Kalimdor

Ryan Mouw

January 28, 2021

1 Welcome to Kalimdor

Welcome to Kalimdor. Kalimdor is one of two original continents that exists in the massive multiplayer online role playing game, World of Warcraft (WoW). Over the years, WoW has seen many new additions (Addons) created by the community that plays the game. One such addon was a more detailed and interactive map than the one that the original development team had created. Below is the original map.

As you can see, there are different regions in Kalimdor, but each region is only distinguished by its boarder, with maybe a slight change in coloration. Kalimdor is a very exciting and diverse place, and deserves a map that reflects it.

2 The Problem

The amateur game developers in the community have approached us and asked for our help in designing the new map.

The first thing the developers ask of us is to come up with a map that uses as few colors as possible, while ensuring that each area in Kalimdor does not share a color with one of its neighbors.

Before we apply the developer's constraint, we first number the regions within Kalimdor.

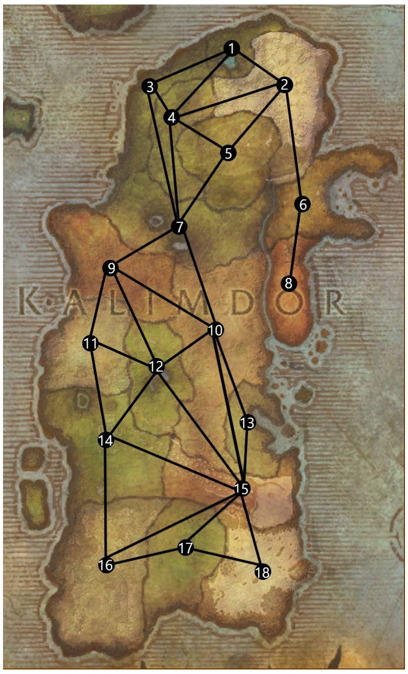
[1]

We call the numbered circle in the center of each region a vertex. Next, we draw lines between two vertices if the regions they represent share a border. We call these edges. We will create a set of edges (E) between the vertex pairs v_i and v_j, if the corresponding regions share a border. E is shown below.

$$
(from\ Map\ 2) \quad where\ E =
\begin{gathered}
\{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_4\}, \\
\{v_2, v_4\}, \{v_2, v_5\}, \{v_2, v_6\}, \\
\{v_3, v_4\}, \{v_3, v_7\}, \\
\{v_4, v_5\}, \{v_4, v_7\}, \\
\{v_5, v_7\}, \\
\{v_6, v_8\}, \\
\{v_7, v_9\}, \{v_7, v_{10}\}, \\
\{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_9, v_{12}\}, \\
\{v_{10}, v_{12}\}, \{v_{10}, v_{13}\}, \{v_{10}, v_{15}\}, \\
\{v_{11}, v_{12}\}, \{v_{11}, v_{14}\}, \\
\{v_{12}, v_{14}\}, \{v_{12}, v_{15}\}, \\
\{v_{13}, v_{15}\}, \\
\{v_{14}, v_{15}\}, \{v_{14}, v_{16}\}, \\
\{v_{15}, v_{16}\}, \{v_{15}, v_{17}\}, \{v_{15}, v_{18}\}, \\
\{v_{16}, v_{17}\}, \\
\{v_{17}, v_{18}\}\}
\end{gathered}
$$

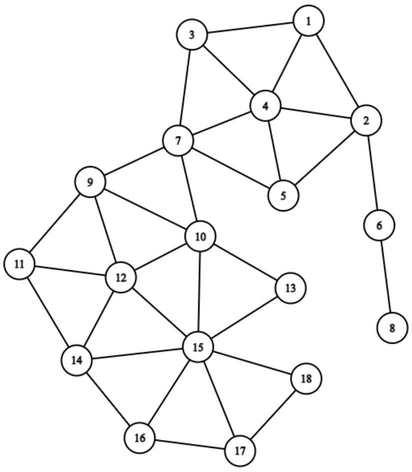
[9]

The edges and vertices are shown below.



[2]

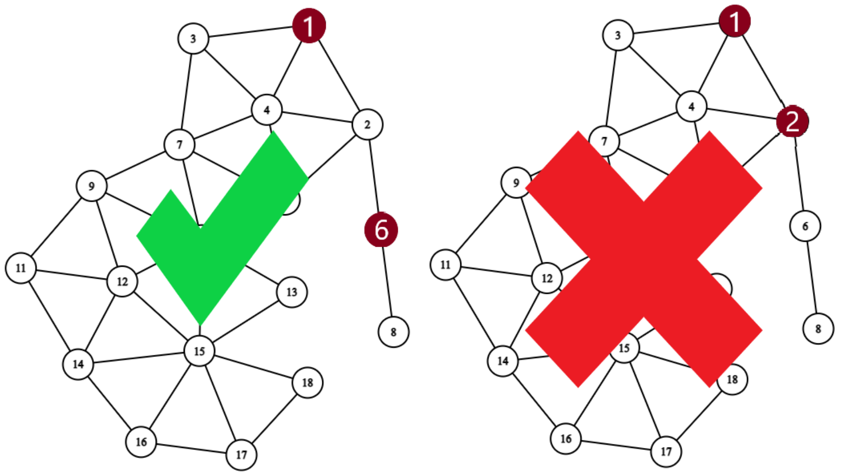
We are now ready to represent our map as a graph. A graph is a set of vertices and edges. Each edge must have a vertex at either end, but a vertex does not need to have an edge. The corresponding graph of map 2 is below.



[3]

The developer's constraint can now be stated in a different way. No two vertices that share an edge can be the same color.

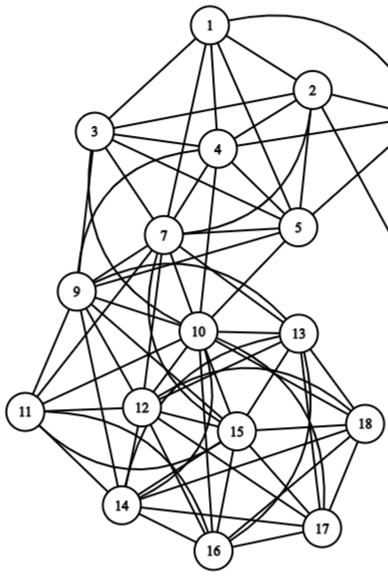
For example, vertex 1 and vertex 2 cannot share the same color, but vertex 1 and vertex 6 can share a color. Reference Map 2 to verify that region 1 and region 6 do not share a border.



The second part of the problem requires that any two regions that border the same region cannot share a color. To represent this, we create a new graph in which we add an edge between any two vertices that are connected to a similar vertex. The additional set of edges is shown below.

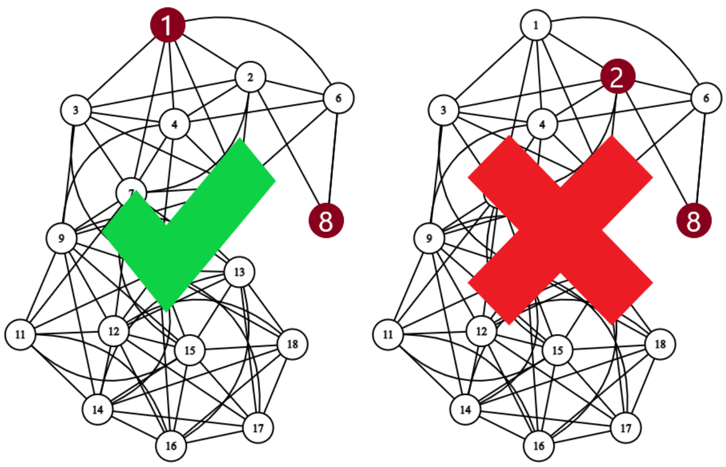
$$\begin{gathered}
\textit{Additional pairs of vertices} = \\
\{\{v_1, v_5\}, \{v_1, v_6\}, \{v_1, v_7\}, \\
\{v_2, v_3\}, \{v_2, v_7\}, \{v_2, v_8\}, \\
\{v_3, v_5\}, \{v_3, v_9\}, \{v_3, v_{10}\}, \\
\{v_4, v_6\}, \{v_4, v_9\}, \{v_4, v_{10}\}, \\
\{v_5, v_6\}, \{v_5, v_9\}, \{v_5, v_{10}\}, \\
\{v_6, v_8\}, \\
\{v_7, v_1 1\}, \{v_7, v_1 2\},, \{v_7, v_1 3\}, \{v_7, v_{15}\}, \\
\{v_9, v_{13}\}, \{v_9, v_{14}\}, \{v_9, v_{15}\}, \\
\{v_{10}, v_{11}\}, \{v_{10}, v_{14}\}, \{v_{10}, v_{16}\}, \{v_{10}, v_{17}\}, \{v_{10}, v_{18}\}, \\
\{v_{11}, v_{15}\}, \{v_{11}, v_{16}\}, \\
\{v_{12}, v_{13}\}, \{v_{12}, v_{16}\}, \{v_{12}, v_{17}\}, \{v_{12}, v_{18}\}, \\
\{v_{13}, v_{14}\}, \{v_{13}, v_{16}\}, \{v_{13}, v_{17}\}, \{v_{13}, v_{18}\}, \\
\{v_{14}, v_{17}\}, \{v_{14}, v_{18}\} \\
\{v_{16}, v_{18}\}\}
\end{gathered} \tag{10}$$

Below is graph 4, which has incorporated the additional edges.



[4]

Clearly this graph becomes quite busy, but the same rule as before still applies. No two vertices that share an edge can share a color.



To solve this problem we will use linear programming.

3 Objective Functions and Constraints

We need to know the particular color that each region needs to be in order to satisfy the developer's constraints.

First we introduce a set of variables x_{ik} where $i = 1, 2, ..., 18$ for each of the 18 regions in mainland Kalimdor and $k = 1, 2, ..., 18$ for up to 18 different colors. If $x_{ik} = 1$ then region i will have color k. If $x_{ik} = 0$ then region i will **NOT** have color k.

Note: Notice that an immediate solution is to color each region with a unique color. This would require 18 colors because there are 18 regions. This fact is what gives both i and k the upper bound of 18.

Each region can only be assigned a single color. To enforce this, we will require that the sum of x_{ik} for $i = 1, 2, ..., 18$ be equal to 1.

$$\sum_{k=1}^{18} x_{ik} = 1;\ for\ i = 1, 2, 3, ..., 18 \qquad [5]$$

The LPSolve equations will then take on the form;

```
+x_1_1 +x_1_2 +x_1_3...+x_1_18 = 1;
+x_2_1 +x_2_2 +x_2_3...+x_2_18 = 1;
...
...
...
+x_18_1 +x_18_2 +x_18_3...+x_18_18 = 1;
```

Next we will create a set of variables that indicate whether or not a certain color is used. As stated before, the maximum number of colors needed is 18. We will create a set of variables y_i where $i = 1, 2, 3..., 18$. If $y_i = 1$ then the color i is used. If $y_i = 0$ then the color i is **NOT** used. By minimizing the sum of y_i we can determine the minimum number colors needed to color the map. This will be our objective function.

$$min : \sum_{i=1}^{18} y_i \qquad [6]$$

The LPSolve objective function will then take on the form;

```
min:  +y_1 +y_2 +y_3...+y_18;
```

To enforce the binary nature of the x_{ik} and y_i variable in LPSolve, we must list the variables with the appropriate row header, "*bin*." The LPSolve constraint will then take on the form;

```
bin  y_1, x_1_1, x_1_2, x_1_3,...y_18, x_18_1...x_18_16, x_18_17, x_18_18;
```

Next, we need a way to connect the x_{ik} and y_i variables. We can imply y_i from x_{ik} with a simple inequality.

$$x_{ik} \leq y_i;\ for\ i, k = 1, 2, 3..., 18 \qquad [7]$$

This inequality means that if $x_{ik} = 1$ then $y_k = 1$. This is how we connect the x_{ik} and y_i variables.

The LPSolve inequalities will then take on the form;

```
+x_1_1 <= +y_1;
+x_1_2 <= +y_2;
+x_1_3 <= +y_3;
...
...
...
+x_18_16 <= +y_16;
+x_18_17 <= +y_17;
+x_18_18 <= +y_18;
```

Next we must create an inequality that prohibits any two vertices that share an edge from having the same color k. To do this we will create a set of edges (E) between the appropriate vertex pairs v_i and v_j, where vertices v_i and v_j share an edge. We can ensure that v_i and v_j do not have the same color by requiring the x_{ik} and x_{jk} variables associated with the vertex pairs in E are less than or equal to 1.

$$x_{ik} + x_{jk} \leq 1;\ for\ all\ (v_i, v_j) \in E\ and\ k = 1, 2, 3..., 18 \qquad [8]$$

The LPSolve inequalities then take on the form;

```
+x_1_1 +x_2_1 <= 1;
+x_1_1 +x_3_1 <= 1;
+x_1_1 +x_4_1 <= 1;
...
...
...
+x_16_18 +x_17_18 <= 1;
+x_16_18 +x_18_18 <= 1;
+x_17_18 +x_18_18 <= 1;
```

The next constraint is not necessary to solve the problem, but helps LPSolve reduce the number of options that need to be computed. We will require that the smallest index of colors (k) be used. So if only four colors are necessary to complete the entire map, then LPSolve will only use colors $k = 1, 2, 3, 4$. To implement this, we will require that color k must be used before color $k + 1$. This is expressed in the inequality below.

$$y_k \leq y_{k-1},\ where\ k = 1, 2, 3..., 18 \qquad [11]$$

The final set of LPSolve inequalities then take on the form;

```
+y_2 <= +y_1;
+y_3 <= +y_2;
```

```
+y_4 <= +y_3;
...
...
...
+y_16 <= +y_15;
+y_17 <= +y_16;
+y_18 <= +y_17;
```

The final LPSolve input file will then look as follows;

```
min:  +y_1 +y_2 +y_3...+y_18;

+x_1_1 +x_1_2 +x_1_3...+x_1_18 = 1;
+x_2_1 +x_2_2 +x_2_3...+x_2_18 = 1;
...
...
...
+x_18_1 +x_18_2 +x_18_3...+x_18_18 = 1;

+x_1_1 <= +y_1;
+x_1_2 <= +y_2;
+x_1_3 <= +y_3;
...
...
...
+x_18_16 <= +y_16;
+x_18_17 <= +y_17;
+x_18_18 <= +y_18;

+x_1_1 +x_2_1 <= 1;
+x_1_1 +x_3_1 <= 1;
+x_1_1 +x_4_1 <= 1;
...
...
...
+x_16_18 +x_17_18 <= 1;
+x_16_18 +x_18_18 <= 1;
+x_17_18 +x_18_18 <= 1;

+y_2 <= +y_1;
+y_3 <= +y_2;
+y_4 <= +y_3;
...
...
...
+y_16 <= +y_15;
+y_17 <= +y_16;
+y_18 <= +y_17;
```

```
bin  y_1, x_1_1, x_1_2, x_1_3,...y_18, x_18_1...x_18_17, x_18_18;
```

4 Solutions

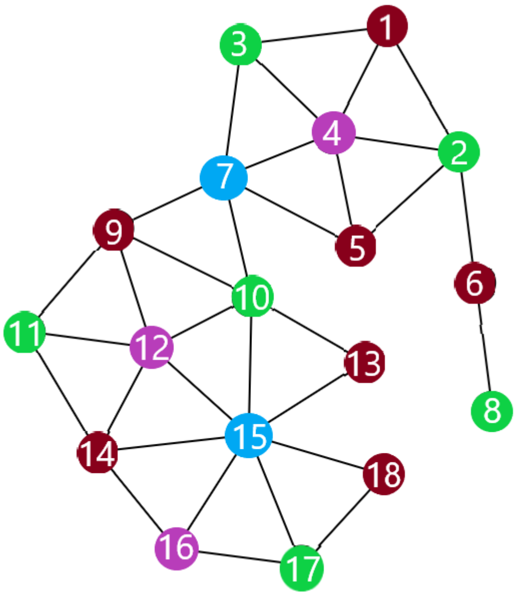
The first problem involved the unaugmented E, and produced the following output file;

```
Value of objective function: 4.00000000

Actual values of the variables:
y_1                                1
y_2                                1
y_3                                1
y_4                                1
x_1_1                              1
x_2_2                              1
x_3_2                              1
x_4_3                              1
x_5_1                              1
x_6_1                              1
x_7_4                              1
x_8_2                              1
x_9_1                              1
x_10_2                             1
x_11_2                             1
x_12_3                             1
x_13_1                             1
x_14_1                             1
x_15_4                             1
x_16_3                             1
x_17_2                             1
x_18_1                             1
```

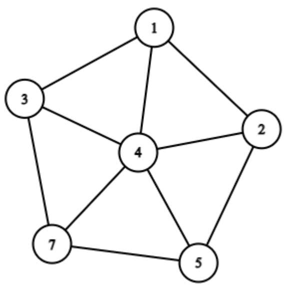
Note: Only nonzero variables are shown in the output file. All other variables are suppressed.

The output file indicates that four colors (y_1, y_2, y_3 and y_4) are used to complete the map. The x_{ik} variables shown indicate where each color should be placed. The solution is shown below.

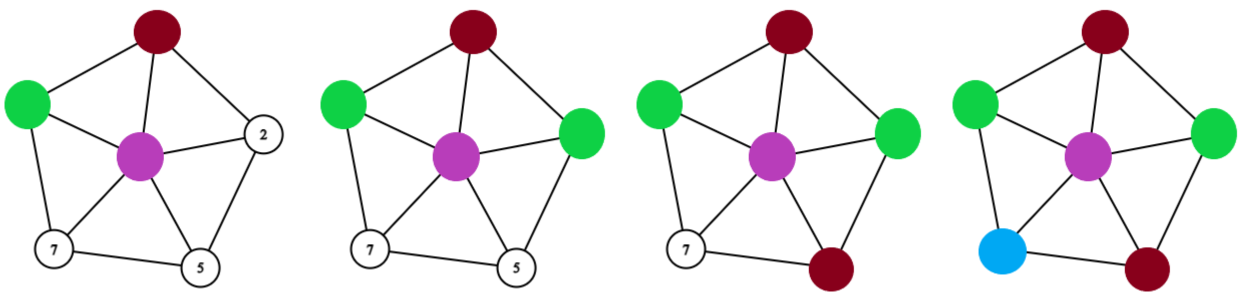


[12]

A way to verify that graph 3 requires at least 4 colors is to show that a subgraph of graph 3 requires at least 4 colors. Below is a subgraph of graph 3.



To color this subgragh, we first pick three connected vertices and give them each a color. We then go around the "wheel" and alternate colors, until the last vertex is left. It is clear that the last vertex must be an entirely new color in order to not share an edge with a vertex of similar color.



This subgraph requires at least four colors, therefore graph 3 will require at least four colors. Since we found a solution for graph 3 with four colors, we know we have the best possible solution.

Below is graph 3 superimposed onto the map of Kalimdor. Verify that no two regions that share a border have the same color.

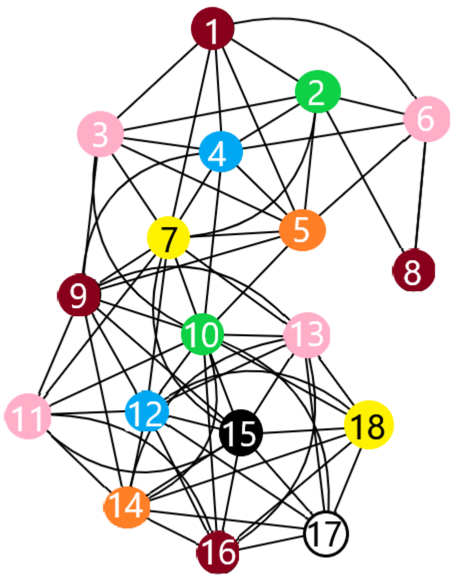
The second problem was a bit more complicated and involved added constraints in the form of vertex pairs. The output file is shown below.

```
    Value of objective function: 8.00000000

Actual values of the variables:
y_1                             1
y_2                             1
y_3                             1
y_4                             1
y_5                             1
y_6                             1
y_7                             1
y_8                             1
x_1_1                           1
x_2_2                           1
x_3_3                           1
x_4_4                           1
x_5_5                           1
x_6_3                           1
x_7_6                           1
x_8_1                           1
x_9_1                           1
x_10_2                          1
x_11_3                          1
x_12_4                          1
```

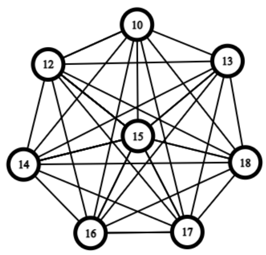
x_13_3	1
x_14_5	1
x_15_8	1
x_16_1	1
x_17_7	1
x_18_6	1

As can be seen, y_1 through y_8 have been given a nonzero value, indicating that 8 different colors were necessary to fill in this graph. An image of the graph based on the above output is provided. Feel free to verify the solution below.



[13]

An interesting way to verify that graph 4 requires at least 8 colors is to look at one of it's subgraphs, shown below;



This is a complete subgraph of graph 4, with 8 vertices. This means we will need 8 colors to fill this subgraph, and therefore will need at least 8 colors to fill in graph 4. Because we found a solution for graph 4 with 8 colors, we now know we have the best possible solution.

Also, feel free to verify that no two regions who share a border with a common region share a color in the map below.


K·A·L·I·M·D·O·R

5 Python Code

The below code produces and LPSolve input file.

```
    ## Objective Function

#y_i i=1:18 represents 18 colors that could be used.
obj = ""
for i in range(18):
    obj=obj + " +y_"+str(i+1)
obj = "min: " + obj + ";"

## Constraint 0

# This will ensure that every vertex, i, will have exactly one color
con_0 = ""
for i in range(18):
    for k in range(18):
        con_0=con_0 + " +x_"+str(i+1)+"_"+str(k+1)
    con_0=con_0 + " = 1;" + "\n"



## Constraint 1

# This connects the x variable to the y variable
con_1 = ""
for i in range(18):
    for k in range(18):
        con_1=con_1 + " +x_"+str(i+1)+"_"+str(k+1)+" <= "+"+y_"+str(k+1)+";"+"\n'



## Constraint 2

#This loop will create coloring constraints for all vertices that are in my edge
#There are 31 eges in my edge set, so 31 pairs of vertices.
con_2=""
for k in range(18):
    #v1
    # v1 + v2
    con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + "
    +x_" + str(2) + "_" + str(k+1)
    + " <= 1;"+"\n"
    #v1 + v3
    con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + "
    +x_" + str(3) + "_" + str(k+1)
```

```
+ " <= 1;" +"\n"
#v1 + v4
con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + "
+x_" + str(4) + "_" + str(k+1) + " <= 1;" +"\n"


#added constraints from v2 to v1
#v1 + v5
con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + "
+x_" + str(5) + "_" + str(k+1) + " <= 1;" +"\n"
#v1 + v6
con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + "
+x_" + str(6) + "_" + str(k+1) + " <= 1;" +"\n"


#added constraints from v3 to v1
#v1 + v7
con_2= con_2 + " +x_" + str(1) + "_" + str(k+1) + " +x_"
+ str(7) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v4 to v1
#none that havent already been satisfied



#v2
#v2 + v4
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + "
+x_" + str(4) + "_" + str(k+1) + " <= 1;" +"\n"
#v2 + v5
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + " +x_"
+ str(5) + "_" + str(k+1) + " <= 1;" +"\n"
#v2 + v6
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + "
+x_" + str(6) + "_" + str(k+1) + " <= 1;" +"\n"


#added constraints from v1 to v2
#v2 + v3
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + " +x_"
+ str(3) + "_" + str(k+1) + " <= 1;" +"\n"
#added constraints from v4 to v2
#v2 + v7
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + "
+x_" + str(7) + "_" + str(k+1) + " <= 1;" +"\n"
#added constraints from v6 to v2
#v2 + v8
con_2= con_2 + " +x_" + str(2) + "_" + str(k+1) + " +x_"
```

```
+ str(8) + "_" + str(k+1) + " <= 1;" +"\n"



#v3
#v3 + v4
con_2= con_2 + " +x_" + str(3) + "_" + str(k+1) + "
+x_" + str(4) + "_" + str(k+1) + " <= 1;" +"\n"
#v3 + v7
con_2= con_2 + " +x_" + str(3) + "_" + str(k+1) + "
+x_" + str(7) + "_" + str(k+1) + " <= 1;" +"\n"


#added constraints from v4 to v3
#v3 + v5
con_2= con_2 + " +x_" + str(3) + "_" + str(k+1) + "
+x_" + str(5) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v7 to v3
#v3 + v9
con_2= con_2 + " +x_" + str(3) + "_" + str(k+1) + "
+x_" + str(9) + "_" + str(k+1) + " <= 1;" +"\n"
#v3 + v10
con_2= con_2 + " +x_" + str(3) + "_" + str(k+1) + " +x_"
+ str(10) + "_" + str(k+1) + " <= 1;" +"\n"




#v4
#v4 + v5
con_2= con_2 + " +x_" + str(4) + "_" + str(k+1) + " +x_"
+ str(5) + "_" + str(k+1) + " <= 1;" +"\n"
#v4 + v7
con_2= con_2 + " +x_" + str(4) + "_" + str(k+1) + " +x_"
+ str(7) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v2 to v4
#v4 + v6
con_2= con_2 + " +x_" + str(4) + "_" + str(k+1) + " +x_"
+ str(6) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v7 to v4
#v4 + v9
con_2= con_2 + " +x_" + str(4) + "_" + str(k+1) + " +x_"
+ str(9) + "_" + str(k+1) + " <= 1;" +"\n"
```

```
#v4 + v10
con_2= con_2 + " +x_" + str(4) + "_" + str(k+1) + " +x_"
+ str(10) + "_" + str(k+1) + " <= 1;" +"\n"




#v5
#v5 + v7
con_2= con_2 + " +x_" + str(5) + "_" + str(k+1) + " +x_"
+ str(7) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v2 to v5
#v5 + v6
con_2= con_2 + " +x_" + str(5) + "_" + str(k+1) + " +x_"
+ str(6) + "_" + str(k+1) + " <= 1;" +"\n"
#added constraints from v7 to v5
# v5 + v9
con_2= con_2 + " +x_" + str(5) + "_" + str(k+1) + "
+x_" + str(9) + "_" + str(k+1) + " <= 1;"+"\n"
# v5 + v10
con_2= con_2 + " +x_" + str(5) + "_" + str(k+1) + " +x_"
+ str(10) + "_" + str(k+1) + " <= 1;"+"\n"



#v6
#v6 + v8
con_2= con_2 + " +x_" + str(6) + "_" + str(k+1) + "
+x_" + str(8) + "_" + str(k+1) + " <= 1;" +"\n"



#v7
#v7 + v9
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1) + " +x_"
+ str(9) + "_" + str(k+1) + " <= 1;" +"\n"
#v7 + v10
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1) + "
+x_" + str(10) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v9 to v7
```

```
# v7 + v11
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1) + "
+x_" + str(11) + "_" + str(k+1) + " <= 1;"+"\n"
# v7 + v12
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1) + "
+x_" + str(12) + "_" + str(k+1) + " <= 1;"+"\n"

#added constraints from v10 to v7
# v7 + v13
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1)
+ " +x_" + str(13) + "_" + str(k+1) + " <= 1;"+"\n"
# v7 + v15
con_2= con_2 + " +x_" + str(7) + "_" + str(k+1) + "
+x_" + str(15) + "_" + str(k+1) + " <= 1;"+"\n"




#v9
#v9 + v10
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(10) + "_" + str(k+1) + " <= 1;" +"\n"
#v9 + v11
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(11) + "_" + str(k+1) + " <= 1;" +"\n"
#v9 + v12
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(12) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v10 to v9
# v9 + v13
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(13) + "_" + str(k+1) + " <= 1;"+"\n"

#added constraints from v11 to v9
# v9 + v14
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(14) + "_" + str(k+1) + " <= 1;"+"\n"

#added constraints from v12 to v9
# v9 + v15
con_2= con_2 + " +x_" + str(9) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;"+"\n"
```

```
#v10
#v10 + v12
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(12) + "_" + str(k+1) + " <= 1;" +"\n"
#v10 + v13
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(13) + "_" + str(k+1) + " <= 1;" +"\n"
#v10 to 15
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v12 to v10
# v10 + v11
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(11) + "_" + str(k+1) + " <= 1;"+"\n"
# v10 + v14
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(14) + "_" + str(k+1) + " <= 1;"+"\n"

#added constraints from v15 to v10
# v10 + v16
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;"+"\n"
# v10 + v17
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;"+"\n"
# v10 + v18
con_2= con_2 + " +x_" + str(10) + "_" + str(k+1)
+ " +x_" + str(18) + "_" + str(k+1) + " <= 1;"+"\n"




#v11
#v11 + v12
con_2= con_2 + " +x_" + str(11) + "_" + str(k+1)
+ " +x_" + str(12) + "_" + str(k+1) + " <= 1;" +"\n"
#v11 + v14
con_2= con_2 + " +x_" + str(11) + "_" + str(k+1)
+ " +x_" + str(14) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v12 to v11
# v11 + v15
con_2= con_2 + " +x_" + str(11) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;"+"\n"
#added constraints from v14 to v11
# v11 + v16
con_2= con_2 + " +x_" + str(11) + "_" + str(k+1)
```

```
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;"+"\n"



#v12
#v12 + v14
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(14) + "_" + str(k+1) + " <= 1;" +"\n"
#v12 + v15
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v14 to v12
# v12 + v16
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;"+"\n"
#added constraints from v15 to v12
# v12 + v13
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(13) + "_" + str(k+1) + " <= 1;"+"\n"
# v12 + v17
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;"+"\n"
# v12 + v18
con_2= con_2 + " +x_" + str(12) + "_" + str(k+1)
+ " +x_" + str(18) + "_" + str(k+1) + " <= 1;"+"\n"




#v13
#v13 + v15
con_2= con_2 + " +x_" + str(13) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v15 to v13
# v13 + v14
con_2= con_2 + " +x_" + str(13) + "_" + str(k+1)
+ " +x_" + str(14) + "_" + str(k+1) + " <= 1;"+"\n"
# v13 + v16
con_2= con_2 + " +x_" + str(13) + "_" + str(k+1)
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;"+"\n"
# v13 + v17
con_2= con_2 + " +x_" + str(13) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;"+"\n"
# v13 + v18
con_2= con_2 + " +x_" + str(13) + "_" + str(k+1)
```

```
+ " +x_" + str(18) + "_" + str(k+1) + " <= 1;"+"\n"




#v14
#v14 + v15
con_2= con_2 + " +x_" + str(14) + "_" + str(k+1)
+ " +x_" + str(15) + "_" + str(k+1) + " <= 1;" +"\n"
#v14 + v16
con_2= con_2 + " +x_" + str(14) + "_" + str(k+1)
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;" +"\n"

#added constraints from v15 to v14
# v14 + v17
con_2= con_2 + " +x_" + str(14) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;"+"\n"
# v14 + v18
con_2= con_2 + " +x_" + str(14) + "_" + str(k+1)
+ " +x_" + str(18) + "_" + str(k+1) + " <= 1;"+"\n"




#v15
#v15 + v16
con_2= con_2 + " +x_" + str(15) + "_" + str(k+1)
+ " +x_" + str(16) + "_" + str(k+1) + " <= 1;" +"\n"
#v15 + v17
con_2= con_2 + " +x_" + str(15) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;" +"\n"
#v15 + v18
con_2= con_2 + " +x_" + str(15) + "_" + str(k+1) + " +x_"
+ str(18) + "_" + str(k+1) + " <= 1;" +"\n"




#v16
#v16 + v17
con_2= con_2 + " +x_" + str(16) + "_" + str(k+1)
+ " +x_" + str(17) + "_" + str(k+1) + " <= 1;" +"\n"
#added constraints from v17 to v16
#v16 + v18
con_2= con_2 + " +x_" + str(16) + "_" + str(k+1)
+ " +x_" + str(18) + "_" + str(k+1) + " <= 1;" +"\n"




#v17
```

```
    #v17 + v18
    con_2= con_2 + " +x_" + str(17) + "_" + str(k+1)
    + " +x_" + str(18) + "_" + str(k+1) + " <= 1;" +"\n"


## Constraint 3

# This constraint will limit the color space, requiring that the next color in th
con_3=""
for k in range(2,19,1):
    con_3 = con_3 + " +y_"+str(k)+ " <= "+ "+y_"+str(k-1)+";"+"\n"

## Constraint 4

#y_i i=1:18 represents 18 colors that could be used.
con_4 = ""
for i in range(18):
    con_4=con_4 + " +y_"+str(i+1)
con_4 = con_4 + ">=8;"

## Binary constraint

con_bin = ""
for i in range(1,19,1):
    con_bin = con_bin + " y_"+str(i)+", "
    for k in range(1,19,1):
        con_bin = con_bin + "x_"+str(i)+"_"+str(k)+ ", "
con_bin = "bin "+con_bin[:-2] + ";" +"\n"

with open("kalimdor_added_constraint_1_2.txt", "w") as text_file:
    print(obj, file = text_file)
    print(con_0, file = text_file)
    print(con_1, file = text_file)
    print(con_2, file = text_file)
    print(con_3, file = text_file)
    print(con_4, file = text_file)
    print(con_bin, file = text_file)
```