```
begin Lestrade execution

                  >>> define linex14 D2 : Ug \
                      linea13

                  linex14 : [(D2_1 : obj) =>
                      (--- : that Forall
                      ([(x'_2 : obj) =>
                         ({def} ((D2_1
                         <<= Cuts2) & x'_2
                         E D2_1) -> (D2_1
                         Intersection x'_2) E Cuts2
                         : prop)]))]

                  {move 5}

                  >>> close

               {move 5}

               >>> define linex15 : Ug linex14

               linex15 : that Forall ([(x'_2
                   : obj) =>
                   ({def} Forall ([(x'_3
                      : obj) =>
                      ({def} ((x'_2 <<=
                      Cuts2) & x'_3 E x'_2) ->
                      (x'_2 Intersection
                      x'_3) E Cuts2 : prop)]) : prop)])

               {move 4}
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 linex17 bhyp : Fixform \
                (thetachain Cuts2, Conj (line19, Conj \
                (line21, Conj (line78, linex15))))

            linex17 : [(bhyp_1 : that B E Cuts) =>
                (--- : that thetachain (Mbold
                Set [(Y_3 : obj) =>
                   ({def} cutsh2 (Y_3) : prop)]))]

            {move 3}

            >>> save

            {move 4}

            >>> close

         {move 3}

         >>> declare bhyp10 that B E Cuts

         bhyp10 : that B E Cuts

         {move 3}

         >>> define linea17 bhyp10 : linex17 \
             bhyp10

         linea17 : [(.B_1 : obj), (bhyp10_1
```

```
        : that .B_1 E Cuts) => (---
        : that thetachain (Mbold Set
        [(Y_3 : obj) =>
           ({def} .B_1 cutsg2 Y_3 : prop)]))]

    {move 2}

    >>> 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

lineb17 : [(.B11_1 : obj), (bhyp11_1
    : that .B11_1 E Cuts) => (---
    : that thetachain (Mbold Set [(Y_3
       : obj) =>
       ({def} .B11_1 cutsf2 Y_3 : prop)]))]

{move 1}
```

```
   >>> 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

linec17 : [(.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)]), (.B12_1
    : obj), (bhyp12_1 : that .B12_1
    E .Misset_1 Cuts3 .thelawchooses_1) =>
    ({def} thetachain1 (.M_1, .thelaw_1, .Misset_1
    Mbold2 .thelawchooses_1 Set [(Y_4
       : obj) =>
       ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_4) : prop)]) Fi
    ((.M_1 E .Misset_1 Mbold2 .thelawchooses_1
    Set [(Y_6 : obj) =>
```

```
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_6) : prop)]) Fi
Simp1 (.Misset_1 Mboldtheta2 .thelawchooses_1) Conj
(.M_1 <<= prime2 (.thelaw_1, .B12_1)) Add2
Simp1 (bhyp12_1 Iff1 .B12_1 Ui .Misset_1
Mbold2 .thelawchooses_1 Separation
[(C_13 : obj) =>
   ({def} cuts2 (.Misset_1, .thelawchooses_1, C_13) : prop)]) Mp
.B12_1 Ui Simp1 (Simp1 (Simp2 (.Misset_1
Mboldtheta2 .thelawchooses_1))) Iff1
.B12_1 Ui Scthm (.M_1) Iff2 .M_1
Ui Separation4 (Refleq (.Misset_1
Mbold2 .thelawchooses_1 Set [(Y_9
   : obj) =>
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_9) : prop)])))
(((.Misset_1 Mbold2 .thelawchooses_1
Set [(Y_8 : obj) =>
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_8) : prop)]) <<
.Misset_1 Mbold2 .thelawchooses_1) Fixform
Separation3 (Refleq (.Misset_1 Mbold2
.thelawchooses_1)) Sepsub2 Refleq
(.Misset_1 Mbold2 .thelawchooses_1
Set [(Y_9 : obj) =>
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_9) : prop)])) T
Simp1 (Simp2 (.Misset_1 Mboldtheta2
.thelawchooses_1)) Conj linee78 (.Misset_1, .thelawchooses_1, bhyp12_1)
Ug ([(D2_6 : obj) =>
   ({def} Ug ([(F2_7 : obj) =>
      ({def} Ded ([(intev_8 : that
         (D2_6 <<= .Misset_1 Mbold2
         .thelawchooses_1 Set [(Y_12
            : obj) =>
            ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_12) :
         E D2_6) =>
         ({def} ((D2_6 Intersection
         F2_7) E .Misset_1 Mbold2
         .thelawchooses_1 Set [(Y_11
            : obj) =>
            ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_11) :
```

```
Simp1 (intev_8) Transsub
((.Misset_1 Mbold2 .thelawchooses_1
Set [(Y_17 : obj) =>
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_17) :
.Misset_1 Mbold2 .thelawchooses_1) Fixform
Separation3 (Refleq (.Misset_1
Mbold2 .thelawchooses_1)) Sepsub2
Refleq (.Misset_1 Mbold2
.thelawchooses_1 Set [(Y_18
   : obj) =>
   ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_18) :
Simp2 (intev_8) Mp F2_7
Ui D2_6 Ui Simp2 (Simp2 (Simp2
(.Misset_1 Mboldtheta2 .thelawchooses_1))) Conj
Cases (Excmid (Forall ([(K_14
   : obj) =>
   ({def} (K_14 E D2_6) ->
   .B12_1 <<= K_14 : prop)])), [(casehyp1_12
   : that Forall ([(K1_14
      : obj) =>
      ({def} (K1_14 E D2_6) ->
      .B12_1 <<= K1_14 : prop)])) =>
   ({def} ((D2_6 Intersection
   F2_7) <<= prime2 (.thelaw_1, .B12_1)) Add2
   (.B12_1 <<= D2_6 Intersection
   F2_7) Fixform Ug ([(K2_16
      : obj) =>
      ({def} Ded ([(khyp_17
         : that K2_16 E .B12_1) =>
         ({def} (K2_16 E D2_6
         Intersection F2_7) Fixform
         Simp2 (intev_8) Mp
         F2_7 Ui Ug ([(B2_23
            : obj) =>
            ({def} Ded ([(bhyp2_24
               : that B2_23
               E D2_6) =>
               ({def} khyp_17
```

```
            Mpsubs bhyp2_24
            Mp B2_23 Ui
            casehyp1_12
            : that K2_16
            E B2_23)]) : that
         (B2_23 E D2_6) ->
         K2_16 E B2_23)]) Conj
      Ug ([(B2_21 : obj) =>
         ({def} Ded ([(bhyp2_22
            : that B2_21
            E D2_6) =>
            ({def} khyp_17
            Mpsubs bhyp2_22
            Mp B2_21 Ui
            casehyp1_12
            : that K2_16
            E B2_21)]) : that
         (B2_21 E D2_6) ->
         K2_16 E B2_21)]) Iff2
      K2_16 Ui Separation4
      (Refleq (D2_6 Intersection
      F2_7)) : that K2_16
      E D2_6 Intersection
      F2_7)]) : that
   (K2_16 E .B12_1) ->
   K2_16 E D2_6 Intersection
   F2_7)]) Conj Setsinchains2
(.Misset_1, .thelawchooses_1, .Misset_1
Mboldtheta2 .thelawchooses_1, Simp1
(bhyp12_1 Iff1 .B12_1
Ui .Misset_1 Mbold2 .thelawchooses_1
Separation [(C_21 : obj) =>
   ({def} cuts2 (.Misset_1, .thelawchooses_1, C_21) : prop)]
Separation3 (Refleq (D2_6
Intersection F2_7)) : that
((D2_6 Intersection F2_7) <<=
prime2 (.thelaw_1, .B12_1)) V .B12_1
<<= D2_6 Intersection F2_7)], [(casehyp2_12
```

```
: that ~ (Forall ([(K1_15
   : obj) =>
   ({def} (K1_15 E D2_6) ->
   .B12_1 <<= K1_15 : prop)]))) =>
({def} (.B12_1 <<= D2_6
Intersection F2_7) Add1
((D2_6 Intersection F2_7) <<=
prime2 (.thelaw_1, .B12_1)) Fixform
Ug ([(K2_16 : obj) =>
   ({def} Ded ([(khyp2_17
      : that K2_16 E D2_6
      Intersection F2_7) =>
      ({def} Counterexample
      (casehyp2_12) Eg
      [(.F3_18 : obj), (fhyp3_18
         : that Counterexample
         (casehyp2_12) Witnesses
         .F3_18) =>
         ({def} Notimp2
         (fhyp3_18) Mp
         .F3_18 Ui Simp2
         (khyp2_17 Iff1
         K2_16 Ui Separation4
         (Refleq (D2_6
         Intersection F2_7))) Mpsubs
         Simp2 (Notimp2
         (fhyp3_18) Mpsubs
         Simp1 (intev_8) Iff1
         .F3_18 Ui Separation4
         (Refleq (.Misset_1
         Mbold2 .thelawchooses_1
         Set [(Y_26 : obj) =>
            ({def} cutse2
            (.Misset_1, .thelawchooses_1, .B12_1, Y_26) : pr
         Notimp1 (fhyp3_18) : that
         K2_16 E prime2
         ([(S'_20 : obj) =>
            ({def} .thelaw_1
```

```
                    (S'_20) : obj)], .B12_1))] : that
                 K2_16 E prime2 ([(S'_19
                    : obj) =>
                    ({def} .thelaw_1
                    (S'_19) : obj)], .B12_1))]) : that
              (K2_16 E D2_6 Intersection
              F2_7) -> K2_16 E prime2
              ([(S'_19 : obj) =>
                 ({def} .thelaw_1
                 (S'_19) : obj)], .B12_1))]) Conj
           Separation3 (Refleq (D2_6
           Intersection F2_7)) Conj
           Separation3 (Refleq (prime2
           (.thelaw_1, .B12_1))) : that
           ((D2_6 Intersection F2_7) <<=
           prime2 (.thelaw_1, .B12_1)) V .B12_1
           <<= D2_6 Intersection F2_7)]) Iff2
        (D2_6 Intersection F2_7) Ui
        Separation4 (Refleq (.Misset_1
        Mbold2 .thelawchooses_1 Set
        [(Y_14 : obj) =>
           ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_14) :
        (D2_6 Intersection F2_7) E .Misset_1
        Mbold2 .thelawchooses_1 Set
        [(Y_10 : obj) =>
           ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_10) :
     ((D2_6 <<= .Misset_1 Mbold2
     .thelawchooses_1 Set [(Y_11
        : obj) =>
        ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_11) : pro
     E D2_6) -> (D2_6 Intersection
     F2_7) E .Misset_1 Mbold2 .thelawchooses_1
     Set [(Y_10 : obj) =>
        ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_10) : pro
  Forall ([(x'_7 : obj) =>
     ({def} ((D2_6 <<= .Misset_1
     Mbold2 .thelawchooses_1 Set [(Y_11
        : obj) =>
```

```
            ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_11) : pro
          E D2_6) -> (D2_6 Intersection
          x'_7) E .Misset_1 Mbold2 .thelawchooses_1
          Set [(Y_10 : obj) =>
            ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_10) : pro
    thetachain1 (.M_1, .thelaw_1, .Misset_1
    Mbold2 .thelawchooses_1 Set [(Y_3
      : obj) =>
      ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_3) : prop)]))]

linec17 : [(.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)]), (.B12_1
    : obj), (bhyp12_1 : that .B12_1
    E .Misset_1 Cuts3 .thelawchooses_1) =>
    (--- : that thetachain1 (.M_1, .thelaw_1, .Misset_1
    Mbold2 .thelawchooses_1 Set [(Y_3
      : obj) =>
      ({def} cutse2 (.Misset_1, .thelawchooses_1, .B12_1, Y_3) : prop)]))]

{move 0}

>>> open

   {move 2}

   >>> define lined17 bhyp11 : linec17 \
       bhyp11

   lined17 : [(.B11_1 : obj), (bhyp11_1
       : that .B11_1 E Cuts) => (---
       : that thetachain1 (M, [(S'_2
         : obj) =>
```

```
      ({def} thelaw (S'_2) : obj)], Misset
    Mbold2 thelawchooses Set [(Y_3
      : obj) =>
      ({def} cutse2 (Misset, thelawchooses, .B11_1, Y_3) : prop)]))]

{move 1}

>>> 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

   linee17 : [(.B13_1 : obj), (bhyp13_1
       : that .B13_1 E Cuts) => (---
       : that thetachain1 (M, [(S'_2
          : obj) =>
          ({def} thelaw (S'_2) : obj)], Misset
       Mbold2 thelawchooses Set [(Y_3
          : obj) =>
          ({def} cutse2 (Misset, thelawchooses, .B13_1, Y_3) : prop)]))]

   {move 2}

   >>> open
```

```
{move 4}

>>> define Line17 bhyp : linee17 \
    bhyp

Line17 : [(bhyp_1 : that B E Cuts) =>
    (--- : that thetachain1 (M, [(S'_2
       : obj) =>
       ({def} thelaw (S'_2) : obj)], Misset
    Mbold2 thelawchooses Set [(Y_3
       : obj) =>
       ({def} cutse2 (Misset, thelawchooses, B, Y_3) : prop)]))]

{move 3}

>>> 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 linex18 khyp \
          : Ui Cuts2, Simp2 (Iff1 \
```

```
        (khyp, Ui K, Separation4 \
        Refleq Mbold))

    linex18 : [(khyp_1 : that
        K E Mbold) => (---
        : that (Cuts2 E Sc
        (Sc (M)) Set [(C_4
           : obj) =>
           ({def} thetachain1
           (M, [(S'_5 : obj) =>
              ({def} thelaw
              (S'_5) : obj)], C_4) : prop)]) ->
        K E Cuts2)]

    {move 5}

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

    linea18 : that Cuts2 E Sc
     (Sc (M))

    {move 5}

    >>> define linex19 : Fixform \
        (Cuts2 E Thetachain, Iff2 \
        (Conj (linea18, Line17 \
        bhyp), Ui Cuts2, Separation4 \
        Refleq Thetachain))

    linex19 : that Cuts2 E Thetachain

    {move 5}
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 (linex19, linex18 \
                      khyp)

                  line110 : [(khyp_1 : that
                      K E Mbold) => (---
                      : that K E Cuts2)]

                  {move 5}

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

                  line111 : [(khyp_1 : that
                      K E Mbold) => (---
                      : that (K E Mbold) & cutsi2
                      (K))]

                  {move 5}

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

                  line112 : that prime (B) <<=
                   B

                  {move 5}

                  >>> define line113 khyp \
```

```
    : Simp2 line111 khyp

line113 : [(khyp_1 : that
    K E Mbold) => (---
    : that cutsi2 (K))]

{move 5}

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

      {move 6}

      >>> close

   {move 6}

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

   line114 : [(khyp_1 : that
       K E Mbold) => (---
       : that (K <<= prime
       (B)) V prime (B) <<=
       K)]

   {move 5}

   >>> close

{move 5}

>>> define line115 K : Ded \
    line114

line115 : [(K_1 : obj) =>
```

```
      (--- : that (K_1 E Mbold) ->
      (K_1 <<= prime (B)) V prime
      (B) <<= K_1)]

   {move 4}

   >>> close

{move 4}

>>> define line116 bhyp : Ug \
    line115

line116 : [(bhyp_1 : that B E Cuts) =>
    (--- : that Forall ([(x'_2
       : obj) =>
       ({def} (x'_2 E Mbold) ->
       (x'_2 <<= prime (B)) V prime
       (B) <<= x'_2 : prop)]))]

{move 3}

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

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)))

        line117 : [(bhyp_1 : that B E Cuts) =>
            (--- : that prime (B) E Cuts)]

        {move 3}

        >>> close

      {move 3}

      >>> define line118 B : Ded line117

      line118 : [(B_1 : obj) => (---
          : that (B_1 E Cuts) -> prime
          (B_1) E Cuts)]

      {move 2}

      >>> close

   {move 2}

   >>> define Linea119 : Ug line118

   Linea119 : that Forall ([(x'_2 : obj) =>
       ({def} (x'_2 E Cuts) -> prime
       (x'_2) E Cuts : prop)])

   {move 1}

   >>> close

{move 1}

>>> define Lineb119 Misset, thelawchooses \
    : Linea119
```

```
Lineb119 : [(.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 ([(B_2 : obj) =>
       ({def} Ded ([(bhyp_3 : that
          B_2 E Misset_1 Cuts3 thelawchooses_1) =>
          ({def} (prime2 (.thelaw_1, B_2) E Misset_1
          Cuts3 thelawchooses_1) Fixform
          Simp1 (bhyp_3 Iff1 B_2 Ui Misset_1
          Mbold2 thelawchooses_1 Separation
          [(C_11 : obj) =>
             ({def} cuts2 (Misset_1, thelawchooses_1, C_11) : prop)]) Mp
          B_2 Ui Simp1 (Simp2 (Simp2
          (Misset_1 Mboldtheta2 thelawchooses_1))) Conj
          Simp1 (bhyp_3 Iff1 B_2 Ui Misset_1
          Mbold2 thelawchooses_1 Separation
          [(C_12 : obj) =>
             ({def} cuts2 (Misset_1, thelawchooses_1, C_12) : prop)]) Mp
          B_2 Ui Simp1 (Simp2 (Simp2
          (Misset_1 Mboldtheta2 thelawchooses_1))) Conj
          Ug ([(K_8 : obj) =>
             ({def} Ded ([(khyp_9 : that
                K_8 E Misset_1 Mbold2 thelawchooses_1) =>
                ({def} Cases (Simp2 ((((Misset_1
                Mbold2 thelawchooses_1
                Set [(Y_16 : obj) =>
                   ({def} cutse2 (Misset_1, thelawchooses_1, B_2, Y_16) : pr
                (Sc (.M_1)) Set [(C_16
                   : obj) =>
                   ({def} thetachain1
                   (.M_1, .thelaw_1, C_16) : prop)]) Fixform
                Simp1 (Simp2 (linec17
```

```
(bhyp_3))) Iff2 (Misset_1
Mbold2 thelawchooses_1
Set [(Y_19 : obj) =>
   ({def} cutse2 (Misset_1, thelawchooses_1, B_2, Y_19) : pr
Scthm (Sc (.M_1)) Conj
linec17 (bhyp_3) Iff2
(Misset_1 Mbold2 thelawchooses_1
Set [(Y_17 : obj) =>
   ({def} cutse2 (Misset_1, thelawchooses_1, B_2, Y_17) : pr
Separation4 (Refleq (Sc
(Sc (.M_1)) Set [(C_19
   : obj) =>
   ({def} thetachain1
   (.M_1, .thelaw_1, C_19) : prop)]))) Mp
(Misset_1 Mbold2 thelawchooses_1
Set [(Y_15 : obj) =>
   ({def} cutse2 (Misset_1, thelawchooses_1, B_2, Y_15) : pr
Simp2 (khyp_9 Iff1 K_8
Ui Separation4 (Refleq
(Misset_1 Mbold2 thelawchooses_1))) Iff1
K_8 Ui Separation4 (Refleq
(Misset_1 Mbold2 thelawchooses_1
Set [(Y_16 : obj) =>
   ({def} cutse2 (Misset_1, thelawchooses_1, B_2, Y_16) : pr
   : that K_8 <<= prime2
   (.thelaw_1, B_2)) =>
   ({def} (prime2 (.thelaw_1, B_2) <<=
   K_8) Add1 casehyp1_10
   : that (K_8 <<= prime2
   (.thelaw_1, B_2)) V prime2
   (.thelaw_1, B_2) <<=
   K_8)], [(casehyp2_10
   : that B_2 <<= K_8) =>
   ({def} (K_8 <<= prime2
   (.thelaw_1, B_2)) Add2
   ((prime2 (.thelaw_1, B_2) <<=
   B_2) Fixform Setsinchains2
   (Misset_1, thelawchooses_1, Misset_1
```

```
                    Mboldtheta2 thelawchooses_1, Simp1
                    (bhyp_3 Iff1 B_2 Ui
                    Misset_1 Mbold2 thelawchooses_1
                    Separation [(C_19
                       : obj) =>
                       ({def} cuts2 (Misset_1, thelawchooses_1, C_19) : prop)
                    Refleq (prime2 (.thelaw_1, B_2))) Transsub
                    casehyp2_10 : that (K_8
                    <<= prime2 (.thelaw_1, B_2)) V prime2
                    (.thelaw_1, B_2) <<=
                    K_8)]) : that (K_8
                 <<= prime2 (.thelaw_1, B_2)) V prime2
                 (.thelaw_1, B_2) <<=
                 K_8)]) : that (K_8
              E Misset_1 Mbold2 thelawchooses_1) ->
              (K_8 <<= prime2 (.thelaw_1, B_2)) V prime2
              (.thelaw_1, B_2) <<= K_8)]) Iff2
           prime2 (.thelaw_1, B_2) Ui
           Separation4 (Refleq (Misset_1
           Cuts3 thelawchooses_1)) : that
           prime2 (.thelaw_1, B_2) E Misset_1
           Cuts3 thelawchooses_1)]) : that
        (B_2 E Misset_1 Cuts3 thelawchooses_1) ->
        prime2 (.thelaw_1, B_2) E Misset_1
        Cuts3 thelawchooses_1)]) : that
     Forall ([(x'_2 : obj) =>
        ({def} (x'_2 E Misset_1 Cuts3
        thelawchooses_1) -> prime2 (.thelaw_1, x'_2) E Misset_1
        Cuts3 thelawchooses_1 : prop)]))]

Lineb119 : [(.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} (x'_2 E Misset_1 Cuts3
       thelawchooses_1) -> prime2 (.thelaw_1, x'_2) E Misset_1
       Cuts3 thelawchooses_1 : prop)]))]

   {move 0}

   >>> open

      {move 2}

      >>> define Line119 : Lineb119 Misset, thelawchooses

      Line119 : that Forall ([(x'_2 : obj) =>
          ({def} (x'_2 E Misset Cuts3 thelawchooses) ->
          prime2 ([(S'_5 : obj) =>
             ({def} thelaw (S'_5) : obj)], x'_2) E Misset
          Cuts3 thelawchooses : prop)])

      {move 1}
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 $\mathbf{M}$).

```
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}

   >>> comment test Ui (D4 \
       Intersection F4, Separation4 \
       Refleq Cuts)

   {move 6}

   >>> comment goal that D4 \
       Intersection F4 E Mbold

   {move 6}

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

   {move 6}

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

   line120 : that D4 <<= Mbold
```

```
{move 5}

>>> define line121 fhyp4 \
    : Mpsubs fhyp4 line120

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) => (--- : 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}

>>> comment test Simp1 \
    (testing)

{move 6}

>>> comment test Simp2 \
    (testing)

{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 : 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}

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

{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) =>
        (---
        : that
        G E B1)]

    {move
     12}

    >>> \
        close

{move
 12}

>>> define \
    line125 \
    B1 : Ded \
    line124

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) =>
    (---
    : 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) =>
    (---
    : that
    G E F4)]
```

```
{move 10}

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

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) =>
          (---
          : that
          G E D4
          Intersection
          F4)]

      {move 10}

      >>> close

   {move 10}

   >>> define \
       line130 G : Ded \
       line129

   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)])) =>
    (--- : 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)])) =>
    (--- : 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
 : 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) =>
    (---
    : that
    ~ (D5
```

```
    <<=
    .H_1))]

{move
 11}

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

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) =>
    (---
    : 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) =>
    (---
    : 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) =>
    (---
    : 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) =>
    (---
    : 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) =>
    (---
    : that
    G E D5)]
```

```
      {move
       11}

      >>> close

   {move 11}

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

   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) =>
    (--- : 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)]))) =>
    (--- : 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)]))) =>
          (--- : 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) =>
       (--- : that (D5
       <<= D4 Intersection
       F4) V (D4 Intersection
       F4) <<= D5)]

   {move 7}

   >>> close

{move 7}
```

```
   >>> define line146 D5 \
       : Ded line145

   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) => (--- : 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) =>
            (--- : 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) =>
         (--- : 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) => (--- : 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) => (---
       : 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 : 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) =>
       (--- : that D9 <<= Cuts)]

   {move 4}

   >>> define secondhyp conjhyp \
       : Simp2 conjhyp

   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) =>
       (--- : 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) =>
```

```
          (--- : 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) => (---
       : 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 : 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 : 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)

      Cutstheta2 : that thetachain (Cuts)

      {move 1}

      >>> close

   {move 1}

   >>> define Cutstheta Misset, thelawchooses \
       : Cutstheta2

   Cutstheta : [(.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} thetachain1 (.M_1, .thelaw_1, Misset_1
   Cuts3 thelawchooses_1) Fixform ((.M_1
   E Misset_1 Cuts3 thelawchooses_1) Fixform
   Simp1 (Misset_1 Mboldtheta2 thelawchooses_1) Conj
   cuts2 (Misset_1, thelawchooses_1, .M_1) Fixform
   Simp1 (Misset_1 Mboldtheta2 thelawchooses_1) Conj
   Ug ([(F_9 : obj) =>
      ({def} Ded ([(finmbold_10 : that
         F_9 E Misset_1 Mbold2 thelawchooses_1) =>
         ({def} (.M_1 <<= F_9) Add1
         finmbold_10 Mp F_9 Ui Simp1 (Simp1
         (Simp2 (Misset_1 Mboldtheta2
         thelawchooses_1))) Iff1 F_9
         Ui Scthm (.M_1) : that (F_9
         <<= .M_1) V .M_1 <<= F_9)]) : that
      (F_9 E Misset_1 Mbold2 thelawchooses_1) ->
      (F_9 <<= .M_1) V .M_1 <<= F_9)]) Iff2
   .M_1 Ui Misset_1 Mbold2 thelawchooses_1
   Separation [(C_7 : obj) =>
      ({def} cuts2 (Misset_1, thelawchooses_1, C_7) : prop)]) Conj
   ((Misset_1 Cuts3 thelawchooses_1
   <<= Misset_1 Mbold2 thelawchooses_1) Fixform
   Sepsub (Misset_1 Mbold2 thelawchooses_1, [(C_7
      : obj) =>
      ({def} cuts2 (Misset_1, thelawchooses_1, C_7) : prop)], Inhabited
   (Simp1 (Misset_1 Mboldtheta2 thelawchooses_1)))) Transsub
   (Misset_1 Mbold2 thelawchooses_1 <<=
   Sc (.M_1)) Fixform Sc2 (.M_1) Sepsub2
   Refleq (Misset_1 Mbold2 thelawchooses_1) Conj
   Misset_1 Lineb119 thelawchooses_1 Conj
   Misset_1 lineb155 thelawchooses_1 : that
   thetachain1 (.M_1, .thelaw_1, Misset_1
   Cuts3 thelawchooses_1))]

Cutstheta : [(.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 thetachain1 (.M_1, .thelaw_1, Misset_1
      Cuts3 thelawchooses_1))]

   {move 0}

   >>> 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

   Mbold : obj

   {move 2}

   >>> declare X obj

   X : obj

   {move 3}

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

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

   {move 2}

   >>> define Thetachain : Set (Sc \
       (Sc M), 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)) =>
       (--- : 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)) =>
       (--- : 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) =>
       (--- : that thetachain (.Y_1))]

   {move 3}

   >>> close

{move 3}

>>> define Cutstheta1 : Cutstheta \
    Misset, thelawchooses

Cutstheta1 : that thetachain1 (M, [(S''_2
    : obj) =>
    ({def} thelaw (S''_2) : obj)], Misset
 Cuts3 thelawchooses)

{move 2}

>>> define Cuts : Misset Cuts3 thelawchooses

Cuts : obj
```

```
{move 2}

>>> declare A obj

A : obj

{move 3}

>>> declare B obj

B : obj

{move 3}

>>> declare aev that A E Mbold

aev : that A E Mbold

{move 3}

>>> declare bev that B E Mbold

bev : that B E Mbold

{move 3}

>>> 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)))
```

```
line1 : [(.A_1 : obj), (aev_1
    : that .A_1 E Mbold) => (---
    : that Forall ([(X_2 : obj) =>
       ({def} (X_2 E Thetachain) ->
       .A_1 E X_2 : prop)]))]

{move 2}

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

Mboldtotal : [(.A_1 : obj), (.B_1
    : obj), (aev_1 : that .A_1
    E Mbold), (bev_1 : that .B_1
    E Mbold) => (--- : that (.B_1
    <<= .A_1) V .A_1 <<= .B_1)]

{move 2}

>>> define prime A : prime2 thelaw, A

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)))

Mboldstrongtotal : [(.A_1 : obj), (.B_1
    : obj), (aev_1 : that .A_1
```

```
        E Mbold), (bev_1 : that .B_1
        E Mbold) => (--- : that (.B_1
        <<= prime (.A_1)) V .A_1 <<=
        .B_1)]

    {move 2}

    >>> 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

aev1 : that A1 E Mbold

{move 2}

>>> declare bev1 that B1 E Mbold

bev1 : that B1 E Mbold

{move 2}
```

```
   >>> define Mboldtotal1 aev1 bev1 : Mboldtotal \
       aev1 bev1

   Mboldtotal1 : [(.A1_1 : obj), (.B1_1
       : obj), (aev1_1 : that .A1_1
       E Misset Mbold2 thelawchooses), (bev1_1
       : that .B1_1 E Misset Mbold2 thelawchooses) =>
       (--- : that (.B1_1 <<= .A1_1) V .A1_1
       <<= .B1_1)]

   {move 1}

   >>> define Mboldstrongtotal1 aev1 bev1 \
       : Mboldstrongtotal aev1 bev1

   Mboldstrongtotal1 : [(.A1_1 : obj), (.B1_1
       : obj), (aev1_1 : that .A1_1
       E Misset Mbold2 thelawchooses), (bev1_1
       : that .B1_1 E Misset Mbold2 thelawchooses) =>
       (--- : that (.B1_1 <<= prime2
       (thelaw, .A1_1)) V .A1_1 <<=
       .B1_1)]

   {move 1}

   >>> 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)

aev2 : that A2 E Misset Mbold2 thelawchooses

{move 1}

>>> declare bev2 that B2 E (Mbold2 Misset, thelawchooses)

bev2 : that B2 E Misset Mbold2 thelawchooses

{move 1}

>>> define Mboldtotal2 Misset, thelawchooses, aev2 \
    bev2 : Mboldtotal1 aev2 bev2

Mboldtotal2 : [(.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)]), (.A2_1
    : obj), (.B2_1 : obj), (aev2_1
    : that .A2_1 E Misset_1 Mbold2 thelawchooses_1), (bev2_1
    : that .B2_1 E Misset_1 Mbold2 thelawchooses_1) =>
    ({def} bev2_1 Mp .B2_1 Ui Simp2 (Simp2
    (Simp1 (Simp2 (Misset_1 Cutstheta
    thelawchooses_1)) Iff2 Misset_1 Cuts3
    thelawchooses_1 Ui Scthm (Sc (.M_1)) Conj
```

```
    Misset_1 Cutstheta thelawchooses_1
    Iff2 Misset_1 Cuts3 thelawchooses_1
    Ui Separation4 (Refleq (Sc (Sc (.M_1)) Set
    [(X_12 : obj) =>
       ({def} thetachain1 (.M_1, .thelaw_1, X_12) : prop)])) Mp
    Misset_1 Cuts3 thelawchooses_1 Ui Forall
    ([(X_10 : obj) =>
       ({def} (X_10 E Sc (Sc (.M_1)) Set
       [(X_13 : obj) =>
          ({def} thetachain1 (.M_1, .thelaw_1, X_13) : prop)]) ->
       .A2_1 E X_10 : prop)]) Fixform
    Simp2 (aev2_1 Iff1 .A2_1 Ui Separation4
    (Refleq (Misset_1 Mbold2 thelawchooses_1))) Iff1
    .A2_1 Ui Separation4 (Refleq (Misset_1
    Cuts3 thelawchooses_1)))) : that
    (.B2_1 <<= .A2_1) V .A2_1 <<= .B2_1)]

Mboldtotal2 : [(.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)]), (.A2_1
    : obj), (.B2_1 : obj), (aev2_1
    : that .A2_1 E Misset_1 Mbold2 thelawchooses_1), (bev2_1
    : that .B2_1 E Misset_1 Mbold2 thelawchooses_1) =>
    (--- : that (.B2_1 <<= .A2_1) V .A2_1
    <<= .B2_1)]

{move 0}

>>> define Mboldstrongtotal2 Misset, thelawchooses, aev2 \
    bev2 : Mboldstrongtotal1 aev2 bev2

Mboldstrongtotal2 : [(.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)]), (.A2_1
: obj), (.B2_1 : obj), (aev2_1
: that .A2_1 E Misset_1 Mbold2 thelawchooses_1), (bev2_1
: that .B2_1 E Misset_1 Mbold2 thelawchooses_1) =>
({def} ((.B2_1 <<= prime2 (.thelaw_1, .A2_1)) V .A2_1
<<= .B2_1) Fixform Simp2 (Separation5
(Simp1 (Simp2 (linec17 (Simp1 (Simp2
(Misset_1 Cutstheta thelawchooses_1)) Iff2
Misset_1 Cuts3 thelawchooses_1 Ui Scthm
(Sc (.M_1)) Conj Misset_1 Cutstheta
thelawchooses_1 Iff2 Misset_1 Cuts3
thelawchooses_1 Ui Separation4 (Refleq
(Sc (Sc (.M_1)) Set [(X_17 : obj) =>
   ({def} thetachain1 (.M_1, .thelaw_1, X_17) : prop)])) Mp
Misset_1 Cuts3 thelawchooses_1 Ui Forall
([(X_15 : obj) =>
   ({def} (X_15 E Sc (Sc (.M_1)) Set
   [(X_18 : obj) =>
      ({def} thetachain1 (.M_1, .thelaw_1, X_18) : prop)]) ->
   .A2_1 E X_15 : prop)]) Fixform
Simp2 (aev2_1 Iff1 .A2_1 Ui Separation4
(Refleq (Misset_1 Mbold2 thelawchooses_1)))))) Iff2
(Misset_1 Mbold2 thelawchooses_1 Set
[(Y_10 : obj) =>
   ({def} cutse2 (Misset_1, thelawchooses_1, .A2_1, Y_10) : prop)]) Ui
Scthm (Sc (.M_1)) Conj linec17
(Simp1 (Simp2 (Misset_1 Cutstheta
thelawchooses_1)) Iff2 Misset_1 Cuts3
thelawchooses_1 Ui Scthm (Sc (.M_1)) Conj
Misset_1 Cutstheta thelawchooses_1
Iff2 Misset_1 Cuts3 thelawchooses_1
Ui Separation4 (Refleq (Sc (Sc (.M_1)) Set
[(X_14 : obj) =>
```

```
      ({def} thetachain1 (.M_1, .thelaw_1, X_14) : prop)])) Mp
    Misset_1 Cuts3 thelawchooses_1 Ui Forall
    ([(X_12 : obj) =>
      ({def} (X_12 E Sc (Sc (.M_1)) Set
      [(X_15 : obj) =>
         ({def} thetachain1 (.M_1, .thelaw_1, X_15) : prop)]) ->
      .A2_1 E X_12 : prop)]) Fixform
    Simp2 (aev2_1 Iff1 .A2_1 Ui Separation4
    (Refleq (Misset_1 Mbold2 thelawchooses_1)))) Iff2
    (Misset_1 Mbold2 thelawchooses_1 Set
    [(Y_8 : obj) =>
      ({def} cutse2 (Misset_1, thelawchooses_1, .A2_1, Y_8) : prop)]) Ui
    Separation4 (Refleq (Sc (Sc (.M_1)) Set
    [(X_10 : obj) =>
      ({def} thetachain1 (.M_1, .thelaw_1, X_10) : prop)])) Univcheat
    Forall ([(X_7 : obj) =>
      ({def} (X_7 E Sc (Sc (.M_1)) Set
      [(X_10 : obj) =>
         ({def} thetachain1 (.M_1, .thelaw_1, X_10) : prop)]) ->
      .B2_1 E X_7 : prop)]) Fixform
    Simp2 (bev2_1 Iff1 .B2_1 Ui Separation4
    (Refleq (Misset_1 Mbold2 thelawchooses_1))))) : that
    (.B2_1 <<= prime2 (.thelaw_1, .A2_1)) V .A2_1
    <<= .B2_1)]

Mboldstrongtotal2 : [(.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)]), (.A2_1
    : obj), (.B2_1 : obj), (aev2_1
    : that .A2_1 E Misset_1 Mbold2 thelawchooses_1), (bev2_1
    : that .B2_1 E Misset_1 Mbold2 thelawchooses_1) =>
    (--- : that (.B2_1 <<= prime2 (.thelaw_1, .A2_1)) V .A2_1
    <<= .B2_1)]
```

```
   {move 0}
end Lestrade execution
```

We deliver results on the total linear ordering of $\mathbf{M}$ by the inclusion relation. Notice that we also prove the stronger result embodied in `Cuts2`.