      (D2)) Fixform
      otherdir_8
      Mp
      G_7
      Ui
      Simp1
      (casehyp2_2) Conj
      Negintro
      ([(eqhyp2_12
         : that
         G_7
         E Usc
         (thelaw
         (D2))) =>
         ({def} Refleq
```

```
(thelaw
(D2)) Mp
casehyp2_2
Antisymsub
(D2
<<=
B) Fixform
Ug
([(H_18
   : obj) =>
   ({def} Ded
   ([(hhyp_19
      : that
      H_18
      E D2) =>
      ({def} Cases
      (Excmid
      (H_18
      = thelaw
      (D2)), [(casehhyp1_20
         : that
         H_18
         = thelaw
         (D2)) =>
         ({def} Eqsymm
         (casehhyp1_20) Subs1
         Oridem
         (eqhyp2_12
         Iff1
         G_7
         Ui
         thelaw
         (D2) Pair
         thelaw
         (D2)) Subs1
         otherdir_8
         : that
         H_18
```

```
E B)], [(casehhyp2_20
: that
~ (H_18
= thelaw
(D2))) =>
({def} ((H_18
E prime
(D2)) Fixform
hhyp_19
Conj
Negintro
([(sillyhyp_25
   : that
   H_18
   E Usc
   (thelaw
   (D2))) =>
   ({def} Oridem
   (sillyhyp_25
   Iff1
   H_18
   Ui
   thelaw
   (D2) Pair
   thelaw
   (D2)) Mp
   casehhyp2_20
   : that
   ??)]) Iff2
H_18
Ui
Separation4
(Refleq
(prime
(D2)))) Mp
H_18
Ui
Simp1
```

```
   (line29
   (dhyp) Ds1
   Negintro
   ([(impossiblesub_25
      : that
      B <<=
      prime
      (D2)) =>
      ({def} Inusc2
      (thelaw
      (D2)) Mp
      Simp2
      (Oridem
      (eqhyp2_12
      Iff1
      G_7
      Ui
      thelaw
      (D2) Pair
      thelaw
      (D2)) Subs1
      otherdir_8
      Mp
      thelaw
      (D2) Ui
      Simp1
      (impossiblesub_25) Iff1
      thelaw
      (D2) Ui
      Separation4
      (Refleq
      (prime
      (D2)))) : that
      ??)])) : that
   H_18
   E B)]) : that
H_18
E B)]) : that
```

```
                              (H_18
                              E D2) ->
                              H_18 E B)]) Conj
                           Simp2 (Simp2
                           (casehyp2_2)) Conj
                           linea14 (bhyp) Subs1
                           casehypa2_3
                           : that ??)]) Iff2
                        G_7 Ui Separation4
                        (Refleq (prime
                        (D2))) : that
                        G_7 E prime
                        (D2))]) : that
                     (G_7 E B) ->
                     G_7 E prime
                     (D2))]) Conj
                  linea14
                  (bhyp) Conj
                  Separation3
                  (Refleq
                  (prime
                  (D2))) : that
                  (prime
                  (D2) <<=
                  prime (B)) V B <<=
                  prime (D2))]) : that
               (prime (D2) <<=
               prime (B)) V B <<=
               prime (D2))]) : that
            (prime (D2) <<=
            prime (B)) V B <<=
            prime (D2))]


        line74 : [(Casehyp2_1
            : that Exists
            ([(U_3 : obj) =>
               ({def} U_3
```

```
                      E B : prop)])) =>
                    (--- : that (prime
                    (D2) <<= prime
                    (B)) V B <<=
                    prime (D2))]


                {move 7}

                >>> close


             {move 7}

             >>> define line75 dhyp \
                 : Cases (linea14 bhyp, linea29, line74)

[dhyp => Cases (linea14 bhyp, linea29, line74)] is not well-formed

(paused, type something to continue) >

             >>> define line76 dhyp \
                 : Fixform ((prime \
                 D2) E Cuts2, Iff2 \
                 (Conj (line28 dhyp, line75 \
                 dhyp), Ui prime D2, Separation4 \
                 Refleq Cuts2))

[dhyp => Fixform ((prime D2) E Cuts2, Iff2 (Conj (line28 dhyp, line75 dhyp), Ui

(paused, type something to continue) >

             >>> close


          {move 6}

          >>> define line77 D2 : Ded \
```

```
                        line76

[D2 => Ded line76] is not well-formed

(paused, type something to continue) >

                  >>> close


               {move 5}

               >>> define linea78 : Ug line77

Ug line77 is not well-formed

(paused, type something to continue) >

               >>> save


               {move 5}

               >>> close


            {move 4}

            >>> define lineb78 bhyp : linea78

[bhyp => linea78] is not well-formed

(paused, type something to continue) >

            >>> save


            {move 4}
```

```
      >>> close


   {move 3}

   >>> declare bhypa1 that B E Cuts


   bhypa1 : that B E Cuts


   {move 3}

   >>> define linec78 bhypa1 : lineb78 \
       bhypa1

[bhypa1 => lineb78 bhypa1] is not well-formed

(paused, type something to continue) >

   >>> save


   {move 3}

   >>> close


 {move 2}

 >>> declare B111 obj


 B111 : obj


 {move 2}
```

```
      >>> declare bhypa2 that B111 E Cuts


      bhypa2 : that B111 E Cuts


      {move 2}

      >>> define lined78 bhypa2 : linec78 \
          bhypa2

[bhypa2 => linec78 bhypa2] is not well-formed

(paused, type something to continue) >

      >>> save


      {move 2}

      >>> close


   {move 1}

   >>> declare B112 obj


   B112 : obj


   {move 1}

   >>> declare bhypa3 that B112 E Cuts


   bhypa3 : that B112 E Cuts
```

```
   {move 1}

   >>> define linee78 Misset, thelawchooses, bhypa3 \
       : lined78 bhypa3

[Misset, thelawchooses, bhypa3 => lined78 bhypa3] is not well-formed

(paused, type something to continue) >

   >>> open


      {move 2}

      >>> define linead78 bhypa2 : linee78 \
          Misset, thelawchooses, bhypa2

[bhypa2 => linee78 Misset, thelawchooses, bhypa2] is not well-formed

(paused, type something to continue) >

      >>> open


         {move 3}

         >>> define lineac78 bhypa1 : linead78 \
             bhypa1

[bhypa1 => linead78 bhypa1] is not well-formed

(paused, type something to continue) >

         >>> open


            {move 4}
```

```
            >>> define lineab78 bhyp : lineac78 \
                bhyp

[bhyp => lineac78 bhyp] is not well-formed

(paused, type something to continue) >

            >>> open


               {move 5}

               >>> define line78 : lineab78 \
                   bhyp

lineab78 bhyp is not well-formed

(paused, type something to continue) >
end Lestrade execution
```

This is the third component of the proof that Cuts2 is a Θ-chain. I want to examine the proof strategy; I also want to see if the size of the term and the slowness of generation of the term can be improved by exporting some intermediate stages to move 0.

```
begin Lestrade execution

               >>> goal that Forall [D1 \
                      => Forall [F1 => ((D1 \
                         <<= Cuts2) & F1 E D1) -> \
                         (D1 Intersection F1) E Cuts2]]


               that Forall ([(D1 : obj) =>
                   ({def} Forall ([(F1
                      : obj) =>
```

```
      ({def} ((D1 <<= Cuts2) & F1
      E D1) -> (D1 Intersection
      F1) E Cuts2 : prop)]) : prop)])


{move 5}

>>> open


   {move 6}

   >>> declare D2 obj


   D2 : obj


   {move 6}

   >>> open


      {move 7}

      >>> declare F2 obj


      F2 : obj


      {move 7}

      >>> open


         {move 8}
```

```
>>> declare intev \
    that (D2 <<= Cuts2) & F2 \
    E D2
```

```
intev : that (D2
 <<= Cuts2) & F2
 E D2
```

```
{move 8}
```

```
>>> goal that (D2 \
    Intersection F2) E Cuts2
```

```
that (D2 Intersection
 F2) E Cuts2
```

```
{move 8}
```

```
>>> define line79 \
    : Ui D2 Intersection \
    F2, Separation4 \
    Refleq Cuts2
```

```
line79 : (D2 Intersection
 F2) Ui Separation4
 (Refleq (Cuts2))
```

```
line79 : that ((D2
 Intersection F2) E Mbold
 Set cutsi2) == ((D2
 Intersection F2) E Mbold) & cutsi2
 (D2 Intersection
```

```
 F2)


{move 7}

>>> goal that (D2 \
    Intersection F2) E Mbold


that (D2 Intersection
 F2) E Mbold


{move 8}

>>> define line80 \
    : Ui F2, Ui D2, Simp2 \
    (Simp2 (Simp2 Mboldtheta))


line80 : F2 Ui D2
 Ui Simp2 (Simp2
 (Simp2 (Mboldtheta)))


line80 : that ((D2
 <<= Misset Mbold2
 thelawchooses) & F2
 E D2) -> (D2 Intersection
 F2) E Misset Mbold2
 thelawchooses


{move 7}

>>> define line81 \
    intev : Mp (Conj \
    (Transsub (Simp1 \
```

```
    intev, line20), Simp2 \
    intev), line80)


line81 : [(intev_1
    : that (D2 <<=
    Cuts2) & F2 E D2) =>
    ({def} Simp1
    (intev_1) Transsub
    line20 Conj Simp2
    (intev_1) Mp
    line80 : that
    (D2 Intersection
    F2) E Misset
    Mbold2 thelawchooses)]


line81 : [(intev_1
    : that (D2 <<=
    Cuts2) & F2 E D2) =>
    (--- : that (D2
    Intersection F2) E Misset
    Mbold2 thelawchooses)]


{move 7}

>>> goal that ((D2 \
    Intersection F2) <<= \
    prime B) V B <<= \
    D2 Intersection F2


that ((D2 Intersection
 F2) <<= prime (B)) V B <<=
 D2 Intersection F2
```

```
{move 8}

>>> declare K obj


K : obj


{move 8}

>>> define line82 \
    : Excmid Forall [K => \
       (K E D2) -> \
       B <<= K]


line82 : [
    ({def} Excmid
    (Forall ([(K_3
       : obj) =>
       ({def} (K_3
       E D2) -> B <<=
       K_3 : prop)])) : that
    Forall ([(K_3
       : obj) =>
       ({def} (K_3
       E D2) -> B <<=
       K_3 : prop)]) V ~ (Forall
    ([(K_4 : obj) =>
       ({def} (K_4
       E D2) -> B <<=
       K_4 : prop)])))]


line82 : that Forall
 ([(K_3 : obj) =>
    ({def} (K_3
    E D2) -> B <<=
```

```
    K_3 : prop)]) V ~ (Forall
 ([(K_4 : obj) =>
    ({def} (K_4
    E D2) -> B <<=
    K_4 : prop)]))


{move 7}

>>> open


   {move 9}

   >>> goal that \
       ((D2 Intersection \
       F2) <<= prime \
       B) V B <<= D2 \
       Intersection F2


   that ((D2 Intersection
    F2) <<= prime
    (B)) V B <<=
    D2 Intersection
    F2


   {move 9}

   >>> declare K1 \
       obj


   K1 : obj


   {move 9}
```

```
>>> declare casehyp1 \
    that Forall [K1 \
       => (K1 E D2) -> \
       B <<= K1]


casehyp1 : that
 Forall ([(K1_2
    : obj) =>
    ({def} (K1_2
    E D2) -> B <<=
    K1_2 : prop)])


{move 9}

>>> goal that \
    B <<= D2 Intersection \
    F2


that B <<= D2
 Intersection F2


{move 9}

>>> open


   {move 10}

   >>> declare \
       K2 obj


   K2 : obj
```

```
{move 10}

>>> open


   {move 11}

   >>> declare \
       khyp that \
       K2 E B


   khyp : that
    K2 E B


   {move 11}

   >>> open


      {move
       12}

      >>> declare \
          B2 obj


      B2 : obj


      {move
       12}

      >>> open
```

```
{move
 13}

>>> \
    declare \
    bhyp2 \
    that \
    B2 \
    E D2


bhyp2
 : that
 B2
 E D2


{move
 13}

>>> \
    define \
    line83 \
    bhyp2 \
    : Mpsubs \
    (khyp, Mp \
    (bhyp2, Ui \
    B2, casehyp1))


line83
 : [(bhyp2_1
    : that
    B2
    E D2) =>
    ({def} khyp
    Mpsubs
```

```
        bhyp2_1
        Mp
        B2
        Ui
        casehyp1
        : that
        K2
        E B2)]


    line83
     : [(bhyp2_1
        : that
        B2
        E D2) =>
        (---
        : that
        K2
        E B2)]


    {move
     12}

    >>> \
        close


{move
 12}

>>> define \
    line84 \
    B2 : Ded \
    line83


line84
```

```
 : [(B2_1
    : obj) =>
    ({def} Ded
    ([(bhyp2_2
       : that
       B2_1
       E D2) =>
       ({def} khyp
       Mpsubs
       bhyp2_2
       Mp
       B2_1
       Ui
       casehyp1
       : that
       K2
       E B2_1)]) : that
    (B2_1
    E D2) ->
    K2
    E B2_1)]


line84
 : [(B2_1
    : obj) =>
    (---
    : that
    (B2_1
    E D2) ->
    K2
    E B2_1)]


{move
 11}

>>> close
```

```
{move 11}

>>> define \
    line85 khyp \
    : Ug line84


line85 : [(khyp_1
    : that
    K2 E B) =>
    ({def} Ug
    ([(B2_2
       : obj) =>
       ({def} Ded
       ([(bhyp2_3
          : that
          B2_2
          E D2) =>
          ({def} khyp_1
          Mpsubs
          bhyp2_3
          Mp
          B2_2
          Ui
          casehyp1
          : that
          K2
          E B2_2)]) : that
       (B2_2
       E D2) ->
       K2
       E B2_2)]) : that
    Forall
    ([(x'_2
       : obj) =>
       ({def} (x'_2
```

```
        E D2) ->
        K2
        E x'_2
        : prop)]))]


line85 : [(khyp_1
    : that
    K2 E B) =>
    (---
    : that
    Forall
    ([(x'_2
        : obj) =>
        ({def} (x'_2
        E D2) ->
        K2
        E x'_2
        : prop)]))]


{move 10}

>>> define \
    line86 khyp \
    : Mp (Simp2 \
    intev, Ui \
    F2, line85 \
    khyp)


line86 : [(khyp_1
    : that
    K2 E B) =>
    ({def} Simp2
    (intev) Mp
    F2 Ui
    line85
```

```
    (khyp_1) : that
    K2 E F2)]


line86 : [(khyp_1
    : that
    K2 E B) =>
    (---
    : that
    K2 E F2)]


{move 10}

>>> define \
    line87 khyp \
    : Fixform \
    (K2 E D2 \
    Intersection \
    F2, Iff2 \
    (Conj (line86 \
    khyp, line85 \
    khyp), Ui \
    K2, Separation4 \
    Refleq (D2 \
    Intersection \
    F2)))


line87 : [(khyp_1
    : that
    K2 E B) =>
    ({def} (K2
    E D2
    Intersection
    F2) Fixform
    line86
    (khyp_1) Conj
```

```
        line85
        (khyp_1) Iff2
        K2 Ui
        Separation4
        (Refleq
        (D2
        Intersection
        F2)) : that
        K2 E D2
        Intersection
        F2)]


    line87 : [(khyp_1
        : that
        K2 E B) =>
        (---
        : that
        K2 E D2
        Intersection
        F2)]


    {move 10}

    >>> close


{move 10}

>>> define \
    line88 K2 : Ded \
    line87


line88 : [(K2_1
    : obj) =>
    ({def} Ded
```

```
([(khyp_2
   : that
   K2_1
   E B) =>
   ({def} (K2_1
   E D2
   Intersection
   F2) Fixform
   Simp2
   (intev) Mp
   F2 Ui
   Ug ([(B2_8
      : obj) =>
      ({def} Ded
      ([(bhyp2_9
         : that
         B2_8
         E D2) =>
         ({def} khyp_2
         Mpsubs
         bhyp2_9
         Mp
         B2_8
         Ui
         casehyp1
         : that
         K2_1
         E B2_8)]) : that
      (B2_8
      E D2) ->
      K2_1
      E B2_8)]) Conj
   Ug ([(B2_6
      : obj) =>
      ({def} Ded
      ([(bhyp2_7
         : that
         B2_6
```

```
                  E D2) =>
                  ({def} khyp_2
                  Mpsubs
                  bhyp2_7
                  Mp
                  B2_6
                  Ui
                  casehyp1
                  : that
                  K2_1
                  E B2_6)]) : that
               (B2_6
               E D2) ->
               K2_1
               E B2_6)]) Iff2
            K2_1
            Ui Separation4
            (Refleq
            (D2
            Intersection
            F2)) : that
            K2_1
            E D2
            Intersection
            F2)]) : that
         (K2_1 E B) ->
         K2_1 E D2
         Intersection
         F2)]


line88 : [(K2_1
         : obj) =>
         (--- : that
         (K2_1 E B) ->
         K2_1 E D2
         Intersection
         F2)]
```

```
   {move 9}

   >>> close


{move 9}

>>> define line89 \
    casehyp1 : Fixform \
    (B <<= D2 Intersection \
    F2, Conj (Ug \
    line88, Conj \
    (linea14 bhyp, Separation3 \
    Refleq (D2 Intersection \
    F2))))


line89 : [(casehyp1_1
    : that Forall
    ([(K1_3
       : obj) =>
       ({def} (K1_3
       E D2) ->
       B <<= K1_3
       : prop)])) =>
    ({def} (B <<=
    D2 Intersection
    F2) Fixform
    Ug ([(K2_4
       : obj) =>
       ({def} Ded
       ([(khyp_5
          : that
          K2_4
          E B) =>
          ({def} (K2_4
```

```
E D2
Intersection
F2) Fixform
Simp2
(intev) Mp
F2 Ui
Ug ([(B2_11
   : obj) =>
   ({def} Ded
   ([(bhyp2_12
      : that
      B2_11
      E D2) =>
      ({def} khyp_5
      Mpsubs
      bhyp2_12
      Mp
      B2_11
      Ui
      casehyp1_1
      : that
      K2_4
      E B2_11)]) : that
   (B2_11
   E D2) ->
   K2_4
   E B2_11)]) Conj
Ug ([(B2_9
   : obj) =>
   ({def} Ded
   ([(bhyp2_10
      : that
      B2_9
      E D2) =>
      ({def} khyp_5
      Mpsubs
      bhyp2_10
      Mp
```

```
                  B2_9
                  Ui
                  casehyp1_1
                  : that
                  K2_4
                  E B2_9)]) : that
               (B2_9
               E D2) ->
               K2_4
               E B2_9)]) Iff2
            K2_4
            Ui Separation4
            (Refleq
            (D2
            Intersection
            F2)) : that
            K2_4
            E D2
            Intersection
            F2)]) : that
         (K2_4 E B) ->
         K2_4 E D2
         Intersection
         F2)]) Conj
      linea14 (bhyp) Conj
      Separation3
      (Refleq (D2
      Intersection
      F2)) : that
      B <<= D2 Intersection
      F2)]


   line89 : [(casehyp1_1
      : that Forall
      ([(K1_3
         : obj) =>
         ({def} (K1_3
```

```
       E D2) ->
       B <<= K1_3
       : prop)])) =>
    (--- : that
    B <<= D2 Intersection
    F2)]


{move 8}

>>> define line90 \
    casehyp1 : Add2 \
    ((D2 Intersection \
    F2) <<= prime \
    B, line89 casehyp1)


line90 : [(casehyp1_1
    : that Forall
    ([(K1_3
       : obj) =>
       ({def} (K1_3
       E D2) ->
       B <<= K1_3
       : prop)])) =>
    ({def} ((D2
    Intersection
    F2) <<= prime
    (B)) Add2
    line89 (casehyp1_1) : that
    ((D2 Intersection
    F2) <<= prime
    (B)) V B <<=
    D2 Intersection
    F2)]


line90 : [(casehyp1_1
```

```
    : that Forall
    ([(K1_3
       : obj) =>
       ({def} (K1_3
       E D2) ->
       B <<= K1_3
       : prop)])) =>
    (--- : that
    ((D2 Intersection
    F2) <<= prime
    (B)) V B <<=
    D2 Intersection
    F2)]


{move 8}

>>> declare casehyp2 \
    that ~ (Forall \
    [K1 => (K1 E D2) -> \
       B <<= K1])


casehyp2 : that
 ~ (Forall ([(K1_3
    : obj) =>
    ({def} (K1_3
    E D2) -> B <<=
    K1_3 : prop)]))


{move 9}

>>> goal that \
    ((D2 Intersection \
    F2) <<= prime \
    B)
```

```
that (D2 Intersection
 F2) <<= prime
 (B)


{move 9}

>>> open


   {move 10}

   >>> declare \
       K2 obj


   K2 : obj


   {move 10}

   >>> open


      {move 11}

      >>> declare \
          khyp2 that \
          K2 E D2 \
          Intersection \
          F2


      khyp2 : that
       K2 E D2
       Intersection
       F2
```

```
{move 11}

>>> define \
    line91 : Counterexample \
    casehyp2


line91 : [
    ({def} Counterexample
    (casehyp2) : that
    Exists
    ([(z_2
       : obj) =>
       ({def} ~ ((z_2
       E D2) ->
       B <<=
       z_2) : prop)]))]


line91 : that
 Exists ([(z_2
    : obj) =>
    ({def} ~ ((z_2
    E D2) ->
    B <<=
    z_2) : prop)])


{move 10}

>>> open


   {move
    12}
```

```
>>> declare \
    F3 obj


F3 : obj


{move
 12}

>>> declare \
    fhyp3 \
    that \
    Witnesses \
    line91 \
    F3


fhyp3
 : that
 line91
 Witnesses
 F3


{move
 12}

>>> define \
    line92 \
    fhyp3 \
    : Notimp2 \
    fhyp3


line92
 : [(.F3_1
    : obj), (fhyp3_1
```

```
    : that
    line91
    Witnesses
    .F3_1) =>
    ({def} Notimp2
    (fhyp3_1) : that
    .F3_1
    E D2)]


line92
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    .F3_1
    E D2)]


{move
 11}

>>> define \
    line93 \
    fhyp3 \
    : Notimp1 \
    fhyp3


line93
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
```

```
    Witnesses
    .F3_1) =>
    ({def} Notimp1
    (fhyp3_1) : that
    ~ (B <<=
    .F3_1))]


line93
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    ~ (B <<=
    .F3_1))]


{move
 11}

>>> define \
    line94 \
    fhyp3 \
    : Simp2 \
    (Iff1 \
    (Mpsubs \
    (line92 \
    fhyp3, Simp1 \
    intev), Ui \
    F3, Separation4 \
    Refleq \
    Cuts2))
```

```
line94
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    ({def} Simp2
    (line92
    (fhyp3_1) Mpsubs
    Simp1
    (intev) Iff1
    .F3_1
    Ui
    Separation4
    (Refleq
    (Cuts2))) : that
    cutsi2
    (.F3_1))]


line94
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    cutsi2
    (.F3_1))]


{move
 11}

>>> define \
```

```
    line95 \
    fhyp3 \
    : Ds1 \
    (line94 \
    fhyp3, line93 \
    fhyp3)


line95
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    ({def} line94
    (fhyp3_1) Ds1
    line93
    (fhyp3_1) : that
    .F3_1
    <<=
    prime2
    ([(S'_3
       : obj) =>
       ({def} thelaw
       (S'_3) : obj)], B))]


line95
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    .F3_1
```

```
    <<=
    prime2
    ([(S'_3
       : obj) =>
       ({def} thelaw
       (S'_3) : obj)], B))]


{move
 11}

>>> define \
    line96 \
    fhyp3 \
    : Mp \
    line92 \
    fhyp3, Ui \
    F3, Simp2 \
    (Iff1 \
    khyp2, Ui \
    K2, Separation4 \
    Refleq \
    (D2 \
    Intersection \
    F2))


line96
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    ({def} line92
    (fhyp3_1) Mp
    .F3_1
    Ui
```

```
    Simp2
    (khyp2
    Iff1
    K2
    Ui
    Separation4
    (Refleq
    (D2
    Intersection
    F2))) : that
    K2
    E .F3_1)]


line96
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    K2
    E .F3_1)]


{move
 11}

>>> define \
    line97 \
    fhyp3 \
    : Mpsubs \
    line96 \
    fhyp3 \
    line95 \
    fhyp3
```

```
line97
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    ({def} line96
    (fhyp3_1) Mpsubs
    line95
    (fhyp3_1) : that
    K2
    E prime2
    ([(S'_3
       : obj) =>
       ({def} thelaw
       (S'_3) : obj)], B))]
```

```
line97
 : [(.F3_1
    : obj), (fhyp3_1
    : that
    line91
    Witnesses
    .F3_1) =>
    (---
    : that
    K2
    E prime2
    ([(S'_3
       : obj) =>
       ({def} thelaw
       (S'_3) : obj)], B))]
```

```
   {move
    11}

   >>> close


{move 11}

>>> define \
    line98 khyp2 \
    : Eg line91 \
    line97


line98 : [(khyp2_1
    : that
    K2 E D2
    Intersection
    F2) =>
    ({def} line91
    Eg [(.F3_2
       : obj), (fhyp3_2
       : that
       line91
       Witnesses
       .F3_2) =>
       ({def} Notimp2
       (fhyp3_2) Mp
       .F3_2
       Ui
       Simp2
       (khyp2_1
       Iff1
       K2
       Ui
       Separation4
       (Refleq
       (D2
```

```
                Intersection
                F2))) Mpsubs
                Simp2
                (Notimp2
                (fhyp3_2) Mpsubs
                Simp1
                (intev) Iff1
                .F3_2
                Ui
                Separation4
                (Refleq
                (Cuts2))) Ds1
                Notimp1
                (fhyp3_2) : that
                K2
                E prime2
                ([(S'_4
                   : obj) =>
                   ({def} thelaw
                   (S'_4) : obj)], B))] : that
             K2 E prime2
             ([(S'_3
                : obj) =>
                ({def} thelaw
                (S'_3) : obj)], B))]


         line98 : [(khyp2_1
             : that
             K2 E D2
             Intersection
             F2) =>
             (---
             : that
             K2 E prime2
             ([(S'_3
                : obj) =>
                ({def} thelaw
```

```
        (S'_3) : obj)], B))]


   {move 10}

   >>> close


{move 10}

>>> define \
    line99 K2 : Ded \
    line98


line99 : [(K2_1
    : obj) =>
    ({def} Ded
    ([(khyp2_2
       : that
       K2_1
       E D2
       Intersection
       F2) =>
       ({def} Counterexample
       (casehyp2) Eg
       [(.F3_3
          : obj), (fhyp3_3
          : that
          Counterexample
          (casehyp2) Witnesses
          .F3_3) =>
          ({def} Notimp2
          (fhyp3_3) Mp
          .F3_3
          Ui
          Simp2
          (khyp2_2
```

```
            Iff1
            K2_1
            Ui
            Separation4
            (Refleq
            (D2
            Intersection
            F2))) Mpsubs
            Simp2
            (Notimp2
            (fhyp3_3) Mpsubs
            Simp1
            (intev) Iff1
            .F3_3
            Ui
            Separation4
            (Refleq
            (Cuts2))) Ds1
            Notimp1
            (fhyp3_3) : that
            K2_1
            E prime2
            ([(S'_5
               : obj) =>
               ({def} thelaw
               (S'_5) : obj)], B))] : that
         K2_1
         E prime2
         ([(S'_4
            : obj) =>
            ({def} thelaw
            (S'_4) : obj)], B))]) : that
      (K2_1 E D2
      Intersection
      F2) ->
      K2_1 E prime2
      ([(S'_4
         : obj) =>
```

```
            ({def} thelaw
            (S'_4) : obj)], B))]


   line99 : [(K2_1
       : obj) =>
       (--- : that
       (K2_1 E D2
       Intersection
       F2) ->
       K2_1 E prime2
       ([(S'_4
          : obj) =>
          ({def} thelaw
          (S'_4) : obj)], B))]


   {move 9}

   >>> close


{move 9}

>>> define line10 \
    casehyp2 : Fixform \
    ((D2 Intersection \
    F2) <<= prime \
    B, Conj (Ug \
    line99, Conj \
    (Separation3 \
    Refleq (D2 Intersection \
    F2), Separation3 \
    Refleq (prime \
    B))))

line10 is badly formed or already reserved or declared
```

```
(paused, type something to continue) >

                           >>> define line11 \
                               casehyp2 : Add1 \
                               (B <<= D2 Intersection \
                               F2, line10 casehyp2)

line11 is badly formed or already reserved or declared

(paused, type something to continue) >

                           >>> close


                        {move 8}

                        >>> define line12 \
                            intev : Cases line82 \
                            line90, line11

Failure in comparing that Mbold <<= Sc(M) to [(qq_7 : that ~(Forall([(K_10 : ob

(paused, type something to continue) >
Object type error in Cases(line82, line90, line11)

(paused, type something to continue) >
general failure of objectsort line 2989

(paused, type something to continue) >
theobjectsort inapplicable line 3077

(paused, type something to continue) >
failure to compute object sort in fixtypes

(paused, type something to continue) >
implicitarglist failure line 1905

(paused, type something to continue) >
```

```
Parse or typefix error in[(intev : that (D2 <<= Cuts2) & F2 E D2) => Cases()]

(paused, type something to continue) >

                        >>> define linea12 \
                            intev : Conj (line81 \
                            intev, line12 intev)

[intev => Conj (line81 intev, line12 intev)] is not well-formed

(paused, type something to continue) >

                        >>> define lineb12 \
                            intev : Fixform ((D2 \
                            Intersection F2) E Cuts2, Iff2 \
                            (linea12 intev, Ui \
                            (D2 Intersection \
                            F2, Separation4 \
                            Refleq Cuts2)))

[intev => Fixform ((D2 Intersection F2) E Cuts2, Iff2 (linea12 intev, Ui (D2 In

(paused, type something to continue) >

                        >>> close


                     {move 7}

                     >>> define line13 F2 \
                         : Ded lineb12

line13 is badly formed or already reserved or declared

(paused, type something to continue) >

                     >>> close
```

```
                  {move 6}

                  >>> define line14 D2 : Ug \
                      line13

line14 is badly formed or already reserved or declared

(paused, type something to continue) >

                  >>> close


               {move 5}

               >>> define line15 : Ug line14

line15 is badly formed or already reserved or declared

(paused, type something to continue) >
end Lestrade execution
```

This is the fourth component of the proof that `Cuts` is a Θ-chain. I wonder whether this has common features with the fourth component of the larger proof which can be used to shorten the file. This also might be worth exporting to move 0.

```
begin Lestrade execution

               >>> close


            {move 4}

            >>> define line17 bhyp : Fixform \
                (thetachain Cuts2, Conj (line19, Conj \
                (line21, Conj (line78, line15))))
```

```
line17 is badly formed or already reserved or declared

(paused, type something to continue) >

            >>> save


            {move 4}

            >>> close


         {move 3}

         >>> declare bhyp10 that B E Cuts


         bhyp10 : that B E Cuts


         {move 3}

         >>> define linea17 bhyp10 : line17 \
             bhyp10

[bhyp10 => line17 bhyp10] is not well-formed

(paused, type something to continue) >

         >>> save


         {move 3}

         >>> close
```

```
      {move 2}

      >>> declare B11 obj


      B11 : obj


      {move 2}

      >>> declare bhyp11 that B11 E Cuts


      bhyp11 : that B11 E Cuts


      {move 2}

      >>> define lineb17 bhyp11 : linea17 \
          bhyp11

[bhyp11 => linea17 bhyp11] is not well-formed

(paused, type something to continue) >

      >>> save


      {move 2}

      >>> close


   {move 1}

   >>> declare B12 obj
```

```
   B12 : obj


   {move 1}

   >>> declare bhyp12 that B12 E Cuts


   bhyp12 : that B12 E Cuts


   {move 1}

   >>> define linec17 bhyp12 : lineb17 bhyp12

[bhyp12 => lineb17 bhyp12] is not well-formed

(paused, type something to continue) >

   >>> open


      {move 2}

      >>> define lined17 bhyp11 : linec17 \
          bhyp11

[bhyp11 => linec17 bhyp11] is not well-formed

(paused, type something to continue) >

      >>> open


         {move 3}

         >>> declare B13 obj
```

```
         B13 : obj


         {move 3}

         >>> declare bhyp13 that B13 E Cuts


         bhyp13 : that B13 E Cuts


         {move 3}

         >>> define linee17 bhyp13 : lined17 \
             bhyp13

[bhyp13 => lined17 bhyp13] is not well-formed

(paused, type something to continue) >

         >>> open


            {move 4}

            >>> define Line17 bhyp : linee17 \
                bhyp

[bhyp => linee17 bhyp] is not well-formed

(paused, type something to continue) >

            >>> open


               {move 5}
```

```
>>> declare K obj


K : obj


{move 5}

>>> open


   {move 6}

   >>> declare khyp that K E Mbold


   khyp : that K E Mbold


   {move 6}

   >>> define line18 khyp \
       : Ui Cuts2, Simp2 (Iff1 \
       (khyp, Ui K, Separation4 \
       Refleq Mbold))

line18 is badly formed or already reserved or declared

(paused, type something to continue) >

   >>> define linea18 : Iff2 \
       (Simp1 (Simp2 Line17 \
       bhyp), Ui Cuts2, Scthm \
       (Sc M))

Iff2 (Simp1 (Simp2 Line17 bhyp), Ui Cuts2, Scthm (Sc M)) is not well-formed

(paused, type something to continue) >
```

```
>>> define line19 : Fixform \
    (Cuts2 E Thetachain, Iff2 \
    (Conj (linea18, Line17 \
    bhyp), Ui Cuts2, Separation4 \
    Refleq Thetachain))

line19 is badly formed or already reserved or declared

(paused, type something to continue) >
end Lestrade execution
```

Here we have line 107 to the effect that Cuts2 is a Θ-chain and line 109 to the effect that it belongs to the set of Θ-chains.

```
begin Lestrade execution

>>> define line110 khyp \
    : Mp (line19, line18 \
    khyp)

[khyp => Mp (line19, line18 khyp)] is not well-formed

(paused, type something to continue) >

>>> define line111 khyp \
    : Iff1 (line110 khyp, Ui \
    K, Separation4 Refleq \
    Cuts2)

[khyp => Iff1 (line110 khyp, Ui K, Separation4 Refleq Cuts2)] is not well-forme

(paused, type something to continue) >

>>> define line112 : Fixform \
    ((prime B) <<= B, Sepsub2 \
    (linea14 bhyp, Refleq \
```

```
    prime B))


line112 : [
    ({def} (prime (B) <<=
    B) Fixform linea14
    (bhyp) Sepsub2 Refleq
    (prime (B)) : that
    prime (B) <<= B)]


line112 : that prime (B) <<=
 B


{move 5}

>>> define line113 khyp \
    : Simp2 line111 khyp
```

```
[khyp => Simp2 line111 khyp] is not well-formed

(paused, type something to continue) >
```

```
>>> open


   {move 7}

   >>> declare casehyp1 \
       that K <<= prime B


   casehyp1 : that K <<=
    prime (B)


   {move 7}
```

```
>>> declare casehyp2 \
    that B <<= K


casehyp2 : that B <<=
 K


{move 7}

>>> define case1 casehyp1 \
    : Add1 ((prime B) <<= \
    K, casehyp1)


case1 : [(casehyp1_1
    : that K <<= prime
    (B)) =>
    ({def} (prime (B) <<=
    K) Add1 casehyp1_1
    : that (K <<= prime
    (B)) V prime (B) <<=
    K)]


case1 : [(casehyp1_1
    : that K <<= prime
    (B)) => (---
    : that (K <<= prime
    (B)) V prime (B) <<=
    K)]


{move 6}

>>> define case2 casehyp2 \
    : Add2 (K <<= prime \
```

```
        B, Transsub line112, casehyp2)


    case2 : [(casehyp2_1
        : that B <<= K) =>
        ({def} (K <<= prime
        (B)) Add2 line112
        Transsub casehyp2_1
        : that (K <<= prime
        (B)) V prime (B) <<=
        K)]


    case2 : [(casehyp2_1
        : that B <<= K) =>
        (--- : that (K <<=
        prime (B)) V prime
        (B) <<= K)]


    {move 6}

    >>> close


   {move 6}

   >>> define line114 khyp \
       : Cases (line113 khyp, case1, case2)

[khyp => Cases (line113 khyp, case1, case2)] is not well-formed

(paused, type something to continue) >

   >>> close


  {move 5}
```

```
              >>> define line115 K : Ded \
                  line114

[K => Ded line114] is not well-formed

(paused, type something to continue) >

              >>> close


           {move 4}

           >>> define line116 bhyp : Ug \
               line115

[bhyp => Ug line115] is not well-formed

(paused, type something to continue) >

           >>> define linea116 bhyp : Mp \
               (line14 bhyp, Ui B, Simp1 \
               Simp2 Simp2 Mboldtheta)


           linea116 : [(bhyp_1 : that
               B E Cuts) =>
               ({def} line14 (bhyp_1) Mp
               B Ui Simp1 (Simp2 (Simp2
               (Mboldtheta))) : that
               prime2 ([(S'_3 : obj) =>
                  ({def} thelaw (S'_3) : obj)], B) E Misset
               Mbold2 thelawchooses)]


           linea116 : [(bhyp_1 : that
               B E Cuts) => (--- : that
               prime2 ([(S'_3 : obj) =>
```

```
          ({def} thelaw (S'_3) : obj)], B) E Misset
        Mbold2 thelawchooses)]


      {move 3}

      >>> define line117 bhyp : Fixform \
          ((prime B) E Cuts, Iff2 (Conj \
          (linea116 bhyp, Conj (linea116 \
          bhyp, line116 bhyp)), Ui \
          (prime B, Separation4 Refleq \
          Cuts)))

[bhyp => Fixform ((prime B) E Cuts, Iff2 (Conj (linea116 bhyp, Conj (linea116 b

(paused, type something to continue) >

      >>> close


    {move 3}

    >>> define line118 B : Ded line117

[B => Ded line117] is not well-formed

(paused, type something to continue) >

    >>> close


  {move 2}

  >>> define Linea119 : Ug line118

Ug line118 is not well-formed

(paused, type something to continue) >
```

```
      >>> close


   {move 1}

   >>> define Lineb119 Misset thelawchooses \
       : Linea119

[Misset thelawchooses => Linea119] is not well-formed

(paused, type something to continue) >

   >>> open


      {move 2}

      >>> define Line119 : Lineb119 Misset \
          thelawchooses

Lineb119 Misset thelawchooses is not well-formed

(paused, type something to continue) >
end Lestrade execution
```

This is the third component of the proof that `Cuts` is a Θ-chain, proved with the aid of the result that `Cuts2` is a Θ-chain (and so coincides with **M**).

```
begin Lestrade execution

      >>> declare D3 obj


      D3 : obj
```

```
{move 2}

>>> declare F3 obj


F3 : obj


{move 2}

>>> goal that Forall [D3 => [F3 => \
          ((D3 <<= Cuts) & F3 E D3) -> \
          (D3 Intersection F3) E Cuts]]


{error type}


{move 2}

>>> open


   {move 3}

   >>> declare D4 obj


   D4 : obj


   {move 3}

   >>> open


      {move 4}
```

```
>>> declare dhyp4 that D4 <<= \
    Cuts


dhyp4 : that D4 <<= Cuts


{move 4}

>>> open


   {move 5}

   >>> declare F4 obj


   F4 : obj


   {move 5}

   >>> open


      {move 6}

      >>> declare fhyp4 that \
          F4 E D4


      fhyp4 : that F4 E D4


      {move 6}

      >>> test Ui (D4 Intersection \
          F4, Separation4 Refleq \
```

```
                    Cuts)

{function error}

general failure of functionsort line 3030

(paused, type something to continue) >

                  {move 6}

                  >>> goal that D4 Intersection \
                      F4 E Mbold

Failure in comparing prop to obj line 3073

(paused, type something to continue) >
Object type error in D4 Intersection F4 E Mbold

(paused, type something to continue) >
general failure of objectsort line 2989

(paused, type something to continue) >
bad proof/evidence type, body not prop line 3913

(paused, type something to continue) >

                  {error type}


                  {move 6}

                  >>> test Fixform (Cuts \
                      <<= Mbold, Sepsub2 (Separation3 \
                      Refleq Mbold, Refleq Cuts))

{function error}

general failure of functionsort line 3030
```

```
(paused, type something to continue) >

                  {move 6}

                  >>> define line120 : Transsub \
                      (dhyp4, Fixform (Cuts \
                      <<= Mbold, Sepsub2 (Separation3 \
                      Refleq Mbold, Refleq Cuts)))


                  line120 : [
                      ({def} dhyp4 Transsub
                      (Cuts <<= Mbold) Fixform
                      Separation3 (Refleq
                      (Mbold)) Sepsub2
                      Refleq (Cuts) : that
                      D4 <<= Mbold)]


                  line120 : that D4 <<= Mbold


                  {move 5}

                  >>> define line121 fhyp4 \
                      : Mpsubs fhyp4 line120


                  line121 : [(fhyp4_1 : that
                      F4 E D4) =>
                      ({def} fhyp4_1 Mpsubs
                      line120 : that F4 E Mbold)]


                  line121 : [(fhyp4_1 : that
                      F4 E D4) => (--- : that
                      F4 E Mbold)]
```

```
{move 5}

>>> define line122 fhyp4 \
    : Mp (line120 Conj fhyp4, Ui \
    F4, Ui D4, Simp2 Simp2 \
    Simp2 Mboldtheta)


line122 : [(fhyp4_1 : that
    F4 E D4) =>
    ({def} line120 Conj
    fhyp4_1 Mp F4 Ui D4
    Ui Simp2 (Simp2 (Simp2
    (Mboldtheta))) : that
    (D4 Intersection F4) E Misset
    Mbold2 thelawchooses)]


line122 : [(fhyp4_1 : that
    F4 E D4) => (--- : that
    (D4 Intersection F4) E Misset
    Mbold2 thelawchooses)]


{move 5}

>>> goal that cuts (D4 \
    Intersection F4)


that cuts (D4 Intersection
 F4)


{move 6}
```

```
                   >>> declare testing that \
                       cuts (D4 Intersection \
                       F4)


                   testing : that cuts (D4
                    Intersection F4)


                   {move 6}

                   >>> test Simp1 (testing)

{function error}

general failure of functionsort line 3030

(paused, type something to continue) >

                   {move 6}

                   >>> test Simp2 (testing)

{function error}

general failure of functionsort line 3030

(paused, type something to continue) >

                   {move 6}

                   >>> open


                      {move 7}

                      >>> declare D5 obj
```

```
D5 : obj


{move 7}

>>> open


   {move 8}

   >>> declare dhyp5 \
       that D5 E Mbold


   dhyp5 : that D5 E Mbold


   {move 8}

   >>> goal that (D5 \
       <<= D4 Intersection \
       F4) V (D4 Intersection \
       F4) <<= D5


   that (D5 <<= D4
    Intersection F4) V (D4
    Intersection F4) <<=
    D5


   {move 8}

   >>> declare D6 obj


   D6 : obj
```

```
{move 8}

>>> define line123 \
    : Excmid (Forall \
    [D6 => (D6 E D4) -> \
       D5 <<= D6])


line123 : [
    ({def} Excmid
    (Forall ([(D6_3
       : obj) =>
       ({def} (D6_3
       E D4) -> D5
       <<= D6_3 : prop)])) : that
    Forall ([(D6_3
       : obj) =>
       ({def} (D6_3
       E D4) -> D5
       <<= D6_3 : prop)]) V ~ (Forall
    ([(D6_4 : obj) =>
       ({def} (D6_4
       E D4) -> D5
       <<= D6_4 : prop)])))]


line123 : that Forall
 ([(D6_3 : obj) =>
    ({def} (D6_3
    E D4) -> D5 <<=
    D6_3 : prop)]) V ~ (Forall
 ([(D6_4 : obj) =>
    ({def} (D6_4
    E D4) -> D5 <<=
    D6_4 : prop)]))
```

```
{move 7}

>>> open


   {move 9}

   >>> declare D7 \
       obj


   D7 : obj


   {move 9}

   >>> declare casehyp1 \
       that Forall [D7 \
          => (D7 E D4) -> \
          D5 <<= D7]


   casehyp1 : that
    Forall ([(D7_2
       : obj) =>
       ({def} (D7_2
       E D4) -> D5
       <<= D7_2 : prop)])


   {move 9}

   >>> open


      {move 10}
```

```
>>> declare \
    G obj


G : obj


{move 10}

>>> open


   {move 11}

   >>> declare \
       ghyp that \
       G E D5


   ghyp : that
    G E D5


   {move 11}

   >>> goal \
       that G E D4 \
       Intersection \
       F4


   that G E D4
    Intersection
    F4


   {move 11}
```

```
>>> test \
    Ui G, Separation4 \
    Refleq (D4 \
    Intersection \
    F4)
```

```
{function error}

general failure of functionsort line 3030

(paused, type something to continue) >
```

```
{move 11}

>>> open


   {move
    12}

   >>> declare \
       B1 obj


   B1 : obj


   {move
    12}

   >>> open


      {move
       13}

      >>> \
          declare \
```

```
    bhyp1 \
    that \
    B1 \
    E D4


bhyp1
 : that
 B1
 E D4


{move
 13}

>>> \
    goal \
    that \
    G E B1


that
 G E B1


{move
 13}

>>> \
    define \
    line124 \
    bhyp1 \
    : Mpsubs \
    ghyp, Mp \
    bhyp1, Ui \
    B1 \
    casehyp1
```

```
   line124
    : [(bhyp1_1
       : that
       B1
       E D4) =>
       ({def} ghyp
       Mpsubs
       bhyp1_1
       Mp
       B1
       Ui
       casehyp1
       : that
       G E B1)]


   line124
    : [(bhyp1_1
       : that
       B1
       E D4) =>
       (---
       : that
       G E B1)]


   {move
    12}

   >>> \
       close


{move
 12}

>>> define \
```

```
    line125 \
    B1 : Ded \
    line124


line125
 : [(B1_1
    : obj) =>
    ({def} Ded
    ([(bhyp1_2
       : that
       B1_1
       E D4) =>
       ({def} ghyp
       Mpsubs
       bhyp1_2
       Mp
       B1_1
       Ui
       casehyp1
       : that
       G E B1_1)]) : that
    (B1_1
    E D4) ->
    G E B1_1)]


line125
 : [(B1_1
    : obj) =>
    (---
    : that
    (B1_1
    E D4) ->
    G E B1_1)]


{move
```

```
    11}

   >>> close


{move 11}

>>> define \
    line126 \
    ghyp : Ug \
    line125


line126
 : [(ghyp_1
    : that
    G E D5) =>
    ({def} Ug
    ([(B1_2
       : obj) =>
       ({def} Ded
       ([(bhyp1_3
          : that
          B1_2
          E D4) =>
          ({def} ghyp_1
          Mpsubs
          bhyp1_3
          Mp
          B1_2
          Ui
          casehyp1
          : that
          G E B1_2)]) : that
       (B1_2
       E D4) ->
       G E B1_2)]) : that
    Forall
```

```
    ([(x'_2
       : obj) =>
       ({def} (x'_2
       E D4) ->
       G E x'_2
       : prop)]))]


line126
 : [(ghyp_1
    : that
    G E D5) =>
    (---
    : that
    Forall
    ([(x'_2
       : obj) =>
       ({def} (x'_2
       E D4) ->
       G E x'_2
       : prop)]))]


{move 10}

>>> define \
    line127 \
    ghyp : Mp \
    fhyp4, Ui \
    F4, line126 \
    ghyp


line127
 : [(ghyp_1
    : that
    G E D5) =>
    ({def} fhyp4
```

```
    Mp F4
    Ui line126
    (ghyp_1) : that
    G E F4)]


line127
 : [(ghyp_1
    : that
    G E D5) =>
    (---
    : that
    G E F4)]


{move 10}

>>> define \
    line128 \
    ghyp : Conj \
    (line127 \
    ghyp, line126 \
    ghyp)


line128
 : [(ghyp_1
    : that
    G E D5) =>
    ({def} line127
    (ghyp_1) Conj
    line126
    (ghyp_1) : that
    (G E F4) & Forall
    ([(x'_3
       : obj) =>
       ({def} (x'_3
       E D4) ->
```

```
       G E x'_3
       : prop)]))]


line128
 : [(ghyp_1
    : that
    G E D5) =>
    (---
    : that
    (G E F4) & Forall
    ([(x'_3
       : obj) =>
       ({def} (x'_3
       E D4) ->
       G E x'_3
       : prop)]))]


{move 10}

>>> define \
    line129 \
    ghyp : Fixform \
    (G E D4 \
    Intersection \
    F4, Iff2 \
    (line128 \
    ghyp, Ui \
    G, Separation4 \
    Refleq (D4 \
    Intersection \
    F4)))


line129
 : [(ghyp_1
    : that
```

```
        G E D5) =>
        ({def} (G E D4
        Intersection
        F4) Fixform
        line128
        (ghyp_1) Iff2
        G Ui
        Separation4
        (Refleq
        (D4
        Intersection
        F4)) : that
        G E D4
        Intersection
        F4)]


    line129
     : [(ghyp_1
        : that
        G E D5) =>
        (---
        : that
        G E D4
        Intersection
        F4)]


    {move 10}

    >>> close


{move 10}

>>> define \
    line130 G : Ded \
    line129
```

```
line130 : [(G_1
    : obj) =>
    ({def} Ded
    ([(ghyp_2
       : that
       G_1 E D5) =>
       ({def} (G_1
       E D4
       Intersection
       F4) Fixform
       fhyp4
       Mp F4
       Ui Ug
       ([(B1_8
          : obj) =>
          ({def} Ded
          ([(bhyp1_9
             : that
             B1_8
             E D4) =>
             ({def} ghyp_2
             Mpsubs
             bhyp1_9
             Mp
             B1_8
             Ui
             casehyp1
             : that
             G_1
             E B1_8)]) : that
          (B1_8
          E D4) ->
          G_1
          E B1_8)]) Conj
       Ug ([(B1_6
          : obj) =>
```

```
      ({def} Ded
      ([(bhyp1_7
         : that
         B1_6
         E D4) =>
         ({def} ghyp_2
         Mpsubs
         bhyp1_7
         Mp
         B1_6
         Ui
         casehyp1
         : that
         G_1
         E B1_6)]) : that
      (B1_6
      E D4) ->
      G_1
      E B1_6)]) Iff2
   G_1 Ui
   Separation4
   (Refleq
   (D4
   Intersection
   F4)) : that
   G_1 E D4
   Intersection
   F4)]) : that
(G_1 E D5) ->
G_1 E D4
Intersection
F4)]


line130 : [(G_1
    : obj) =>
    (--- : that
    (G_1 E D5) ->
```

```
        G_1 E D4
        Intersection
        F4)]


    {move 9}

    >>> close


{move 9}

>>> define line131 \
    casehyp1 : Fixform \
    (D5 <<= D4 Intersection \
    F4, Conj (Ug \
    line130, Conj \
    (Setsinchains \
    Mboldtheta, dhyp5, Separation3 \
    Refleq (D4 Intersection \
    F4))))


line131 : [(casehyp1_1
    : that Forall
    ([(D7_3
       : obj) =>
       ({def} (D7_3
       E D4) ->
       D5 <<= D7_3
       : prop)])) =>
    ({def} (D5
    <<= D4 Intersection
    F4) Fixform
    Ug ([(G_4
       : obj) =>
       ({def} Ded
       ([(ghyp_5
```

```
: that
G_4 E D5) =>
({def} (G_4
E D4
Intersection
F4) Fixform
fhyp4
Mp F4
Ui Ug
([(B1_11
   : obj) =>
   ({def} Ded
   ([(bhyp1_12
      : that
      B1_11
      E D4) =>
      ({def} ghyp_5
      Mpsubs
      bhyp1_12
      Mp
      B1_11
      Ui
      casehyp1_1
      : that
      G_4
      E B1_11)]) : that
   (B1_11
   E D4) ->
   G_4
   E B1_11)]) Conj
Ug ([(B1_9
   : obj) =>
   ({def} Ded
   ([(bhyp1_10
      : that
      B1_9
      E D4) =>
      ({def} ghyp_5
```

```
                    Mpsubs
                    bhyp1_10
                    Mp
                    B1_9
                    Ui
                    casehyp1_1
                    : that
                    G_4
                    E B1_9)]) : that
                 (B1_9
                 E D4) ->
                 G_4
                 E B1_9)]) Iff2
              G_4 Ui
              Separation4
              (Refleq
              (D4
              Intersection
              F4)) : that
              G_4 E D4
              Intersection
              F4)]) : that
           (G_4 E D5) ->
           G_4 E D4
           Intersection
           F4)]) Conj
        Mboldtheta
        Setsinchains
        dhyp5 Conj
        Separation3
        (Refleq (D4
        Intersection
        F4)) : that
        D5 <<= D4 Intersection
        F4)]


    line131 : [(casehyp1_1
```

```
    : that Forall
    ([(D7_3
       : obj) =>
       ({def} (D7_3
       E D4) ->
       D5 <<= D7_3
       : prop)])) =>
    (--- : that
    D5 <<= D4 Intersection
    F4)]


{move 8}

>>> define line132 \
    casehyp1 : Add1 \
    ((D4 Intersection \
    F4) <<= D5, line131 \
    casehyp1)


line132 : [(casehyp1_1
    : that Forall
    ([(D7_3
       : obj) =>
       ({def} (D7_3
       E D4) ->
       D5 <<= D7_3
       : prop)])) =>
    ({def} ((D4
    Intersection
    F4) <<= D5) Add1
    line131 (casehyp1_1) : that
    (D5 <<= D4
    Intersection
    F4) V (D4
    Intersection
    F4) <<= D5)]
```

```
line132 : [(casehyp1_1
    : that Forall
    ([(D7_3
       : obj) =>
       ({def} (D7_3
       E D4) ->
       D5 <<= D7_3
       : prop)])) =>
    (--- : that
    (D5 <<= D4
    Intersection
    F4) V (D4
    Intersection
    F4) <<= D5)]


{move 8}

>>> declare casehyp2 \
    that ~ (Forall \
    [D7 => (D7 E D4) -> \
       D5 <<= D7])


casehyp2 : that
 ~ (Forall ([(D7_3
    : obj) =>
    ({def} (D7_3
    E D4) -> D5
    <<= D7_3 : prop)]))


{move 9}

>>> open
```

```
{move 10}

>>> declare \
    G obj


G : obj


{move 10}

>>> open


   {move 11}

   >>> declare \
       ghyp that \
       G E D4 Intersection \
       F4


   ghyp : that
    G E D4 Intersection
    F4


   {move 11}

   >>> goal \
       that G E D5


   that G E D5


   {move 11}
```

```
>>> define \
    line133 \
    : Counterexample \
    casehyp2


line133
 : [
    ({def} Counterexample
    (casehyp2) : that
    Exists
    ([(z_2
       : obj) =>
       ({def} ~ ((z_2
       E D4) ->
       D5
       <<=
       z_2) : prop)]))]


line133
 : that Exists
 ([(z_2
    : obj) =>
    ({def} ~ ((z_2
    E D4) ->
    D5 <<=
    z_2) : prop)])


{move 10}

>>> open


   {move
    12}
```

```
>>> declare \
    H obj


H : obj


{move
 12}

>>> declare \
    hhyp \
    that \
    Witnesses \
    line133 \
    H


hhyp
 : that
 line133
 Witnesses
 H


{move
 12}

>>> define \
    line134 \
    hhyp \
    : Notimp1 \
    hhyp


line134
 : [(.H_1
```

```
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} Notimp1
    (hhyp_1) : that
    ~ (D5
    <<=
    .H_1))]


line134
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    ~ (D5
    <<=
    .H_1))]


{move
 11}

>>> define \
    line135 \
    hhyp \
    : Notimp2 \
    hhyp


line135
 : [(.H_1
```

```
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} Notimp2
    (hhyp_1) : that
    .H_1
    E D4)]


line135
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    .H_1
    E D4)]


{move
 11}

>>> define \
    line136 \
    hhyp \
    : Mp \
    line135 \
    hhyp, Ui \
    H, Simp2 \
    (Iff1 \
    (ghyp, Ui \
    G, Separation4 \
    Refleq \
```

```
    (D4 \
    Intersection \
    F4)))


line136
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} line135
    (hhyp_1) Mp
    .H_1
    Ui
    Simp2
    (ghyp
    Iff1
    G Ui
    Separation4
    (Refleq
    (D4
    Intersection
    F4))) : that
    G E .H_1)]


line136
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    G E .H_1)]
```

```
{move
 11}

>>> define \
    line137 \
    hhyp \
    : Mpsubs \
    line135 \
    hhyp, dhyp4


line137
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} line135
    (hhyp_1) Mpsubs
    dhyp4
    : that
    .H_1
    E Cuts)]


line137
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    .H_1
```

```
    E Cuts)]


{move
 11}

>>> define \
    line138 \
    hhyp \
    : Mp \
    dhyp5, Ui \
    D5, Simp2 \
    (Simp2 \
    (Iff1 \
    (line137 \
    hhyp, Ui \
    H, Separation4 \
    Refleq \
    Cuts)))


line138
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} dhyp5
    Mp
    D5
    Ui
    Simp2
    (Simp2
    (line137
    (hhyp_1) Iff1
    .H_1
    Ui
```

```
    Separation4
    (Refleq
    (Cuts)))) : that
    (D5
    <<=
    .H_1) V .H_1
    <<=
    D5)]


line138
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    (D5
    <<=
    .H_1) V .H_1
    <<=
    D5)]


{move
 11}

>>> define \
    line139 \
    hhyp \
    : Ds2 \
    (line138 \
    hhyp, line134 \
    hhyp)
```

```
line139
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    ({def} line138
    (hhyp_1) Ds2
    line134
    (hhyp_1) : that
    .H_1
    <<=
    D5)]


line139
 : [(.H_1
    : obj), (hhyp_1
    : that
    line133
    Witnesses
    .H_1) =>
    (---
    : that
    .H_1
    <<=
    D5)]


{move
 11}

>>> define \
    line140 \
    hhyp \
    : Mpsubs \
    (line136 \
```

```
        hhyp, line139 \
        hhyp)


    line140
     : [(.H_1
        : obj), (hhyp_1
        : that
        line133
        Witnesses
        .H_1) =>
        ({def} line136
        (hhyp_1) Mpsubs
        line139
        (hhyp_1) : that
        G E D5)]


    line140
     : [(.H_1
        : obj), (hhyp_1
        : that
        line133
        Witnesses
        .H_1) =>
        (---
        : that
        G E D5)]


    {move
     11}

    >>> close


{move 11}
```

```
>>> define \
    line141 \
    ghyp : Eg \
    line133 \
    line140


line141
 : [(ghyp_1
    : that
    G E D4
    Intersection
    F4) =>
    ({def} line133
    Eg [(.H_2
       : obj), (hhyp_2
       : that
       line133
       Witnesses
       .H_2) =>
       ({def} Notimp2
       (hhyp_2) Mp
       .H_2
       Ui
       Simp2
       (ghyp_1
       Iff1
       G Ui
       Separation4
       (Refleq
       (D4
       Intersection
       F4))) Mpsubs
       dhyp5
       Mp
       D5
       Ui
       Simp2
```

```
                (Simp2
                (Notimp2
                (hhyp_2) Mpsubs
                dhyp4
                Iff1
                .H_2
                Ui
                Separation4
                (Refleq
                (Cuts)))) Ds2
                Notimp1
                (hhyp_2) : that
                G E D5)] : that
             G E D5)]


   line141
    : [(ghyp_1
       : that
       G E D4
       Intersection
       F4) =>
       (---
       : that
       G E D5)]


   {move 10}

   >>> close


{move 10}

>>> define \
    line142 G : Ded \
    line141
```

```
line142 : [(G_1
    : obj) =>
    ({def} Ded
    ([(ghyp_2
       : that
       G_1 E D4
       Intersection
       F4) =>
       ({def} Counterexample
       (casehyp2) Eg
       [(.H_3
          : obj), (hhyp_3
          : that
          Counterexample
          (casehyp2) Witnesses
          .H_3) =>
          ({def} Notimp2
          (hhyp_3) Mp
          .H_3
          Ui
          Simp2
          (ghyp_2
          Iff1
          G_1
          Ui
          Separation4
          (Refleq
          (D4
          Intersection
          F4))) Mpsubs
          dhyp5
          Mp
          D5
          Ui
          Simp2
          (Simp2
          (Notimp2
```

```
                    (hhyp_3) Mpsubs
                    dhyp4
                    Iff1
                    .H_3
                    Ui
                    Separation4
                    (Refleq
                    (Cuts)))) Ds2
                    Notimp1
                    (hhyp_3) : that
                    G_1
                    E D5)] : that
                 G_1 E D5)]) : that
              (G_1 E D4
              Intersection
              F4) ->
              G_1 E D5)]


   line142 : [(G_1
              : obj) =>
              (--- : that
              (G_1 E D4
              Intersection
              F4) ->
              G_1 E D5)]


   {move 9}

   >>> close


{move 9}

>>> define line143 \
    casehyp2 : Fixform \
    ((D4 Intersection \
```

```
    F4) <<= D5, Conj \
    (Ug line142, Conj \
    (Separation3 \
    Refleq (D4 Intersection \
    F4), Setsinchains \
    Mboldtheta, dhyp5)))


line143 : [(casehyp2_1
    : that ~ (Forall
    ([(D7_4
       : obj) =>
       ({def} (D7_4
       E D4) ->
       D5 <<= D7_4
       : prop)]))) =>
    ({def} ((D4
    Intersection
    F4) <<= D5) Fixform
    Ug ([(G_4
       : obj) =>
       ({def} Ded
       ([(ghyp_5
          : that
          G_4 E D4
          Intersection
          F4) =>
          ({def} Counterexample
          (casehyp2_1) Eg
          [(.H_6
             : obj), (hhyp_6
             : that
             Counterexample
             (casehyp2_1) Witnesses
             .H_6) =>
             ({def} Notimp2
             (hhyp_6) Mp
             .H_6
```

```
            Ui
            Simp2
            (ghyp_5
            Iff1
            G_4
            Ui
            Separation4
            (Refleq
            (D4
            Intersection
            F4))) Mpsubs
            dhyp5
            Mp
            D5
            Ui
            Simp2
            (Simp2
            (Notimp2
            (hhyp_6) Mpsubs
            dhyp4
            Iff1
            .H_6
            Ui
            Separation4
            (Refleq
            (Cuts)))) Ds2
            Notimp1
            (hhyp_6) : that
            G_4
            E D5)] : that
         G_4 E D5)]) : that
      (G_4 E D4
      Intersection
      F4) ->
      G_4 E D5)]) Conj
   Separation3
   (Refleq (D4
   Intersection
```

```
    F4)) Conj
    Mboldtheta
    Setsinchains
    dhyp5 : that
    (D4 Intersection
    F4) <<= D5)]


line143 : [(casehyp2_1
    : that ~ (Forall
    ([(D7_4
       : obj) =>
       ({def} (D7_4
       E D4) ->
       D5 <<= D7_4
       : prop)]))) =>
    (--- : that
    (D4 Intersection
    F4) <<= D5)]


{move 8}

>>> define line144 \
    casehyp2 : Add2 \
    (D5 <<= D4 Intersection \
    F4, line143 casehyp2)


line144 : [(casehyp2_1
    : that ~ (Forall
    ([(D7_4
       : obj) =>
       ({def} (D7_4
       E D4) ->
       D5 <<= D7_4
       : prop)]))) =>
    ({def} (D5
```

```
      <<= D4 Intersection
      F4) Add2 line143
      (casehyp2_1) : that
      (D5 <<= D4
      Intersection
      F4) V (D4
      Intersection
      F4) <<= D5)]


   line144 : [(casehyp2_1
      : that ~ (Forall
      ([(D7_4
         : obj) =>
         ({def} (D7_4
         E D4) ->
         D5 <<= D7_4
         : prop)]))) =>
      (--- : that
      (D5 <<= D4
      Intersection
      F4) V (D4
      Intersection
      F4) <<= D5)]


   {move 8}

   >>> close


{move 8}

>>> define line145 \
    dhyp5 : Cases line123, line132, line144


line145 : [(dhyp5_1
```

```
: that D5 E Mbold) =>
({def} Cases
(line123, [(casehyp1_2
   : that Forall
   ([(D7_4
      : obj) =>
      ({def} (D7_4
      E D4) ->
      D5 <<= D7_4
      : prop)])) =>
   ({def} ((D4
   Intersection
   F4) <<= D5) Add1
   (D5 <<= D4
   Intersection
   F4) Fixform
   Ug ([(G_6
      : obj) =>
      ({def} Ded
      ([(ghyp_7
         : that
         G_6 E D5) =>
         ({def} (G_6
         E D4
         Intersection
         F4) Fixform
         fhyp4
         Mp F4
         Ui Ug
         ([(B1_13
            : obj) =>
            ({def} Ded
            ([(bhyp1_14
               : that
               B1_13
               E D4) =>
               ({def} ghyp_7
               Mpsubs
```

```
        bhyp1_14
        Mp
        B1_13
        Ui
        casehyp1_2
        : that
        G_6
        E B1_13)]) : that
     (B1_13
     E D4) ->
     G_6
     E B1_13)]) Conj
  Ug ([(B1_11
     : obj) =>
     ({def} Ded
     ([(bhyp1_12
        : that
        B1_11
        E D4) =>
        ({def} ghyp_7
        Mpsubs
        bhyp1_12
        Mp
        B1_11
        Ui
        casehyp1_2
        : that
        G_6
        E B1_11)]) : that
     (B1_11
     E D4) ->
     G_6
     E B1_11)]) Iff2
  G_6 Ui
  Separation4
  (Refleq
  (D4
  Intersection
```

```
      F4)) : that
      G_6 E D4
      Intersection
      F4)]) : that
   (G_6 E D5) ->
   G_6 E D4
   Intersection
   F4)]) Conj
Mboldtheta
Setsinchains
dhyp5_1 Conj
Separation3
(Refleq (D4
Intersection
F4)) : that
(D5 <<= D4
Intersection
F4) V (D4
Intersection
F4) <<= D5)], [(casehyp2_2
: that ~ (Forall
([(D7_5
   : obj) =>
   ({def} (D7_5
   E D4) ->
   D5 <<= D7_5
   : prop)]))) =>
({def} (D5
<<= D4 Intersection
F4) Add2 ((D4
Intersection
F4) <<= D5) Fixform
Ug ([(G_6
   : obj) =>
   ({def} Ded
   ([(ghyp_7
      : that
      G_6 E D4
```

```
Intersection
F4) =>
({def} Counterexample
(casehyp2_2) Eg
[(.H_8
   : obj), (hhyp_8
   : that
   Counterexample
   (casehyp2_2) Witnesses
   .H_8) =>
   ({def} Notimp2
   (hhyp_8) Mp
   .H_8
   Ui
   Simp2
   (ghyp_7
   Iff1
   G_6
   Ui
   Separation4
   (Refleq
   (D4
   Intersection
   F4))) Mpsubs
   dhyp5_1
   Mp
   D5
   Ui
   Simp2
   (Simp2
   (Notimp2
   (hhyp_8) Mpsubs
   dhyp4
   Iff1
   .H_8
   Ui
   Separation4
   (Refleq
```

```
                  (Cuts)))) Ds2
                  Notimp1
                  (hhyp_8) : that
                  G_6
                  E D5)] : that
               G_6 E D5)]) : that
            (G_6 E D4
            Intersection
            F4) ->
            G_6 E D5)]) Conj
         Separation3
         (Refleq (D4
         Intersection
         F4)) Conj
         Mboldtheta
         Setsinchains
         dhyp5_1 : that
         (D5 <<= D4
         Intersection
         F4) V (D4
         Intersection
         F4) <<= D5)]) : that
    (D5 <<= D4 Intersection
    F4) V (D4 Intersection
    F4) <<= D5)]


line145 : [(dhyp5_1
    : that D5 E Mbold) =>
    (--- : that (D5
    <<= D4 Intersection
    F4) V (D4 Intersection
    F4) <<= D5)]


{move 7}

>>> close
```

```
{move 7}

>>> define line146 D5 \
    : Ded line145


line146 : [(D5_1 : obj) =>
    ({def} Ded ([(dhyp5_2
       : that D5_1 E Mbold) =>
       ({def} Cases
       (Excmid (Forall
       ([(D6_5 : obj) =>
          ({def} (D6_5
          E D4) -> D5_1
          <<= D6_5 : prop)])), [(casehyp1_3
          : that Forall
          ([(D7_5
             : obj) =>
             ({def} (D7_5
             E D4) ->
             D5_1 <<=
             D7_5 : prop)])) =>
          ({def} ((D4
          Intersection
          F4) <<= D5_1) Add1
          (D5_1 <<=
          D4 Intersection
          F4) Fixform
          Ug ([(G_7
             : obj) =>
             ({def} Ded
             ([(ghyp_8
                : that
                G_7 E D5_1) =>
                ({def} (G_7
                E D4
```

```
Intersection
F4) Fixform
fhyp4
Mp F4
Ui Ug
([(B1_14
   : obj) =>
   ({def} Ded
   ([(bhyp1_15
      : that
      B1_14
      E D4) =>
      ({def} ghyp_8
      Mpsubs
      bhyp1_15
      Mp
      B1_14
      Ui
      casehyp1_3
      : that
      G_7
      E B1_14)]) : that
   (B1_14
   E D4) ->
   G_7
   E B1_14)]) Conj
Ug ([(B1_12
   : obj) =>
   ({def} Ded
   ([(bhyp1_13
      : that
      B1_12
      E D4) =>
      ({def} ghyp_8
      Mpsubs
      bhyp1_13
      Mp
      B1_12
```

```
                  Ui
                  casehyp1_3
                  : that
                  G_7
                  E B1_12)]) : that
               (B1_12
               E D4) ->
               G_7
               E B1_12)]) Iff2
            G_7 Ui
            Separation4
            (Refleq
            (D4
            Intersection
            F4)) : that
            G_7 E D4
            Intersection
            F4)]) : that
         (G_7 E D5_1) ->
         G_7 E D4
         Intersection
         F4)]) Conj
      Mboldtheta
      Setsinchains
      dhyp5_2 Conj
      Separation3
      (Refleq (D4
      Intersection
      F4)) : that
      (D5_1 <<=
      D4 Intersection
      F4) V (D4
      Intersection
      F4) <<= D5_1)], [(casehyp2_3
      : that ~ (Forall
      ([(D7_6
         : obj) =>
         ({def} (D7_6
```

```
   E D4) ->
   D5_1 <<=
   D7_6 : prop)]))) =>
({def} (D5_1
<<= D4 Intersection
F4) Add2 ((D4
Intersection
F4) <<= D5_1) Fixform
Ug ([(G_7
   : obj) =>
   ({def} Ded
   ([(ghyp_8
      : that
      G_7 E D4
      Intersection
      F4) =>
      ({def} Counterexample
      (casehyp2_3) Eg
      [(.H_9
         : obj), (hhyp_9
         : that
         Counterexample
         (casehyp2_3) Witnesses
         .H_9) =>
         ({def} Notimp2
         (hhyp_9) Mp
         .H_9
         Ui
         Simp2
         (ghyp_8
         Iff1
         G_7
         Ui
         Separation4
         (Refleq
         (D4
         Intersection
         F4))) Mpsubs
```

```
            dhyp5_2
            Mp
            D5_1
            Ui
            Simp2
            (Simp2
            (Notimp2
            (hhyp_9) Mpsubs
            dhyp4
            Iff1
            .H_9
            Ui
            Separation4
            (Refleq
            (Cuts)))) Ds2
            Notimp1
            (hhyp_9) : that
            G_7
            E D5_1)] : that
         G_7 E D5_1)]) : that
      (G_7 E D4
      Intersection
      F4) ->
      G_7 E D5_1)]) Conj
   Separation3
   (Refleq (D4
   Intersection
   F4)) Conj
   Mboldtheta
   Setsinchains
   dhyp5_2 : that
   (D5_1 <<=
   D4 Intersection
   F4) V (D4
   Intersection
   F4) <<= D5_1)]) : that
(D5_1 <<= D4
Intersection F4) V (D4
```

```
            Intersection F4) <<=
            D5_1)]) : that
         (D5_1 E Mbold) ->
         (D5_1 <<= D4 Intersection
         F4) V (D4 Intersection
         F4) <<= D5_1)]


   line146 : [(D5_1 : obj) =>
         (--- : that (D5_1
         E Mbold) -> (D5_1
         <<= D4 Intersection
         F4) V (D4 Intersection
         F4) <<= D5_1)]


   {move 6}

   >>> close


{move 6}

>>> define line147 fhyp4 \
    : Conj (line122 fhyp4, Conj \
    (line122 fhyp4, Ug line146))


line147 : [(fhyp4_1 : that
    F4 E D4) =>
    ({def} line122 (fhyp4_1) Conj
    line122 (fhyp4_1) Conj
    Ug ([(D5_4 : obj) =>
       ({def} Ded ([(dhyp5_5
          : that D5_4 E Mbold) =>
          ({def} Cases
          (Excmid (Forall
          ([(D6_8 : obj) =>
```

```
({def} (D6_8
E D4) -> D5_4
<<= D6_8 : prop)])), [(casehyp1_6
: that Forall
([(D7_8
   : obj) =>
   ({def} (D7_8
   E D4) ->
   D5_4 <<=
   D7_8 : prop)])) =>
({def} ((D4
Intersection
F4) <<= D5_4) Add1
(D5_4 <<=
D4 Intersection
F4) Fixform
Ug ([(G_10
   : obj) =>
   ({def} Ded
   ([(ghyp_11
      : that
      G_10
      E D5_4) =>
      ({def} (G_10
      E D4
      Intersection
      F4) Fixform
      fhyp4_1
      Mp F4
      Ui Ug
      ([(B1_17
         : obj) =>
         ({def} Ded
         ([(bhyp1_18
            : that
            B1_17
            E D4) =>
            ({def} ghyp_11
```

```
      Mpsubs
      bhyp1_18
      Mp
      B1_17
      Ui
      casehyp1_6
      : that
      G_10
      E B1_17)]) : that
   (B1_17
   E D4) ->
   G_10
   E B1_17)]) Conj
Ug ([(B1_15
   : obj) =>
   ({def} Ded
   ([(bhyp1_16
      : that
      B1_15
      E D4) =>
      ({def} ghyp_11
      Mpsubs
      bhyp1_16
      Mp
      B1_15
      Ui
      casehyp1_6
      : that
      G_10
      E B1_15)]) : that
   (B1_15
   E D4) ->
   G_10
   E B1_15)]) Iff2
G_10
Ui Separation4
(Refleq
(D4
```

```
      Intersection
      F4)) : that
      G_10
      E D4
      Intersection
      F4)]) : that
   (G_10 E D5_4) ->
   G_10 E D4
   Intersection
   F4)]) Conj
Mboldtheta
Setsinchains
dhyp5_5 Conj
Separation3
(Refleq (D4
Intersection
F4)) : that
(D5_4 <<=
D4 Intersection
F4) V (D4
Intersection
F4) <<= D5_4)], [(casehyp2_6
: that ~ (Forall
([(D7_9
   : obj) =>
   ({def} (D7_9
   E D4) ->
   D5_4 <<=
   D7_9 : prop)]))) =>
({def} (D5_4
<<= D4 Intersection
F4) Add2 ((D4
Intersection
F4) <<= D5_4) Fixform
Ug ([(G_10
   : obj) =>
   ({def} Ded
   ([(ghyp_11
```

```
: that
G_10
E D4
Intersection
F4) =>
({def} Counterexample
(casehyp2_6) Eg
[(.H_12
   : obj), (hhyp_12
   : that
   Counterexample
   (casehyp2_6) Witnesses
   .H_12) =>
   ({def} Notimp2
   (hhyp_12) Mp
   .H_12
   Ui
   Simp2
   (ghyp_11
   Iff1
   G_10
   Ui
   Separation4
   (Refleq
   (D4
   Intersection
   F4)) Mpsubs
   dhyp5_5
   Mp
   D5_4
   Ui
   Simp2
   (Simp2
   (Notimp2
   (hhyp_12) Mpsubs
   dhyp4
   Iff1
   .H_12
```

```
                    Ui
                    Separation4
                    (Refleq
                    (Cuts)))) Ds2
                    Notimp1
                    (hhyp_12) : that
                    G_10
                    E D5_4)] : that
                 G_10
                 E D5_4)]) : that
              (G_10 E D4
              Intersection
              F4) ->
              G_10 E D5_4)]) Conj
           Separation3
           (Refleq (D4
           Intersection
           F4)) Conj
           Mboldtheta
           Setsinchains
           dhyp5_5 : that
           (D5_4 <<=
           D4 Intersection
           F4) V (D4
           Intersection
           F4) <<= D5_4)]) : that
        (D5_4 <<= D4
        Intersection F4) V (D4
        Intersection F4) <<=
        D5_4)]) : that
     (D5_4 E Mbold) ->
     (D5_4 <<= D4 Intersection
     F4) V (D4 Intersection
     F4) <<= D5_4)]) : that
  ((D4 Intersection
  F4) E Misset Mbold2
  thelawchooses) & ((D4
  Intersection F4) E Misset
```

```
    Mbold2 thelawchooses) & Forall
    ([(x'_4 : obj) =>
       ({def} (x'_4 E Mbold) ->
       (x'_4 <<= D4 Intersection
       F4) V (D4 Intersection
       F4) <<= x'_4 : prop)]))]


line147 : [(fhyp4_1 : that
    F4 E D4) => (--- : that
    ((D4 Intersection
    F4) E Misset Mbold2
    thelawchooses) & ((D4
    Intersection F4) E Misset
    Mbold2 thelawchooses) & Forall
    ([(x'_4 : obj) =>
       ({def} (x'_4 E Mbold) ->
       (x'_4 <<= D4 Intersection
       F4) V (D4 Intersection
       F4) <<= x'_4 : prop)]))]


{move 5}

>>> define linea147 fhyp4 \
    : Iff2 (line147 fhyp4, Ui \
    (D4 Intersection F4, Separation4 \
    Refleq Cuts))


linea147 : [(fhyp4_1
    : that F4 E D4) =>
    ({def} line147 (fhyp4_1) Iff2
    (D4 Intersection F4) Ui
    Separation4 (Refleq
    (Cuts)) : that (D4
    Intersection F4) E Misset
    Mbold2 thelawchooses
```

```
        Set [(C_3 : obj) =>
           ({def} cuts2 (Misset, thelawchooses, C_3) : prop)])]


   linea147 : [(fhyp4_1
        : that F4 E D4) =>
        (--- : that (D4 Intersection
        F4) E Misset Mbold2
        thelawchooses Set [(C_3
           : obj) =>
           ({def} cuts2 (Misset, thelawchooses, C_3) : prop)])]


   {move 5}

   >>> close


{move 5}

>>> define line148 F4 : Ded \
    linea147


line148 : [(F4_1 : obj) =>
    ({def} Ded ([(fhyp4_2
        : that F4_1 E D4) =>
        ({def} dhyp4 Transsub
        (Cuts <<= Mbold) Fixform
        Separation3 (Refleq
        (Mbold)) Sepsub2
        Refleq (Cuts) Conj
        fhyp4_2 Mp F4_1 Ui D4
        Ui Simp2 (Simp2 (Simp2
        (Mboldtheta))) Conj
        dhyp4 Transsub (Cuts
        <<= Mbold) Fixform
        Separation3 (Refleq
```

```
(Mbold)) Sepsub2
Refleq (Cuts) Conj
fhyp4_2 Mp F4_1 Ui D4
Ui Simp2 (Simp2 (Simp2
(Mboldtheta))) Conj
Ug ([(D5_6 : obj) =>
   ({def} Ded ([(dhyp5_7
      : that D5_6 E Mbold) =>
      ({def} Cases
      (Excmid (Forall
      ([(D6_10 : obj) =>
         ({def} (D6_10
         E D4) -> D5_6
         <<= D6_10 : prop)])), [(casehyp1_8
         : that Forall
         ([(D7_10
            : obj) =>
            ({def} (D7_10
            E D4) ->
            D5_6 <<=
            D7_10 : prop)])) =>
         ({def} ((D4
         Intersection
         F4_1) <<=
         D5_6) Add1
         (D5_6 <<=
         D4 Intersection
         F4_1) Fixform
         Ug ([(G_12
            : obj) =>
            ({def} Ded
            ([(ghyp_13
               : that
               G_12
               E D5_6) =>
               ({def} (G_12
               E D4
               Intersection
```

```
F4_1) Fixform
fhyp4_2
Mp F4_1
Ui Ug
([(B1_19
   : obj) =>
   ({def} Ded
   ([(bhyp1_20
      : that
      B1_19
      E D4) =>
      ({def} ghyp_13
      Mpsubs
      bhyp1_20
      Mp
      B1_19
      Ui
      casehyp1_8
      : that
      G_12
      E B1_19)]) : that
   (B1_19
   E D4) ->
   G_12
   E B1_19)]) Conj
Ug ([(B1_17
   : obj) =>
   ({def} Ded
   ([(bhyp1_18
      : that
      B1_17
      E D4) =>
      ({def} ghyp_13
      Mpsubs
      bhyp1_18
      Mp
      B1_17
      Ui
```

```
              casehyp1_8
              : that
              G_12
              E B1_17)]) : that
           (B1_17
           E D4) ->
           G_12
           E B1_17)]) Iff2
        G_12
        Ui Separation4
        (Refleq
        (D4
        Intersection
        F4_1)) : that
        G_12
        E D4
        Intersection
        F4_1)]) : that
     (G_12 E D5_6) ->
     G_12 E D4
     Intersection
     F4_1)]) Conj
  Mboldtheta
  Setsinchains
  dhyp5_7 Conj
  Separation3
  (Refleq (D4
  Intersection
  F4_1)) : that
  (D5_6 <<=
  D4 Intersection
  F4_1) V (D4
  Intersection
  F4_1) <<=
  D5_6)], [(casehyp2_8
  : that ~ (Forall
  ([(D7_11
     : obj) =>
```

```
   ({def} (D7_11
   E D4) ->
   D5_6 <<=
   D7_11 : prop)]))) =>
({def} (D5_6
<<= D4 Intersection
F4_1) Add2
((D4 Intersection
F4_1) <<=
D5_6) Fixform
Ug ([(G_12
   : obj) =>
   ({def} Ded
   ([(ghyp_13
      : that
      G_12
      E D4
      Intersection
      F4_1) =>
      ({def} Counterexample
      (casehyp2_8) Eg
      [(.H_14
         : obj), (hhyp_14
         : that
         Counterexample
         (casehyp2_8) Witnesses
         .H_14) =>
         ({def} Notimp2
         (hhyp_14) Mp
         .H_14
         Ui
         Simp2
         (ghyp_13
         Iff1
         G_12
         Ui
         Separation4
         (Refleq
```

```
         (D4
         Intersection
         F4_1))) Mpsubs
         dhyp5_7
         Mp
         D5_6
         Ui
         Simp2
         (Simp2
         (Notimp2
         (hhyp_14) Mpsubs
         dhyp4
         Iff1
         .H_14
         Ui
         Separation4
         (Refleq
         (Cuts)))) Ds2
         Notimp1
         (hhyp_14) : that
         G_12
         E D5_6)] : that
      G_12
      E D5_6)]) : that
   (G_12 E D4
   Intersection
   F4_1) ->
   G_12 E D5_6)]) Conj
Separation3
(Refleq (D4
Intersection
F4_1)) Conj
Mboldtheta
Setsinchains
dhyp5_7 : that
(D5_6 <<=
D4 Intersection
F4_1) V (D4
```

```
                Intersection
                F4_1) <<=
                D5_6)]) : that
             (D5_6 <<= D4
             Intersection F4_1) V (D4
             Intersection F4_1) <<=
             D5_6)]) : that
          (D5_6 E Mbold) ->
          (D5_6 <<= D4 Intersection
          F4_1) V (D4 Intersection
          F4_1) <<= D5_6)]) Iff2
       (D4 Intersection F4_1) Ui
       Separation4 (Refleq
       (Cuts)) : that (D4
       Intersection F4_1) E Misset
       Mbold2 thelawchooses
       Set [(C_4 : obj) =>
          ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])])
    (F4_1 E D4) -> (D4 Intersection
    F4_1) E Misset Mbold2
    thelawchooses Set [(C_4
       : obj) =>
       ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])]


line148 : [(F4_1 : obj) =>
    (--- : that (F4_1 E D4) ->
    (D4 Intersection F4_1) E Misset
    Mbold2 thelawchooses Set
    [(C_4 : obj) =>
       ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])]


{move 4}

>>> close
```

```
{move 4}

>>> define line149 dhyp4 : Ug \
    line148


line149 : [(dhyp4_1 : that
    D4 <<= Cuts) =>
    ({def} Ug ([(F4_2 : obj) =>
       ({def} Ded ([(fhyp4_3
          : that F4_2 E D4) =>
          ({def} dhyp4_1 Transsub
          (Cuts <<= Mbold) Fixform
          Separation3 (Refleq
          (Mbold)) Sepsub2
          Refleq (Cuts) Conj
          fhyp4_3 Mp F4_2 Ui D4
          Ui Simp2 (Simp2 (Simp2
          (Mboldtheta))) Conj
          dhyp4_1 Transsub (Cuts
          <<= Mbold) Fixform
          Separation3 (Refleq
          (Mbold)) Sepsub2
          Refleq (Cuts) Conj
          fhyp4_3 Mp F4_2 Ui D4
          Ui Simp2 (Simp2 (Simp2
          (Mboldtheta))) Conj
          Ug ([(D5_7 : obj) =>
             ({def} Ded ([(dhyp5_8
                : that D5_7 E Mbold) =>
                ({def} Cases
                (Excmid (Forall
                ([(D6_11 : obj) =>
                   ({def} (D6_11
                   E D4) -> D5_7
                   <<= D6_11 : prop)])), [(casehyp1_9
                   : that Forall
                   ([(D7_11
```

```
   : obj) =>
   ({def} (D7_11
   E D4) ->
   D5_7 <<=
   D7_11 : prop)])) =>
({def} ((D4
Intersection
F4_2) <<=
D5_7) Add1
(D5_7 <<=
D4 Intersection
F4_2) Fixform
Ug ([(G_13
   : obj) =>
   ({def} Ded
   ([(ghyp_14
      : that
      G_13
      E D5_7) =>
      ({def} (G_13
      E D4
      Intersection
      F4_2) Fixform
      fhyp4_3
      Mp F4_2
      Ui Ug
      ([(B1_20
         : obj) =>
         ({def} Ded
         ([(bhyp1_21
            : that
            B1_20
            E D4) =>
            ({def} ghyp_14
            Mpsubs
            bhyp1_21
            Mp
            B1_20
```

```
      Ui
      casehyp1_9
      : that
      G_13
      E B1_20)]) : that
   (B1_20
   E D4) ->
   G_13
   E B1_20)]) Conj
Ug ([(B1_18
   : obj) =>
   ({def} Ded
   ([(bhyp1_19
      : that
      B1_18
      E D4) =>
      ({def} ghyp_14
      Mpsubs
      bhyp1_19
      Mp
      B1_18
      Ui
      casehyp1_9
      : that
      G_13
      E B1_18)]) : that
   (B1_18
   E D4) ->
   G_13
   E B1_18)]) Iff2
G_13
Ui Separation4
(Refleq
(D4
Intersection
F4_2)) : that
G_13
E D4
```

```
      Intersection
      F4_2)]) : that
   (G_13 E D5_7) ->
   G_13 E D4
   Intersection
   F4_2)]) Conj
Mboldtheta
Setsinchains
dhyp5_8 Conj
Separation3
(Refleq (D4
Intersection
F4_2)) : that
(D5_7 <<=
D4 Intersection
F4_2) V (D4
Intersection
F4_2) <<=
D5_7)], [(casehyp2_9
: that ~ (Forall
([(D7_12
   : obj) =>
   ({def} (D7_12
   E D4) ->
   D5_7 <<=
   D7_12 : prop)]))) =>
({def} (D5_7
<<= D4 Intersection
F4_2) Add2
((D4 Intersection
F4_2) <<=
D5_7) Fixform
Ug ([(G_13
   : obj) =>
   ({def} Ded
   ([(ghyp_14
      : that
      G_13
```

```
E D4
Intersection
F4_2) =>
({def} Counterexample
(casehyp2_9) Eg
[(.H_15
   : obj), (hhyp_15
   : that
   Counterexample
   (casehyp2_9) Witnesses
   .H_15) =>
   ({def} Notimp2
   (hhyp_15) Mp
   .H_15
   Ui
   Simp2
   (ghyp_14
   Iff1
   G_13
   Ui
   Separation4
   (Refleq
   (D4
   Intersection
   F4_2))) Mpsubs
   dhyp5_8
   Mp
   D5_7
   Ui
   Simp2
   (Simp2
   (Notimp2
   (hhyp_15) Mpsubs
   dhyp4_1
   Iff1
   .H_15
   Ui
   Separation4
```

```
                    (Refleq
                    (Cuts)))) Ds2
                    Notimp1
                    (hhyp_15) : that
                    G_13
                    E D5_7)] : that
                 G_13
                 E D5_7)]) : that
              (G_13 E D4
              Intersection
              F4_2) ->
              G_13 E D5_7)]) Conj
           Separation3
           (Refleq (D4
           Intersection
           F4_2)) Conj
           Mboldtheta
           Setsinchains
           dhyp5_8 : that
           (D5_7 <<=
           D4 Intersection
           F4_2) V (D4
           Intersection
           F4_2) <<=
           D5_7)]) : that
        (D5_7 <<= D4
        Intersection F4_2) V (D4
        Intersection F4_2) <<=
        D5_7)]) : that
     (D5_7 E Mbold) ->
     (D5_7 <<= D4 Intersection
     F4_2) V (D4 Intersection
     F4_2) <<= D5_7)]) Iff2
  (D4 Intersection F4_2) Ui
  Separation4 (Refleq
  (Cuts)) : that (D4
  Intersection F4_2) E Misset
  Mbold2 thelawchooses
```

```
            Set [(C_5 : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_5) : prop)])])
         (F4_2 E D4) -> (D4 Intersection
         F4_2) E Misset Mbold2
         thelawchooses Set [(C_5
            : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_5) : prop)])]) : t
      Forall ([(x'_2 : obj) =>
         ({def} (x'_2 E D4) ->
         (D4 Intersection x'_2) E Misset
         Mbold2 thelawchooses Set
         [(C_5 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_5) : prop)] : prop


   line149 : [(dhyp4_1 : that
       D4 <<= Cuts) => (--- : that
       Forall ([(x'_2 : obj) =>
          ({def} (x'_2 E D4) ->
          (D4 Intersection x'_2) E Misset
          Mbold2 thelawchooses Set
          [(C_5 : obj) =>
             ({def} cuts2 (Misset, thelawchooses, C_5) : prop)] : prop


   {move 3}

   >>> close


{move 3}

>>> define line150 D4 : Ded line149


line150 : [(D4_1 : obj) =>
    ({def} Ded ([(dhyp4_2 : that
       D4_1 <<= Cuts) =>
```

```
({def} Ug ([(F4_3 : obj) =>
   ({def} Ded ([(fhyp4_4
      : that F4_3 E D4_1) =>
      ({def} dhyp4_2 Transsub
      (Cuts <<= Mbold) Fixform
      Separation3 (Refleq
      (Mbold)) Sepsub2
      Refleq (Cuts) Conj
      fhyp4_4 Mp F4_3 Ui D4_1
      Ui Simp2 (Simp2 (Simp2
      (Mboldtheta))) Conj
      dhyp4_2 Transsub (Cuts
      <<= Mbold) Fixform
      Separation3 (Refleq
      (Mbold)) Sepsub2
      Refleq (Cuts) Conj
      fhyp4_4 Mp F4_3 Ui D4_1
      Ui Simp2 (Simp2 (Simp2
      (Mboldtheta))) Conj
      Ug ([(D5_8 : obj) =>
         ({def} Ded ([(dhyp5_9
            : that D5_8 E Mbold) =>
            ({def} Cases
            (Excmid (Forall
            ([(D6_12 : obj) =>
               ({def} (D6_12
               E D4_1) ->
               D5_8 <<= D6_12
               : prop)])), [(casehyp1_10
               : that Forall
               ([(D7_12
                  : obj) =>
                  ({def} (D7_12
                  E D4_1) ->
                  D5_8 <<=
                  D7_12 : prop)])) =>
               ({def} ((D4_1
               Intersection
```

```
F4_3) <<=
D5_8) Add1
(D5_8 <<=
D4_1 Intersection
F4_3) Fixform
Ug ([(G_14
   : obj) =>
   ({def} Ded
   ([(ghyp_15
      : that
      G_14
      E D5_8) =>
      ({def} (G_14
      E D4_1
      Intersection
      F4_3) Fixform
      fhyp4_4
      Mp F4_3
      Ui Ug
      ([(B1_21
         : obj) =>
         ({def} Ded
         ([(bhyp1_22
            : that
            B1_21
            E D4_1) =>
            ({def} ghyp_15
            Mpsubs
            bhyp1_22
            Mp
            B1_21
            Ui
            casehyp1_10
            : that
            G_14
            E B1_21)]) : that
         (B1_21
         E D4_1) ->
```

```
            G_14
            E B1_21)]) Conj
         Ug ([(B1_19
            : obj) =>
            ({def} Ded
            ([(bhyp1_20
               : that
               B1_19
               E D4_1) =>
               ({def} ghyp_15
               Mpsubs
               bhyp1_20
               Mp
               B1_19
               Ui
               casehyp1_10
               : that
               G_14
               E B1_19)]) : that
            (B1_19
            E D4_1) ->
            G_14
            E B1_19)]) Iff2
         G_14
         Ui Separation4
         (Refleq
         (D4_1
         Intersection
         F4_3)) : that
         G_14
         E D4_1
         Intersection
         F4_3)]) : that
      (G_14 E D5_8) ->
      G_14 E D4_1
      Intersection
      F4_3)]) Conj
   Mboldtheta
```

```
Setsinchains
dhyp5_9 Conj
Separation3
(Refleq (D4_1
Intersection
F4_3)) : that
(D5_8 <<=
D4_1 Intersection
F4_3) V (D4_1
Intersection
F4_3) <<=
D5_8)], [(casehyp2_10
: that ~ (Forall
([(D7_13
   : obj) =>
   ({def} (D7_13
   E D4_1) ->
   D5_8 <<=
   D7_13 : prop)]))) =>
({def} (D5_8
<<= D4_1 Intersection
F4_3) Add2
((D4_1 Intersection
F4_3) <<=
D5_8) Fixform
Ug ([(G_14
   : obj) =>
   ({def} Ded
   ([(ghyp_15
      : that
      G_14
      E D4_1
      Intersection
      F4_3) =>
      ({def} Counterexample
      (casehyp2_10) Eg
      [(.H_16
         : obj), (hhyp_16
```

```
   : that
   Counterexample
   (casehyp2_10) Witnesses
   .H_16) =>
   ({def} Notimp2
   (hhyp_16) Mp
   .H_16
   Ui
   Simp2
   (ghyp_15
   Iff1
   G_14
   Ui
   Separation4
   (Refleq
   (D4_1
   Intersection
   F4_3))) Mpsubs
   dhyp5_9
   Mp
   D5_8
   Ui
   Simp2
   (Simp2
   (Notimp2
   (hhyp_16) Mpsubs
   dhyp4_2
   Iff1
   .H_16
   Ui
   Separation4
   (Refleq
   (Cuts)))) Ds2
   Notimp1
   (hhyp_16) : that
   G_14
   E D5_8)] : that
G_14
```

```
                  E D5_8)]) : that
                 (G_14 E D4_1
                 Intersection
                 F4_3) ->
                 G_14 E D5_8)]) Conj
              Separation3
              (Refleq (D4_1
              Intersection
              F4_3)) Conj
              Mboldtheta
              Setsinchains
              dhyp5_9 : that
              (D5_8 <<=
              D4_1 Intersection
              F4_3) V (D4_1
              Intersection
              F4_3) <<=
              D5_8)]) : that
           (D5_8 <<= D4_1
           Intersection F4_3) V (D4_1
           Intersection F4_3) <<=
           D5_8)]) : that
        (D5_8 E Mbold) ->
        (D5_8 <<= D4_1 Intersection
        F4_3) V (D4_1 Intersection
        F4_3) <<= D5_8)]) Iff2
     (D4_1 Intersection
     F4_3) Ui Separation4
     (Refleq (Cuts)) : that
     (D4_1 Intersection
     F4_3) E Misset Mbold2
     thelawchooses Set [(C_6
        : obj) =>
        ({def} cuts2 (Misset, thelawchooses, C_6) : prop)])])
  (F4_3 E D4_1) -> (D4_1
  Intersection F4_3) E Misset
  Mbold2 thelawchooses Set
  [(C_6 : obj) =>
```

```
            ({def} cuts2 (Misset, thelawchooses, C_6) : prop)])]) : t
      Forall ([(x'_3 : obj) =>
         ({def} (x'_3 E D4_1) ->
         (D4_1 Intersection x'_3) E Misset
         Mbold2 thelawchooses Set
         [(C_6 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop
   (D4_1 <<= Cuts) -> Forall ([(x'_3
      : obj) =>
      ({def} (x'_3 E D4_1) ->
      (D4_1 Intersection x'_3) E Misset
      Mbold2 thelawchooses Set [(C_6
         : obj) =>
         ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)])


line150 : [(D4_1 : obj) => (---
    : that (D4_1 <<= Cuts) -> Forall
    ([(x'_3 : obj) =>
       ({def} (x'_3 E D4_1) ->
       (D4_1 Intersection x'_3) E Misset
       Mbold2 thelawchooses Set [(C_6
          : obj) =>
          ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)])


{move 2}

>>> close


{move 2}

>>> define line151 : Ug line150


line151 : Ug ([(D4_2 : obj) =>
    ({def} Ded ([(dhyp4_3 : that
```

```
D4_2 <<= Cuts) =>
({def} Ug ([(F4_4 : obj) =>
   ({def} Ded ([(fhyp4_5
      : that F4_4 E D4_2) =>
      ({def} dhyp4_3 Transsub
      (Cuts <<= Mbold) Fixform
      Separation3 (Refleq (Mbold)) Sepsub2
      Refleq (Cuts) Conj fhyp4_5
      Mp F4_4 Ui D4_2 Ui Simp2
      (Simp2 (Simp2 (Mboldtheta))) Conj
      dhyp4_3 Transsub (Cuts
      <<= Mbold) Fixform Separation3
      (Refleq (Mbold)) Sepsub2
      Refleq (Cuts) Conj fhyp4_5
      Mp F4_4 Ui D4_2 Ui Simp2
      (Simp2 (Simp2 (Mboldtheta))) Conj
      Ug ([(D5_9 : obj) =>
         ({def} Ded ([(dhyp5_10
            : that D5_9 E Mbold) =>
            ({def} Cases (Excmid
            (Forall ([(D6_13
               : obj) =>
               ({def} (D6_13
               E D4_2) -> D5_9
               <<= D6_13 : prop)])), [(casehyp1_11
               : that Forall
               ([(D7_13 : obj) =>
                  ({def} (D7_13
                  E D4_2) ->
                  D5_9 <<= D7_13
                  : prop)])) =>
               ({def} ((D4_2
               Intersection F4_4) <<=
               D5_9) Add1 (D5_9
               <<= D4_2 Intersection
               F4_4) Fixform
               Ug ([(G_15
                  : obj) =>
```

```
({def} Ded
([(ghyp_16
   : that G_15
   E D5_9) =>
   ({def} (G_15
   E D4_2 Intersection
   F4_4) Fixform
   fhyp4_5
   Mp F4_4
   Ui Ug ([(B1_22
      : obj) =>
      ({def} Ded
      ([(bhyp1_23
         : that
         B1_22
         E D4_2) =>
         ({def} ghyp_16
         Mpsubs
         bhyp1_23
         Mp
         B1_22
         Ui
         casehyp1_11
         : that
         G_15
         E B1_22)]) : that
      (B1_22
      E D4_2) ->
      G_15
      E B1_22)]) Conj
   Ug ([(B1_20
      : obj) =>
      ({def} Ded
      ([(bhyp1_21
         : that
         B1_20
         E D4_2) =>
         ({def} ghyp_16
```

```
            Mpsubs
            bhyp1_21
            Mp
            B1_20
            Ui
            casehyp1_11
            : that
            G_15
            E B1_20)]) : that
         (B1_20
         E D4_2) ->
         G_15
         E B1_20)]) Iff2
      G_15 Ui
      Separation4
      (Refleq
      (D4_2 Intersection
      F4_4)) : that
      G_15 E D4_2
      Intersection
      F4_4)]) : that
   (G_15 E D5_9) ->
   G_15 E D4_2
   Intersection
   F4_4)]) Conj
Mboldtheta Setsinchains
dhyp5_10 Conj
Separation3 (Refleq
(D4_2 Intersection
F4_4)) : that
(D5_9 <<= D4_2
Intersection F4_4) V (D4_2
Intersection F4_4) <<=
D5_9)], [(casehyp2_11
: that ~ (Forall
([(D7_14 : obj) =>
   ({def} (D7_14
   E D4_2) ->
```

```
   D5_9 <<= D7_14
   : prop)]))) =>
({def} (D5_9
<<= D4_2 Intersection
F4_4) Add2 ((D4_2
Intersection F4_4) <<=
D5_9) Fixform
Ug ([(G_15
   : obj) =>
   ({def} Ded
   ([(ghyp_16
      : that G_15
      E D4_2 Intersection
      F4_4) =>
      ({def} Counterexample
      (casehyp2_11) Eg
      [(.H_17
         : obj), (hhyp_17
         : that
         Counterexample
         (casehyp2_11) Witnesses
         .H_17) =>
         ({def} Notimp2
         (hhyp_17) Mp
         .H_17
         Ui Simp2
         (ghyp_16
         Iff1
         G_15
         Ui Separation4
         (Refleq
         (D4_2
         Intersection
         F4_4))) Mpsubs
         dhyp5_10
         Mp D5_9
         Ui Simp2
         (Simp2
```

```
                  (Notimp2
                  (hhyp_17) Mpsubs
                  dhyp4_3
                  Iff1
                  .H_17
                  Ui Separation4
                  (Refleq
                  (Cuts)))) Ds2
                  Notimp1
                  (hhyp_17) : that
                  G_15
                  E D5_9)] : that
               G_15 E D5_9)]) : that
            (G_15 E D4_2
            Intersection
            F4_4) -> G_15
            E D5_9)]) Conj
         Separation3 (Refleq
         (D4_2 Intersection
         F4_4)) Conj
         Mboldtheta Setsinchains
         dhyp5_10 : that
         (D5_9 <<= D4_2
         Intersection F4_4) V (D4_2
         Intersection F4_4) <<=
         D5_9)]) : that
      (D5_9 <<= D4_2 Intersection
      F4_4) V (D4_2 Intersection
      F4_4) <<= D5_9)]) : that
   (D5_9 E Mbold) ->
   (D5_9 <<= D4_2 Intersection
   F4_4) V (D4_2 Intersection
   F4_4) <<= D5_9)]) Iff2
(D4_2 Intersection F4_4) Ui
Separation4 (Refleq (Cuts)) : that
(D4_2 Intersection F4_4) E Misset
Mbold2 thelawchooses Set
[(C_7 : obj) =>
```

```
                  ({def} cuts2 (Misset, thelawchooses, C_7) : prop)])]) : t
            (F4_4 E D4_2) -> (D4_2
            Intersection F4_4) E Misset
            Mbold2 thelawchooses Set [(C_7
               : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_7) : prop)])]) : that
         Forall ([(x'_4 : obj) =>
            ({def} (x'_4 E D4_2) ->
            (D4_2 Intersection x'_4) E Misset
            Mbold2 thelawchooses Set [(C_7
               : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_7) : prop)] : prop)])
      (D4_2 <<= Cuts) -> Forall ([(x'_4
         : obj) =>
         ({def} (x'_4 E D4_2) -> (D4_2
         Intersection x'_4) E Misset
         Mbold2 thelawchooses Set [(C_7
            : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_7) : prop)] : prop)]))])


line151 : that Forall ([(x'_2 : obj) =>
    ({def} (x'_2 <<= Cuts) -> Forall
    ([(x'_4 : obj) =>
       ({def} (x'_4 E x'_2) -> (x'_2
       Intersection x'_4) E Misset
       Mbold2 thelawchooses Set [(C_7
          : obj) =>
          ({def} cuts2 (Misset, thelawchooses, C_7) : prop)] : prop)]) :


{move 1}

>>> open


   {move 3}
```

```
>>> declare D9 obj


D9 : obj


{move 3}

>>> open


   {move 4}

   >>> declare F9 obj


   F9 : obj


   {move 4}

   >>> open


      {move 5}

      >>> declare conjhyp that (D9 \
          <<= Cuts) & F9 E D9


      conjhyp : that (D9 <<= Cuts) & F9
       E D9


      {move 5}

      >>> define firsthyp conjhyp \
          : Simp1 conjhyp
```

```
firsthyp : [(conjhyp_1 : that
    (D9 <<= Cuts) & F9 E D9) =>
    ({def} Simp1 (conjhyp_1) : that
    D9 <<= Cuts)]


firsthyp : [(conjhyp_1 : that
    (D9 <<= Cuts) & F9 E D9) =>
    (--- : that D9 <<= Cuts)]


{move 4}

>>> define secondhyp conjhyp \
    : Simp2 conjhyp


secondhyp : [(conjhyp_1
    : that (D9 <<= Cuts) & F9
    E D9) =>
    ({def} Simp2 (conjhyp_1) : that
    F9 E D9)]


secondhyp : [(conjhyp_1
    : that (D9 <<= Cuts) & F9
    E D9) => (--- : that
    F9 E D9)]


{move 4}

>>> define line152 conjhyp \
    : Mp secondhyp conjhyp, Ui \
    F9, Mp (firsthyp conjhyp, Ui \
    D9 line151)
```

```
   line152 : [(conjhyp_1 : that
       (D9 <<= Cuts) & F9 E D9) =>
       ({def} secondhyp (conjhyp_1) Mp
       F9 Ui firsthyp (conjhyp_1) Mp
       D9 Ui line151 : that (D9
       Intersection F9) E Misset
       Mbold2 thelawchooses Set
       [(C_3 : obj) =>
          ({def} cuts2 (Misset, thelawchooses, C_3) : prop)])]


   line152 : [(conjhyp_1 : that
       (D9 <<= Cuts) & F9 E D9) =>
       (--- : that (D9 Intersection
       F9) E Misset Mbold2 thelawchooses
       Set [(C_3 : obj) =>
          ({def} cuts2 (Misset, thelawchooses, C_3) : prop)])]


   {move 4}

   >>> close


{move 4}

>>> define line153 F9 : Ded line152


line153 : [(F9_1 : obj) =>
    ({def} Ded ([(conjhyp_2
       : that (D9 <<= Cuts) & F9_1
       E D9) =>
       ({def} Simp2 (conjhyp_2) Mp
       F9_1 Ui Simp1 (conjhyp_2) Mp
       D9 Ui line151 : that (D9
```

```
            Intersection F9_1) E Misset
            Mbold2 thelawchooses Set
            [(C_4 : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])]) : t
         ((D9 <<= Cuts) & F9_1 E D9) ->
         (D9 Intersection F9_1) E Misset
         Mbold2 thelawchooses Set [(C_4
            : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])]


   line153 : [(F9_1 : obj) =>
         (--- : that ((D9 <<= Cuts) & F9_1
         E D9) -> (D9 Intersection
         F9_1) E Misset Mbold2 thelawchooses
         Set [(C_4 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_4) : prop)])]


   {move 3}

   >>> close


{move 3}

>>> define line154 D9 : Ug line153


line154 : [(D9_1 : obj) =>
    ({def} Ug ([(F9_2 : obj) =>
       ({def} Ded ([(conjhyp_3
          : that (D9_1 <<= Cuts) & F9_2
          E D9_1) =>
          ({def} Simp2 (conjhyp_3) Mp
          F9_2 Ui Simp1 (conjhyp_3) Mp
          D9_1 Ui line151 : that
          (D9_1 Intersection F9_2) E Misset
```

```
            Mbold2 thelawchooses Set
            [(C_5 : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_5) : prop)])]) : t
         ((D9_1 <<= Cuts) & F9_2
         E D9_1) -> (D9_1 Intersection
         F9_2) E Misset Mbold2 thelawchooses
         Set [(C_5 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_5) : prop)])]) : that
      Forall ([(x'_2 : obj) =>
         ({def} ((D9_1 <<= Cuts) & x'_2
         E D9_1) -> (D9_1 Intersection
         x'_2) E Misset Mbold2 thelawchooses
         Set [(C_5 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_5) : prop)] : prop)])


   line154 : [(D9_1 : obj) => (---
       : that Forall ([(x'_2 : obj) =>
          ({def} ((D9_1 <<= Cuts) & x'_2
          E D9_1) -> (D9_1 Intersection
          x'_2) E Misset Mbold2 thelawchooses
          Set [(C_5 : obj) =>
             ({def} cuts2 (Misset, thelawchooses, C_5) : prop)] : prop)])


   {move 2}

   >>> close


{move 2}

>>> define linea155 : Ug line154


linea155 : Ug ([(D9_2 : obj) =>
    ({def} Ug ([(F9_3 : obj) =>
       ({def} Ded ([(conjhyp_4 : that
```

```
            (D9_2 <<= Cuts) & F9_3 E D9_2) =>
            ({def} Simp2 (conjhyp_4) Mp
            F9_3 Ui Simp1 (conjhyp_4) Mp
            D9_2 Ui line151 : that (D9_2
            Intersection F9_3) E Misset
            Mbold2 thelawchooses Set [(C_6
               : obj) =>
               ({def} cuts2 (Misset, thelawchooses, C_6) : prop)])]) : that
         ((D9_2 <<= Cuts) & F9_3 E D9_2) ->
         (D9_2 Intersection F9_3) E Misset
         Mbold2 thelawchooses Set [(C_6
            : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_6) : prop)])]) : that
      Forall ([(x'_3 : obj) =>
         ({def} ((D9_2 <<= Cuts) & x'_3
         E D9_2) -> (D9_2 Intersection
         x'_3) E Misset Mbold2 thelawchooses
         Set [(C_6 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)]))])


linea155 : that Forall ([(x'_2 : obj) =>
      ({def} Forall ([(x'_3 : obj) =>
         ({def} ((x'_2 <<= Cuts) & x'_3
         E x'_2) -> (x'_2 Intersection
         x'_3) E Misset Mbold2 thelawchooses
         Set [(C_6 : obj) =>
            ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)]) :


{move 1}

>>> save


{move 2}

>>> close
```

```
{move 1}

>>> define lineb155 Misset, thelawchooses \
    : linea155


lineb155 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
       .S_2 <<= .M_1), (inev_2 : that
       Exists ([(x_4 : obj) =>
          ({def} x_4 E .S_2 : prop)])) =>
       (--- : that .thelaw_1 (.S_2) E .S_2)]) =>
    ({def} Ug ([(D9_2 : obj) =>
       ({def} Ug ([(F9_3 : obj) =>
          ({def} Ded ([(conjhyp_4 : that
             (D9_2 <<= Misset_1 Cuts3
             thelawchooses_1) & F9_3 E D9_2) =>
             ({def} Simp2 (conjhyp_4) Mp
             F9_3 Ui Simp1 (conjhyp_4) Mp
             D9_2 Ui Ug ([(D4_9 : obj) =>
                ({def} Ded ([(dhyp4_10
                   : that D4_9 <<= Misset_1
                   Cuts3 thelawchooses_1) =>
                   ({def} Ug ([(F4_11
                      : obj) =>
                      ({def} Ded ([(fhyp4_12
                         : that F4_11 E D4_9) =>
                         ({def} dhyp4_10
                         Transsub (Misset_1
                         Cuts3 thelawchooses_1
                         <<= Misset_1 Mbold2
                         thelawchooses_1) Fixform
                         Separation3 (Refleq
                         (Misset_1 Mbold2
```

```
thelawchooses_1)) Sepsub2
Refleq (Misset_1
Cuts3 thelawchooses_1) Conj
fhyp4_12 Mp F4_11
Ui D4_9 Ui Simp2
(Simp2 (Simp2
(Misset_1 Mboldtheta2
thelawchooses_1))) Conj
dhyp4_10 Transsub
(Misset_1 Cuts3
thelawchooses_1
<<= Misset_1 Mbold2
thelawchooses_1) Fixform
Separation3 (Refleq
(Misset_1 Mbold2
thelawchooses_1)) Sepsub2
Refleq (Misset_1
Cuts3 thelawchooses_1) Conj
fhyp4_12 Mp F4_11
Ui D4_9 Ui Simp2
(Simp2 (Simp2
(Misset_1 Mboldtheta2
thelawchooses_1))) Conj
Ug ([(D5_16
   : obj) =>
   ({def} Ded
   ([(dhyp5_17
      : that D5_16
      E Misset_1
      Mbold2 thelawchooses_1) =>
      ({def} Cases
      (Excmid
      (Forall
      ([(D6_20
         : obj) =>
         ({def} (D6_20
         E D4_9) ->
         D5_16
```

```
<<= D6_20
: prop)])), [(casehyp1_18
: that
Forall
([(D7_20
   : obj) =>
   ({def} (D7_20
   E D4_9) ->
   D5_16
   <<=
   D7_20
   : prop)])) =>
({def} ((D4_9
Intersection
F4_11) <<=
D5_16) Add1
(D5_16
<<= D4_9
Intersection
F4_11) Fixform
Ug ([(G_22
   : obj) =>
   ({def} Ded
   ([(ghyp_23
      : that
      G_22
      E D5_16) =>
      ({def} (G_22
      E D4_9
      Intersection
      F4_11) Fixform
      fhyp4_12
      Mp
      F4_11
      Ui
      Ug
      ([(B1_29
         : obj) =>
```

```
          ({def} Ded
          ([(bhyp1_30
             : that
             B1_29
             E D4_9) =>
             ({def} ghyp_23
             Mpsubs
             bhyp1_30
             Mp
             B1_29
             Ui
             casehyp1_18
             : that
             G_22
             E B1_29)]) : that
          (B1_29
          E D4_9) ->
          G_22
          E B1_29)]) Conj
       Ug ([(B1_27
          : obj) =>
          ({def} Ded
          ([(bhyp1_28
             : that
             B1_27
             E D4_9) =>
             ({def} ghyp_23
             Mpsubs
             bhyp1_28
             Mp
             B1_27
             Ui
             casehyp1_18
             : that
             G_22
             E B1_27)]) : that
          (B1_27
          E D4_9) ->
```

```
          G_22
          E B1_27)]) Iff2
       G_22
       Ui Separation4
       (Refleq
       (D4_9
       Intersection
       F4_11)) : that
       G_22
       E D4_9
       Intersection
       F4_11)]) : that
    (G_22
    E D5_16) ->
    G_22 E D4_9
    Intersection
    F4_11)]) Conj
Setsinchains2
(Misset_1, thelawchooses_1, Misset_1
Mboldtheta2
thelawchooses_1, dhyp5_17) Conj
Separation3
(Refleq (D4_9
Intersection
F4_11)) : that
(D5_16 <<=
D4_9 Intersection
F4_11) V (D4_9
Intersection
F4_11) <<=
D5_16)], [(casehyp2_18
: that
~ (Forall
([(D7_21
    : obj) =>
    ({def} (D7_21
    E D4_9) ->
    D5_16
```

```
   <<=
   D7_21
   : prop)]))) =>
({def} (D5_16
<<= D4_9
Intersection
F4_11) Add2
((D4_9
Intersection
F4_11) <<=
D5_16) Fixform
Ug ([(G_22
   : obj) =>
   ({def} Ded
   ([(ghyp_23
      : that
      G_22
      E D4_9
      Intersection
      F4_11) =>
      ({def} Counterexample
      (casehyp2_18) Eg
      [(.H_24
         : obj), (hhyp_24
         : that
         Counterexample
         (casehyp2_18) Witnesses
         .H_24) =>
         ({def} Notimp2
         (hhyp_24) Mp
         .H_24
         Ui
         Simp2
         (ghyp_23
         Iff1
         G_22
         Ui
         Separation4
```

```
                  (Refleq
                  (D4_9
                  Intersection
                  F4_11))) Mpsubs
                  dhyp5_17
                  Mp
                  D5_16
                  Ui
                  Simp2
                  (Simp2
                  (Notimp2
                  (hhyp_24) Mpsubs
                  dhyp4_10
                  Iff1
                  .H_24
                  Ui
                  Separation4
                  (Refleq
                  (Misset_1
                  Cuts3
                  thelawchooses_1)))) Ds2
                  Notimp1
                  (hhyp_24) : that
                  G_22
                  E D5_16)] : that
               G_22
               E D5_16)]) : that
            (G_22
            E D4_9
            Intersection
            F4_11) ->
            G_22
            E D5_16)]) Conj
         Separation3
         (Refleq
         (D4_9
         Intersection
         F4_11)) Conj
```

```
                Setsinchains2
                (Misset_1, thelawchooses_1, Misset_1
                Mboldtheta2
                thelawchooses_1, dhyp5_17) : that
                (D5_16
                <<= D4_9
                Intersection
                F4_11) V (D4_9
                Intersection
                F4_11) <<=
                D5_16)]) : that
             (D5_16
             <<= D4_9
             Intersection
             F4_11) V (D4_9
             Intersection
             F4_11) <<=
             D5_16)]) : that
          (D5_16 E Misset_1
          Mbold2 thelawchooses_1) ->
          (D5_16 <<=
          D4_9 Intersection
          F4_11) V (D4_9
          Intersection
          F4_11) <<=
          D5_16)]) Iff2
       (D4_9 Intersection
       F4_11) Ui Separation4
       (Refleq (Misset_1
       Cuts3 thelawchooses_1)) : that
       (D4_9 Intersection
       F4_11) E Misset_1
       Mbold2 thelawchooses_1
       Set [(C_14 : obj) =>
          ({def} cuts2
          (Misset_1, thelawchooses_1, C_14) : prop)])]) :
    (F4_11 E D4_9) ->
    (D4_9 Intersection
```

```
                  F4_11) E Misset_1
                  Mbold2 thelawchooses_1
                  Set [(C_14 : obj) =>
                     ({def} cuts2
                     (Misset_1, thelawchooses_1, C_14) : prop)])]) : tha
               Forall ([(x'_11 : obj) =>
                  ({def} (x'_11 E D4_9) ->
                  (D4_9 Intersection
                  x'_11) E Misset_1
                  Mbold2 thelawchooses_1
                  Set [(C_14 : obj) =>
                     ({def} cuts2
                     (Misset_1, thelawchooses_1, C_14) : prop)] : prop)]
            (D4_9 <<= Misset_1 Cuts3
            thelawchooses_1) -> Forall
            ([(x'_11 : obj) =>
               ({def} (x'_11 E D4_9) ->
               (D4_9 Intersection
               x'_11) E Misset_1 Mbold2
               thelawchooses_1 Set
               [(C_14 : obj) =>
                  ({def} cuts2 (Misset_1, thelawchooses_1, C_14) : prop)
         (D9_2 Intersection F9_3) E Misset_1
         Mbold2 thelawchooses_1 Set
         [(C_6 : obj) =>
            ({def} cuts2 (Misset_1, thelawchooses_1, C_6) : prop)])]) :
      ((D9_2 <<= Misset_1 Cuts3 thelawchooses_1) & F9_3
      E D9_2) -> (D9_2 Intersection
      F9_3) E Misset_1 Mbold2 thelawchooses_1
      Set [(C_6 : obj) =>
         ({def} cuts2 (Misset_1, thelawchooses_1, C_6) : prop)])]) : tha
   Forall ([(x'_3 : obj) =>
      ({def} ((D9_2 <<= Misset_1
      Cuts3 thelawchooses_1) & x'_3
      E D9_2) -> (D9_2 Intersection
      x'_3) E Misset_1 Mbold2 thelawchooses_1
      Set [(C_6 : obj) =>
         ({def} cuts2 (Misset_1, thelawchooses_1, C_6) : prop)] : prop)]
```

```
    Forall ([(x'_2 : obj) =>
       ({def} Forall ([(x'_3 : obj) =>
          ({def} ((x'_2 <<= Misset_1
          Cuts3 thelawchooses_1) & x'_3
          E x'_2) -> (x'_2 Intersection
          x'_3) E Misset_1 Mbold2 thelawchooses_1
          Set [(C_6 : obj) =>
             ({def} cuts2 (Misset_1, thelawchooses_1, C_6) : prop)] : prop)]


lineb155 : [(.M_1 : obj), (Misset_1
    : that Isset (.M_1)), (.thelaw_1
    : [(S_2 : obj) => (--- : obj)]), (thelawchooses_1
    : [(.S_2 : obj), (subsetev_2 : that
       .S_2 <<= .M_1), (inev_2 : that
       Exists ([(x_4 : obj) =>
          ({def} x_4 E .S_2 : prop)])) =>
       (--- : that .thelaw_1 (.S_2) E .S_2)]) =>
    (--- : that Forall ([(x'_2 : obj) =>
       ({def} Forall ([(x'_3 : obj) =>
          ({def} ((x'_2 <<= Misset_1
          Cuts3 thelawchooses_1) & x'_3
          E x'_2) -> (x'_2 Intersection
          x'_3) E Misset_1 Mbold2 thelawchooses_1
          Set [(C_6 : obj) =>
             ({def} cuts2 (Misset_1, thelawchooses_1, C_6) : prop)] : prop)]


{move 0}

>>> open


   {move 2}

   >>> define line155 : lineb155 Misset, thelawchooses
```

```
      line155 : [
          ({def} Misset lineb155 thelawchooses
          : that Forall ([(x'_2 : obj) =>
             ({def} Forall ([(x'_3 : obj) =>
                ({def} ((x'_2 <<= Misset
                Cuts3 thelawchooses) & x'_3
                E x'_2) -> (x'_2 Intersection
                x'_3) E Misset Mbold2 thelawchooses
                Set [(C_6 : obj) =>
                   ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)])


      line155 : that Forall ([(x'_2 : obj) =>
          ({def} Forall ([(x'_3 : obj) =>
             ({def} ((x'_2 <<= Misset Cuts3
             thelawchooses) & x'_3 E x'_2) ->
             (x'_2 Intersection x'_3) E Misset
             Mbold2 thelawchooses Set [(C_6
                : obj) =>
                ({def} cuts2 (Misset, thelawchooses, C_6) : prop)] : prop)]) :


      {move 1}
end Lestrade execution
```

This is the fourth component of the proof that `Cuts` is a Θ-chain.

```
begin Lestrade execution

      >>> define Cutstheta2 : Fixform (thetachain \
          (Cuts), Line9 Conj Line12 Conj Line119 \
          Conj line155)

Fixform (thetachain (Cuts), Line9 Conj Line12 Conj Line119 Conj line155) is not

(paused, type something to continue) >
```

```
   >>> close


{move 1}

>>> define Cutstheta Misset thelawchooses \
    : Cutstheta2

[Misset thelawchooses => Cutstheta2] is not well-formed

(paused, type something to continue) >

   >>> clearcurrent

{move 1}
end Lestrade execution
```

This is the proof that `Cuts` is a Θ-chain. Suppressing definitional expansion of its four components has made it somewhat manageable in size.

Since I clear move 1 above, a number of convenient definitions are restated.

```
begin Lestrade execution

   >>> save


   {move 1}

   >>> declare M obj


   M : obj


   {move 1}
```

```
>>> declare Misset that Isset M

Misset : that Isset (M)

{move 1}

>>> open

   {move 2}

   >>> declare S obj

   S : obj

   {move 2}

   >>> declare x obj

   x : obj

   {move 2}

   >>> declare subsetev that S <<= M

   subsetev : that S <<= M

   {move 2}

   >>> declare inev that Exists [x => \
```

```
       x E S]


inev : that Exists ([(x_2 : obj) =>
    ({def} x_2 E S : prop)])


{move 2}

>>> postulate thelaw S : obj


thelaw : [(S_1 : obj) => (--- : obj)]


{move 1}

>>> postulate thelawchooses subsetev \
    inev : that (thelaw S) E S


thelawchooses : [(.S_1 : obj), (subsetev_1
    : that .S_1 <<= M), (inev_1 : that
    Exists ([(x_3 : obj) =>
       ({def} x_3 E .S_1 : prop)])) =>
    (--- : that thelaw (.S_1) E .S_1)]


{move 1}

>>> open


   {move 3}

   >>> define Mbold : Mbold2 Misset \
       thelawchooses
```

```
Mbold2 Misset thelawchooses is not well-formed

(paused, type something to continue) >

         >>> declare X obj


         X : obj


         {move 3}

         >>> define thetachain X : thetachain1 \
             M thelaw, X


         thetachain : [(X_1 : obj) =>
             ({def} thetachain1 (M, thelaw, X_1) : prop)]


         thetachain : [(X_1 : obj) =>
             (--- : prop)]


         {move 2}

         >>> define Thetachain : Set (Sc \
             (Sc M), thetachain)


         Thetachain : Sc (Sc (M)) Set
          thetachain


         Thetachain : obj


         {move 2}
```

```
>>> open

   {move 4}

   >>> declare Y obj


   Y : obj


   {move 4}

   >>> declare theta1 that thetachain \
       Y


   theta1 : that thetachain (Y)


   {move 4}

   >>> declare theta2 that Y E Thetachain


   theta2 : that Y E Thetachain


   {move 4}

   >>> define thetaa1 theta1 : Iff2 \
       (Simp1 Simp2 theta1, Ui Y, Scthm \
       Sc M)


   thetaa1 : [(.Y_1 : obj), (theta1_1
       : that thetachain (.Y_1)) =>
```

```
    ({def} Simp1 (Simp2 (theta1_1)) Iff2
    .Y_1 Ui Scthm (Sc (M)) : that
    .Y_1 E Sc (Sc (M)))]


thetaa1 : [(.Y_1 : obj), (theta1_1
    : that thetachain (.Y_1)) =>
    (--- : that .Y_1 E Sc (Sc
    (M)))]


{move 3}

>>> define Theta1 theta1 : Iff2 \
    (Conj (thetaa1 theta1, theta1), Ui \
    Y, Separation4 Refleq Thetachain)


Theta1 : [(.Y_1 : obj), (theta1_1
    : that thetachain (.Y_1)) =>
    ({def} thetaa1 (theta1_1) Conj
    theta1_1 Iff2 .Y_1 Ui Separation4
    (Refleq (Thetachain)) : that
    .Y_1 E Sc (Sc (M)) Set
    thetachain)]


Theta1 : [(.Y_1 : obj), (theta1_1
    : that thetachain (.Y_1)) =>
    (--- : that .Y_1 E Sc (Sc
    (M)) Set thetachain)]


{move 3}

>>> define Theta2 theta2 : Simp2 \
    (Iff1 (theta2, Ui Y, Separation4 \
    Refleq Thetachain))
```

```
   Theta2 : [(.Y_1 : obj), (theta2_1
       : that .Y_1 E Thetachain) =>
       ({def} Simp2 (theta2_1 Iff1
       .Y_1 Ui Separation4 (Refleq
       (Thetachain))) : that
       thetachain (.Y_1))]


   Theta2 : [(.Y_1 : obj), (theta2_1
       : that .Y_1 E Thetachain) =>
       (--- : that thetachain (.Y_1))]


   {move 3}

   >>> close


{move 3}

>>> define Cutstheta1 : Cutstheta \
    Misset thelawchooses

Cutstheta Misset thelawchooses is not well-formed

(paused, type something to continue) >

>>> define Cuts : Misset Cuts3 thelawchooses


Cuts : [
    ({def} Misset Cuts3 thelawchooses
    : obj)]


Cuts : obj
```

```
          {move 2}

          >>> declare A obj


          A : obj


          {move 3}

          >>> declare B obj

B is badly formed or already reserved or declared

(paused, type something to continue) >

          >>> declare aev that A E Mbold

{declare command error}

(paused, type something to continue) >

          >>> declare bev that B E Mbold

{declare command error}

(paused, type something to continue) >

          >>> goal that (A <<= B) V B <<= \
              A


          that (A <<= B) V B <<= A


          {move 3}
```

```
          >>> define line1 aev : Fixform (Forall \
              [X => (X E Thetachain) -> A E X], Simp2 \
              (Iff1 (aev, Ui A, Separation4 \
              Refleq Mbold)))

aev : Fixform (Forall [X => (X E Thetachain) -> A E X], Simp2 (Iff1 (aev, Ui A,

(paused, type something to continue) >

          >>> define Mboldtotal aev bev : Mp \
              bev, Ui B, Simp2 (Simp2 (Iff1 \
              (Mp (Theta1 Cutstheta1, Ui Cuts, line1 \
              aev), Ui A, Separation4 Refleq \
              Cuts)))

aev bev : Mp bev, Ui B, Simp2 (Simp2 (Iff1 (Mp (Theta1 Cutstheta1, Ui Cuts, lin

(paused, type something to continue) >

          >>> define prime A : prime2 thelaw, A


          prime : [(A_1 : obj) =>
              ({def} prime2 (thelaw, A_1) : obj)]


          prime : [(A_1 : obj) => (---
              : obj)]


          {move 2}

          >>> define Mboldstrongtotal aev \
              bev : Fixform ((B <<= prime A) V A <<= \
              B, Simp2 (Separation5 Univcheat \
              (Theta1 linec17 Mp (Theta1 Cutstheta1, Ui \
              Cuts, line1 aev), line1 bev)))
```

```
aev bev : Fixform ((B <<= prime A) V A <<= B, Simp2 (Separation5 Univcheat (The

(paused, type something to continue) >

         >>> save


         {move 3}

         >>> close


      {move 2}

      >>> declare A1 obj


      A1 : obj


      {move 2}

      >>> declare B1 obj


      B1 : obj


      {move 2}

      >>> declare aev1 that A1 E Mbold

{declare command error}

(paused, type something to continue) >

      >>> declare bev1 that B1 E Mbold
```

```
{declare command error}

(paused, type something to continue) >

      >>> define Mboldtotal1 aev1 bev1 : Mboldtotal \
          aev1 bev1

aev1 bev1 : Mboldtotal aev1 bev1 is not well-formed

(paused, type something to continue) >

      >>> define Mboldstrongtotal1 aev1 bev1 \
          : Mboldstrongtotal aev1 bev1

aev1 bev1 : Mboldstrongtotal aev1 bev1 is not well-formed

(paused, type something to continue) >

      >>> save


      {move 2}

      >>> close


   {move 1}

   >>> declare A2 obj


   A2 : obj


   {move 1}

   >>> declare B2 obj
```

```
   B2 : obj


   {move 1}

   >>> declare aev2 that A2 E (Mbold2 Misset \
       thelawchooses)

{declare command error}

(paused, type something to continue) >

   >>> declare bev2 that B2 E (Mbold2 Misset \
       thelawchooses)

{declare command error}

(paused, type something to continue) >

   >>> define Mboldtotal2 Misset thelawchooses, aev2 \
       bev2 : Mboldtotal1 aev2 bev2

[Misset thelawchooses, => aev2 bev2 : Mboldtotal1 aev2 bev2] is not well-formed

(paused, type something to continue) >

   >>> define Mboldstrongtotal2 Misset thelawchooses, aev2 \
       bev2 : Mboldstrongtotal1 aev2 bev2

[Misset thelawchooses, => aev2 bev2 : Mboldstrongtotal1 aev2 bev2] is not well-

(paused, type something to continue) >
end Lestrade execution
```

We deliver results on the total linear ordering of **M** by the inclusion relation. Notice that we also prove the stronger result embodied in `Cuts2`.