

```
> L1 := [[0, 3],[3, 0],[1, 2],[2, 1]];
      L1 := [[0, 3],[3, 0],[1, 2],[2, 1]] (1)
```

```
> L2 := [[0, 0], [1, 0], [0, 1], [1, 1], [0, 2], [2, 0],[0, 3],[3, 0],[1, 2],[2, 1]];
      L2 := [[0, 0], [1, 0], [0, 1], [1, 1], [0, 2], [2, 0], [0, 3], [3, 0], [1, 2], [2, 1]] (2)
```

```
> #we study the case that a_11+a_21 = b_11+b_21 = 3 and a_12+a_22 <
= 3
#already done by hand. need to double check
```

```
> p := [];
S1 := [];
S2 := [];
count := 0;
for i from 1 to nops(L1) do
  for j from 1 to nops(L1) do
    for k from 1 to nops(L2) do
      t := [];
      p := [op(L1[i]), op(L1[j]), op(L2[k])];
      #t record the condition type
      if max(p[1], p[3])<p[5] then
        t := [op(t), 1];
      fi:
      if max(p[2], p[4])<p[6] then
        t := [op(t), 2];
      fi:
      if min(p[1], p[3])>p[5] and p[5]>0 then
        t := [op(t), 3];
      fi:
      if min(p[2], p[4])>p[6] and p[6]>0 then
        t := [op(t), 4];
      fi:
      if ((has(t, 1) or has(t, 3)) and p[1]<>p[3])
        or
        ((has(t, 2) or has(t, 4)) and p[2]<>p[4]))
        and p[1]*p[3]*p[5]=0
        and p[2]*p[4]*p[6]=0
      then
        count := count + 1;
        S1 := [op(S1), p];
        S2 := [op(S2), t];
        # print(p); print(t);
      fi:
    od:
  od:
od:
count;
```

```
> sym := {};
for j from 1 to nops(S1) do
  for i from 1 to nops(S1) do
```

```

        p1 := S1[j]:  p2 := S1[i]:
        if p1[1]=p2[2] and p1[2]=p2[1] and
           p1[3]=p2[4] and p1[4]=p2[3] and
           p1[5]=p2[6] and p1[6]=p2[5] then
            sym := sym union {{j, i}}:
        fi:
    od:
od:
> S3 := []:
S4 := []:
for i from 1 to nops(sym) do
    if S2[sym[i][1]]=1 then
        S3 := [op(S3), S1[sym[i][1]]]:
        S4 := [op(S4), S2[sym[i][1]]]:
    else
        S3 := [op(S3), S1[sym[i][2]]]:
        S4 := [op(S4), S2[sym[i][2]]]:
    fi:
od:
#nops(S3);
#for i from 1 to nops(S3) do
#    print(S3[i], S4[i]);
#od:

sym2 := {}:
for j from 1 to nops(S3) do
    for i from 1 to nops(S3) do
        p1 := S3[j]:  p2 := S3[i]:
        if p1[1]=p2[3] and p1[2]=p2[4] and
           p1[3]=p2[1] and p1[4]=p2[2] and
           p1[5]=p2[5] and p1[6]=p2[6]
        then
            sym2 := sym2 union {{j, i}}:
        fi:
    od:
od:
sym2:

S5 := [seq(S3[op(1, t)], t in sym2)]:
S6 := [seq(S4[op(1, t)], t in sym2)]:

for i from 1 to nops(S5) do
    print(S5[i], S6[i]);
od:
nops(S5);
#The 3 elements are listed in the last column of Table 3
#For instance, the first element means [a11, a21, b11, b21, a12,
a22] = [0, 3, 1, 2, 2, 0]
                                [0, 3, 1, 2, 2, 0], [1]
                                [0, 3, 1, 2, 3, 0], [1]
                                [0, 3, 2, 1, 3, 0], [1]
                                    3
> flag := [1$nops(S5)]:

```

```

for i from 1 to nops(S5) do
    if has(S6[i], 1) and S5[i][6]=0
        and (S5[i][3]-S5[i][1])*(S5[i][4]-S5[i][2])<0 then
        flag[i] := 0:
    fi:
    if has(S6[i], 2) and S5[i][5]=0
        and (S5[i][3]-S5[i][1])*(S5[i][4]-S5[i][2])<0 then
        flag[i] := 0:
    fi:
    if flag[i]=1 then
        print(S5[i], S6[i]);
    fi:
    #print(S5[i], S6[i]);
od:
flag;
#This is a program to check if  $\pi_{\sigma}^{-1}(\sigma) \cap C = \emptyset$  in a simple way.
#For the above 3 elements,  $\pi_{\sigma}^{-1}(\sigma) \cap C$  are
all empty.

```

[0, 0, 0]

(5)