Graph Reconstruction Conjecture Project Final Report

Gabriel Diaz, Richard Seddon, David Schmitt

10th December 2017

1 Background

1.1 Purpose

The purpose of this project is to make progress towards determining the legitimacy of the graph reconstruction conjecture, and eventually arriving at a proof or counter-example for it.This group primarily worked at using computer programs to gather data for analysis, and cross examining this data with manually determined data from other groups.

1.2 The Graph Reconstruction Conjecture

Given a simple graph, a card is generated by removing one vertex and all edges connected to that vertex from this simple graph. The deck of a simple graph is the set of all its cards. The graph reconstruction conjecture states that it is possible to reconstruct the original graph using only the deck, with no knowledge of the original graph.

1.3 Kocay Graphs

This project is studying a special case of simple graphs called Kocay Graphs, which are known to be relatively difficult to reconstruct. A Kocay graph is generated by taking a graph of degree n (meaning each vertex is connected to n other vertices) and removing one edge from it. Thus, if k is the number of vertices, Kocay graphs have the degree sequence

$$(n-1)^2, n^{k-2}$$

More specifically, this project is examining Kocay graphs that are generated from the nineteen cubic (having degree three) graphs on ten vertices.

1.4 Labeling

For specific graphs, we will use the following notation: A graphs label starts with either a G, an H, or a V. G means one of the original nineteen cubic graphs. H means a Kocay graph generated from one of the original graphs. V means a card generated from one of the Kocay graphs. The letter is followed by a four digit number, where the first two digits indicate the number of vertices (which is ten for all of the graphs used here) and the second two digits indicate which of the original 19 graphs this graph is generated from, ranging from 0 to 18. If the graph is a Kocay graph, it is then followed by the letter e and two digits. The two digits indicated the vertices of the edge that was deleted to create this Kocay graph. If the graph is a card, it is then followed by an underscore and one more digits, indicating which vertex was removed to create it. Examples:

$$G1001$$

$$H1001e01$$

$$V1001e01\_3$$

Meaning the second cubic graph, the Kocay graph created by removing the edge from vertex 0 to vertex 1 from that cubic graph, and the card created by removing vertex 3 from that Kocay graph, respectively.

1.5 Graph6 Format

This group determined that it would be useful to stored the graphs in graph6 format. Graph6 format is a way of storing graphs as a string of printable ASCII characters. Using SageMath, a graph can be constructed from its graph6 string, and a graph6 string can be generated from any graph. A detailed explanation of graph6 format can be found here. *Note: Graph6 format does preserve labels, or at least the order of them. Since the graph6 listed is the string for the first occurrence of that graph, the labels will be the labels of that first occurrence.*

1.6 Other Terms

- Isomorphic: If two graphs are isomorphic to each other, they have the same structure but may have different labels.
- Cubic Vertex: A cubic vertex is a vertex of degree three.
- Bump: A bump is a vertex of degree two.

- Similar Vertices: Similar vertices are two vertices in a Kocay graph that are not isomorphic but do generate the same card when deleted.

2 Early Work: Examining Cubic Kocay Graphs on 10 Vertices

2.1 Basic Data

First, we worked on generating all the graphs we would be working with. We did this using SageMath by taking the original nineteen cubic graphs, generating all of their Kocay graphs by removing the making removing each edge, then generating all the cards from these graphs by removing each vertex from them. We found 285 Kocay graphs, and 2850 cards originating from them. Contained in this folder is three lists including the graphs label, its graph6 string, and its degree sequence: "MainList.pdf" contains the original nineteen cubic graphs, "KocayList.pdf" contains the 285 Kocay graphs, and "CardList.pdf" contains the 2850 cards.

2.2 Isomorphisms

We then used these lists to group these graphs based on their isomorphisms to each other, as many of the graphs we generated were likely copies of each other and therefore not very interesting. Checking all the Kocay graphs against each other and all of the cards against eacho ther, we determined that there were 91 unique Kocay graphs and 240 unique cards. Contained in this folder are the files "UniqueKocayList.txt" and "UniqueCardList.txt", which give the unique graphs a label (c0-c239 for cards, h0-h90 for Kocay graphs) and list their graph6 strings and all the graphs that belong to that isomorphism group in the labeling system described in section 1.4.

2.3 Appearance Chart

We then used the isomorphism lists and the original lists to generate a 91 X 240 chart showing how many times each card shows up in each Kocay graph. This chart is included in this folder, under the name "GraphDatabaseFinal.xlsx". We also have several smaller versions of this chart specific to each original cubic graph in the Google Drive, in the same format. Below is an example of one of these smaller charts to give an idea of the format for the larger one. As shown, the cards are sorted by degree sequence, with cards created by deleting bumps coming first. Mousing over any of the graph6 cells will show all appearances of that graph in a list of the cards or Kocay graphs that are isomorphic to it. Clicking on this same cell will take you to an online image of that graph.

Cubic Graphs ->		Graph1003 (IhCggU@oG)				
Kocay Graphs ->		IHCggU@oG	IhCggU@OG	I`CggU@oG	IgCggU@oG	Ih?ggU@oG
Vertex Deleted Graphs		h18	h19	h20	h21	h22
[3,3,3,3,3,3,2,2,2]						
c83	HhSggE@	1	2	--	--	--
c84	HHSgkE@	1	--	1	--	--
c90	H`CgkF@	--	--	1	1	--
c94	HgKGkF@	--	--	--	1	2
[3,3,3,3,3,3,2,1,1]						
c87	HHCggU@	1	--	--	--	--
c88	HHSgkC@	--	2	--	--	--
[3,3,3,3,3,2,2,2,1]						
c25	HHSGGMB	--	--	--	1	--
c27	HgKGGMB	--	--	--	--	2
c33	HLSGGKB	--	--	--	--	2
c38	HXSGGKB	--	--	2	--	--
c44	HhGGGMB	--	--	--	2	--
c85	H@CgkF@	1	--	--	--	--
c86	HHCggV?	1	--	2	--	--
c93	H@SgkE@	--	--	--	1	--
[3,3,3,3,2,2,2,2,2]						
c5	HhS?GMB	--	--	2	1	--
c14	Hik?GKB	--	--	1	--	--
c16	HhHW?EB	1	--	--	1	--
c17	HhH[?E@	--	2	--	--	--
c40	Hbcg?EB	--	--	--	--	2
c60	HjGGGIB	2	--	--	--	--
c80	HhGWKCB	2	--	--	--	--
c89	HgKGkD@	--	4	--	--	--
c91	H`CggV?	--	--	1	--	--
c92	H`SggE@	--	--	--	2	--
c95	HgSggE@	--	--	--	--	2

2.4 Kocay Graphs Sharing Cards

We then used this appearance chart to generate another chart, "KocayCardSharing.xlsx", along with the file "KocayCardSharingList.txt", both of which are included in this folder. This chart compares the 91 Kocay graphs against each other and shows how many cards they have in common. KocayCardSharingList is a list of what cards each pair of Kocay graphs shares. Below is an image of a small portion of the chart. The bold diagonal cells are where the Kocay graphs are being compared against themselves, so all entries in that line are 10, and this serves as the line of symmetry for the chart. Pairs of Kocay graphs sharing two cards are highlighted in green. Pairs sharing three are highlighted in yellow.

	h0	h1	h2	h3	h4	h5	h6	h7	h8	h9
h0	10	1	1	1	--	--	--	--	--	--
h1	1	10	2	--	1	--	1	--	--	--
h2	1	2	10	1	1	--	1	--	--	--
h3	1	--	1	10	--	--	--	--	--	--
h4	--	1	1	--	10	1	1	1	1	1
h5	--	--	--	--	1	10	2	1	--	1
h6	--	1	1	--	1	2	10	--	2	--
h7	--	--	--	--	1	1	--	10	1	--
h8	--	--	--	--	1	--	2	1	10	1
h9	--	--	--	--	1	1	--	--	1	10
h10	--	--	1	--	1	--	2	--	--	--
h11	--	--	2	--	--	--	--	--	--	--
h12	--	--	--	--	--	1	1	--	1	--
h13	--	1	1	--	1	3	1	1	1	--
h14	--	--	--	--	--	--	--	--	--	--
h15	--	2	1	--	1	--	1	--	--	--
h16	--	--	--	--	--	1	--	--	--	--
h17	--	--	--	--	--	--	--	--	--	--
h18	--	--	--	1	--	--	--	--	--	--
h19	--	--	--	2	--	--	--	--	--	--
h20	--	2	--	1	--	--	1	2	--	--
h21	--	1	--	1	--	1	1	--	1	--
h22	--	--	--	--	--	1	1	2	1	--
h23	--	1	--	--	1	1	1	--	--	1
h24	--	2	--	--	1	--	1	--	1	1
h25	--	2	--	--	--	--	1	--	1	--
h26	--	--	--	--	1	1	--	1	--	--

2.5 Other Work

We have worked on several smaller parts of this project as well. Though we made the entire chart, our group was also personally responsible for created smaller appearance charts for Kocay graphs generated from G1003, G1007, G1011, and G1015. We compared the entries in these charts to the automorphism groups of the involved Kocay graphs to look for similar vertices. We are also working on a database with detailed information about each graph encountered. Finally, we generated much of the above data for cubic graphs on 8 vertices, to double check work done by hand previously. We intend to try and generate this data for cubic graphs on 12 vertices as well.

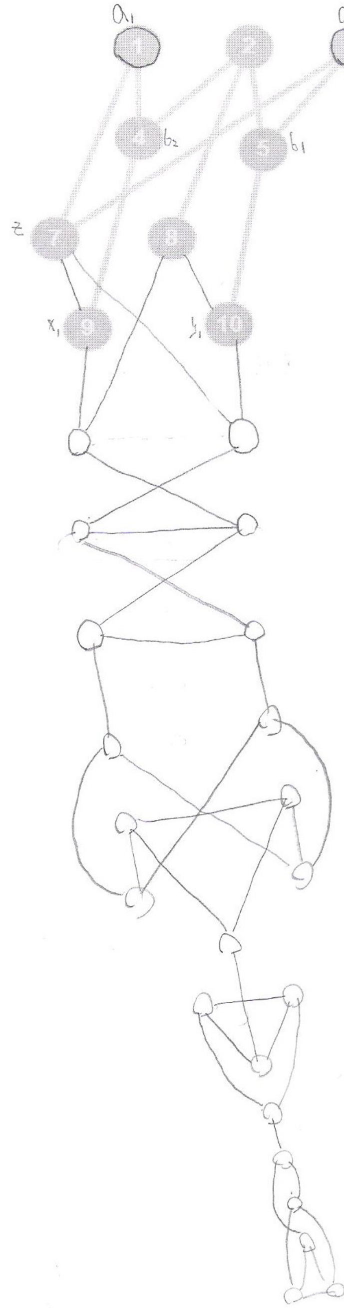
3 Later Work: Examining Partial Automorphisms

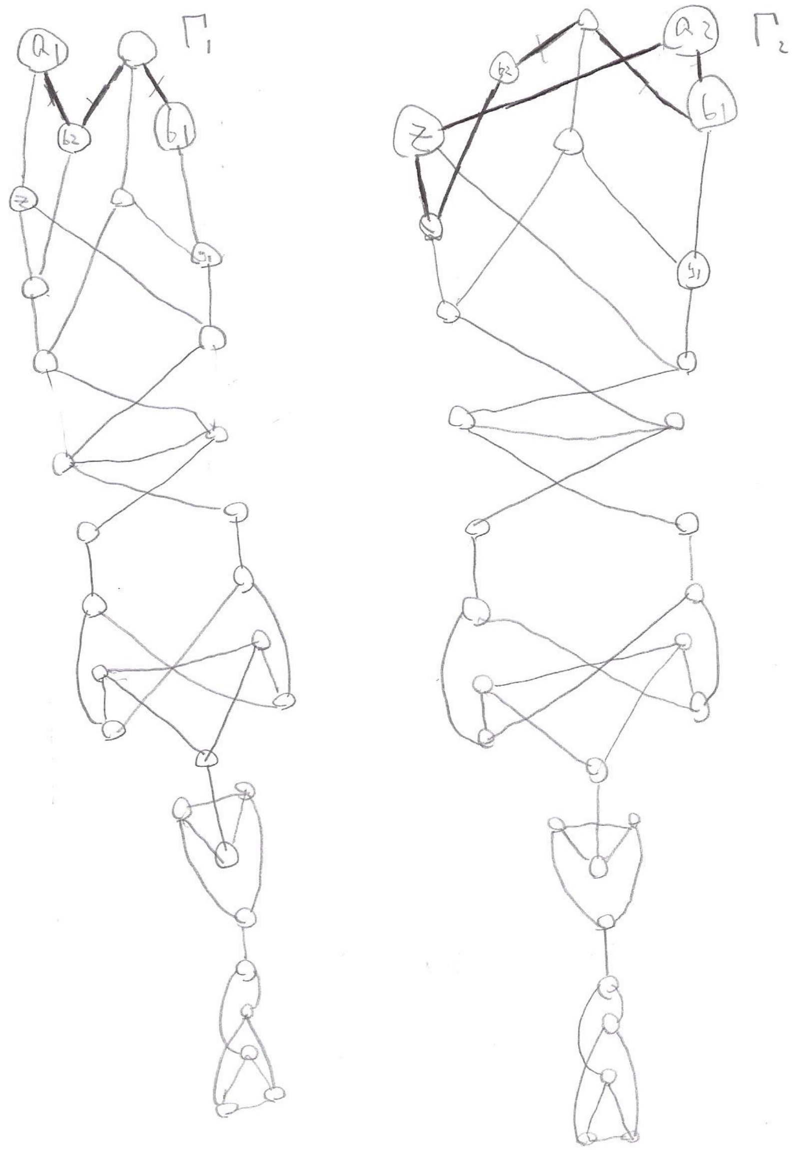
3.1 Partial Automorphisms

Using the data mentioned above, we began studying Kocay graphs which shared two or more cards. We worked for a while finding partial automorphisms that mapped one of these Kocay graphs to the other, and from this made a process to generate pairs of Kocay graphs that had partial automorphisms between them and shared two or more cards. We examined the threads in the partial automorphisms, that is, the mappings in the partial automorphism that aren't cyclical, and focused on using those to generate these new graphs. Our group focused on threads of length 2.

3.2 Graphs and Observations for Threads of Length 2

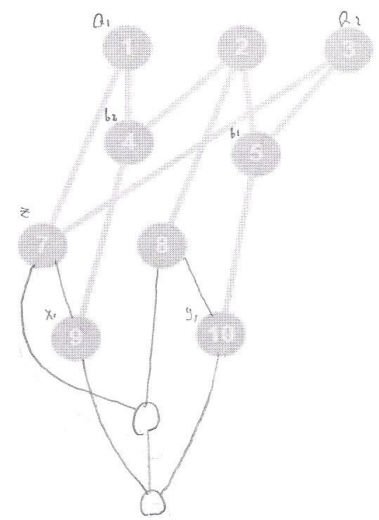
3.2.1

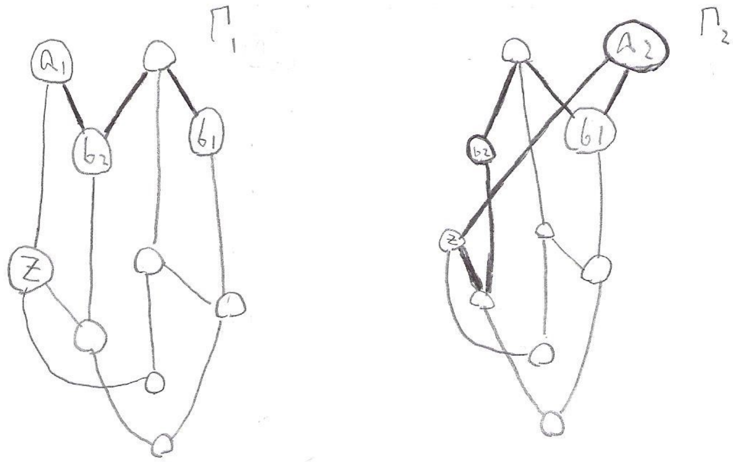




These are not isomorphic because Γ_2 has two paths of length 3 from one bump to another, but Γ_1 has only has one.

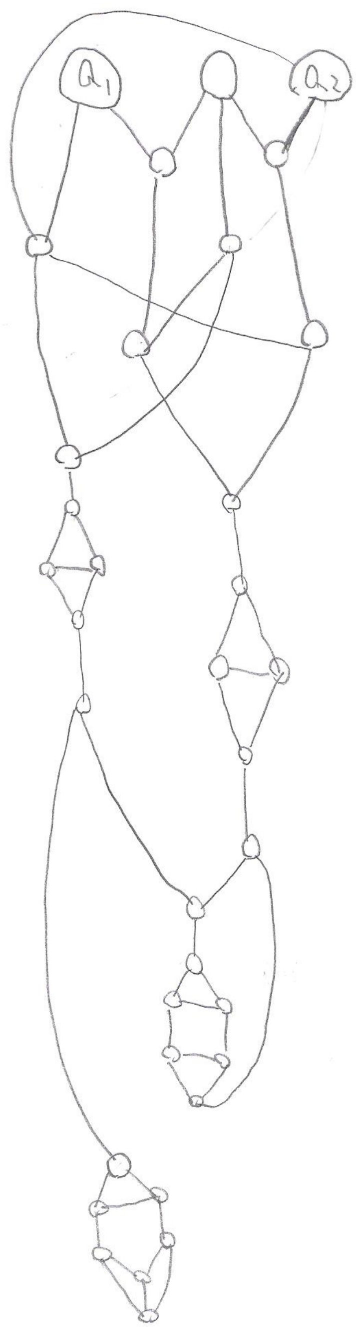
3.2.2

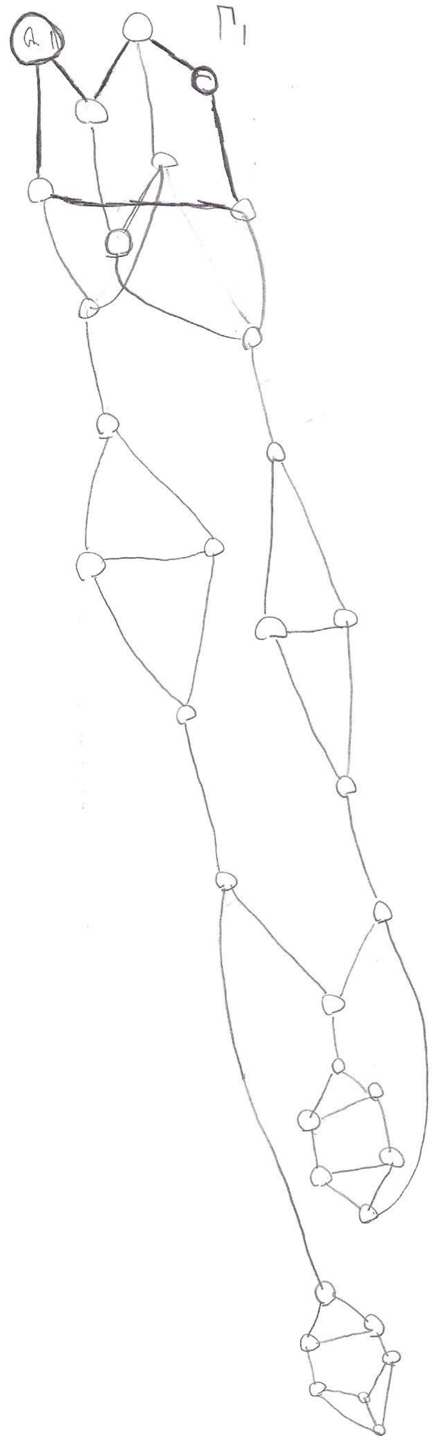


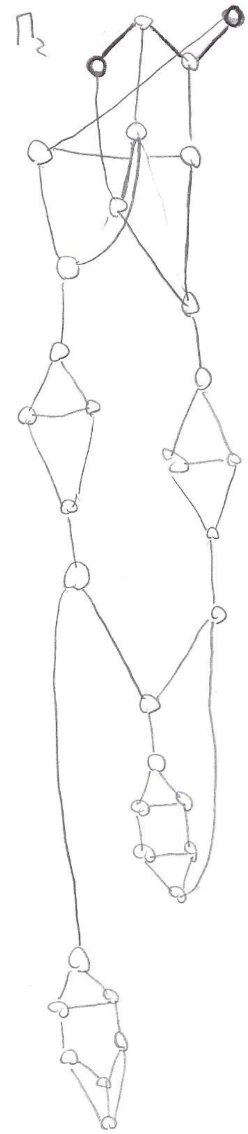


These two are not isomorphic because Γ_2 contains a 6 cycle with both bumps while Γ_1 does not.

3.2.3

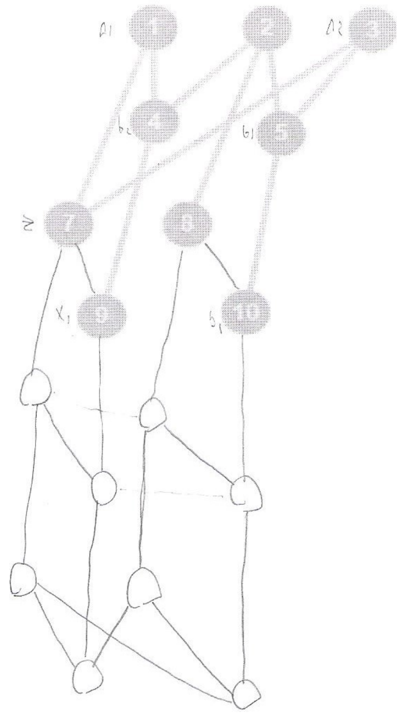


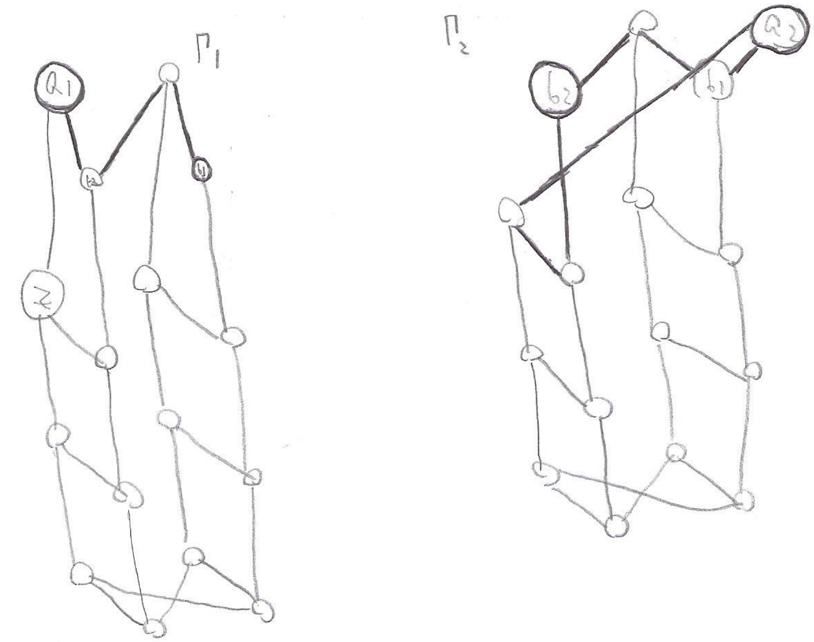




These two graphs are not isomorphic because Γ_1 has a six cycle with both bumps while Γ_2 does not.

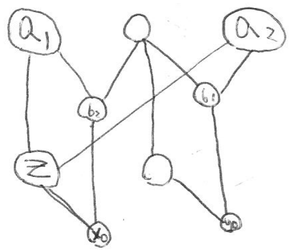
3.2.4





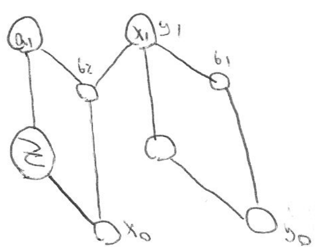
These are not isomorphic because Γ_2 has a 6 cycle with both bumps while Γ_1 does not.

3.2.5

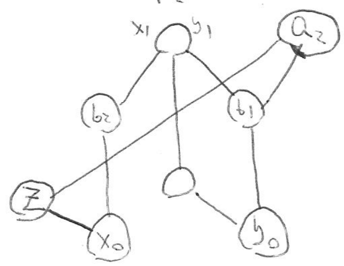


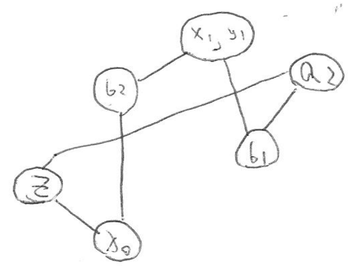
Connecting z to x_0 creates a 6 cycle in Γ_2 that is not present in Γ_1. The <u>only</u> way to make Γ_1 and Γ_2 isomorphic after this is by connecting z to y_0.

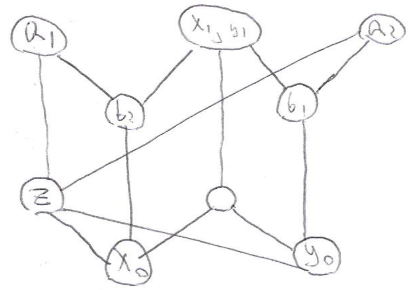
Γ_1



Γ_2

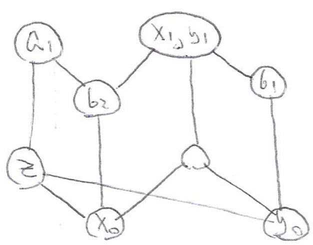




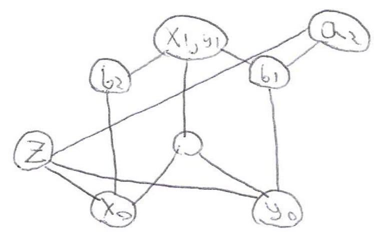


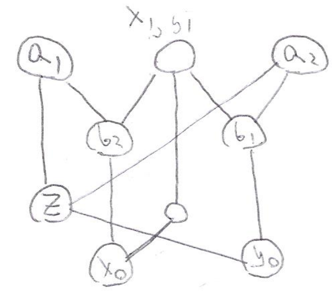
A complete graph that is isomorphic, created by connecting z to X_0 and y_0 and following the thread mappings.

Γ_1



Γ_2

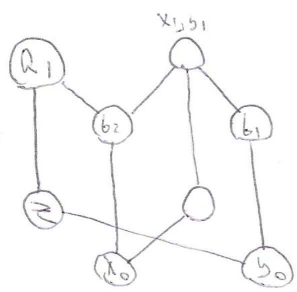


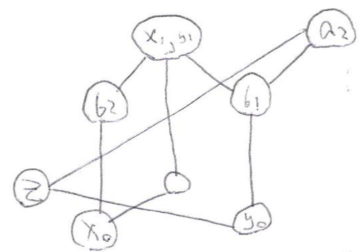


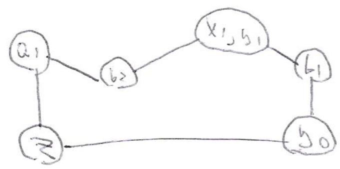
Connecting Z to y_0 creates a 6 cycle in Γ_1 that is not present in Γ_2. The <u>only</u> way to make Γ_1 and Γ_2 isomorphic after this is to also connect Z to X_0, completing the graph.

Γ_1

Γ_2

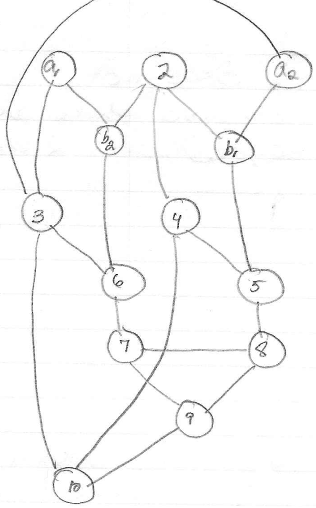




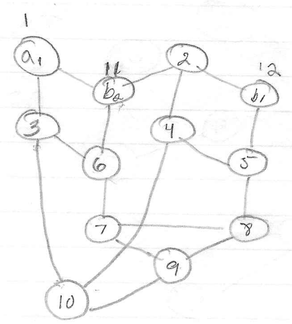


3.2.6

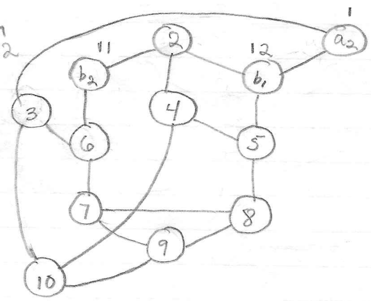
$\Gamma_1 \cup \Gamma_2$



Γ_1



Γ_2



	π	π'	π^2	π^3
a_1	b_2	a_2	b_2	a_2
b_2	6	3	2	b_1
2	3	6	b_1	2
b_1	a_2	b_2	a_2	b_2
3	2	b_1	6	3
5	b_1	2	3	6
6	—	—	—	—

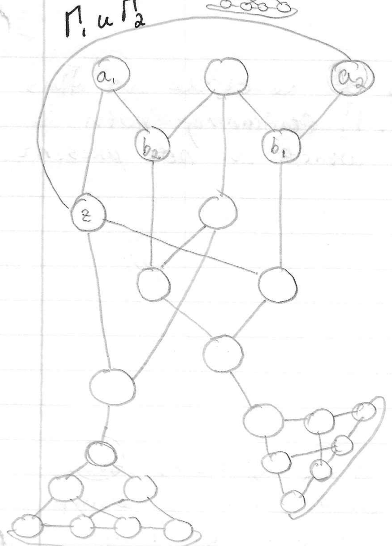
The isomorphism fails when trying to project vertex 6 from Γ_1 to Γ_2.
So we know that Γ_1 and Γ_2 are not isomorphic.
We also know that they are not the same graph from observing the 6-cycle in Γ_2 which contains both bumps. Such a cycle is not present in Γ_1, proving that Γ_1 and Γ_2 are undoubtedly different Kocay graphs.

3.2.7

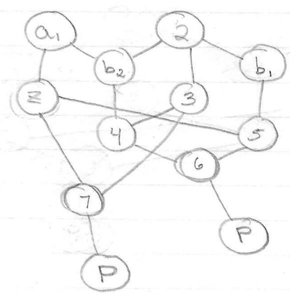
For future reference I will be attaching the pyramid structure  as (P)

These are graphs on 24 vertices

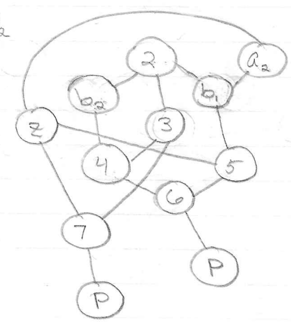
$\Gamma_1 \cup \Gamma_2$



Γ_1



Γ_2



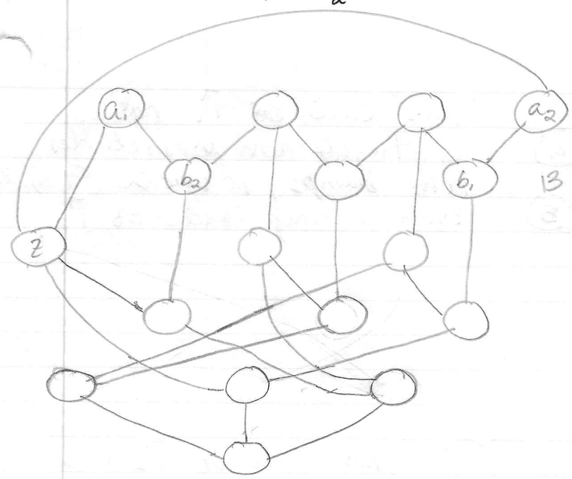
	π	π^2
a_1	a_2	b_2
b_1	b_2	a_2
b_2	b_1	2
2	2	b_1
z	z	4
4	5	z
3	X	5
5	4	z X
6	6	
7		

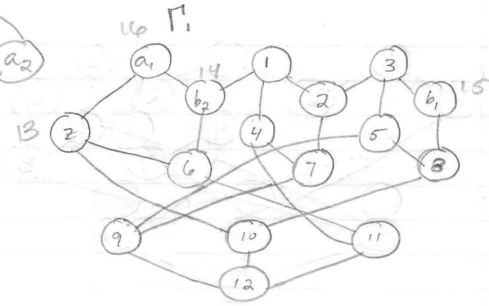
- 3 would project onto a vertex that is connected to 2 + 5 of π which would be b_2 which is already projected by a_1 onto π^2.
- 4 + 5 project onto z of π^2

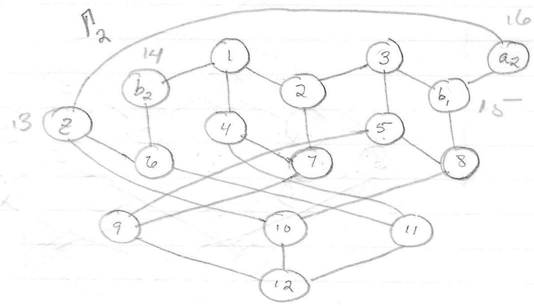
So these graphs are <u>not</u> isomorphic.
There is also a 6-cycle in Γ_1 that includes both bumps which is not present in Γ_2. therefore Γ_1 + Γ_2 are definitely different graphs.

3.2.8

$\Gamma_1 \cup \Gamma_2$







	π	π^2
a_1	b_2	a_2
b_2	1	b_1
1	2	3
2	3	2
b_1	a_2	b_2
z	6	z
6	4	8
7	5	7
3	b_1	1
4	7	5
5	8	4
8	z	6

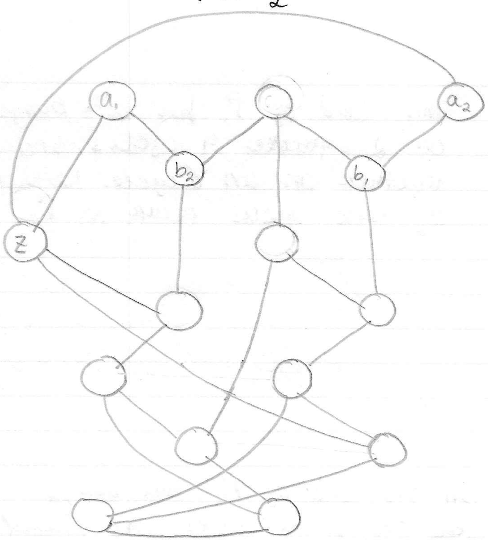
Bump a_1 is on a 4 cycle in Γ_1, there are no 4 cycles in Γ_2 containing a bump.

In π of Γ_1 onto Γ_2 claims that vertex 4 in Γ_1 projects onto 7 in Γ_2, and 7 in Γ_1 projects onto 5 in Γ_2, but this does not hold because in Γ_1 vertices 4 and 7 are one step apart whereas vertices 5 and 7 of Γ_2 are 2 steps apart.

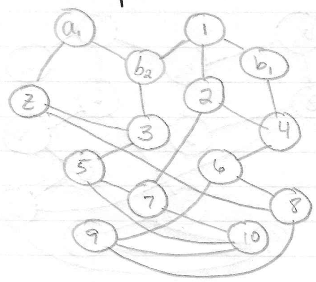
For π^2 of Γ_1 onto Γ_2 we see that in Γ_1 vertices 1, 2, 4, + 7 are on a 4-cycle. The projections onto Γ_2 2, 3, 5, 7 are not.

3.2.9

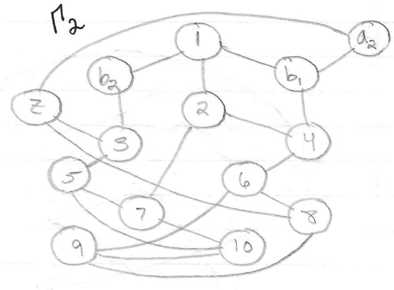
$\Gamma_1 \cup \Gamma_2$



Γ_1



Γ_2



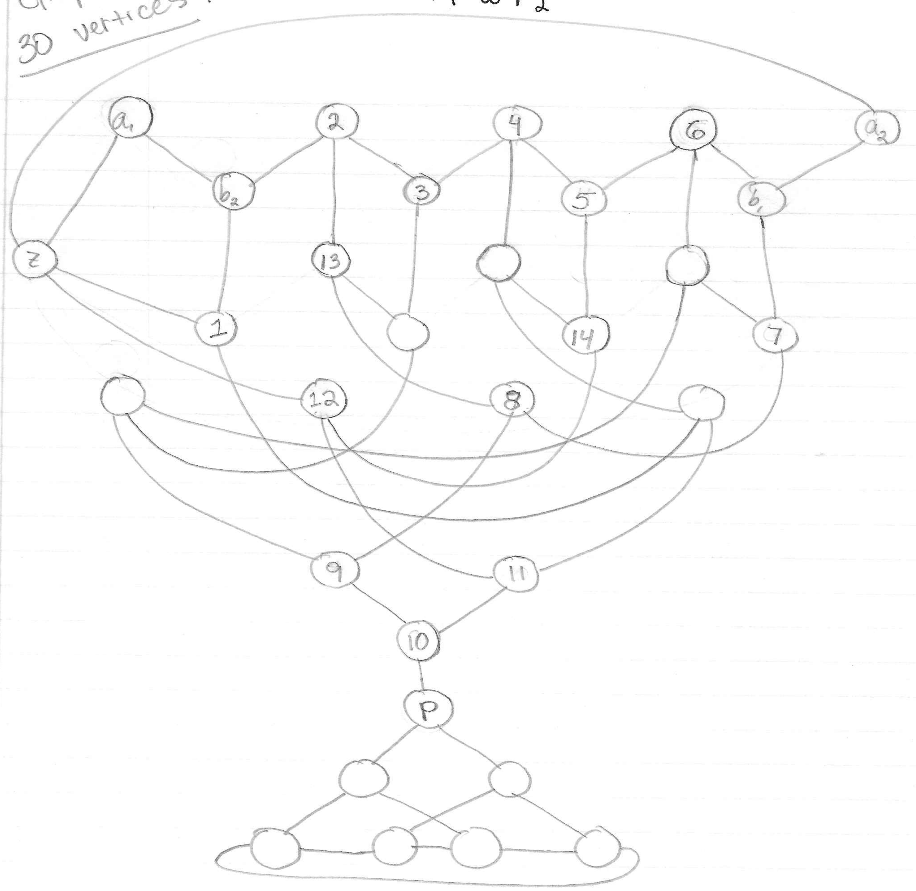
	π	π^2
a_1	b_2	a_2
b_2	1	b_1
1	b_1	1
b_1	a_2	b_2
z	3	z
4	z	3
3	2	4
2	4	2

For both of these projections of π and π^2 from Γ_1 to Γ_2, the vertices 1, b_1, 2, and 4 are on a 4 cycle in Γ_1, but the vertices that they are projected onto are not. In π, these vertices are b_1, a_2, 4, and z. For π^2, these vertices are 1, b_2, 2, and 3. This shows us that Γ_1 and Γ_2 are not isomorphic.

3.2.10

Graph on 30 vertices.

$\Gamma_1 \cup \Gamma_2$



I can irrefutably say that Γ_1 and Γ_2 are not isomorphic AND not the same graph simply by following the shortest path that puts both bumps of Γ_1 and Γ_2 into the same cycle.

For Γ_1 the cycle follows:

$a_1 \to b_2 \to 2 \to 13 \to 8 \to 7 \to b_1 \to 6 \to 5 \to 14 \to 12 \to z \to a_1$

12 steps, therefore a 12 cycle is the minimum required for both bumps in Γ_1.

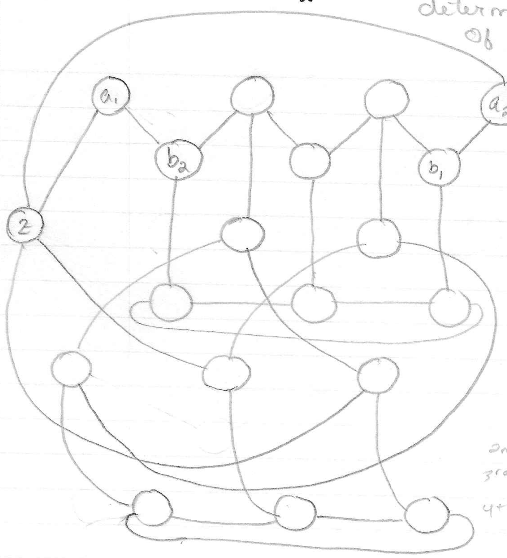
For Γ_2 the cycle follows:

$a_2 \to z \to 1 \to b_2 \to 2 \to 13 \to 8 \to 7 \to b_1 \to a_2$

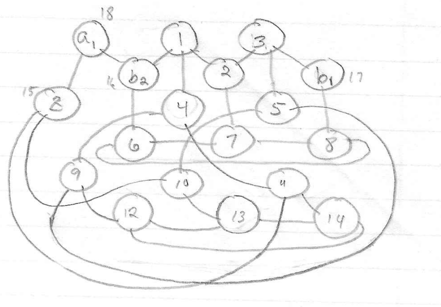
9 steps, therefore a minimum 9 cycle is required to put both bumps of Γ_2 in a cycle.

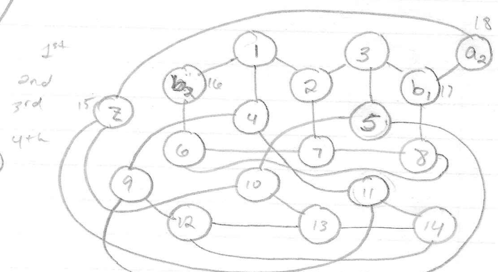
3.2.11

$\Gamma_1 \cup \Gamma_2$



The connections (edges) of Z not connected to a_1 in Γ_1 and a_2 in Γ_2 are pretty deterministic of isomorphism, or not, of Γ_1 and Γ_2



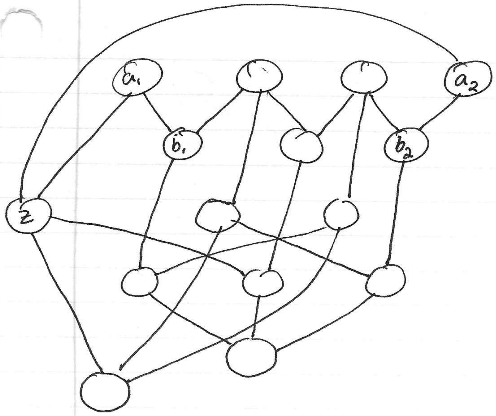


	π	π
a_1	b_2	a_2
b_2	1	b_1
1	2	3
2	3	2
3	b_1	1
b_1	a_2	6
Z	6	Z
6	4	8
7	5	5
5	8	7
8	Z	4

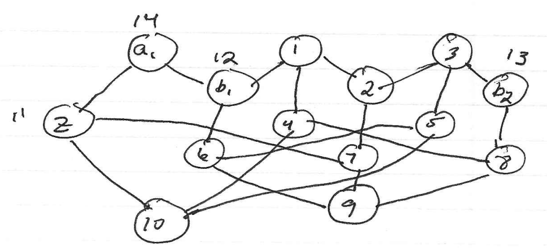
Because Z does not interact with the 4th level of vertices at all, I have undoubtedly created an isomorphism of Γ_1 and Γ_2.
There are no short routes between bumps and no cycles that separate Γ_1 from Γ_2, this has repeatedly been the case any time Z does not have a direct edge to the 4th level of vertices.

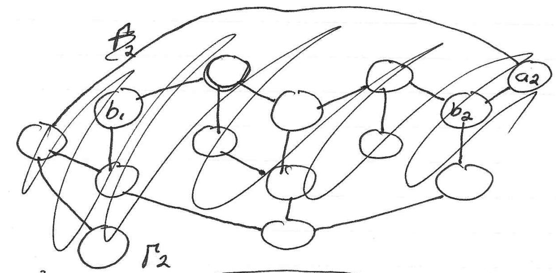
3.2.12

$\Gamma_1 \cup \Gamma_2$



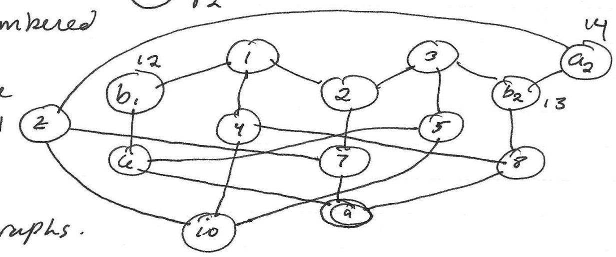
Γ_1





The exception to the ~~ru~~ rule happens when there is a thread of odd numbered vertices. When z connects to the middle vertex of the 4th level, it also creates ~~a isomorphic graphs.~~ mirrored isomorphic graphs.

Γ_2



	π	π^2
a_1	b_1	a_2
b_1	1	b_2
1	2	3
2	3	2
3	b_2	1
b_2	a_2	b_1
z	6	z
~~4~~ 6	4	8
4	7	5
7	5	7

continued	π	π^2
5	8	4
8	z	6
9	10	9
10	9	10

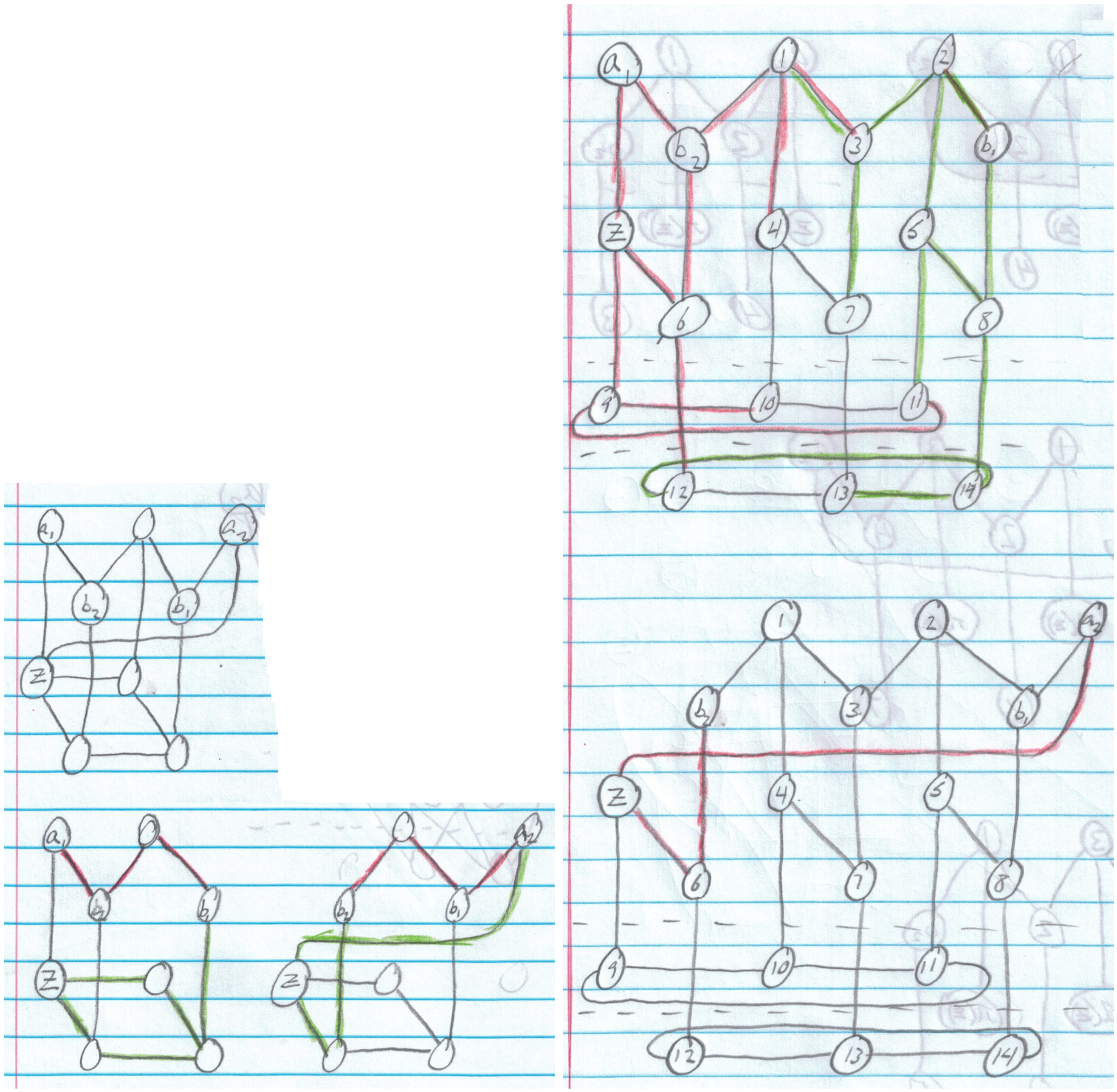
Isomorphic!

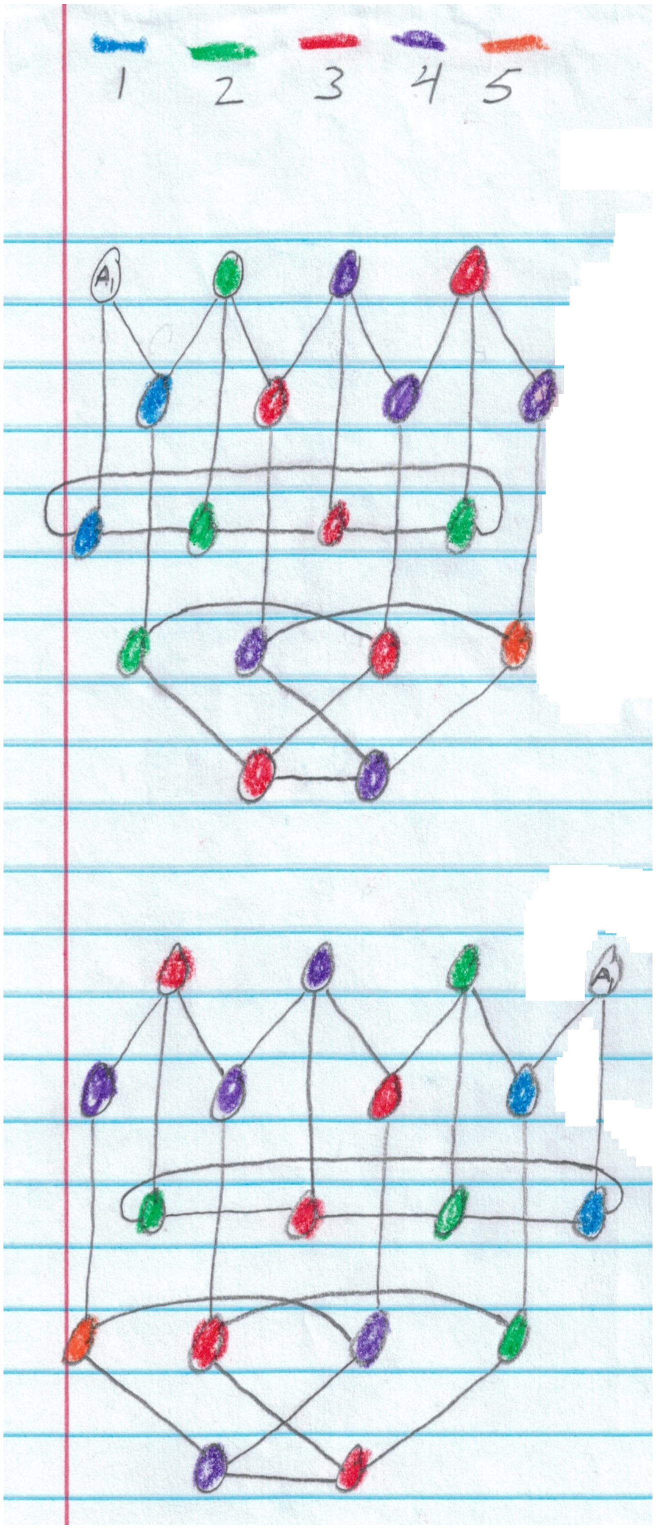
3.2.13

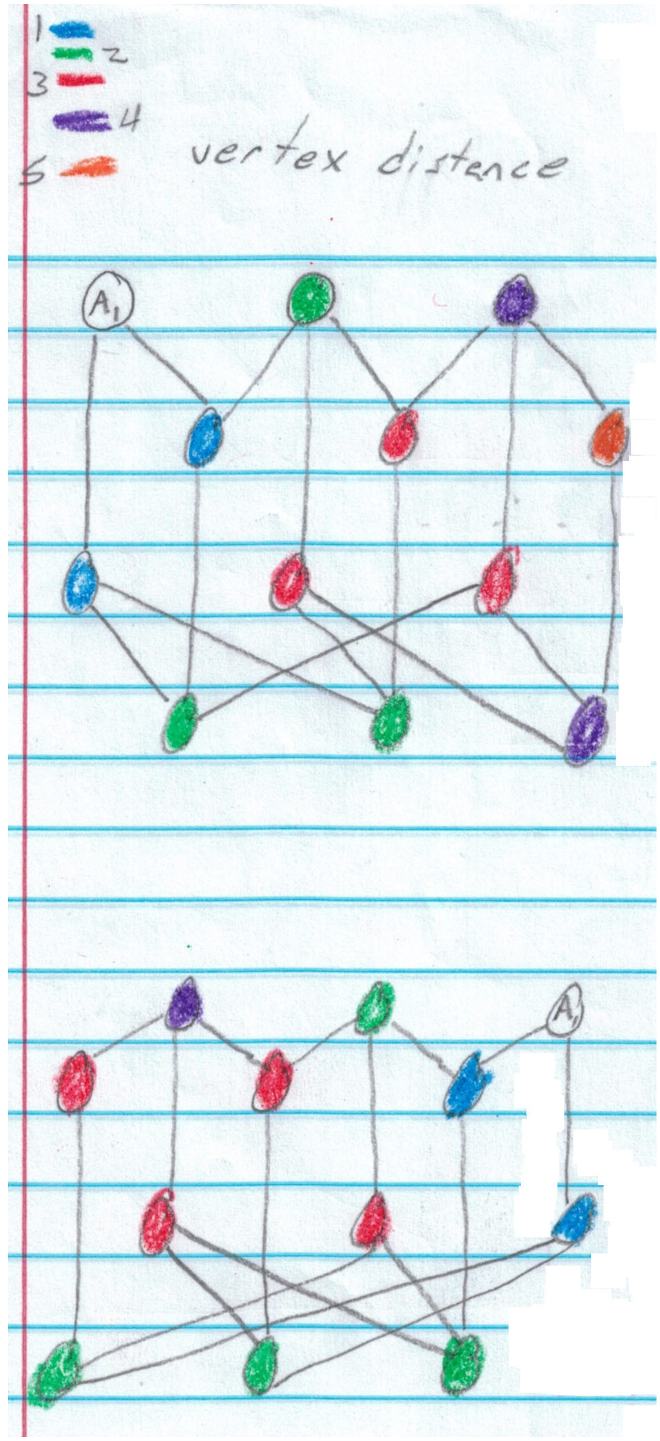
Note of clarification: After writing most of this I have noticed that K is used several times in this class to mean several things. When referring to K, I am doing so in regards to the length of a row/level/line of vertices in Γ_1 or Γ_2, rather than the degree of the vertices or anything else.

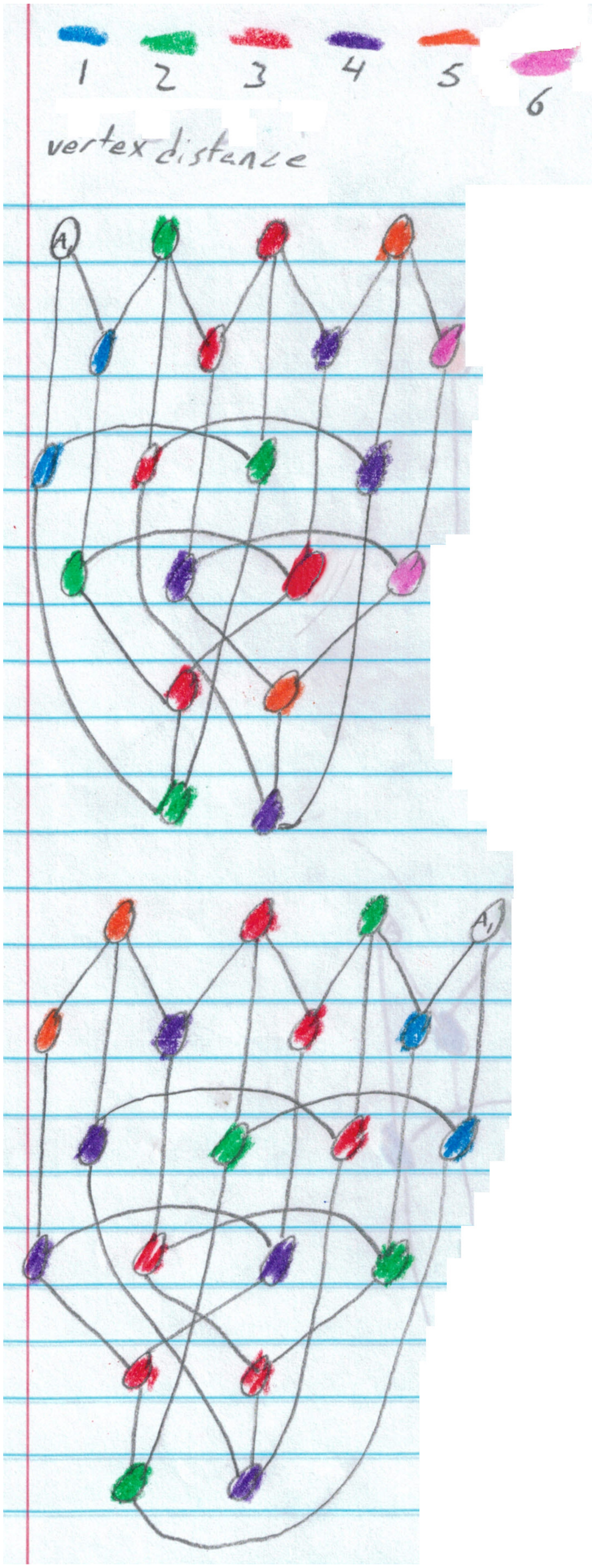
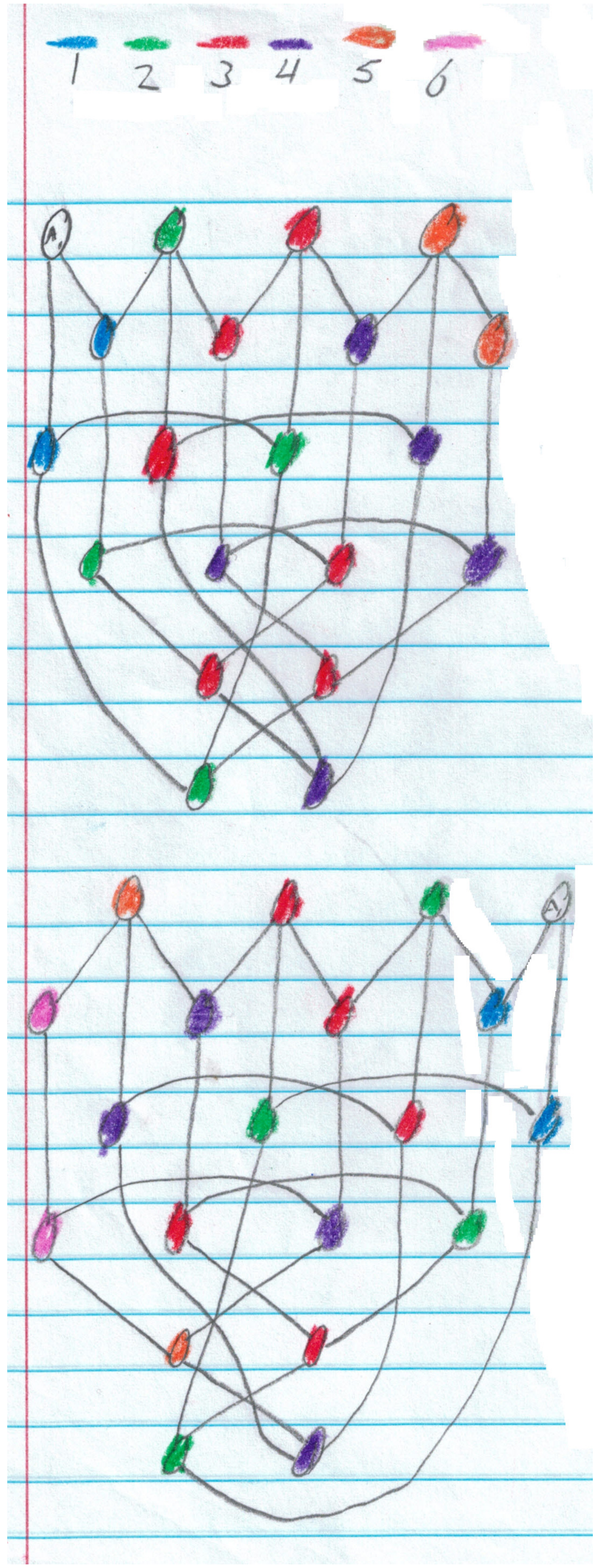
Graphs:

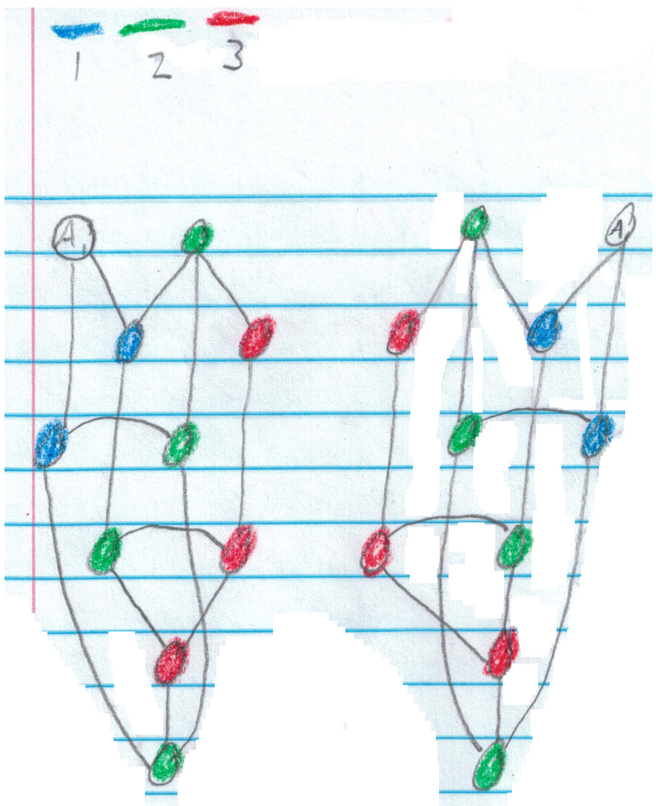
Note: Most of the pages had been written on both sides, I have attempted to remove some of the distracting lines which have bled through from the other side. In the scans below Γ_1 is always above and/or to the left of Γ_2.

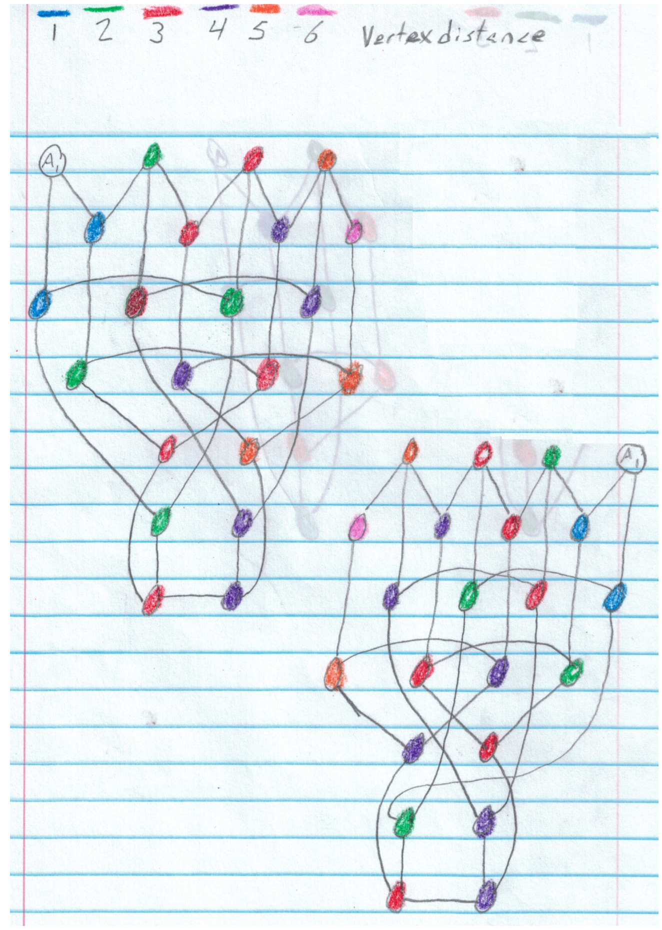



1 2 3 4 5
A1
A1


1 2 3 4 5
vertex distance
A1
A1


1 2 3 4 5 6
vertex distance
A1
A1

1 2 3 4 5 6
A1
A1

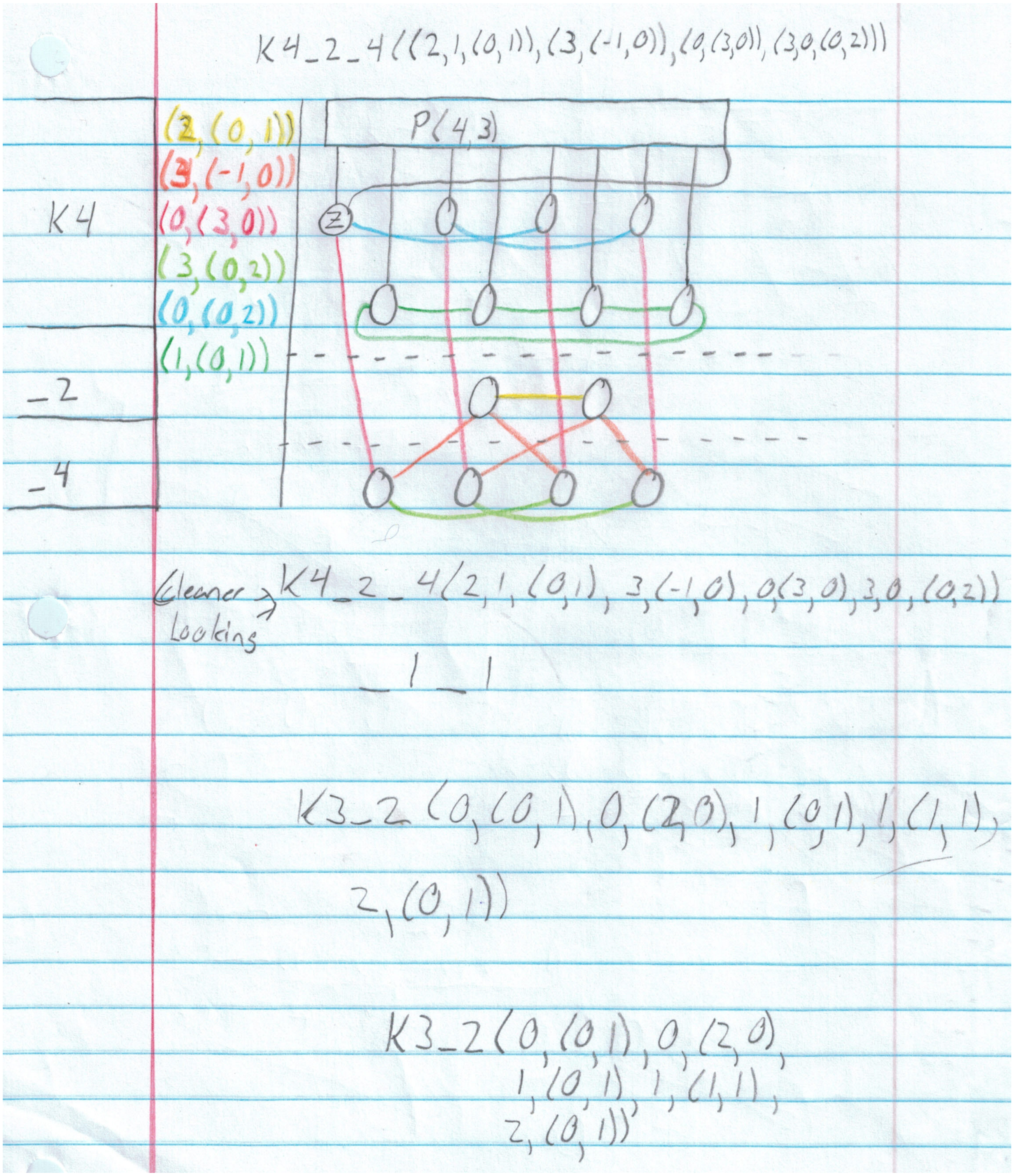




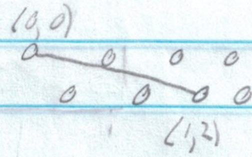
In some of these graphs $\Gamma_1 \cong \Gamma_2$, however I have included them because I had them. In the above graphs I started out tracing the distance between the bumps and ended up settling on coloring all the vertices based on their minimum distance from A_1 instead. In last scan, Γ_1 and Γ_2 are quite similar despite their larger size, I believe that this coloring method saves quite a bit of time in most cases when compared to checking the various possible mappings of π.

Notation:

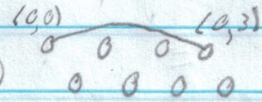
The first two scans below are from when I was working on creating the notation.



Move((0,0),(1,2))

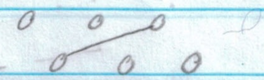


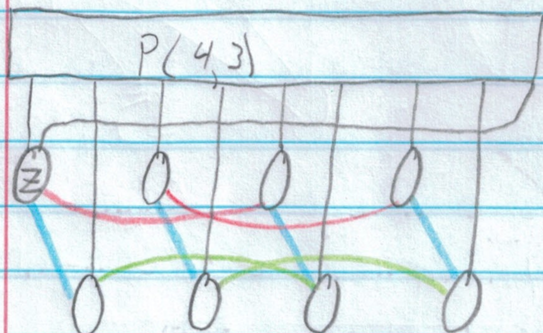
Move((0,0),(0,3))



OR

Move(1,(-1,2))





Move(0,(0,2))
Move(1,(0,2))
Move(0,(1,0))

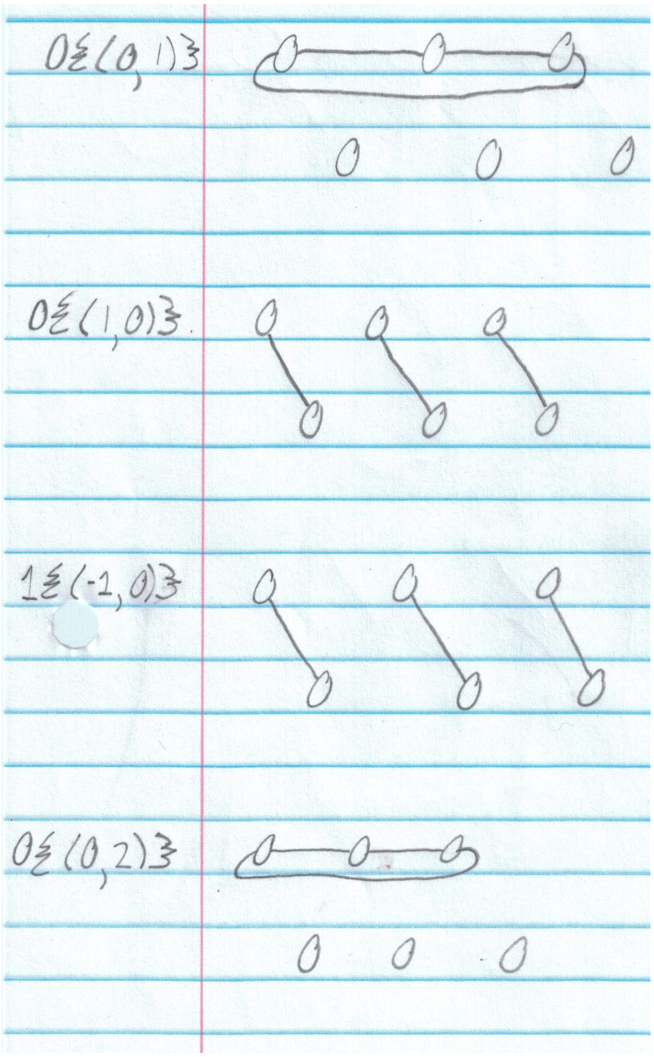
P(4,3) M(0,(0,2)) M(1,(0,2)) M(0,(1,0))

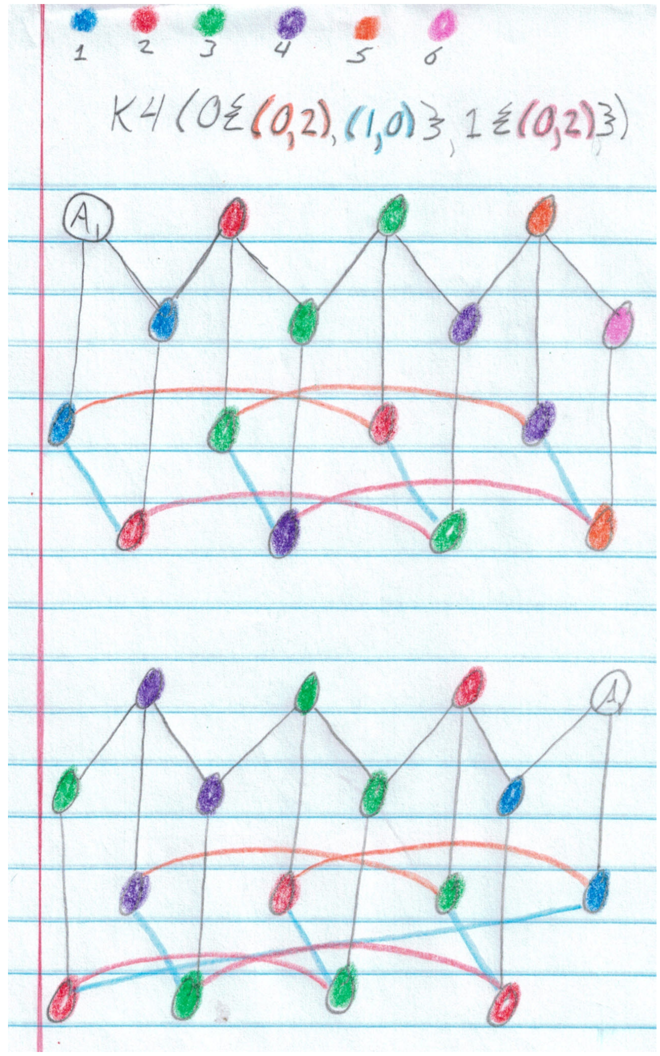
K4((0,(0,2)),(1,(0,2)),(0,(1,0)))

K4((0,1,(0,2)),(0,(1,0)))

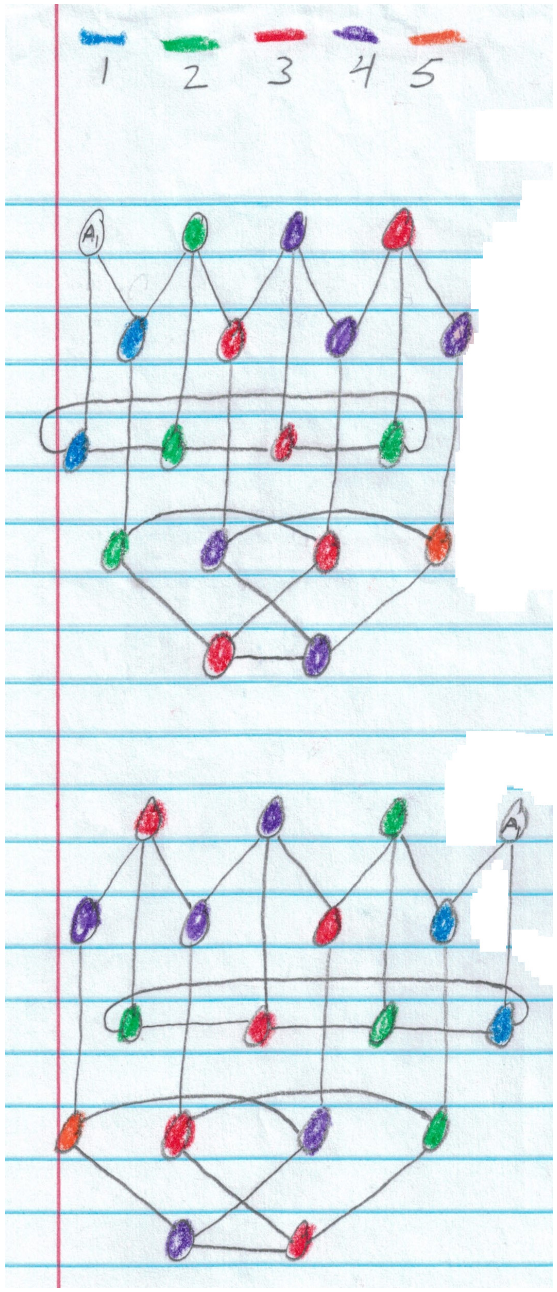
K4_0((0,1,(0,2)),(0,(1,0)))

$$\underbrace{K4}_{K=4}\_\overbrace{2}^{\text{Add row, population 2}}\underbrace{((0,1,(0,2)),(0,(1,0)))}_{\text{Moves}}$$





I have chosen the notation based on your request that each level/row of the graph be separately listed.

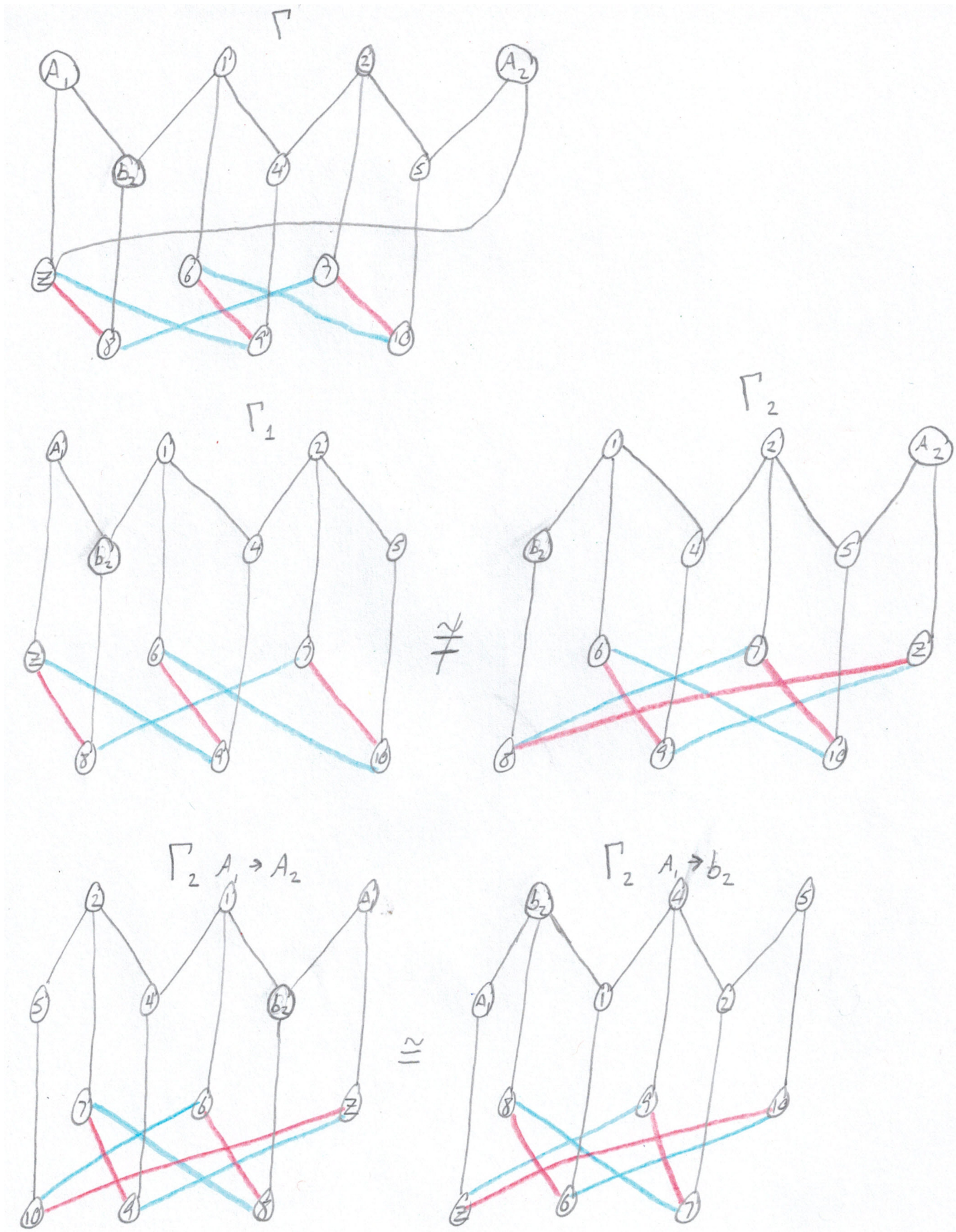


K4_2(0{(0,1)},1{(0,2),(1,0)},2{(0,1)})

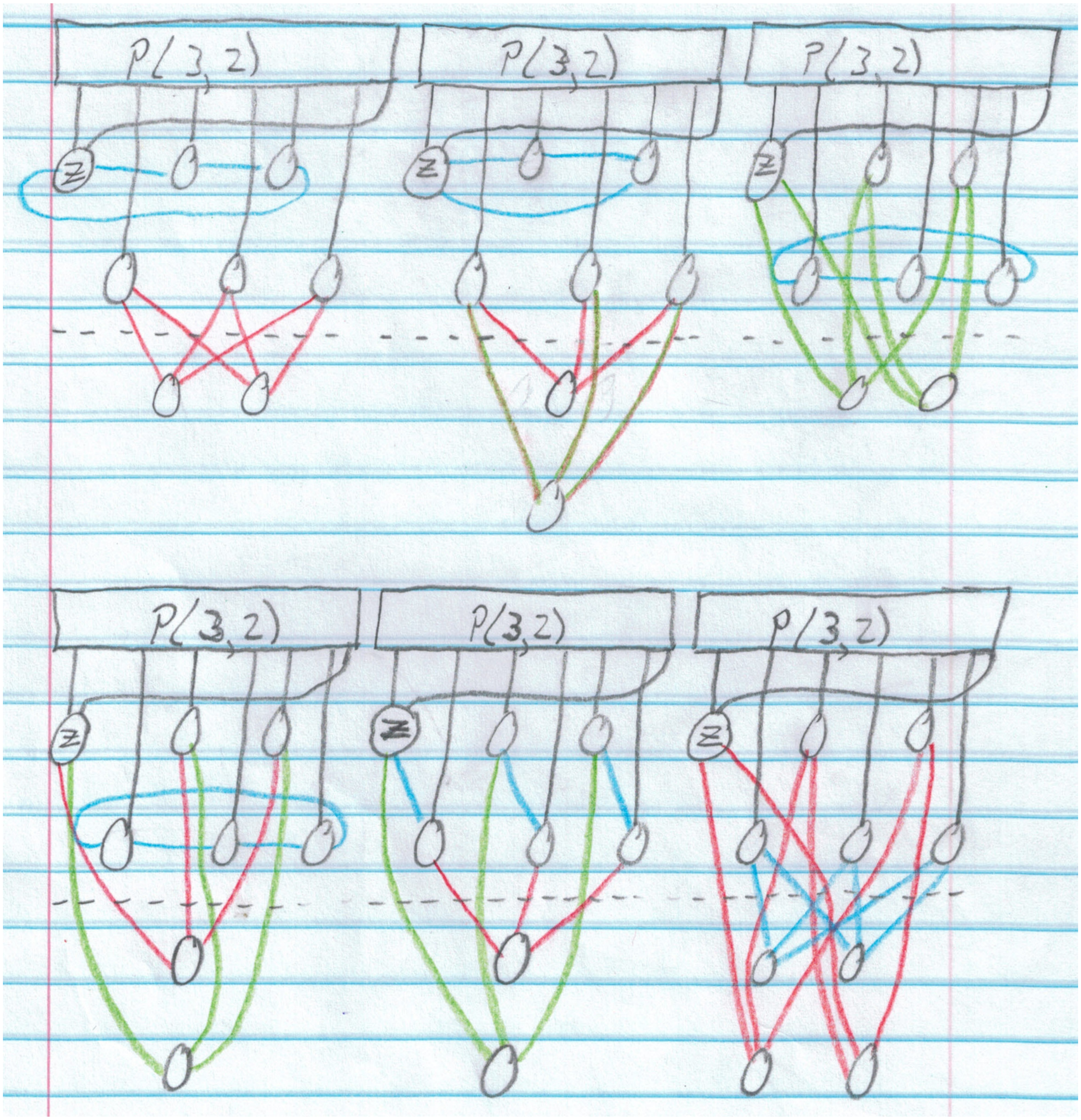
Explanation: The K4 at the beginning of the notation indicates that $K = 4$ in this graph. Following that the _2 means that after you draw the graph you will draw an additional two vertices on a new level below the current bottom one. Then we reach the area between the parentheses, this part of the notation is where the edges are drawn. 0{} directs our attention to the top incomplete row of vertices (the row containing Z). Within the curly braces there is (0,1), this indicates that an edge is drawn to the vertex 0 rows down and 1 position to the right. Add the edge to any vertex in the row, then move over either to the left or right, keep going until you find yourself again adding the first edge you added. Continue doing this for each row (e.g. 1{(0,2),(1,0)} and 2{(0,1)}) until you reach the end of the notation and you should have the graph above.

Note: There are multiple ways to write any given edge when converting from a drawn graph to notation. 0{(1,0)} is the same as writing 1{(-1,0)}. In large graphs this could be a problem, unknowingly writing down the same edge group several times.(a,b) This can be avoided by disallowing a or b to be negative and $b \geq K$.

Thoughts:

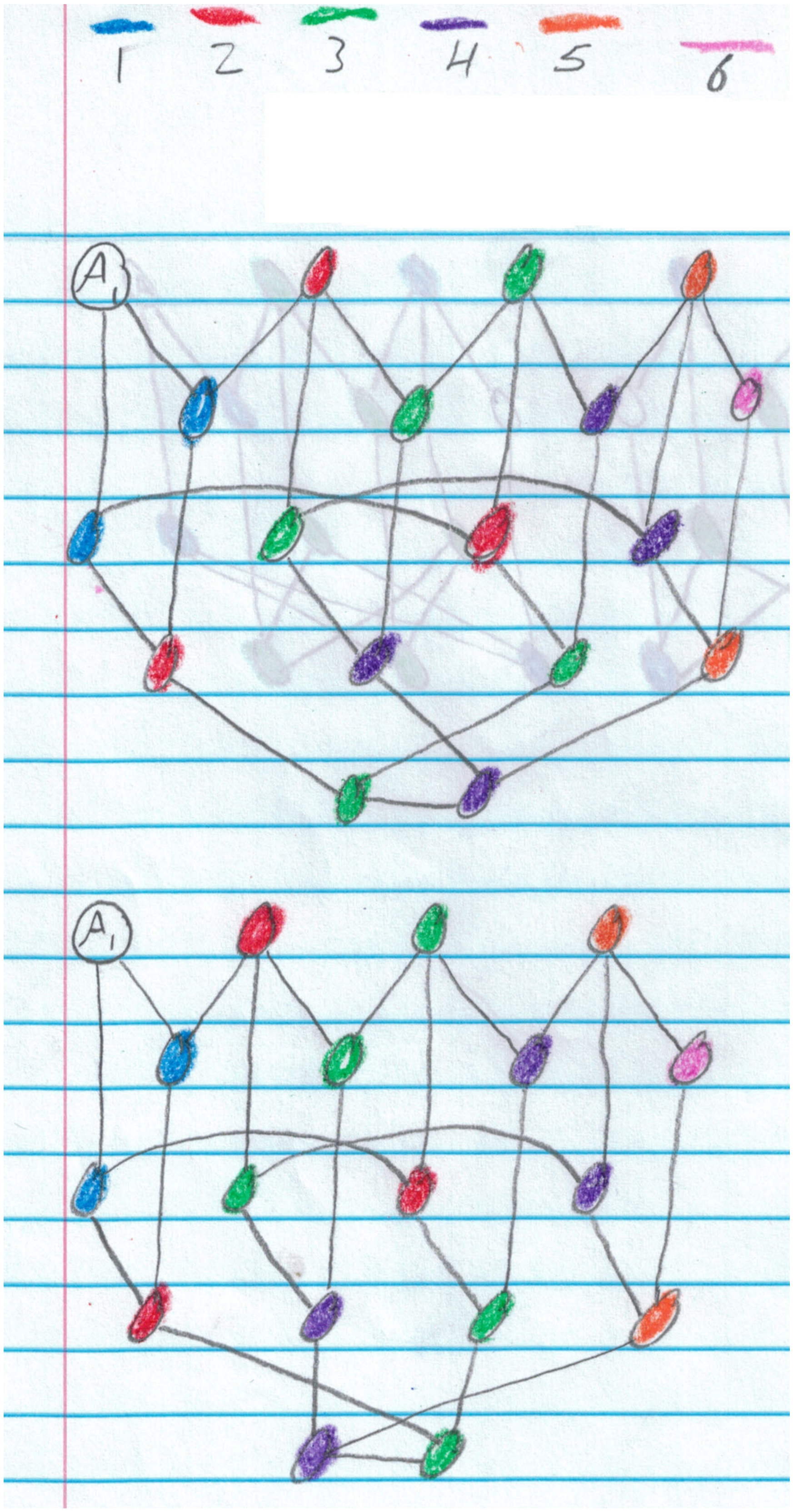


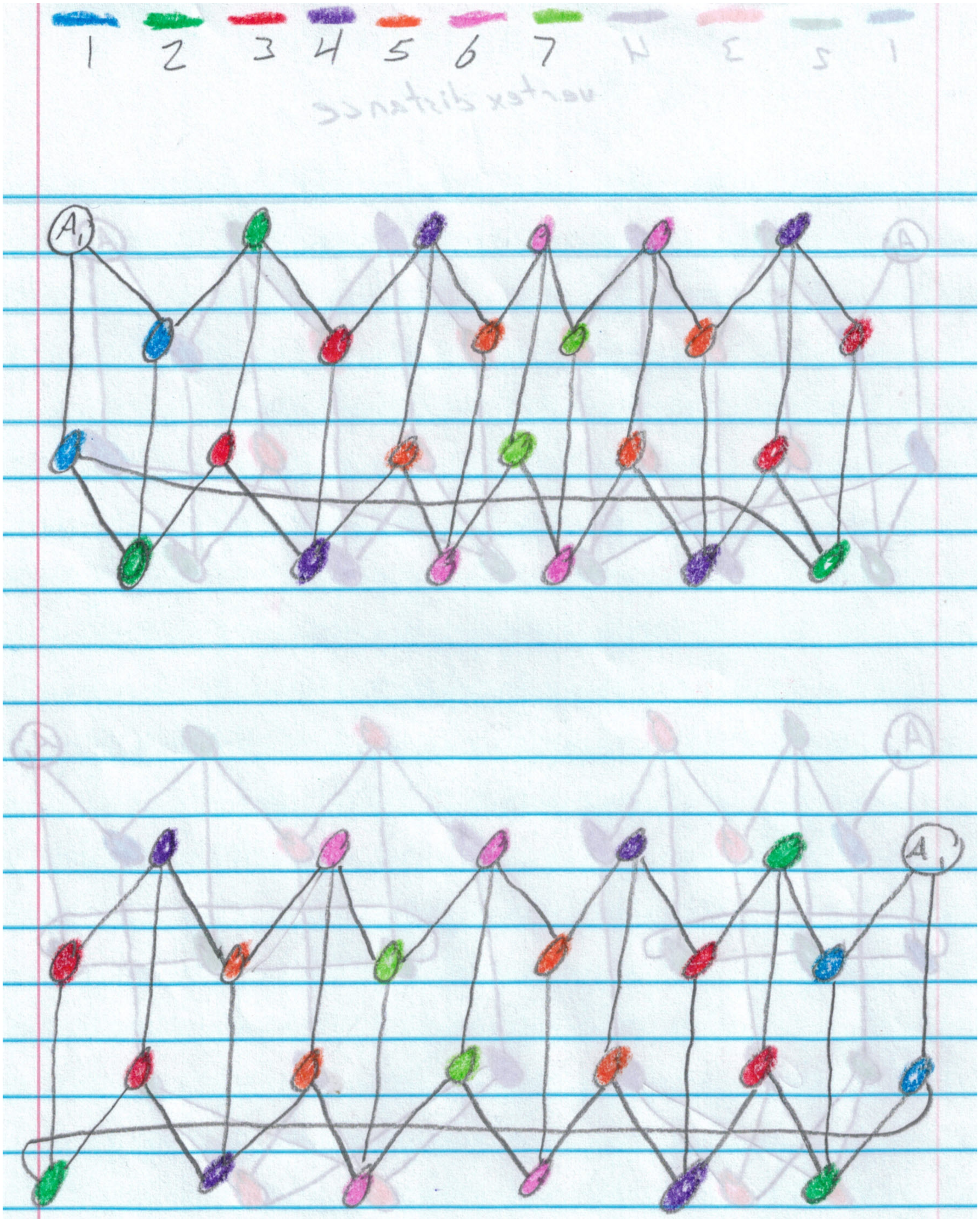
First Observation: Mapping A_1 to A_2 should always be the same as mapping A_1 to B_2. This could already be in your notes or even just limited to certain cases, I'm uncertain how I feel about this claim. If this is true, then it is due to the orbits of Γ_2, the interchangeability of A_2 and B_2.



Second Observation: Some edge placements (moves) result in isomorphic graphs that are easily discerned, as seen in the following scan (The first and second graphs, the third and fourth graphs)

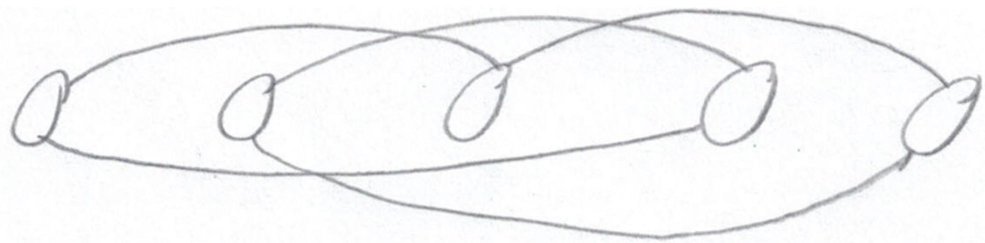
Some are less easily discerned.





Third Observation: Mapping A_1 to A_2 in Γ_2 results in the vertices in Γ_2 mirroring the placement of the vertices in Γ_1, while the edges added onto Z's row and the rows below remain unmirrored. This means that a combination of moves resulting in left/right symmetry result in an isomorphic pair of Γ_1 and Γ_2. An example of this is `O{(1, $n$), (1, $K$-1-$n$)}`; another example that occurs in graphs with an odd numbered K is `O{(1, floor($K$/2))}`. One thing of note though is that indirect interactions between rows 0 and 1 through additional lower rows aren't as simple as direct interactions. In the scan above `O{(1,0),(1,K-1)}`.

(0,2)



Fourth Observation: In $K = 2$ and $K = 3$, the interactions within a row are limited to a single outcome which results in $\Gamma_1 \cong \Gamma_2$. In $K = 4$ we are given the opportunity to have a same row interaction that doesn't yield the same results as (0,1); this new same row move is (0,K/2), this move is possible when K is even. Additional possible moves open up as K grows larger. Example above.

Parting Thoughts:

1. If rows 0 and 1 never interact directly or indirectly $\Gamma_1 \cong \Gamma_2$, at least while $K \leq 5$ (it is possible that as K grows larger and more possible outcomes occur from same row moves, that some or possibly most might result in $\Gamma_1 \ncong \Gamma_2$).

2. When working with these graphs by hand or in Sage, avoiding unnecessary graphs is appreciated. The occurrence of these can be minimized through several methods; following the advice in my note on notation, avoiding graphs with mirrored edge symmetry, and analyzing moves as some moves produce Γ that are isomorphic to other Γ (example: two pages back under "Some are less easily discerned").