* 1. **LINGO Code**

**LINGO:Dragon to the Map of China ( Multipurpose):**

MODEL:

SETS:

fc: a;

mt: b;

mr: f;

LINKS( fc, mt, mr): c, d, e;

ENDSETS

DATA:

a = 1 …1;(477 ones in total)

b = 1 …1;(477 ones in total)

f = 1 …1;(477 ones in total) d=Distance Matrix

e=Including Angle Vector Matrix

ENDDATA

MIN = @SUM( LINKS( I, J): 0.5 \* c( I, J) \* d( I, J) - 0.5 \* c(I, J) \* e( I, J));

@FOR( mt( J):

@SUM( fc( I): c( I, J)) = b(j));

@FOR( mr( J):

@SUM( fc( I): c( I, J)) = f(j));

@FOR( fc( I):

@SUM( mt( J): c( I, J)) < a( I));

@FOR( fc( I):

@SUM( mr( J): c( I, J)) < a( I));

@FOR( mt( J):

@FOR( mr( J):

@FOR( fc( I): @gin(c(i,j))));

END

**LINDO： Dragon to the Map of China, Ground to Ferris Wheel Result( Bee Nest)**

MODEL:

SETS:

fc: a;

mt: b;

LINKS( fc, mt): c, d;

ENDSETS

DATA:

a = 1 …1;(477 ones in total)

b = 1 …1;(477 ones in total)

d=Distance Matrix

ENDDATA

MIN = @SUM( LINKS( I, J): c( I, J) \* d( I, J));

@FOR( mt( J):

@SUM( fc( I): c( I, J)) = b(j));

@FOR( fc( I):

@SUM( mt( J): c( I, J)) < a( I));

@FOR( mt( J):

@FOR( fc( I): @gin(c(i,j))));

END

**LINDO： Ferris Wheel to Dragon Head**

MODEL:

SETS:

fc: a;

mt: b;

LINKS( fc, mt): c, d;

ENDSETS

DATA:

a = 1 …1;(477 ones in total)

b = 1 …1;(78 ones in total)

d=Distance Matrix

ENDDATA

MIN = @SUM( LINKS( I, J): c( I, J) \* d( I, J));

@FOR( mt( J):

@SUM( fc( I): c( I, J)) = b(j));

@FOR( fc( I):

@SUM( mt( J): c( I, J)) < a( I));

@FOR( mt( J):

@FOR( fc( I): @gin(c(i,j))));

END

**LINDO： Ferris Wheel to Dragon Tail**

MODEL:

SETS:

fc: a;

mt: b;

LINKS( fc, mt): c, d;

ENDSETS

DATA:

a = 1 …1;(399 ones in total)

b = 1 …1;(167 ones in total)

d=Distance Matrix

ENDDATA

MIN = @SUM( LINKS( I, J): c( I, J) \* d( I, J));

@FOR( mt( J):

@SUM( fc( I): c( I, J)) = b(j));

@FOR( fc( I):

@SUM( mt( J): c( I, J)) < a( I));

@FOR( mt( J):

@FOR( fc( I): @gin(c(i,j))));

END

**LINDO： Ferris Wheel to Dragon Body**

MODEL:

SETS:

fc: a;

mt: b;

LINKS( fc, mt): c, d;

ENDSETS

DATA:

a = 1 …1;(232 ones in total)

b = 1 …1;(232 ones in total)

d=Distance Matrix

ENDDATA

MIN = @SUM( LINKS( I, J): c( I, J) \* d( I, J));

@FOR( mt( J):

@SUM( fc( I): c( I, J)) = b(j));

@FOR( fc( I):

@SUM( mt( J): c( I, J)) < a( I));

@FOR( mt( J):

@FOR( fc( I): @gin(c(i,j))));

END