Graceful Labeling Necklace Graphs

by

Jacob Yunker

MS Candidate: Applied and Computational Mathematics

Advisor: Dalibor Froncek

Department of Mathematics and Statistics

University of Minnesota Duluth

June 2012

Abstract

A C_4 necklace graph can be described as a graph made up of r beads, where each bead is comprised of one cycle of length four which connects to a path of length l. We construct our necklace graphs by stringing together our beads to form a cycle of beads, where each path is adjoined to non-adjacent vertices of the four-cycle. We denote such a necklace graph as $N(C_4, l, r)$. Now consider a graph G on n vertices, with vertices x and y and corresponding edge xy. A graceful labeling of this graph is a one-to-one mapping between the vertices and a set of non-negative integers. More specifically, a mapping such that the edge length of xy is $min\{x - y, y - x\}$., where subtraction is performed in Z_{2n+1}, the vertex labels are restricted to the set $\{0, 1, 2, ..., n\}$., and the edge lengths are restricted to the set $\{1, 2, ..., n\}$.. In this paper we will investigate several cyclic structures, various graph labelings, and an algorithm which allows all necklace graphs to be gracefully labeled.

Contents

List of Figures

1 Introduction

In order to understand graph labelings, it is imperative to know first what a graph is in the field of graph theory. A graph is defined to be comprised of two finite sets: the vertex set and the edge set. Graphs are often denoted by G and using the previous definition we may write $G = (V(G), E(G))$, where $V(G)$ represents the vertex set (which is nonempty) and $E(G)$ represents the edge set (which may be empty). The next natural definitions then would be defining what a vertex and an edge are. A vertex is simply an element of the vertex set, and an edge represents a connection between two elements of an unordered pair from the vertex set. For example, if we have the edge e which is assigned to the unordered pair (u, v), then this would imply that u and v are the end vertices of the edge e [**?**].

Next we look at graph labelings. A graph labeling is defined to be an assignment of integers to vertices, or sometimes edges, in a graph based upon certain criteria. In this paper however, we will focus only on labelings whose criterion assign integer values to the vertices, and not to the edges. Interest in such graph labelings has increased in the last few decades, giving rise to many different and exotic labelings that have appeared in literature. The significance of graph labelings is that they are used in many fields of study. Some common applications for graph labelings are found in coding theory, x-ray crystallography, radar, astronomy, circuit design, communication network addressing, data base management, secret sharing schemes, and models for constraint programming over finite domains [**?**]. However, of all the graph labelings in existence, there are three that remain very popular. These three graph labelings are the ρ, β, and α labelings, which are largely attributed to the work done by Alex Rosa in the 1960s. Because of his renowned work with these labelings, they are often referred to as Rosa-type labelings.

1.1 Rho-Labelings

The first Rosa-type labeling we will discuss is the ρ-labeling. Let the assignment of integers to vertices be denoted as the function f; this is also how we will define it in the subsequent sections. Now consider a graph G on n edges, and define edge length to be $min\{|f(u)-f(v)|, 2n+1-|f(u)-f(v)|\}$. The ρ-labeling is an injection f from the vertex set, $V(G)$, into the set $\{0, 1, 2, ..., 2n\}$ such that when every edge is labeled with its edge length, then the set of all edge lengths is equal to $\{1, 2, ..., n\}$. Thus, in this type of labeling, there is exactly one edge length in G for every value $1, 2, ..., n$ [**?**].

Although ρ-labelings have tended to be not as popular in recent studies, they do still lead to some significant results and uses. For example, consider the complete graph K_{2n+1} for any n. We de-

fine a cyclic G-decomposition of K_{2n+1} to be a collection of edge-disjoint, isomorphic subgraphs, $G_1, G_2, ..., G_t$, such that every edge of K_{2n+1} belongs uniquely to one copy of G. Furthermore, because it is a cyclic G-decomposition of K_{2n+1}, one copy of G may be *clicked* to obtain the other copies of G. *Clicking* refers to taking one copy G_j, and mapping each vertex label in the following manner: $i \to i+1$. This thus rotates G_j in a cyclic fashion to obtain G_{j+1}. Clicking can be continued until all copies of G have been obtained; thus when all copies of G are put together, we have the graph K_{2n+1}. The connection to ρ-labelings is that there are nice theorems concerning graph decompositions if a graph is known to admit a ρ-labeling.

Theorem 1.1: Let G be a graph with n edges. There exists a purely cyclic G-decomposition of K_{2n+1} if and only if G has a ρ-labeling [**?**].

1.2 Sigma-Labelings

There is also a not very well known labeling called the σ-labeling, which was introduced by Rosa. Consider a graph G on n edges, we say that G has a σ-labeling if $\{|f(u) - f(v)| : uv \in E(G)\} = \{1, 2, ..., n\}$. We see that this definition is very similar to definition of a ρ-labeling, aside from the possibility of an edge lengths being labeled with $2n + 1 - |f(u) - f(v)|$ [**?**]. Hence, the σ-labeling is essentially a special case of the ρ-labeling, the difference being that wrap-around edges are not allowed. That being said, we would place the σ-labeling higher up on the hierarchy of Rosa-type labelings than the ρ-labeling, based in terms of restrictions on the vertex and edge labels. Hence, any graph that admits a σ-labeling automatically admits a ρ-labeling, and thus all the properties and theorems relating to ρ-labelings thereby relate to σ-labelings.

Despite being less popular than the ρ-labeling, the σ-labeling does still contribute a couple interesting theorems. Consider a graph G on n edges:

Theorem 1.2: If G with n edges admits a σ-labeling, then there exists a cyclic G -decomposition of $K_{2n+2} - I$, where I is a 1-factor of K_{2n+2} [**?**].

Theorem 1.3: Let G be a graph of size n and suppose every vertex of G has even degree. If G admits a σ-labeling then $n \equiv 0$ or 3 (mod 4) [**?**].

1.3 Beta-Labelings

The next labeling to consider is the β-labeling, also known as the graceful labeling, of a graph. Consider a graph G on n edges with a ρ-labeling as previously described. When the one-to-one function f is now restricted so that $V(G) \to \{0, 1, 2, ..., n\}$, and edges labeled with their edge lengths result in the set $\{1, 2, ..., n\}$, we obtain a β, or graceful, labeling [**?**]. Hence a graceful labeling is more restrictive than ρ-labeling and σ-labelings, and we begin to see the building of a hierarchy of graph valuations that Rosa introduced.

Graceful labelings have become increasingly popular in graph theory publications over the past decade. One reason is because if G is a graceful graph, then it also has a ρ-labeling, and therefore every property and theorem that applies to a graph with a ρ-labeling also applies to G. That is, Theorem 1.1 from above may be applied to any graceful graph as well. Also, since graceful graphs are more restrictive with vertex labels, while having each edge length still distinct, other nice theorems and properties come about. For example, some decomposition theorems only apply to graceful graphs, and cannot be applied to graphs that only admit a ρ-labeling. One such theorem is the following:

Theorem 1.4: Let G be a graph of size n that has a β-labeling. Then there exists a cyclic $2G$ -decomposition of $K_{2n+2} - I$, where I is a 1-factor [**?**].

Here it is important to note that a cyclic $2G$-decomposition of $K_{2n+2} - I$ refers to taking two vertex-disjoint copies of G, where one copy keeps its labels exactly the same, while the other changes its vertex labels by adding $n + 1$. Again, we go about clicking one copy of G to obtain $K_{2n+2} - I$. Here, the 1-factor will be all edges of length $n + 1$ [**?**].

Another theorem about graceful labelings concerns trees, which are simply connected graphs that contain no cycles. The theorem goes as follows:

Theorem 1.5: If a tree G of size n has a β-labeling, then there exists a purely cyclic G-factorization of ${}^2K_{n+1}$ [**?**].

Here the notation ${}^2K_{n+1}$ refers to taking the graph of K_{n+1} and replacing every single edge with two edges. Now being on the topic of trees, it is also significant to mention here one of the most famous conjectures concerning labelings:

Conjecture 1.6: Every tree has a β-labeling [**?**].

Although many attempts have been made at proving this conjecture, it still remains open. In any case, theorems and conjectures such as those mentioned here are often useful in some applications of graceful labelings that will be discussed in section 1.6.

1.4 Alpha-Labelings

Finally, there is the α-labeling, which is sometimes referred to as a bipartite labeling since it is a labeling restricted only to bipartite graphs. Consider a bipartite graph G on n edges with bipartition (X, Y). If G admits a β-labeling, f, and there exists an integer k such that $f(x) \leq k$ for every $x \in X$ and $f(y) > k$ for every $y \in Y$, then G admits an α-labeling [**?**]. Thus the α-labeling is the most restrictive of the three, and is therefore on top of the hierarchy. We now see that every graph that admits an α-labeling, also admits a β- and ρ-labeling. Furthermore, any graph that admits a β-labeling, also must have a ρ-labeling. The significance of this hierarchy is that each type of labeling has certain properties, and if a graph admits one type of labeling, then it also admits every type of "less restrictive" graph labeling within the hierarchy; and therefore has the properties of those labelings as well.

Since the α-labeling is the highest in the Rosa-type labeling hierarchy, and hence the most restrictive, we would again expect useful theorems that apply only to this labeling, and not to any "less restrictive" labeling. In keeping with the theme of the previously mentioned theorems, we have the following:

Theorem 1.7: If G with n edges admits an α-labeling, then there exists a cyclic G-decomposition of K_{2nx+1} for every positive integer x [**?**].

This theorem is much more robust than the previous theorems regarding cyclic G-decompositions of the "less restrictive" labelings since this one extends to an infinite family of graphs. The previous theorems were restricted to just one value of n, however this theorem applies to all multiples of n since x is any positive integer. However, despite this theorem being much more robust than previously mentioned theorems, one should note that Theorems 1.1, 1.2, 1.3, 1.4, and 1.5 do still apply to graphs that admit an α-labeling due to the hierarchy of Rosa-type labelings.

1.5 Ordered Labelings

Ordered Labelings are a relatively new concept that have only been around for a few decades. However, despite being newer to the field of graph theory, the idea of an ordered labeling yields some nice theorems and results.

Ordered labelings involve further restrictions placed on the labelings that have been previously described here. Consider a bipartite graph G on n edges with bipartition (A, B). If G admits a ρ, σ, or β-labeling, and if $f(a) < f(b)$ for each edge ab with $a \in A$ and $b \in B$, then this labeling is considered to be *ordered* [**?**]. If we have a graph that admits β-labeling, and it happens to be an ordered labeling, it is commonly denoted as a β^+-labeling. In a similar fashion, we have ordered σ-labelings denoted as σ^+.

A further extension of ordered labelings is called "uniformly ordered labelings." A uniformly ordered labeling is a labeling for which $f(a) < f(b)$ for every $a \in A$ and $b \in B$ [**?**]. Such labelings are denoted in a similarly fashion, only we append one more "+" to the superscript. That is, a uniformly ordered β-labeling would be denoted β^{++}. However, note that by definition, a uniformly ordered β-labeling is equivalent to an α-labeling. That being said, sometimes an ordered β-labeling (i.e. a β^+-labeling) is referred to as a "near *alpha*-labeling" or a "gracious labeling" [**?**].

As previously mentioned, these further restrictions imposed on graph labelings yield certain interesting theorems such as the following:

Theorem 1.8: If a bipartite graph G of size n has a β^+-labeling, then there exists a purely cyclic G-decomposition of $K_{n,n}$ [**?**].

Theorem 1.9: If G with n edges admits an β^+-labeling, then there exists a cyclic G-decomposition of K_{2nx+1} for every positive integer x [**?**].

Conjecture 1.10: Every tree admits a β^+-labeling [**?**].

Theorem 1.11: Let G be a bipartite graph with n edges. If G admits a ρ^+-labeling, then there exists a cyclic (K_{2nx+1}, G)-design for all non-negative integers x [**?**].

Note that the hierarchy of Rosa-type labelings we have mentioned here (based on restrictions), continues to work in the same fashion when we introduce ordered labelings. That is, since a β^+-labeling

is more restrictive than $\sigma^{++}, \sigma^{+}, \rho^{++}$, and ρ^{+}-labelings, any theorems that apply to any of these labelings will apply to a graph that admits a β^{+}-labeling (or even a more restrictive bipartite graph, such as one that admits an α-labeling). For example, Theorem 1.11 would apply to a graph that admits any other ordered labeling we have discussed since it is the least restrictive ordered labeling.

1.6 Applications of Graceful Labelings

Of the three Rosa-type labelings mentioned here, graceful labelings (β-labelings) are arguably the most popular, and they are what we will be examining in this paper. Graceful labelings of various structures have been studied in recent years, and interestingly enough do have some industry applications.

1.6.1 Communication Networks

One such application is in communication network labeling. For example, if one had a communication network with a fixed number $n+1$ of communicating centers (i.e. vertices) and they were numbered $0, 1, ..., n$, then the lines between any two centers could be labeled with the difference between the two center labels (i.e. the vertex labels). If the communication center grid was laid out in a fashion conducive to a graceful graph, we would then be able to label the connections between each center such that each connection would have a distinct label. There are many advantages of creating a communications network that is analogous a graceful graph. One advantage is that if a link goes out, a simple algorithm could detect which two centers are no longer linked, since each connection is labeled with the difference between the two communication centers. Another advantage is that this network also would have all the same properties as a graceful graph; such as having cyclic decompositions [**?**].

1.6.2 Coding Theory

Another application for graceful labelings is in the field of Coding Theory. First, it is important to note that with this application, as in many other applications of graceful labelings, from time to time it is necessary to use semi-graceful labelings or even quasi-graceful labelings. A semi-graceful labeling is defined to be one in which the constraint that the edge lengths need to be consecutive is relaxed. That is, one edge length may be skipped in the graph labeling by adding an $n+1$ edge length into the graph. An example of this is would be how K_4 may be gracefully labeled, but not K_5. However, K_5 may be semi-gracefully labeled if the edge length 6 is ignored, and 11 is added as a

possible edge length. When this is done, and the vertices are labeled 0,1,4,9,11, K_5 is semi-gracefully labeled [**?**].

There is also a quasi-graceful labeling, which is defined to be when the vertex labels are allowed to be extended beyond the largest edge length value, however the edge length constraints are left unchanged. Between these two types of graceful labeling variations, the extension to coding theory is made possible. The idea is that once a graph is gracefully, semi-gracefully, or quasi-gracefully labeled, the vertex labels may be used to create a special kind of "ruler." This is done by assigning each vertex label to the ruler, while using no other tick marks on the ruler; this is referred to as a Golomb Ruler [**?**]. It is important to note that the distance between the marks on the Golomb Ruler are left respectively spaced apart, meaning 4 is still three units away from 1; however, these such rulers measure time and not distance in coding theory. Golomb Rulers are used in Radar Type Codes, Self-Orthogonal Codes in Algebraic Coding, and Synch Set Codes such as those utilized in lasers used with rotating disks. Hence, we see yet another application of graceful labelings.

2 Cyclic Structures

Many structures that have been studied in recent years are structures that involve cycles. One reason for this is that Rosa proved that all cycles that are of lengths $n \equiv 0, 3 \pmod 4$ are graceful. Hence, many families of cyclic structures tend to have graceful subfamilies. We will now investigate some of these structures.

2.1 Wheels

A wheel graphs is defined as $W_n = C_n + K_1$, where n is the number of vertices in the cycle. Wheel graphs are a nice structure to work with since they have been shown to be always graceful; that is, they always admit a β-labeling [**?**]. However, note that cycles are graceful if and only if $n \equiv 0, 3 \pmod 4$. That being said, we arrive at an important point: subgraphs of graceful graphs may not be graceful. Below we see some examples of wheel graphs.

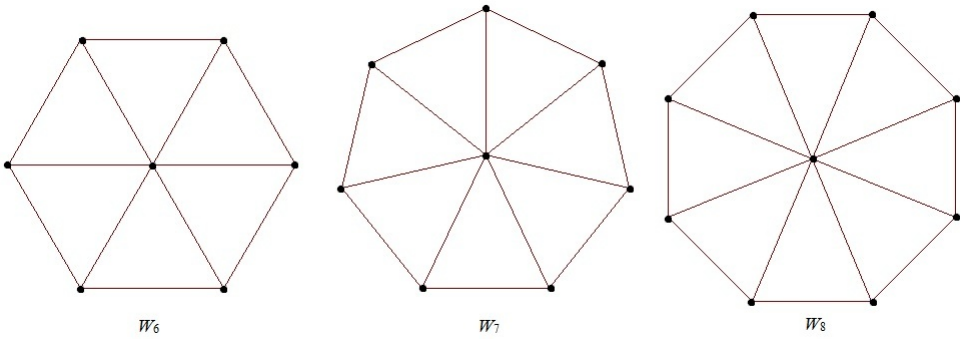

Figure 1: Wheel Graphs

2.2 Gear Graphs

The next cyclic structure to consider is a gear graph. A gear graph is essentially an extension of a wheel graph, being that it is formed by adding a vertex between each pair of adjacent vertices in the n-cycle of a wheel graph [**?**]. We will denote a gear graph here as G_n, where n refers to the number of vertices in the cycle of the wheel graph. Hence, the number of vertices in any gear graph G_n is $2n + 1$. Like wheel graphs, gear graphs have also been proven to be always graceful. Note the following examples of gear graphs [**?**].

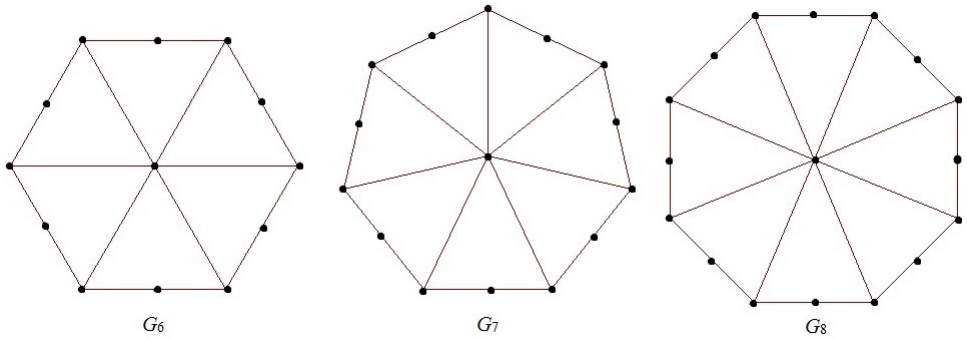

Figure 2: Gear Graphs

2.3 Helms

Helm graphs are another nice extension of wheel graphs. To form a helm graph, denoted here as H_n, we take the wheel graph W_n and append a pendant edge to each vertex of the n-cycle. Helm graphs turn out to be another graceful cyclic structure [**?**].

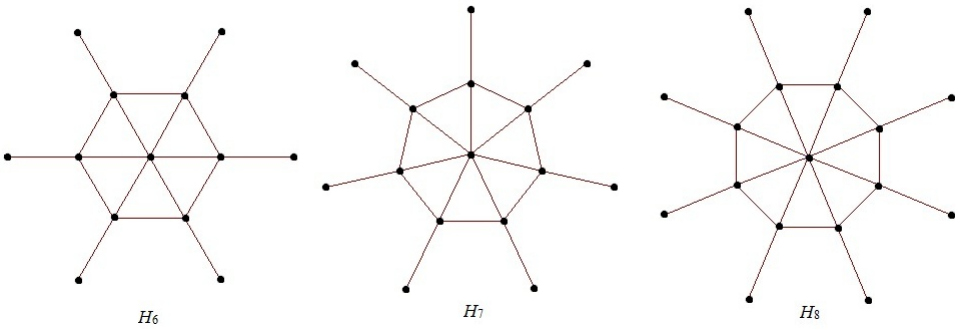

Figure 3: Helm Graphs

2.4 Closed Helms

Closed Helms are denoted here as $\underline{H_n}$, and they are formed by taking the a helm graph, H_n, and placing an edge between each pendant vertex to form a second cycle in the graph [**?**]. Although there are few results concerning closed helms, we include them here to demonstrate an iterative process which leads to our next class of graphs: Web graphs.

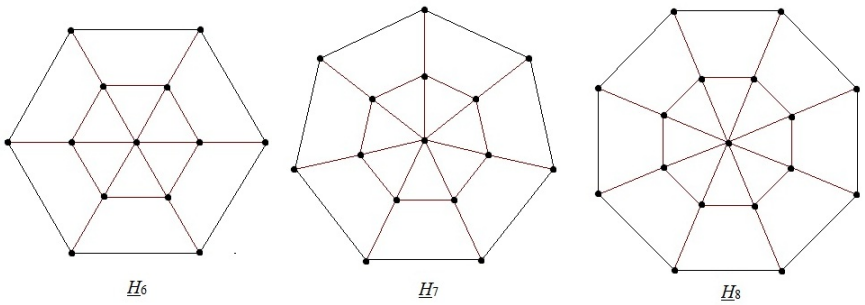

Figure 4: Closed Helm Graphs

2.5 Webs

Next let us look at web graphs. To form a web graph, consider how we went from a wheel to a helm, to a closed helm graph, and then iterate the process one more time. That is, to get a web graph we take a closed helm graph, remove the central vertex, and attach a pendant edge to each vertex of the n-cycle. The result are the examples below. We could continue this iterative process of placing an edge between pendant vertices, and then attaching new pendant edges to each vertex of the n-cycle to get more and more cycles in our web graphs than just two, as shown below. Thus, we denote a generalized web graph as $W(t, n)$, where t is the number of n-cycles. Not all web graphs are necessarily graceful, but $W(3, n)$ and $W(4, n)$ web graphs have been shown to be graceful by Yang [**?**].

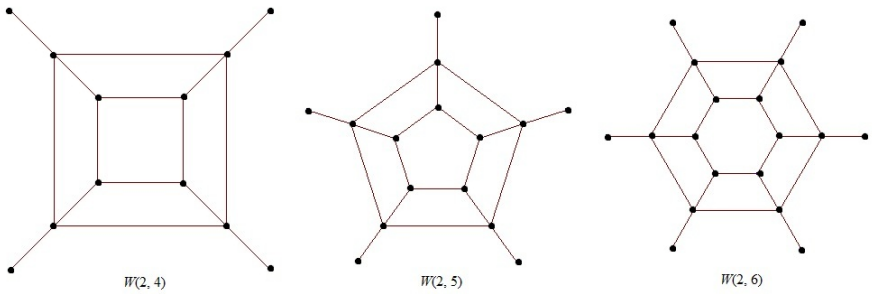

Figure 5: Web Graphs

2.6 Dragons/Kites/Canoe Paddles

The canoe paddle graph (also known as the dragon or kite) is made up of a cycle of length n, (i.e. C_n) with a path of length l adjacent to it. Because of how it is defined here, we will denote canoe paddle graphs as $CP(n, l)$. One nice result known about canoe paddles is that all canoe paddles are graceful, which was proved by Truszczynski in his paper "Graceful Unicyclic Graphs" [**?**].

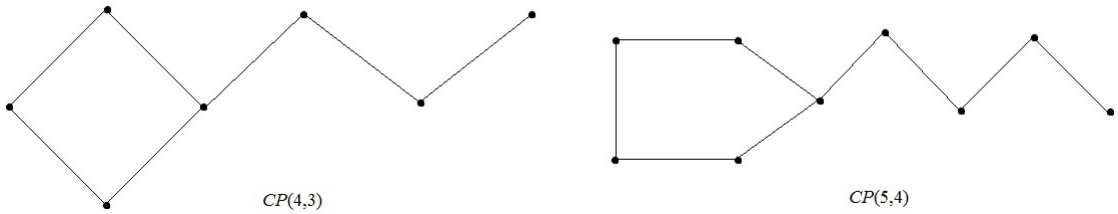

Figure 6: Canoe Paddle Graphs

2.7 Kayak Paddles

From a canoe paddle graph, we can expand the graph and obtain more interesting structures; such as a kayak paddle. A kayak paddle is described as two cycles, C_r and C_s that are joined together by a path of length l. These graphs are denoted here as $KP(r, s, l)$, and Ann Litersky proved in her Masters Thesis that certain classes of kayak paddles are also graceful.

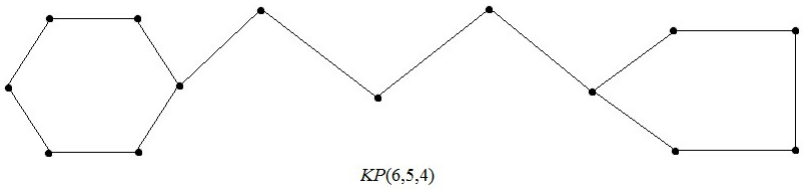

Figure 7: Kayak Paddle Graph Example

2.8 Necklace Graphs

One cycle-related structure to consider is a *necklace graph*. A necklace graph can be described as a graph made up of r beads, where each bead is comprised of one cycle of length k which is incident with a path of length l; hence each bead is a canoe paddle. We construct our necklace graphs by stringing together our beads to form a cycle of beads, where each path is adjoined to *opposite* vertices of the k-cycle. In this context, opposite vertices refer to vertices who have paths of uniform length when traversing the cycle in either direction, and this path length is the maximum length possible. In the case of C_4 necklace graphs, opposite vertices are simply non-adjacent vertices. Also, note that necklace graphs must therefore be comprised of even length cycles. Furthermore, we denote a necklace graph by $N(C_k, l, r)$; where k is the length of every cycle, l is length of the path, and r denotes the number of beads. In this paper, the $N(C_4, l, r)$ family of necklace graphs is investigated, where cycles were held constant with a length of four, while path lengths and number of beads were increased. We found a graceful labeling algorithm for C_4 necklace graphs for any $l > 0$ and any $r > 1$. The key to this algorithm is first breaking all the possible cases into families, which are divided up by the path length between cycles mod 4. Next, we then break each familiy into subfamilies depending on the number of beads mod 4. Thus we are left with sixteen infinite subfamilies of necklace graphs. For example, first we have the family of $N(C_4, l, r)$ where l is congruent to 0 (mod 4), with four subfamilies: one per value of r where $r \equiv 0$, 1, 2, 3 (mod 4). Similarly, the same follows for $l \equiv 1$, 2, 3 (mod 4), each with analogous subfamilies.

This paper will use the notation $\overline{x_i y_j}$ to represent the edge length between the vertices x_i and y_j, and m to represent the total number of edges in a given graph. We will also make use of the fact that there are two ways to gracefully label a cycle of length four when represented as a bipartite graph.

Type I C_4: The edge $\overline{x_0 y_0} = 4$, $\overline{x_0 y_1} = 3$, $\overline{x_1 y_0} = 2$, and $\overline{x_1 y_1} = 1$.
*Note that in a Type I C_4 the two edges adjacent with the vertex x_0 differ by an increment of 1.

Type II C_4: The edge $\overline{x_0 y_0} = 4$, $\overline{x_0 y_1} = 2$, $\overline{x_1 y_0} = 3$, and $\overline{x_1 y_1} = 1$.
*Note that in a Type II C_4 the two edges adjacent with the vertex x_0 differ by an increment of 2.

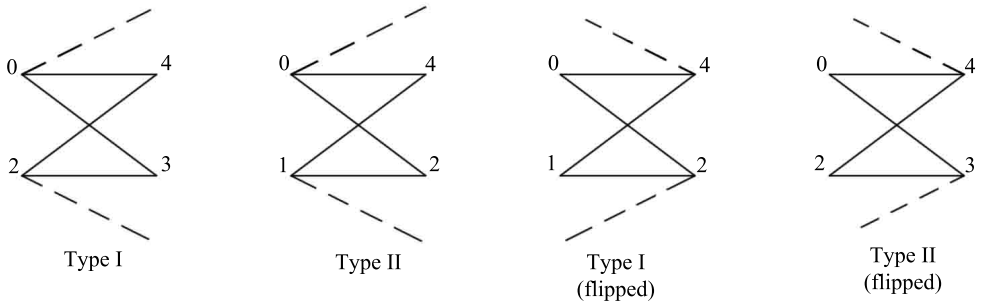

Figure 8: C_4 Labelings

Since we will be using bipartite (or near-bipartite as discussed later) representations of our necklace graphs in the algorithm described in this paper, it is imperative to highlight the way in which a bead is constructed. The C_4 is first drawn as a bipartite graph, followed by the path being attached to the X (left) partition of the four-cycle. This will always be the way in which a bead is being described, unless it is remarked that the bead is *flipped*. If a bead is said to be flipped, then the path is being attached to the Y (right) partition of the four-cycle. It also significant to mention that these algorithms mentioned work with the non-clasp edge lengths strictly decreasing through the bipartite (near-bipartite) representations, the X partition vertex labels strictly increasing, and the Y vertex labels strictly decreasing. Note that a near-bipartite graph refers to a graph that is not bipartite, but by removing one edge the graph is then bipartite. That is, the vertices of the graph may be divided into two distinct sets, say X and Y, such that every edge connects a vertex from X to a vertex in Y. Finally, necklace graphs with only one bead will be described as special cases, the algorithms described will therefore be for $r > 1$.

3 Family: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$

First, notice that this family may be represented with a bipartite graph; consequently that is how each of its subfamilies will be described here.

3.1 Subfamily: $r \equiv 0 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$, $r \equiv 0 \pmod 4$

In order to start our labeling process, we must first determine the length of the *clasp edge* for this infinite subfamily. We define the clasp edge to be the edge that connects the first vertex of the X partition (x_0) to either the last vertex of the Y partition (if our graph is bipartite), or the edge that connects x_0 to the last vertex of the X partition (if our graph is near-bipartite). We give this clasp edge length $\frac{m}{2}$. We know that this expression will always yield an integer since an even length cycle plus an even length path will clearly give an even number of total edges regardless the number of beads; hence divisibility by two. Furthermore, since this family has an even number of beads and the clasp edge is equal to half the total number of edges, we see that we will have *jump* in edge lengths in the path between the $(\frac{r}{2})^{th}$ cycle and the $(\frac{r}{2}+1)^{th}$ cycle in order to save that length for the clasp edge. Now it is important to note that all of our labelings in this paper will start with the vertex $x_0 = 0$ and the vertex $y_0 = m$. We then will describe the edge labels since they will follow a nice, sequential, decreasing pattern. Remember that a graceful labeling is a vertex labeling, not an edge labeling. However, in this algorithm we are fixing the vertex labels of x_0 and y_0; thus the edge labels will force the values of the vertex labels. Finally, this subfamily only uses Type I C_4 labelings.

Base case: $N(C_4, 4, 4)$

Starting with the vertices $x_0 = 0$ and $y_0 = m$, we calculate the clasp edge to have length $\frac{32}{2} = 16$. So then, the first cycle is labeled $\overline{x_0y_0} = 32$, $\overline{x_0y_1} = 31$, $\overline{x_1y_0} = 30$, $\overline{x_1y_1} = 29$. We then continue sequentially through the path, $\overline{x_1y_2} = 28$, $\overline{x_2y_2} = 27$, $\overline{x_2y_3} = 26$, $\overline{x_3y_3} = 25$. The second cycle continues with $\overline{x_3y_4} = 24$, $\overline{x_3y_5} = 23$, $\overline{x_4y_4} = 22$, $\overline{x_4y_5} = 21$. We continue through the second path with $\overline{x_4y_6} = 20$, $\overline{x_5y_6} = 19$, $\overline{x_5y_7} = 18$, $\overline{x_6y_7} = 17$, until now we have reached the clasp jump. Because of this we label the next C_4 starting with $\overline{x_6y_8} = 15$, since our clasp edge will be length 16. Now finishing up the third C_4 we have: $\overline{x_6y_9} = 14$, $\overline{x_7y_8} = 13$, $\overline{x_7y_9} = 12$. The third path is labeled with edge lengths $\overline{x_7y_{10}} = 11$, $\overline{x_8y_{10}} = 10$, $\overline{x_8y_{11}} = 9$, and $\overline{x_9y_{11}} = 8$. The fourth and final C_4 is labeled with edge lengths $\overline{x_9y_{12}} = 7$, $\overline{x_9y_{13}} = 6$, $\overline{x_{10}y_{12}} = 5$, and $\overline{x_{10}y_{13}} = 4$. We then conclude with the last path being $\overline{x_{10}y_{14}} = 3$, $\overline{x_{11}y_{14}} = 2$, $\overline{x_{11}y_{15}} = 1$, and the clasp edge $\overline{x_0y_{15}} = 16$. All that is left is to label the vertices now. So starting with x_0 being labeled 0 and y_0 labeled 32, this is easily accomplished keeping in mind that the X partition in strictly increasing in vertex labels, while the

Y partition is strictly decreasing. Now our first necklace graph in this subfamily is labeled.

More general case: $N(C_4, 4, r)$, $r \equiv 0 \pmod 4$

Again, starting with the vertices $x_0 = 0$ and $y_0 = m$, we calculate the clasp edge to have length $\frac{m}{2}$. The first cycle is then labeled $\overline{x_0y_0} = m$, $\overline{x_0y_1} = m-1$, $\overline{x_1y_0} = m-2$, $\overline{x_1y_1} = m-3$. We then continue sequentially through the path, $\overline{x_1y_2} = m-4$, $\overline{x_2y_2} = m-5$, $\overline{x_2y_3} = m-6$, $\overline{x_3y_3} = m-7$. The second cycle continues with $\overline{x_3y_4} = m-8$, $\overline{x_3y_5} = m-9$, $\overline{x_4y_4} = m-10$, $\overline{x_4y_5} = m-11$. Again, we would continue sequentially through the path and continue using this labeling method until we reach the $(\frac{r}{2})^{th}$ bead. This bead is special because, as previously mentioned, the jump in edge lengths will occur in its path. Aside from the jump though, all continues as previously described until we reach the r^{th} bead. The r^{th} cycle is labeled in the same fashion as earlier, followed by labeling the r^{th} path as follows: $\overline{x_{\frac{m}{2}-(r+2)}y_{\frac{m}{2}+1}} = 3$, $\overline{x_{\frac{m}{2}-(r+1)}y_{\frac{m}{2}+1}} = 2$, $\overline{x_{\frac{m}{2}-(r+1)}y_{\frac{m}{2}}} = 1$, $\overline{x_0y_{\frac{m}{2}}} = \frac{m}{2}$. Finally, we finish by labeling the rest of the vertices. We started with $x_0 = 0$, and from the edge lengths we will have $x_1 = 2$, $x_2 = 3$, $x_3 = 4$, $x_4 = 6$, $x_5 = 7$, etc. Notice that we have almost sequentially increasing vertex labels, we simply skip one vertex label in the X partition every time we are within a cycle. In the Y partition we started with $y_0 = m$, and have sequentially decreasing vertex labels, with the exception of $m - 4(\frac{r}{2})$. This is because y_0 is labeled n, there are four Y partition vertices per bead, and the jump happens after the last edge label in the path from the $(\frac{r}{2})^{th}$ bead; thus the vertex label $n-4(\frac{r}{2})$ is skipped in the Y partition. Notice though that we can simplify $m-4(\frac{r}{2})$ to $m-2r$.

Generalization:

We now have a more general representation of this subfamily gracefully labeled, where the number of beads is allowed to increase, as long as $r \equiv 0 \pmod 4$. However, notice that we can also allow the path length to vary as long as $l \equiv 0 \pmod 4$, and the graph would still be gracefully labeled. To see why this is true, let us consider the "more general" case mentioned above. If we extend the path length to 8, we have merely added two vertices to each partition, by nature of the bipartite representation being utilized by this algorithm. However, this doesn't change anything that was outlined above. We have only stretched our graph out in a very "regular" fashion. So no matter how many of multiples of 4 we add to the path length between every C_4, we still have our graph gracefully labeled by the above algorithm. Hence, we now see that we have every necklace graph in this infinite subfamily labeled by the algorithm.

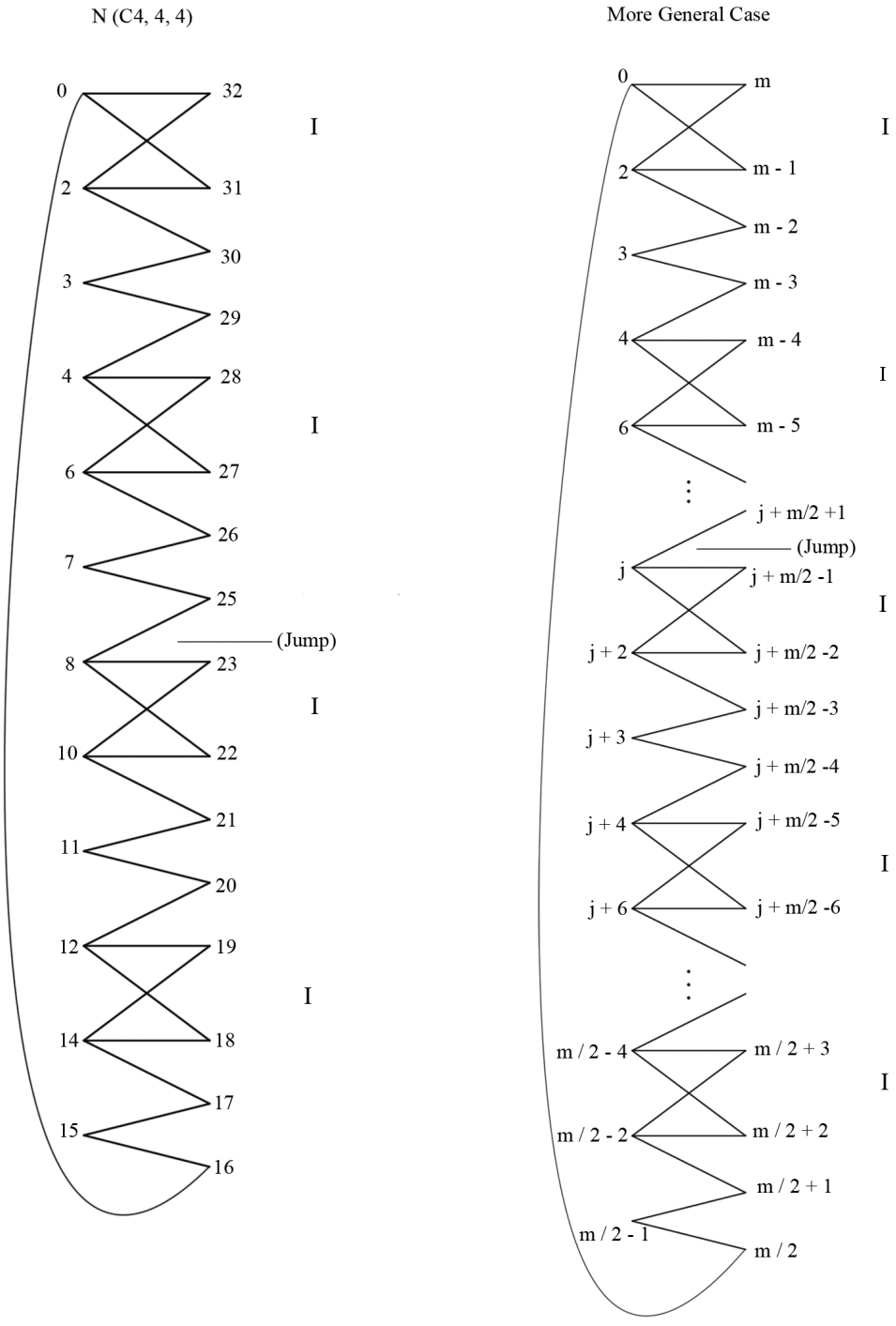

Figure 9: 3.1 Subfamily Illustrations

3.2 Subfamily: $r \equiv 1 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$, $r \equiv 1 \pmod 4$

As before, we will fix the vertex labels $x_0 = 0$ and $y_0 = m$ and start by determining the clasp edge for this infinite subfamily. This clasp edge for this subfamily will be $\frac{m-2}{2}$. Again, this expression will always yield an integer since each bead has an even number of edges, and so regardless of how many beads there are, the graph will always have a even number of total edges. Furthermore, since this family has an even number of beads and the clasp edge is equal to $\frac{m-2}{2}$, we see that the jump in the edge lengths will occur in the path between the $(\frac{r+1}{2})^{th}$ cycle and the $(\frac{r+1}{2}+1)^{th}$ cycle in order to save that length for the clasp edge. Finally, this subfamily starts with a Type II C_4, followed by only Type I four-cycles until we reach the last C_4; this last C_4 will be a Type II.

Base case: $N(C_4, 4, 5)$

With vertex labels $x_0 = 0$ and $y_0 = m = 40$, and clasp edge having length $\frac{40-2}{2} = 19$, we may label the first cycle as: $\overline{x_0y_0} = 40$, $\overline{x_1y_0} = 39$, $\overline{x_0y_1} = 38$, $\overline{x_1y_1} = 37$. We then continue sequentially through the path, $\overline{x_1y_2} = 36$, $\overline{x_2y_2} = 35$, $\overline{x_2y_3} = 34$, $\overline{x_3y_3} = 33$. The second cycle continues with $\overline{x_3y_4} = 32$, $\overline{x_3y_5} = 31$, $\overline{x_4y_4} = 30$, $\overline{x_4y_5} = 29$. We continue through the second path with $\overline{x_4y_6} = 28$, $\overline{x_5y_6} = 27$, $\overline{x_5y_7} = 26$, $\overline{x_6y_7} = 25$. The third cycle is labled with $\overline{x_6y_8} = 24$, $\overline{x_6y_9} = 23$, $\overline{x_7y_8} = 22$, $\overline{x_7y_9} = 21$. Continuing on, we have $\overline{x_7y_10} = 20$, and then we reach what should be 19. However, recall that that is the edge length we are saving for the clasp edge. So the second path continues with edge lengths $\overline{x_8y_{10}} = 18$, $\overline{x_8y_{11}} = 17$, and $\overline{x_9y_{11}} = 16$. Our next C_4 is labeled with edge lengths $\overline{x_9y_{12}} = 15$, $\overline{x_9y_{13}} = 14$, $\overline{x_{10}y_{12}} = 13$, and $\overline{x_{10}y_{13}} = 12$. Continuing into the fourth path, $\overline{x_{10}y_{14}} = 11$, $\overline{x_{11}y_{14}} = 10$, $\overline{x_{11}y_{15}} = 9$, $\overline{x_{12}y_{15}} = 8$. The fifth and final C_4 is labeled as: $\overline{x_{12}y_{16}} = 7$, $\overline{x_{13}y_{16}} = 6$, $\overline{x_{12}y_{17}} = 5$, and $\overline{x_{13}y_{17}} = 4$. Finally, we reach our last path and label it with $\overline{x_{13}y_{18}} = 3$, $\overline{x_{14}y_{18}} = 2$, $\overline{x_{14}y_{19}} = 1$, and the last edge (being the clasp edge) is $\overline{x_0y_{19}} = 19$. The last step now in our graph labeling is to again label the vertices. In the same fashion as before, we start with x_0 being labeled 0 and y_0 labeled 40. Keeping in mind that the X partition in strictly increasing in vertex labels, while the Y partition is strictly decreasing, together with the edge lengths listed above; our base case necklace graph in this subfamily is labeled.

More General case: $N(C_4, 4, r)$, $r \equiv 1 \pmod 4$

The cycle is labeled $\overline{x_0y_0} = m$, $\overline{x_1y_0} = m-1$, $\overline{x_0y_1} = m-2$, $\overline{x_1y_1} = m-3$. We then continue sequentially through the path, $\overline{x_1y_2} = m-4$, $\overline{x_2y_2} = m-5$, $\overline{x_2y_3} = m-6$, $\overline{x_3y_3} = m-7$. The second cycle continues with $\overline{x_3y_4} = m-8$, $\overline{x_3y_5} = m-9$, $\overline{x_4y_4} = m-10$, $\overline{x_4y_5} = m-11$. Again, we would continue sequentially through the path and continue using this labeling method until we reach the $(\frac{r+1}{2})^{th}$ bead. This bead is special because the jump in edge lengths will occur in its path,

just as in the previous subfamily. Aside from the jump though, everything continues as previously described until we reach the r^{th} bead. The r^{th} cycle is again labeled in the same fashion as before, finishing up with the r^{th} path being labeled as before: 3, 2, 1, $\frac{m-2}{2}$. Remember that the last edge of this last path is our clasp edge. Finally, we just have to label the vertices. We started with $x_0 = 0$, and so using the above edge length we see that $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 5$, $x_5 = 6$, etc. Notice that we have strictly increasing vertex labels in the X partition. In the Y partition we start with $y_0 = m$, and have strictly decreasing vertex labels.

Generalization:
As before, we now have a more general representation of this subfamily gracefully labeled, where the number of beads is allowed to increase, as long as $r \equiv 1 \pmod 4$. So using the same logic as previously mentioned, we can conclude that allowing the path length to vary as long as $l \equiv 0 \pmod 4$, does not change the fact that the graph would still be gracefully labeled by the above algorithm. Thus, we now see that we have every necklace graph in this infinite subfamily labeled.

Special Case: $N(C_4, 4, 1)$
Recall how necklace graphs with only one bead do not follow our algorithm. The reason for this is because the graphs are simply not large enough to have the amount of regularity in structure that is required for our algorithm. Therefore, we treat these graphs as special cases. Now we examine how to label this graph.

We first determine the clasp edge, which will have an edge length of 4. The cycle is labeled $\overline{x_0y_0} = 8$, $\overline{x_0y_1} = 7$, $\overline{x_1y_0} = 6$, $\overline{x_1y_1} = 5$, and then we continue sequentially through the path, skipping the edge length 4 since that is saved for our clasp edge. So our path is labeled: $\overline{x_1y_2} = 3$, $\overline{x_2y_2} = 2$, $\overline{x_2y_3} = 1$, $\overline{x_0y_3} = 4$. Note that $\overline{x_0y_3}$ was our clasp edge. Now as with all of our necklace graphs, we will start labeling the vertices with $x_0 = 0$. So, our vertices are thereby labeled, $x_0 = 0$, $x_1 = 2$, $x_2 = 3$, and $y_0 = 8$, $y_1 = 7$, $y_2 = 5$, $y_3 = 4$. Our smallest case in this infinite subfamily is now gracefully labeled as well.

Note though, that if we fix $r = 1$ and extend the path lengths in such a fashion that they are still congruent to 0 (mod 4), our graphs are still graceful by the same argument used above to extend from the "more general case" to a truly general case.

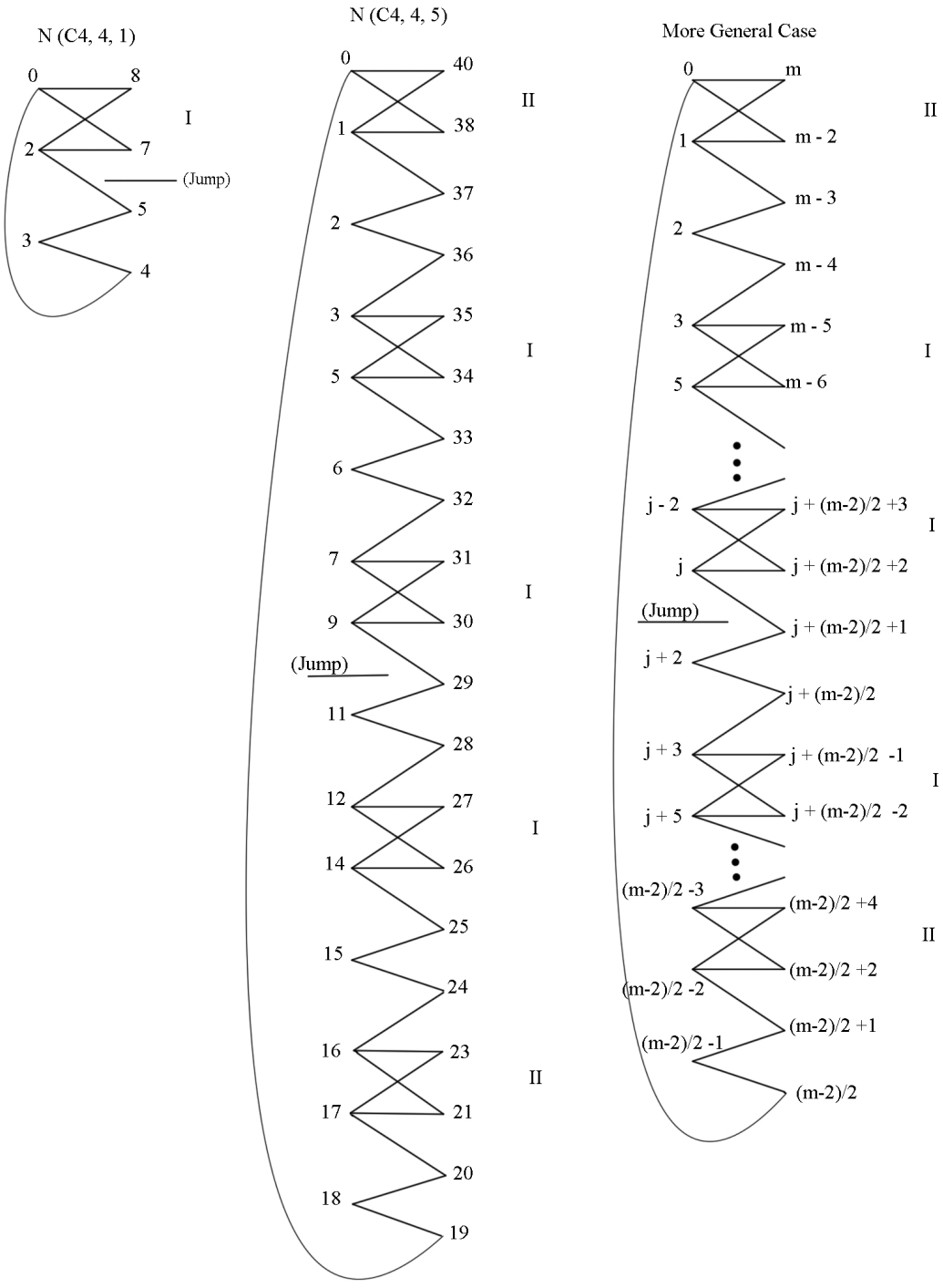

Figure 10: 3.2 Subfamily Illustrations

3.3 Subfamily: $r \equiv 2 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$, $r \equiv 2 \pmod 4$

As in the last two sections, the *first* two vertex labels $x_0 = 0$ and $y_0 = m$ are fixed, and we begin with determining the clasp edge for this infinite subfamily. As with the first subfamily, the clasp edge length is $\frac{m}{2}$, which will always yield an integer since an even number of beads will clearly give an even number of total edges in the graph. Next, consider the fact that since this subfamily has an even number of beads, and since the clasp edge is equal to half the total number of edges, we see that we will have the jump in edge lengths in the path between the $(\frac{r}{2})^{th}$ cycle and the $(\frac{r}{2}+1)^{th}$ cycle in order to save that length for the clasp edge. Finally, as with the first subfamily, this subfamily only uses Type I C_4 labelings.

Small case: $N(C_4, 4, 6)$

We will first look at the graph $N(C_4, 4, 6)$, rather than the base case $N(C_4, 4, 2)$. The reason is to show a larger case that better illustrates what is happening with our algorithm; however, figures for both cases will be included.

After fixing the first two vertices and calculating the length of the clasp edge as before, the first cycle is labeled $\overline{x_0y_0} = 48$, $\overline{x_0y_1} = 47$, $\overline{x_1y_0} = 46$, $\overline{x_1y_1} = 45$. We then continue sequentially through the path, $\overline{x_1y_2} = 44$, $\overline{x_2y_2} = 43$, $\overline{x_2y_3} = 42$, $\overline{x_3y_3} = 41$. The second cycle continues with $\overline{x_3y_4} = 40$, $\overline{x_3y_5} = 39$, $\overline{x_4y_4} = 38$, $\overline{x_4y_5} = 37$. We continue through the second path with $\overline{x_4y_6} = 36$, $\overline{x_5y_6} = 35$, $\overline{x_5y_7} = 34$, $\overline{x_6y_7} = 33$. The third cycle gets labeled $\overline{x_6y_8} = 32$, $\overline{x_6y_9} = 31$, $\overline{x_7y_8} = 30$, $\overline{x_7y_9} = 29$. Continue on through the third path with: $\overline{x_7y_{10}} = 28$, $\overline{x_8y_{10}} = 27$, $\overline{x_8y_{11}} = 26$, $\overline{x_9y_{11}} = 25$. The next edge would be be labeled 24, but note that this is saved for the clasp edge. Hence, the first edge of the fourth cycle is $\overline{x_9y_{12}} = 23$, $\overline{x_9y_{13}} = 22$, $\overline{x_{10}y_{12}} = 21$, and $\overline{x_{10}y_{13}} = 20$. The next path then becomes $\overline{x_{10}y_{14}} = 19$, $\overline{x_{11}y_{14}} = 18$, $\overline{x_{11}y_{15}} = 17$, and $\overline{x_{12}y_{15}} = 16$. The fifth cycle is: $\overline{x_{12}y_{16}} = 15$, $\overline{x_{12}y_{17}} = 14$, $\overline{x_{13}y_{16}} = 13$, and $\overline{x_{13}y_{17}} = 12$. The fifth path then is: $\overline{x_{13}y_{18}} = 11$, $\overline{x_{14}y_{18}} = 10$, $\overline{x_{14}y_{19}} = 9$, and $\overline{x_{15}y_{19}} = 8$. Finally, we reach the sixth cycle: $\overline{x_{15}y_{20}} = 7$, $\overline{x_{15}y_{21}} = 6$, $\overline{x_{16}y_{20}} = 5$, and $\overline{x_{16}y_{21}} = 4$. Then comes the final path: $\overline{x_{16}y_{22}} = 3$, $\overline{x_{17}y_{22}} = 2$, $\overline{x_{17}y_{23}} = 1$, and $\overline{x_0y_{23}} = 24$. All that is left now is to label the rest of the vertices. Recall that x_0 is labeled 0 and y_0 is labeled 48. Keeping in mind that the X partition in strictly increasing in vertex labels, while the Y partition is strictly decreasing, together with the edge lengths listed above; our first necklace graph in this subfamily is labeled.

More General case: $N(C_4, 4, r)$, $r \equiv 2 \pmod 4$

The cycle is labeled $\overline{x_0y_0} = m$, $\overline{x_0y_1} = m-1$, $\overline{x_1y_0} = m-2$, $\overline{x_1y_1} = m-3$. We then continue sequentially through the path, $\overline{x_1y_2} = m-4$, $\overline{x_2y_2} = m-5$, $\overline{x_2y_3} = m-6$, $\overline{x_3y_3} = m-7$. The second cycle continues with $\overline{x_3y_4} = m-8$, $\overline{x_3y_5} = m-9$, $\overline{x_4y_4} = m-10$, $\overline{x_4y_5} = m-11$. Again, we would continue sequentially through the path and continue using this labeling method until we reach the $(\frac{r}{2})^{th}$ bead. This bead is special because, as previously mentioned, the jump in edge lengths will occur in its path. Aside from the jump though, all continues as previously described until we reach the r^{th} bead. We label the r^{th} cycle and then conclude with the r^{th} path being labeled: $\overline{x_{\frac{m}{2}-(r+2)}y_{\frac{m}{2}+1}} = 3$, $\overline{x_{\frac{m}{2}-(r+1)}y_{\frac{m}{2}+1}} = 2$, $\overline{x_{\frac{m}{2}-(r+1)}y_{\frac{m}{2}}} = 1$, $\overline{x_0y_{\frac{m}{2}}} = \frac{m}{2}$. Finally, we just have to label the vertices. We had fixed $x_0 = 0$, so by the edge lengths we obtain $x_1 = 2$, $x_2 = 3$, $x_3 = 4$, $x_4 = 6$, $x_5 = 7$, etc. Notice that we have almost sequentially increasing vertex labels, we simply skip one vertex label in the X partition every time we are within a cycle. In the Y partition we started with $y_0 = m$, and have sequentially decreasing vertex labels, with the exception of $m - 4(\frac{r}{2})$. This is because y_0 is labeled n, there are four Y partition vertices per bead, and the jump happens after the last edge label in the path from the $(\frac{r}{2})^{th}$ bead; thus the vertex label $m-4(\frac{r}{2}) = m-2r$ is skipped in the Y partition.

Generalization:

Once again, we have a more general representation of this subfamily gracefully labeled. So using the same logic as previously mentioned, we can conclude that allowing the path length to vary as long as $l \equiv 0 \pmod 4$, does not change the fact that the graph would still be gracefully labeled by the above algorithm. Therefore, we see that we now have every necklace graph in this infinite subfamily labeled.

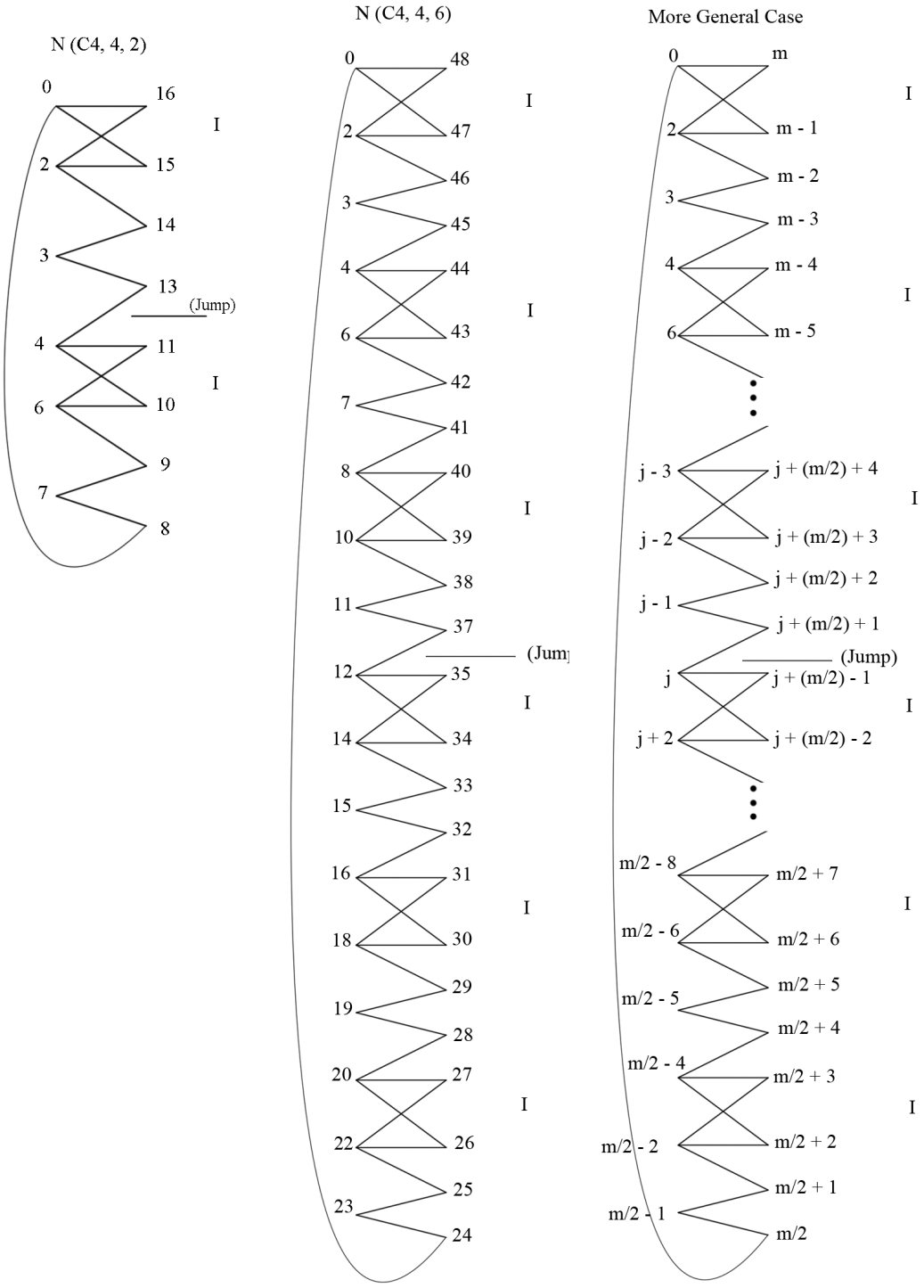

Figure 11: 3.3 Subfamily Illustrations

3.4 Subfamily: $r \equiv 3 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$, $r \equiv 3 \pmod 4$

The graphs in this subfamily, as in the other subfamilies here, will have an even number of edges. Again this is a consequence of each bead having an even number of edges, so no matter the number of beads, the graph always has an even number of edges. This is important since our clasp edge is $\frac{m-2}{2}$. Since we always have an even number of edges, this expression will always be an integer. The jump in edge lengths will be in the path between the $(\frac{r+1}{2})^{th}$ cycle and the $(\frac{r+1}{2}+1)^{th}$ cycle, just as it was for subfamily $r \equiv 1 \pmod 4$. Furthermore, we will again fix vertex labels $x_0 = 0$ and $y_0 = m$, and each four-cycle will be a Type I C_4 labeling except for the initial C_4 and the first C_4 after the path with the jump. These two four-cycles will by Type II.

Base case: $N(C_4, 4, 3)$

Starting by fixing vertex labels to the first two vertices, and calculating the clasp edge length, we can then label the first cycle: $\overline{x_0y_0} = 24$, $\overline{x_1y_0} = 23$, $\overline{x_0y_1} = 22$, $\overline{x_1y_1} = 21$. We then continue sequentially through the path, $\overline{x_1y_2} = 20$, $\overline{x_2y_2} = 19$, $\overline{x_2y_3} = 18$, $\overline{x_3y_3} = 17$. The second cycle continues with $\overline{x_3y_4} = 16$, $\overline{x_3y_5} = 15$, $\overline{x_4y_4} = 14$, $\overline{x_4y_5} = 13$. Continuing through our path we have: $\overline{x_4y_6} = 12$, we save 11 for our clasp edge and thus have $\overline{x_5y_6} = 10$, $\overline{x_5y_7} = 9$, and finish this path with $\overline{x_6y_7} = 8$. The third and final cycle is labeled: $\overline{x_6y_8} = 7$, $\overline{x_7y_8} = 6$, $\overline{x_6y_9} = 5$, $\overline{x_7y_9} = 4$. Finally, The third path is labeled with edge lengths: $\overline{x_7y_{10}} = 3$, $\overline{x_8y_{10}} = 2$, $\overline{x_8y_{11}} = 1$, and $\overline{x_0y_{11}} = 11$. Now we just have to label the vertices. We recall that we had started with x_0 being labeled 0 and y_0 labeled 24. Furthermore, as before, the X partition in strictly increasing in vertex labels, while the Y partition is strictly decreasing. Together with the edge lengths listed above, our first necklace graph in this last subfamily is labeled.

More General case: $N(C_4, 4, r)$, $r \equiv 3 \pmod 4$

The cycle is labeled $\overline{x_0y_0} = m$, $\overline{x_1y_0} = m-1$, $\overline{x_0y_1} = m-2$, $\overline{x_1y_1} = m-3$. We then continue sequentially through the path, $\overline{x_1y_2} = m-4$, $\overline{x_2y_2} = m-5$, $\overline{x_2y_3} = m-6$, $\overline{x_3y_3} = m-7$. The second cycle continues with $\overline{x_3y_4} = m-8$, $\overline{x_3y_5} = m-9$, $\overline{x_4y_4} = m-10$, $\overline{x_4y_5} = m-11$. Again, we would continue sequentially through the path and continue using this labeling method until we reach the $(\frac{r+1}{2})^{th}$ bead. This bead is special because the jump in edge lengths will occur in its path, just as in the previous subfamily. Aside from the jump though, everything continues as previously described until we reach the r^{th} bead. The r^{th} cycle is labeled as before, followed by labeling the r^{th} path with edge lengths 3, 2, 1, $\frac{m-2}{2}$. Finally, we just have to label the vertices. We had started with $x_0 = 0$, so with the edge lengths listed above we obtain: $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 5$, $x_5 = 6$, etc. Notice that we have strictly increasing vertex labels in the X partition. In the Y partition

we started with $y_0 = m$, and have strictly decreasing vertex labels.

Generalization:

Again, with this more general description we can see how allowing the path lengths to vary as long as $l \equiv 0 \pmod 4$, does not change the fact that the graph would still be gracefully labeled. Hence, by the above algorithm we can conclude that every necklace graph in this infinite subfamily is able to be gracefully labeled.

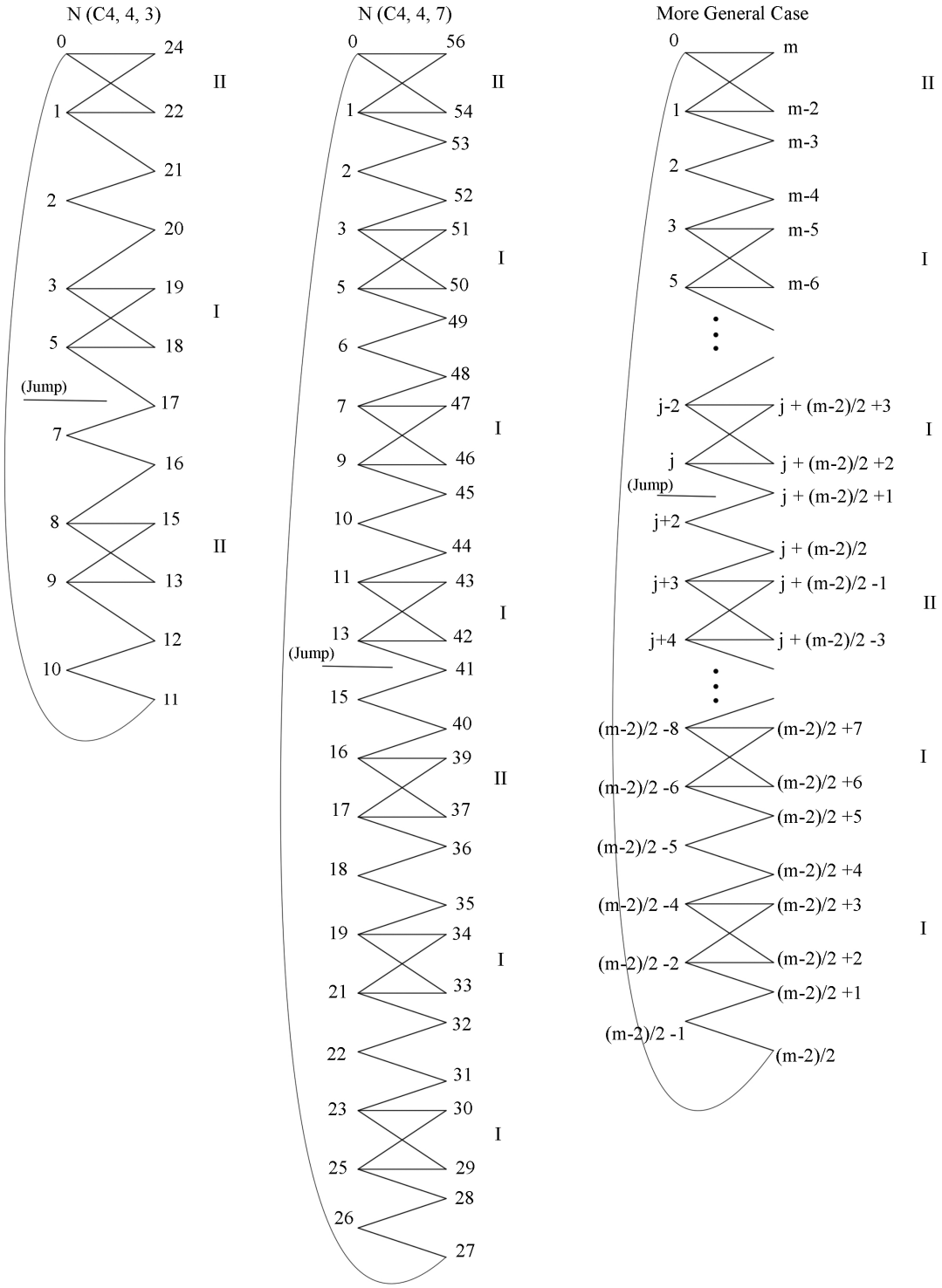

Figure 12: 3.4 Subfamily Illustrations

3.5 Summary of Family: $N(C_4, l, r)$, $l \equiv 0 \pmod 4$

From the above algorithms, we start to see certain regularities within subfamilies. For example, every necklace graph in this family with even number of beads uses only Type I C_4 labelings, with clasp edge length equal to $\frac{m}{2}$. Necklace graphs in this subfamily with an odd number of beads uses two Type II four-cycles, and the rest are Type I with a clasp edge length equal to $\frac{m-2}{2}$. Finally, from these algorithms we also have proved the following theorem.

Theorem 3.1: All necklace graphs $N(C_4, l, r)$, $l \equiv 0 \pmod 4$ are graceful.

4 Family: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$

Note, that unlike the previous family of necklace graphs, this family may not always be represented as bipartite. For example, the graph $N(C_4, 5, 5)$ will have beads with 9 edges each, so a sort of "flipping" will occur with each consecutive bead. Meaning the first bead will start out as before with the vertex labels $x_0 = 0$ and $y_0 = m$, so we will have 4 vertices in the X partition and 5 vertices in the Y partition. However, by the nature of the way we have been connecting paths and cycles, our second bead will have 5 vertices in X partition and 4 vertices in the Y partition. This flipping from one bead to the next will actually occur throughout this entire family (due to the odd path length), and is the reason why we will not have a bipartite representation for every graph; including $N(C_4, 5, 5)$. The problem with this specific necklace graph is that the flipping causes the clasp edge to be incident with x_0 and another vertex from the X partition. This will actually be the case for all necklace graphs that have an odd path length *and* an odd number of beads. Thus, we will use a $near - bipartite$ representation of the graph. A near-bipartite graph is one such that the removal on one edge causes the graph to be bipartite. In this family, it will be the removal of the clasp edge that will cause the graph to be bipartite. Hence, our graphs will all look as they did in the previous family, except in the case where we have an odd path length and an odd number of beads. In these instances, we will see that the first and last vertices of the X partition will be joined by the clasp edge.

4.1 Subfamily: $r \equiv 0 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$, $r \equiv 0 \pmod 4$

As with the last family we start by fixing the vertex labels $x_0 = 0$ and $y_0 = m$, and by calculating the clasp edge length. For this particular subfamily we will use the expression $\frac{m+2}{2}$ to find the clasp edge length. Note that the clasp edge will occur between the $(\frac{r}{2})^{th}$ cycle and the $(\frac{r}{2} + 1)^{th}$ cycle. We also will use both Type I and Type II labelings for the four-cycles. Which type of labeling we use for a given C_4 within this subfamily is determined by the alternating pattern: II, I, II, I, II,..., I (jump), I,..., II, I, II, I, II. Meaning, that once we figure out where the jump edge falls, the type of C_4 labelings used are actually mirrored across this jump in edge lengths. Since we will again treat the case where $r = 1$ as a special case, let us look at the smallest relevant case.

Small case: $N(C_4, 5, 4)$

We will first look at the graph $N(C_4, 5, 4)$, rather than the base case $N(C_4, 1, 4)$. The reason is to show a larger case that better illustrates what is happening with our algorithm; however, figures for both cases will be included.

This graph has 4 beads with 9 edges each, so we have 36 edges total ($m = 36$). So we start with labeling the vertices $x_0 = 0$ and $y_0 = m$, and then we calculate the clasp edge to have length $\frac{36+2}{2} = 19$. Now we may proceed with a similar algorithm as with the last family, the first cycle (Type II) is labeled $\overline{x_0y_0} = 36$, $\overline{x_1y_0} = 35$, $\overline{x_0y_1} = 34$, $\overline{x_1y_1} = 33$. We then continue sequentially through the path, $\overline{x_1y_2} = 32$, $\overline{x_2y_2} = 31$, $\overline{x_2y_3} = 30$, $\overline{x_3y_3} = 29$, $\overline{x_3y_4} = 28$. The second cycle (Type I, but flipped as mentioned previously) continues with $\overline{x_4y_4} = 27$, $\overline{x_5y_4} = 26$, $\overline{x_4y_5} = 25$, $\overline{x_5y_5} = 24$. We continue through the second path with $\overline{x_6y_5} = 23$, $\overline{x_6y_6} = 22$, $\overline{x_7y_6} = 21$, $\overline{x_7y_7} = 20$, and now we have reached the clasp edge length, so we have our jump and end our path with $\overline{x_8y_7} = 18$. We have now crossed our jump and so we must mirror the types of C_4 labelings used. Therfore, we label the next C_4 as a Type I (but flipped compared to the previous C_4). The labeling proceeds as follows: $\overline{x_8y_8} = 17$, $overlinex_8y_9 = 16$, $\overline{x_9y_8} = 15$, $\overline{x_9y_9} = 14$. The third path is labeled with edge lengths $\overline{x_9y_{10}} = 13$, $\overline{x_{10}y_{10}} = 12$, $\overline{x_{10}y_{11}} = 11$, $\overline{x_{11}y_{11}} = 10$, and $\overline{x_{11}y_{12}} = 9$. The fourth and final C_4 is a Type II (flipped compared to the first Type II) and labeled with edge lengths $\overline{x_{12}y_{12}} = 8$, $\overline{x_{12}y_{13}} = 7$, $\overline{x_{13}y_{12}} = 6$, and $\overline{x_{13}y_{13}} = 5$. We then conclude with the last path being labeled: $\overline{x_{14}y_{13}} = 4$, $\overline{x_{14}y_{14}} = 3$, $\overline{x_{15}y_{14}} = 2$, $\overline{x_{15}y_{15}} = 1$, and the clasp edge $\overline{x_0y_{15}} = 19$. All that is left is to label the remaining vertices now. Since we started with x_0 being labeled 0 and y_0 labeled 36, this is easily accomplished since the edge lengths will now force the values of the vertex labels. Note that as before in the other family, we will still have the vertex labels in the X partition strictly increasing, while the Y partition is strictly decreasing. Now our first necklace graph in this subfamily is labeled.

More General case: $N(C_4, 5, r)$, $r \equiv 0 \pmod 4$

Again, starting with the vertices $x_0 = 0$ and $y_0 = m$, we calculate the clasp edge to have length $\frac{m+2}{2}$. The first cycle (Type II) is then labeled $\overline{x_0y_0} = m$, $\overline{x_1y_0} = m - 1$, $\overline{x_0y_1} = m - 2$, $\overline{x_1y_1} = m - 3$. We then continue sequentially through the path, $\overline{x_1y_2} = m - 4$, $\overline{x_2y_2} = m - 5$, $\overline{x_2y_3} = m - 6$, $\overline{x_3y_3} = m - 7$, $\overline{x_3y_4} = m - 8$. The second cycle (Type I) continues with $\overline{x_4y_4} = m - 9$, $\overline{x_5y_4} = m - 10$, $\overline{x_4y_5} = m - 11$, $\overline{x_5y_5} = m - 12$. Again, we would continue sequentially through the path and continue using this labeling method, remembering the pattern of C_4 types as well as the bead flipping nature of this family, until we reach the $(\frac{r}{2})^{th}$ bead. This bead is special because, as previously mentioned, the jump in edge lengths will occur in its path. Aside from the jump though, all continues as previously described until we reach the r^{th} bead. Once again, remember that now that we have crossed the jump in edge lengths, we must mirror the type of C_4 labeling as the described pattern showed. The r^{th} cycle is labeled using a Type II C_4 labeling, followed by labeling the r^{th} path in the following manner: $\overline{x_{4r-2}y_{4r-3}} = 4$, $\overline{x_{4r-2}y_{4r-2}} = 3$, $\overline{x_{4r-1}y_{4r-2}} = 2$, $\overline{x_{4r-1}y_{4r-1}} = 1$, $\overline{x_0y_{4r-1}} = \frac{m+2}{2}$. Finally, we finish by labeling the rest of the vertices. We started with $x_0 = 0$, and from the edge lengths we will have $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, $x_4 = 4$, $x_5 = 5$, etc. Notice that we have sequentially increasing vertex labels up to the jump in edge lengths. After

the jump, only one vertex label is skipped per bead and it alternates in partition location due to the flipping nature of this family. Again, flipping of beads is due to the odd path length of this family.

Generalization:

We now have a more general representation of this subfamily gracefully labeled, where the number of beads is allowed to increase, as long as $r \equiv 0 \pmod 4$. However, notice that we can also allow the path length to vary as long as $l \equiv 1 \pmod 4$, and the graph would still be gracefully labeled. To see why this is true, consider the "more general" case above for example. If we extend the path length to 9, we have merely added two vertices to each partition, by nature of the bipartite representation being utilized by this algorithm. However, this doesn't change anything that was outlined above. We have only stretched our graph out in a very "regular" fashion. So no matter how many of multiples of 4 we add to the path length between every C_4, we still have our graph gracefully labeled by the above algorithm. Hence, we now see that we have every necklace graph in this infinite subfamily labeled by the algorithm.

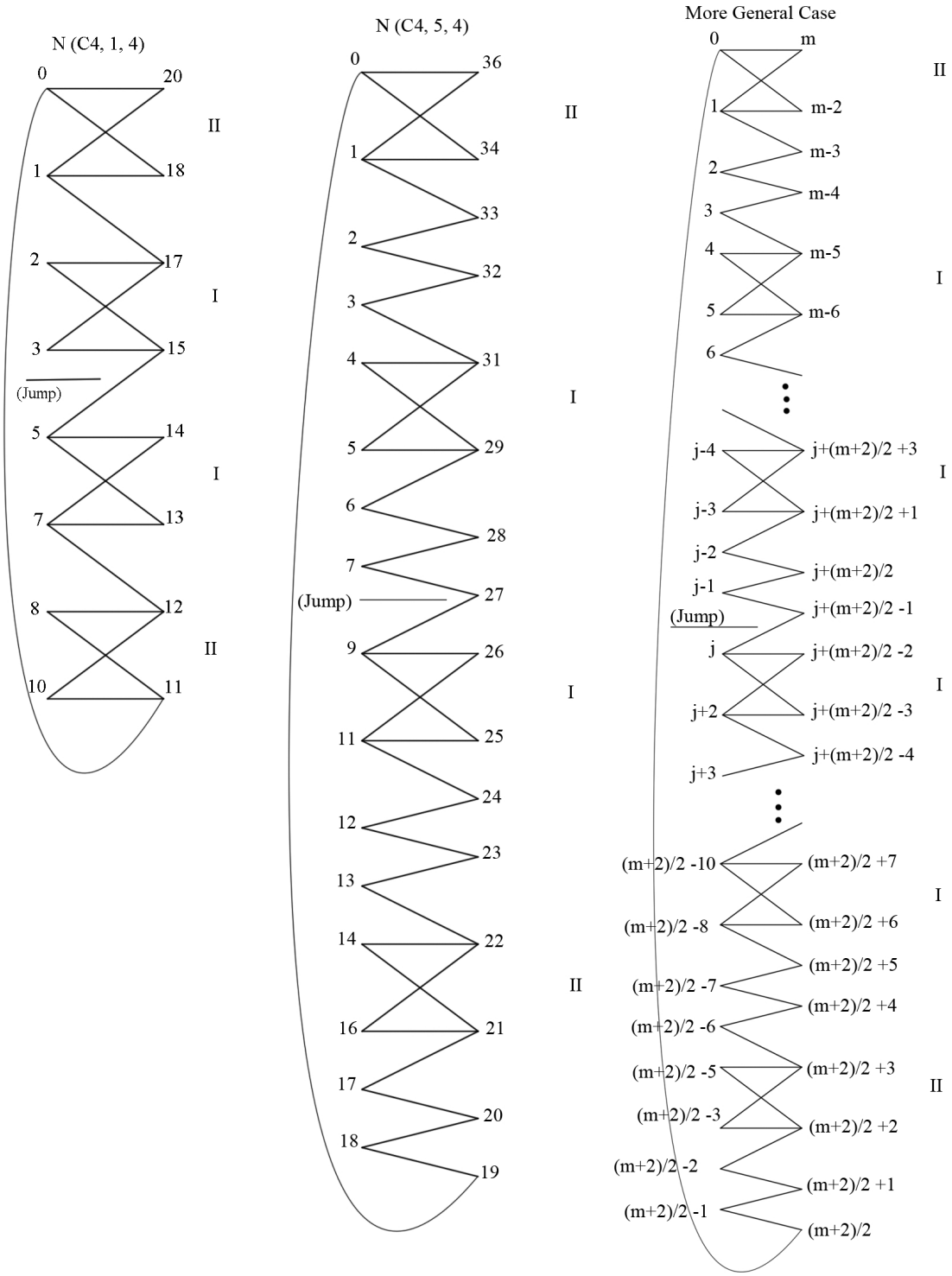

Figure 13: 4.1 Subfamily Illustrations

4.2 Subfamily: $r \equiv 1 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$, $r \equiv 1 \pmod 4$

As before, we will fix the vertex labels $x_0 = 0$ and $y_0 = m$ and start by determining the clasp edge for this infinite subfamily. This subfamily will have a clasp edge whose length is: $\frac{m-5}{2}$. Again, this expression will always yield an integer since each bead has 9 edges, and so regardless of how many beads there are, this expression will always yield an integer. Furthermore, since this family has an odd number of beads, and each bead has 9 edges, we see that we will have a jump in edge lengths that will be in the path between the $(\frac{r+1}{2})^{th}$ cycle and the $(\frac{r+1}{2}+1)^{th}$ cycle. This subfamily will also be only near-bipartite due to it always having an odd path length and an odd number of beads. Finally, we will again use both Type I and Type II labelings for the four-cycles. The pattern for this subfamily will be: II, I, I, I,..., II (jump), I,..., I, I, II. Notice that with our near-bipartite representation, only the first C_4, last C_4, and the C_4 in the bead where the jump occurs is Type II. Every other C_4 is labeled with the Type I labeling.

Small case: $N(C_4, 5, 5)$

With vertex labels $x_0 = 0$ and $y_0 = m = 45$, and clasp edge having length $\frac{45-5}{2} = 20$, we may label the first cycle (Type II) as: $\overline{x_0y_0} = 45$, $\overline{x_1y_0} = 44$, $\overline{x_0y_1} = 43$, $\overline{x_1y_1} = 42$. We then continue sequentially through the path, $\overline{x_1y_2} = 41$, $\overline{x_2y_2} = 40$, $\overline{x_2y_3} = 39$, $\overline{x_3y_3} = 38$, $\overline{x_3y_4} = 37$. The second cycle (Type I flipped) continues with $\overline{x_4y_4} = 36$, $\overline{x_5y_4} = 35$, $\overline{x_4y_5} = 34$, $\overline{x_5y_5} = 33$. We continue through the second path with $\overline{x_6y_5} = 32$, $\overline{x_6y_6} = 31$, $\overline{x_7y_6} = 30$, $\overline{x_7y_7} = 29$, $\overline{x_8y_7} = 28$. The third cycle (Type II since the jump will occur in this bead's path) is labled with $\overline{x_8y_8} = 27$, $\overline{x_9y_8} = 26$, $\overline{x_8y_9} = 25$, $\overline{x_9y_9} = 24$. Continuing into the path we have $\overline{x_9y_{10}} = 23$, $\overline{x_{10}y_{10}} = 22$, $\overline{x_{10}y_{11}} = 21$, and now we have reached our jump, so we skip edge length 20 and continue on to $\overline{x_{11}y_{11}} = 19$, and finally $\overline{x_{11}y_{12}} = 18$. Our next C_4 (Type I flipped) is labeled with edge lengths $\overline{x_{12}y_{12}} = 17$, $\overline{x_{13}y_{12}} = 16$, $\overline{x_{13}y_{12}} = 15$, and $\overline{x_{13}y_{13}} = 14$. Continuing into the fourth path, $\overline{x_{14}y_{13}} = 13$, $\overline{x_{14}y_{14}} = 12$, $\overline{x_{15}y_{14}} = 11$, $\overline{x_{15}y_{15}} = 10$, $\overline{x_{16}y_{15}} = 9$. The fifth and final C_4 (Type II) is labeled as: $\overline{x_{16}y_{16}} = 8$, $\overline{x_{17}y_{16}} = 7$, $\overline{x_{16}y_{17}} = 6$, and $\overline{x_{17}y_{17}} = 5$. Finally, we reach our last path and label it with $\overline{x_{17}y_{18}} = 4$, $\overline{x_{18}y_{18}} = 3$, $\overline{x_{18}y_{19}} = 2$, $\overline{x_{19}y_{19}} = 1$, and finally the clasp edge $\overline{x_0y_{19}} = 20$. The last step, as before, is to label the remaining vertices with their forced values. Our base case necklace graph in this subfamily is labeled.

More General case: $N(C_4, 5, r)$, $r \equiv 1 \pmod 4$

Like always, we fix $x_0 = 0$ and $y_0 = 0$ and calculate the clasp edge length $\frac{m-5}{2}$. The first cycle (Type II) is then labeled $\overline{x_0y_0} = m$, $\overline{x_1y_0} = m-1$, $\overline{x_0y_1} = m-2$, $\overline{x_1y_1} = m-3$. We then continue sequentially through the path, $\overline{x_1y_2} = m-4$, $\overline{x_2y_2} = m-5$, $\overline{x_2y_3} = m-6$, $\overline{x_3y_3} = m-7$, $\overline{x_3y_4} = m-8$. The second cycle (Type I) continues with $\overline{x_3y_4} = m-9$, $\overline{x_3y_5} = m-10$, $\overline{x_4y_4} = m-11$, $\overline{x_4y_5}$

$= m - 12$. Again, we would continue sequentially through the path and continue using this labeling method until we reach the $(\frac{r+1}{2})^{th}$ bead. This bead is special because the jump in edge lengths will occur in its path, just as in the previous subfamily. Aside from the jump though, everything continues as previously described until we reach the r^{th} bead. The r^{th} cycle is labeled with a Type II C_4, followed by the r^{th} path being labeled: $\overline{x_{4r-2}y_{4r-1}} = 4$, $\overline{x_{4r-1}y_{4r-1}} = 3$, $\overline{x_{4r-1}y_{4r}} = 2$, $\overline{x_{4r}y_{4r}} = 1$, $\overline{x_0x_{4r}} = \frac{m}{2}$. Finally, we just have to label the remaining vertices, whose values are forced by the first two vertex labels and the labeled edges.

Generalization:

As before, we now have a more general representation of this subfamily gracefully labeled, where the number of beads is allowed to increase, as long as $r \equiv 1 \pmod 4$. So using the same logic as previously mentioned, we can conclude that allowing the path length to vary as long as $l \equiv 1 \pmod 4$, does not change the fact that the graph would still be gracefully labeled by the above algorithm. Thus, we now see that we have every necklace graph in this infinite subfamily labeled.

Special Case: $N(C_4, 5, 1)$

We first fix $x_0 = 0$ and $y_0 = m$, and then determine the clasp edge, which will have an edge length of 4. The cycle (Type II) is labeled $\overline{x_0y_0} = 9$, $\overline{x_1y_0} = 8$, $\overline{x_0y_1} = 7$, $\overline{x_1y_1} = 6$, and then we continue sequentially through the path, skipping the edge length 4 since that is saved for our clasp edge. So our path is labeled: $\overline{x_1y_2} = 5$, $\overline{x_2y_2} = 3$, $\overline{x_2y_3} = 2$, $\overline{x_3y_3} = 1$, and $\overline{x_0x_3} = 4$. Last, we go back and label the other vertices: $x_1 = 1$, $x_2 = 3$, $x_3 = 4$, $y_1 = 7$, $y_2 = 6$, and $y_3 = 5$. Note that this graph is not bipartite, but rather it is near-bipartite. By removing our clasp edge of $\overline{x_0y_3}$ we would obtain a bipartite graph.

Furthermore, this really only takes care of the case where we have one bead and a path of length 5, even though the smallest cases to consider in this subfamily would be the graphs with path length $l = 1$. Therefore, the graphs of $N(C_4, 1, 1)$ and $N(C_4, 1, 5)$ will also be included. However, following the logic as we did to extend from the "more general case" to a completely generalized case, we can see that by adding multiples of 4 edges to the path length, the graphs in this subfamily will remain graceful. Thus, any necklace graph $N(C_4, l, r)$ where $l \equiv 1 \pmod 4$ and $r \equiv 1 \pmod 4$, will also be graceful.

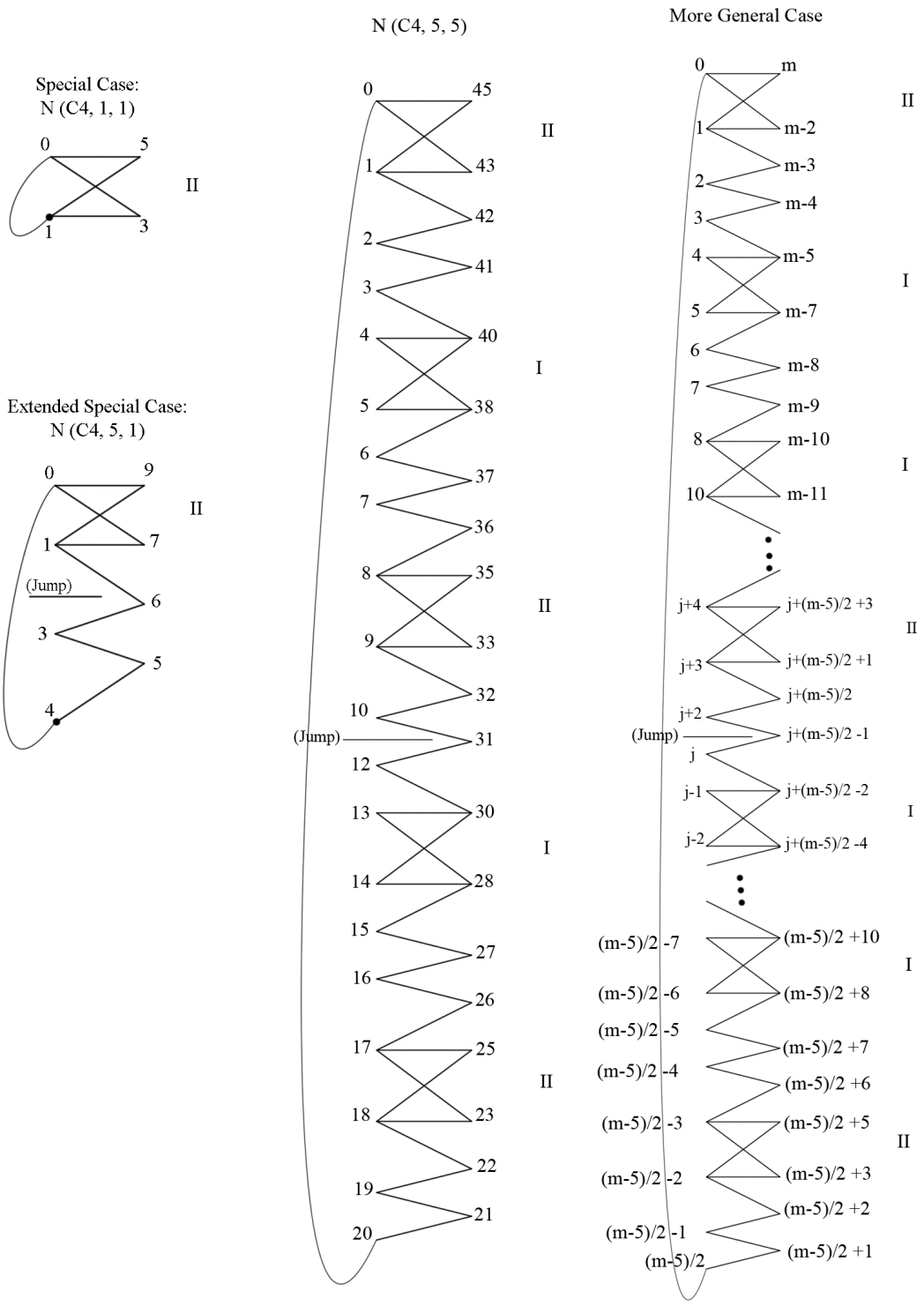

Figure 14: 4.2 Subfamily Illustrations

4.3 Subfamily: $r \equiv 2 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$, $r \equiv 2 \pmod 4$

We start to notice some patterns within this family when we continue labeling in the fashion mention in the last two sections. Actually, we start to see a lot of regularities in the algorithms used for each of the past six subfamilies.

Let us summarize what we have been doing to reduce repetition of ideas. The first step when labeling a necklace graph is first to count how long the path between four-cycles is mod 4. That gives us the family. Next we count how many beads we have mod 4, which gives us the subfamily. Once in the subfamily we have a expression that gives us the length of the clasp edge, since we are using bipartite, or near-bipartite, representations of these necklace graphs. Once we have have the clasp edge length, we fix the vertex values $x_0 = 0$ and $y_0 = m$. This forces $\overline{x_0 y_m} = m$, and recall that we always have strictly decreasing edge lengths as we continue through our bipartite (or near-bipartite) representations. Further, we see that unlike the first family, not all families utilize only one type of method to label the four-cycles. That is, we can have a mixture of Type I and Type II C_4 labelings. Hence, the third step after assigning the first two vertex labels and the clasp edge length, is to see what pattern to use in labeling the four-cycles. After this step, every edge length will now be forced by the type of C_4 labeling for which the algorithm requires of each bead. Finally, the remaining vertex labels will then be forced once all the edges are labeled. The reason why these expressions in the described algorithms work is based on the regularity within each bead, and the defined way in which we draw each bead. Hence, some beads may be flipped back and forth depending on the length of the path in each bead. If the path length is odd, the beads will flip back and forth consecutively. That is, the first bead will have the path incident to the C_4 in the X (left) partition, while the second bead will have the path incident to the C_4 in the Y (right) partition, etc. It is only by all of this mentioned structure and regularity of the graph that this algorithm works. With all that being said, we will now more concisely visit the remaining subfamilies.

For this subfamily, where the path length is congruent to 1 (mod 4) and the number of beads is congruent to 2 (mod 4), we again start by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m}{2}$. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is: II, I, II, I, II, I,..., II (jump), I, I, II, I,..., II, I, II. What is happening here is that we start with a Type II C_4 and alternate until we reach the bead that will have the clasp edge in its path (i.e. the $(\frac{r}{2})^{th}$ bead). After this bead, we have two Type I four-cycles and then continue alternating. Remember, the path length in this family is odd, so when we are drawing the near-bipartite representations of this subfamily, beads will be flipping as well. We now have every-

thing in place to force the non-clasp edge lengths. We start with $\overline{x_0 y_0} = m$, and continue with the edge lengths strictly decreasing. Once the edges are finished being labeled, we label the remaining vertices. Recall, the X partition vertex labels are strictly increasing, while the Y partition vertex labels are strictly decreasing. We are now done with labeling this infinite subfamily.

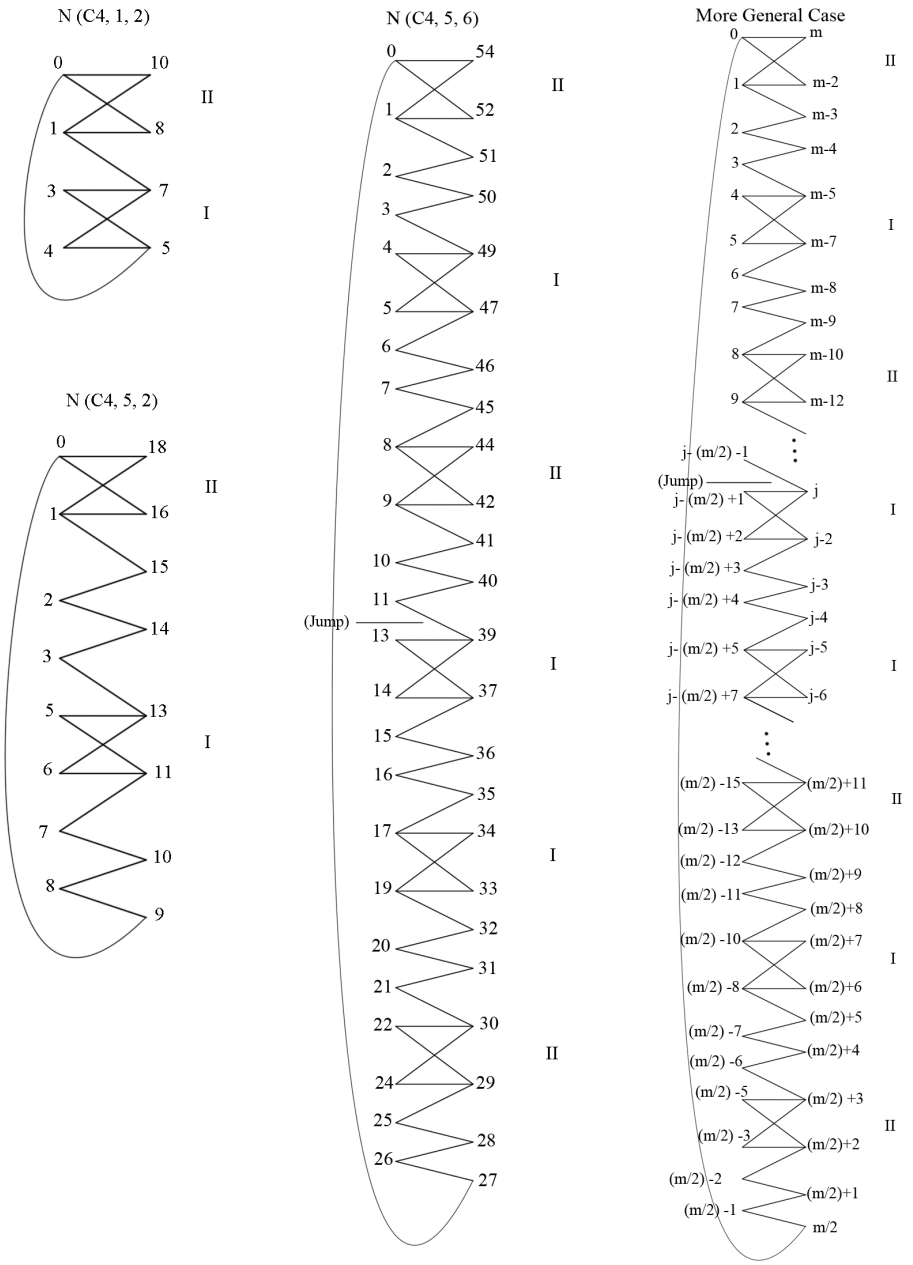

Figure 15: 4.3 Subfamily Illustrations

4.4 Subfamily: $r \equiv 3 \pmod 4$

Cases for $r > 1$: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$, $r \equiv 3 \pmod 4$

For this subfamily, where the path length is congruent to 1 (mod 4) and the number of beads is congruent to 2 (mod 4), we again start by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m-3}{2}$. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is: II, I, I, I, I,..., II (jump), II, I, I,..., I, II. What is happening here is that we use Type II C_4 labelings like bookends. That is we start with a Type II, and then only use Type I labelings until we reach the bead that contains the jump. This bead then uses a Type II C_4 and the bead that immediately proceeds the jump bead uses a Type II C_4. We then continue with Type I labelings until we reach the r^{th} bead, which uses a Type II C_4. Again, since the path length in this family is odd, we will have beads consecutively flipping. We now have everything in place to force the non-clasp edge lengths. We start with $\overline{x_0 y_0} = m$, and continue with the edge lengths strictly decreasing. Once the edges are finished being labeled, we label the remaining vertices. Recall, the X partition vertex labels are strictly increasing, while the Y partition vertex labels are strictly decreasing. We are now done with labeling this infinite subfamily.

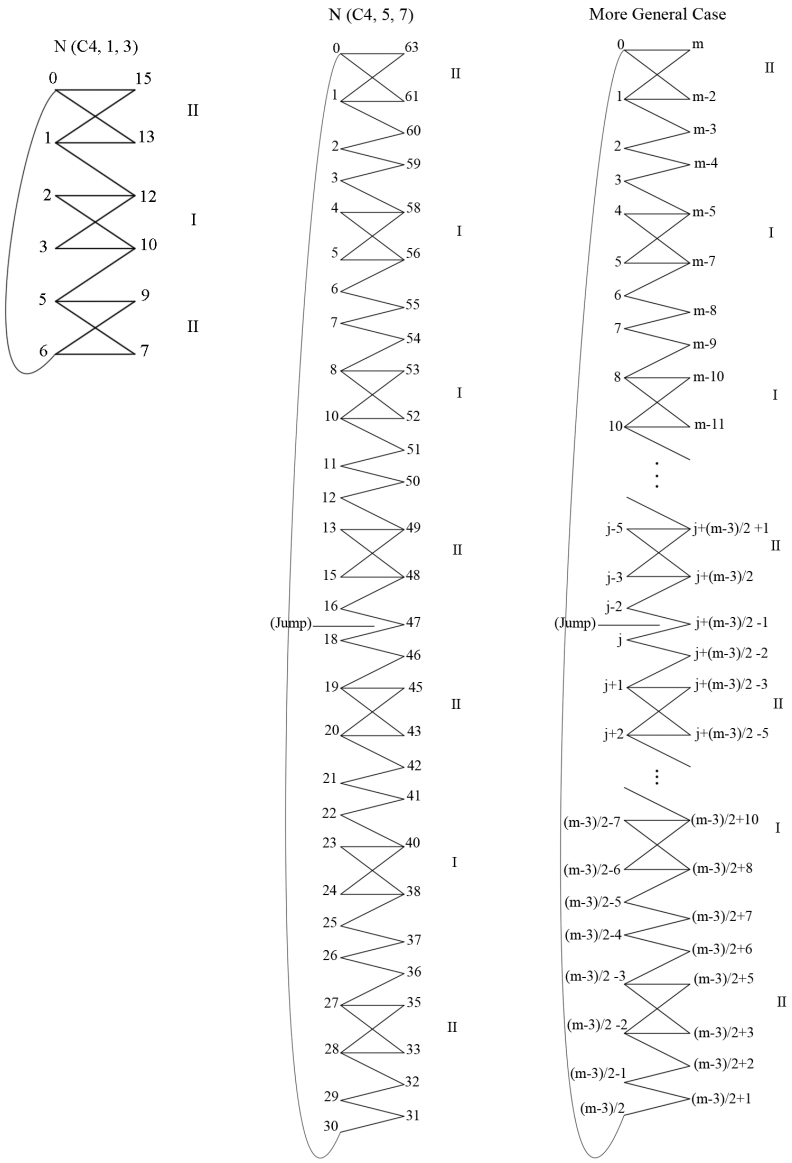

Figure 16: 4.4 Subfamily Illustrations

4.5 Summary of Family: $N(C_4, l, r)$, $l \equiv 1 \pmod 4$

From the above algorithms, we notice something unique to this family compared to the previous family. Each subfamily within this family has a distinct expression for the clasp edge length, as well as a distinct pattern for which type of C_4 labeling to use. In any case though, we have now proved the following theorem.

Theorem 4.1: All necklace graphs $N(C_4, l, r)$, $l \equiv 1 \pmod 4$ are graceful.

5 Family: $N(C_4, l, r)$ $l \equiv 2 \pmod 4$

This family of necklace graphs will always have an even number of edges per bead, hence we will not have to concern ourselves with any kind of flipping of beads, nor using a near-bipartite representation. That being said, this family may always be represented as bipartite as so will be described as such.

5.1 Subfamily: $r \equiv 0 \pmod 4$

In this subfamily the path length is congruent to 2 (mod 4) and the number of beads is congruent to 0 (mod 4), but we still begin by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m}{2}$. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is very regular and simple in that, as with the first family, we only use Type I C_4 labelings in each bead. What is happening here is that each bead has an equal number of vertices in both partitions. As investigated previously with the Type I C_4 labeling, the Y partition vertex labels will be consecutively, strictly decreasing until we reach the edge length that will be saved for the clasp edge. This will occur in the path of the $\frac{r}{2}^{th}$ bead. At this point, there will be a jump in vertex labels, but then they will continue on consecutively decreasing in value. Meanwhile in the X partition, the vertex labels will be strictly increasing and skipping a value in each C_4. Once the first two vertices have their vertex labels set, which thereby forces the first edge length, and we have set the clasp edge length, we are ready to label the remaining edges. Again, we keep in mind that the edge lengths will be consecutively decreasing until the jump, and then they will continue onward. Once the edges are labeled, we must only label the remaining vertices whose labels are now forced as before. It is all by the regularity of this structure that we can see that this will always work, and so we are now done labeling this infinite subfamily.

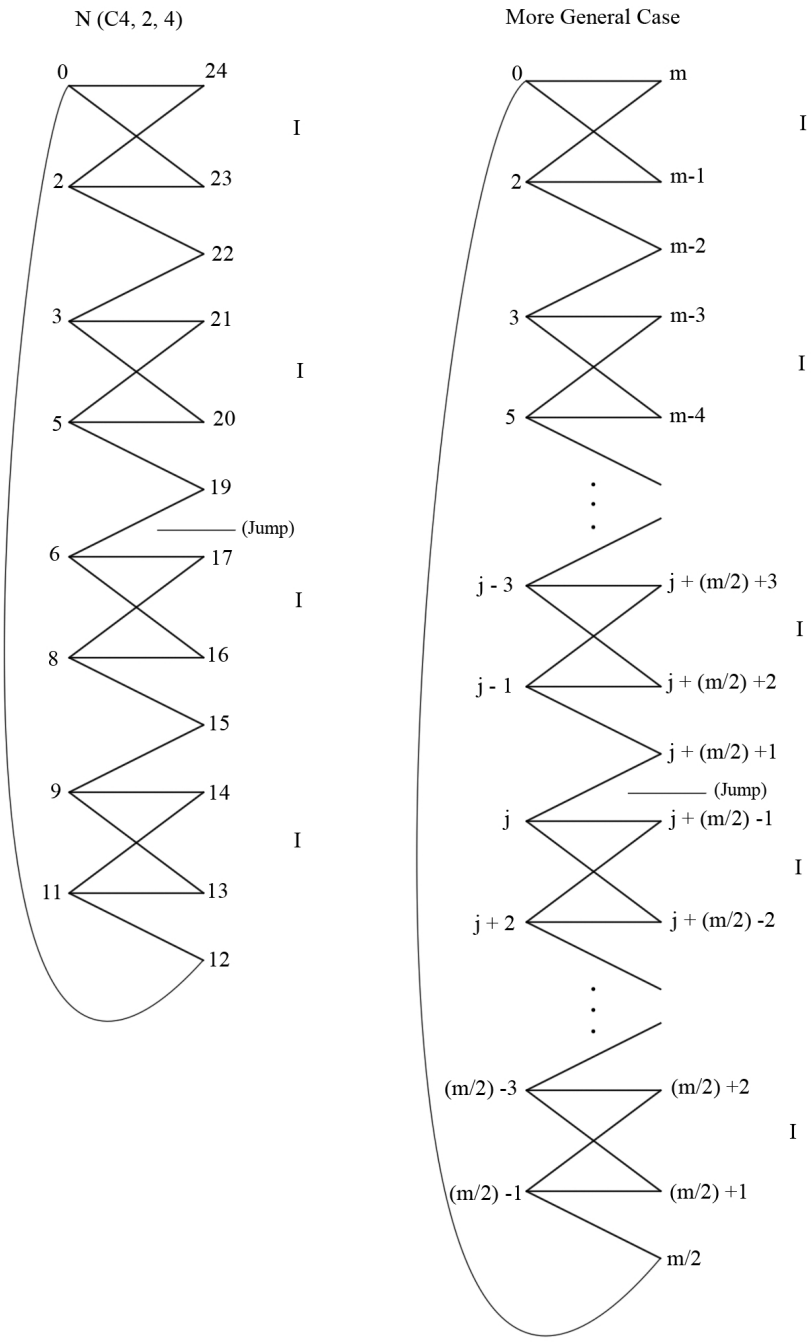

Figure 17: 5.1 Subfamily Illustrations

5.2 Subfamily: $r \equiv 1 \pmod 4$

This subfamily has all of the necklace graphs whose number of beads is congruent to 1 (mod 4). As always we begin by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m-2}{2}$. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is also very regular and simple in that the only difference from the last subfamily is that we start with a Type II C_4, and then continue on only using Type I C_4 labelings in each of the following beads. Here, each bead still has an equal number of vertices in both partitions; however, now we have an odd number of beads. The significance of this is that even though every member of this family of graphs will always have an even number of edges, when we have an odd number of beads the number of edges is not divisible by four. To see this, we can think of prime factorization. Every bead in this family will be divisible by only two if there is an odd number of beads since each bead is congruent to 2 (mod 4), and two multiplied by an odd number is divisible by two and never four. However, if we have an even number of beads, we pick up another factor of two and so we will have a number of edges that is divisible by four. So because of the number of edges, and consequently the number of vertices we will have to label, we start with a Type II C_4 since the jump in vertex labels will start in the Y partition, and then move to just the X partition as before. Meaning, after the first C_4 we will have consecutively decreasing vertex labels in the Y partition up to, and after, the jump. We will also have strictly increasing vertex labels in the X partition, and consecutively decreasing edge lengths as we proceed through our bipartite graph. Therefore, once the first two vertices have their vertex labels set as $x_0 = 0$ and $y_0 = m$, the corresponding edge $\overline{x_0 y_0}$ has length m. We have also already determined the clasp edge length, and so we are ready to label the remaining edges. Again, we keep in mind that the edge lengths will be consecutively decreasing until the jump, and then they will continue onward. The jump in this subfamily will occur in the path of the $\frac{r+1}{2}^{th}$ bead. Once the edges are labeled, we must only label the remaining vertices whose labels are now forced as before. Thus, even though this graph is not quite as regular as the previous subfamily, the regularity that does exist verifies that this algorithm will always work, and so we are now done labeling this infinite subfamily.

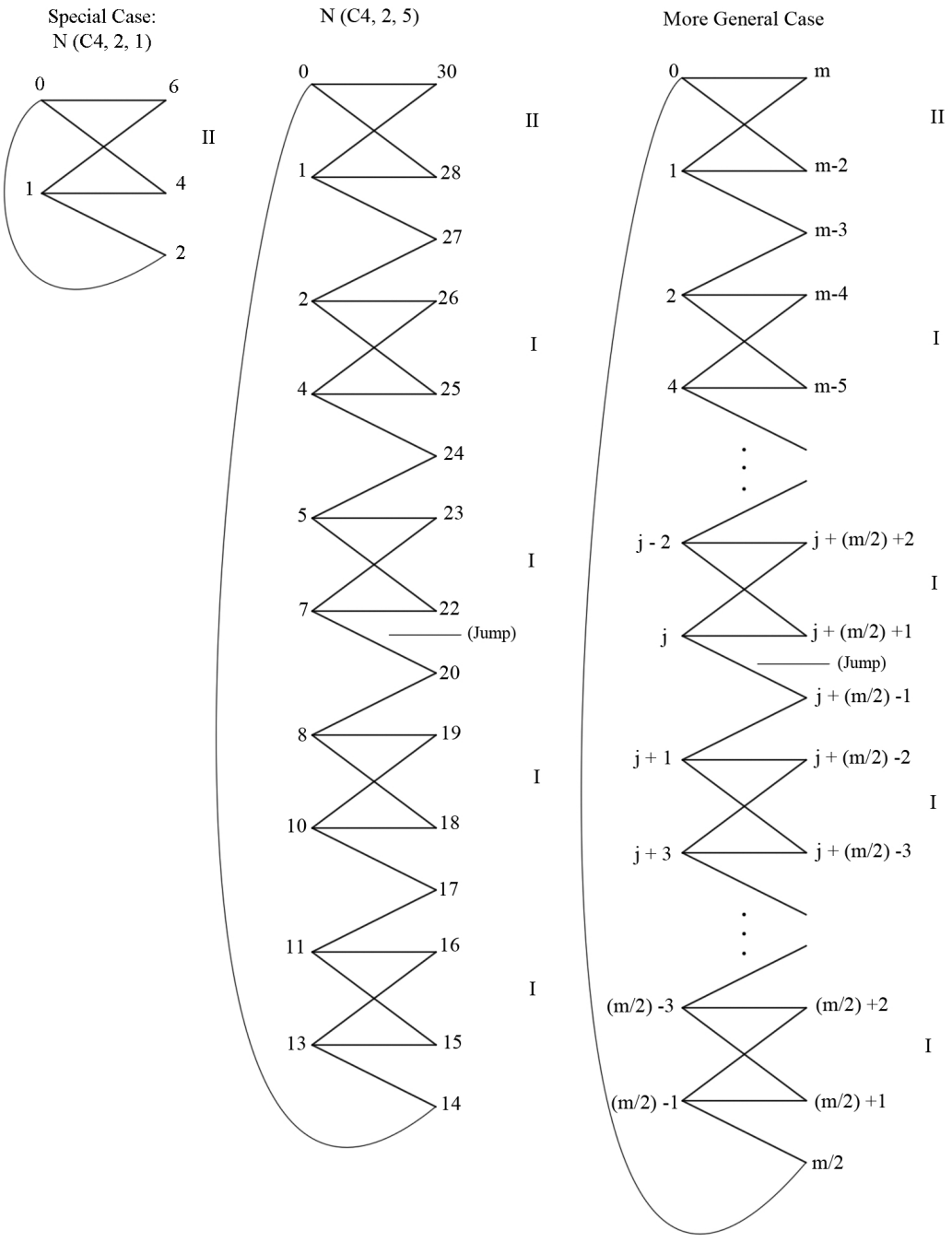

Figure 18: 5.2 Subfamily Illustrations

5.3 Subfamily: $r \equiv 2 \pmod 4$

In this subfamily we indeed have an even number of beads, with an even number of edges each, and so we will have a total number of edges that is divisible by four. As always, we begin by labeling $x_0 = 0$ and $y_0 = m$, and we will once again have the clasp edge length be equal to $\frac{m}{2}$. In this subfamily, the edge length that is saved will cause a jump in thepath of the $\frac{r}{2}^{th}$ bead. The third step of labeling is determining the types of C_4 labelings to use, which is fairly simple now. We again will only use Type I C_4 labelings in each bead. This is again decided based on the regularity of the structure, the fact that since we have m divisible by four, and because Type I C_4 labelings will have vertex labels being skipped only in the X partition. All of these facts come together like puzzle pieces to allow this subfamily of graphs to be gracefully labeled. Remember that by starting with the first two vertices of each partition being labeled as mentioned, by defining the clasp edge length, and through labeling the types of C_4 labelings to use, we are forcing the remaining edge lengths, and ultimately the remaining vertex labels. That is, observing the same patterns before in edge lengths and vertex labels decreasing and increasing accordingly, along with remembering about the jump in the graph labels due to the clasp edge, we are now done labeling this infinite subfamily.

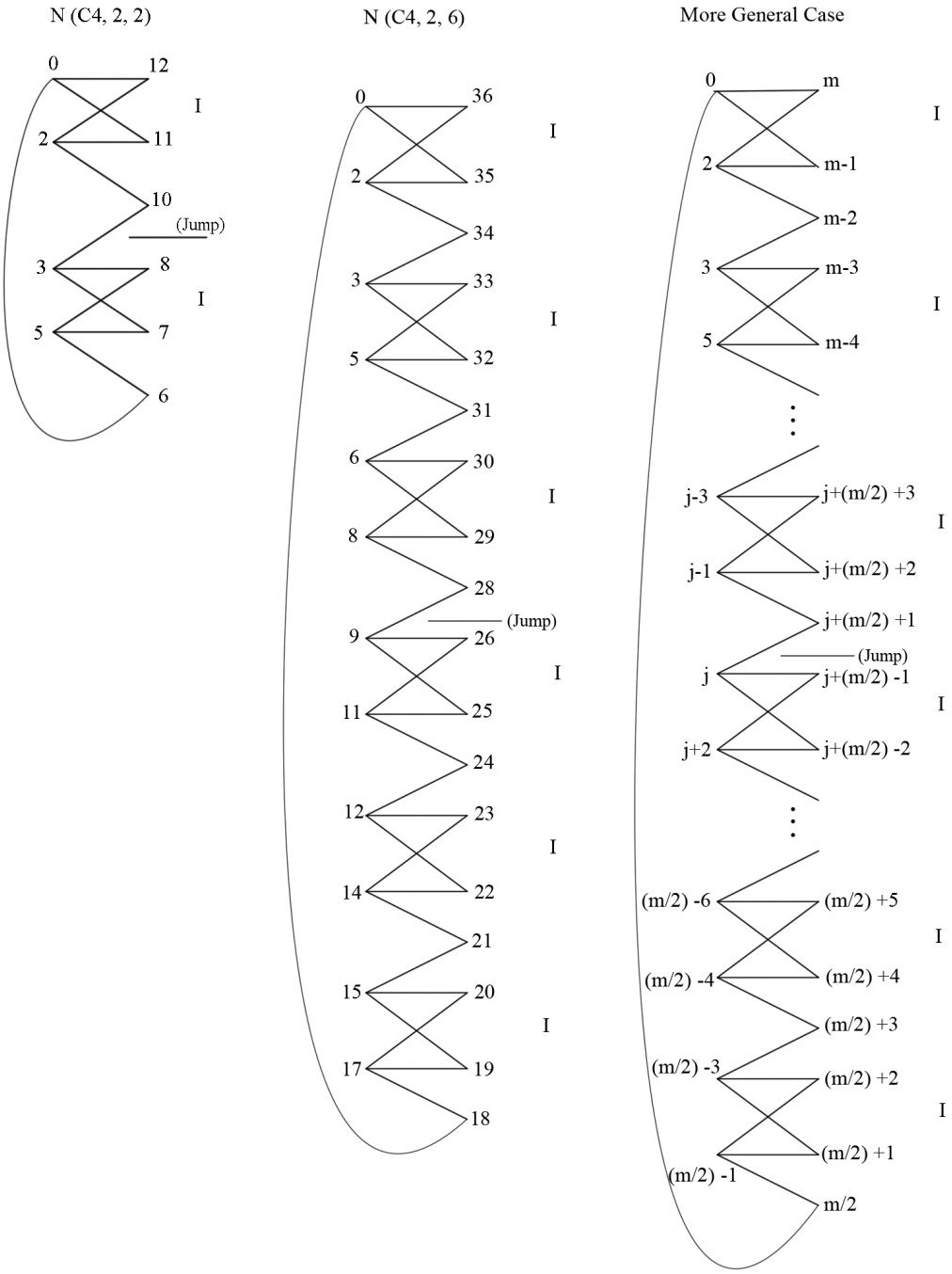

Figure 19: 5.3 Subfamily Illustrations

5.4 Subfamily: $r \equiv 3 \pmod 4$

As with the subfamily mentioned in 5.2, this subfamily will have an odd number of beads, with an even number of edges each. That is, we will not have divisibility by four, only by two. In any case, we start by labeling $x_0 = 0$ and $y_0 = m$, and we will have the clasp edge length be equal to $\frac{m-2}{2}$. The third step of labeling is to label the types of C_4 labelings to use. This subfamily will start off with a Type II C_4 labeling, followed by only Type I C_4 labelings in each of the remaining beads. It should be no real surprise that this subfamily is strikingly similar to the subfamily in 5.2, since both subfamilies have an odd number of beads with each bead being congruent to 2 (mod 4). So using the same logic and reasoning as we did in 5.2, we see that the Y partition vertex labels will still be consecutively decreasing after the initial C_4 until we reach the edge length that is saved for the clasp edge. As in the $r \equiv 1 \pmod 4$ subfamily, the edge length that is saved for the clasp edge will cause a jump in the vertex labels within the path of the $\frac{r+1}{2}^{th}$ bead. However, after we reach this jump, the vertex labels will continue on consecutively decreasing in value in the Y partition. Meanwhile in the X partition, the vertex labels will be strictly increasing and skipping a value in each C_4, aside from the first C_4. Once the first two vertices have their vertex labels set, which forces the first edge length, and we have set the clasp edge length, we are ready to label the remaining edges. Again, we keep in mind that the edge lengths will be consecutively decreasing until the jump, and then they will continue onward. Once the edges are labeled, we must only label the remaining vertices whose labels are now forced as before. We are now done labeling this infinite subfamily.

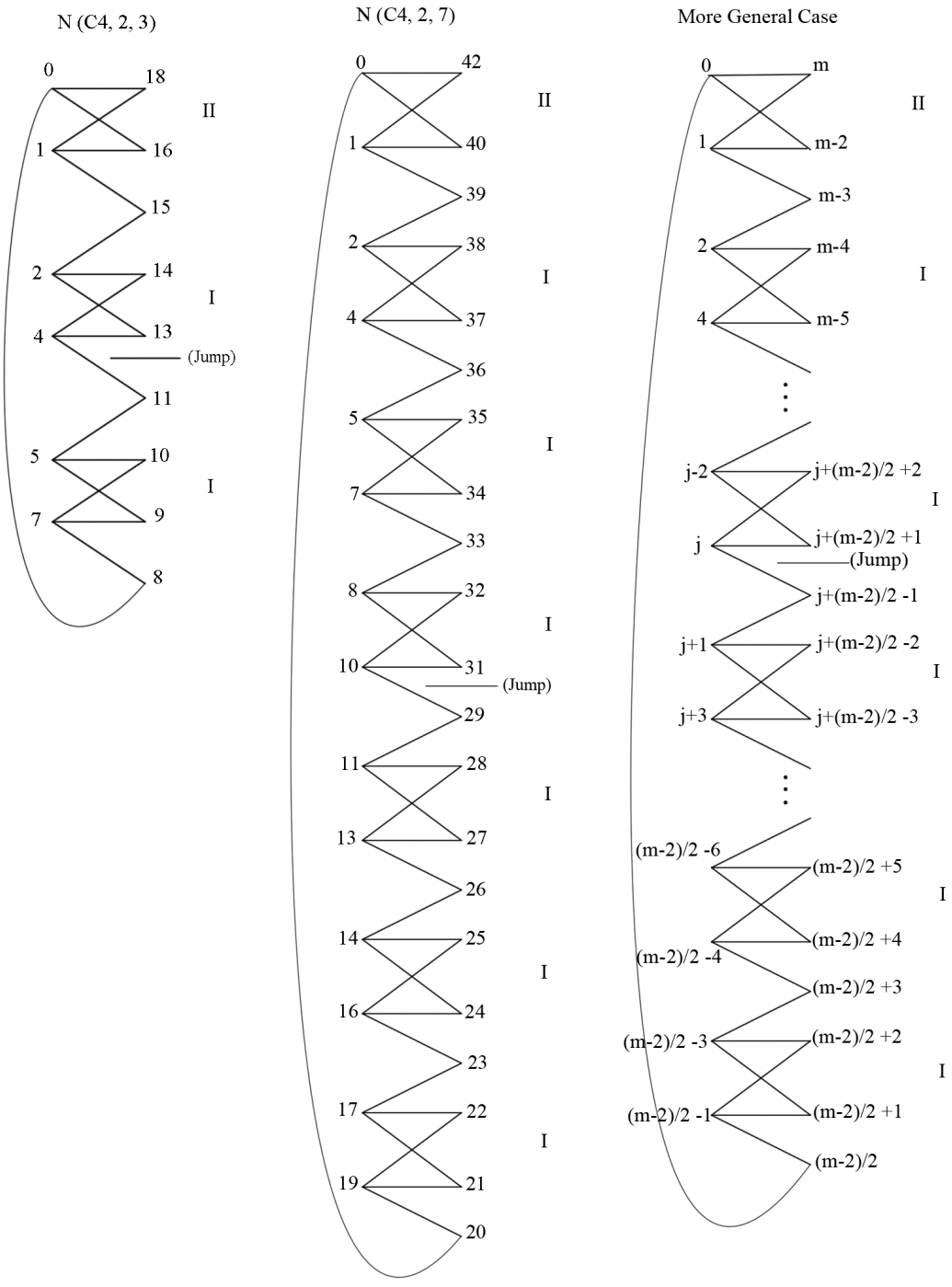

Figure 20: 5.4 Subfamily Illustrations

5.5 Summary of Family: $N(C_4, l, r)$, $l \equiv 2 \pmod 4$

These subfamilies were interesting in that we really had two pairs of subfamilies. One where we had an even number of beads, and one with an odd number of beads. Both subfamilies with even number of beads had the same expression for determining the clasp edge length. They both also only utilized Type I C_4 labelings in every bead. The two subfamilies with odd number of beads also acted alike. They too had the same expression for determining the clasp edge length: $\frac{m-2}{2}$. Similarly, they also both used the same pattern for determining which type of C_4 labeling to use. That is, both "odd bead" subfamilies here started with a Type II C_4 labeling, and then only used Type I C_4 labelings from that point onward. Looking at these algorithms though, and why they always work, we have now proved the following theorem.

Theorem 5.1: All necklace graphs $N(C_4, l, r)$, $l \equiv 2 \pmod 4$ are graceful.

6 Family: $N(C_4, l, r)$ $l \equiv 3 \pmod 4$

This family of necklace graphs will always have an odd number of edges per bead, hence we will once again have to pay attention to the flipping nature of the beads due to the odd path length in each bead. Furthermore, as with the family in Chapter 4, we will be using a near-bipartite representation of our necklace graphs when we have an odd number of beads, where upon removing the clasp edge, our graph is bipartite. That is, the clasp edge is incident to the first and last vertices of the X partition due to how we have constructed our necklace graphs. When we in fact have an even number of beads, our graph will be bipartite simply by construction.

6.1 Subfamily: $r \equiv 0 \pmod 4$

In this subfamily the path length is congruent to 3 (mod 4) and the number of beads is congruent to 0 (mod 4), so we will have a bipartite graph. As always, we start by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m+2}{2}$. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is actually the same as in the one used for the subfamily in 4.1. The pattern used here is the alternating pattern: II, I, II, I, II,..., I (jump), I,..., II, I, II, I, II. Meaning, that once we figure out where the jump edge falls, the type of C_4 labelings used are actually mirrored across this jump in edge lengths. The reason we want an alternating pattern here is that we have an unequal number of vertices in the two partitions within each bead. We also have an odd path length within each bead, so the flipping of beads will occur again just as previously described in Chapter 4. So, by using this alternating pattern, we are able to account for these two actions and gracefully label our necklace graphs with the above clasp edge length. We still have strictly decreasing edge lengths as we move down our bipartite graph; along with strictly increasing vertex labels in the X partition and strictly decreasing vertex labels in the Y partition. Furthermore, by the way we have constructed this subfamily, using alternating types combined with the flipping nature of the beads, we will once again have consecutively increasing vertex labels up to the jump; which will occur in the path of the $\frac{r}{2}^{th}$ bead. Thus, since we had the first two vertices set, thereby forcing one edge length, and we know the clasp edge length, using everything outlined above we can then label the remaining edge lengths and so force the remaining vertex labels. This subfamily has wound up to be very similar to the subfamily in 4.1, and so using similar logic we have now gracefully labeled this infinite subfamily.

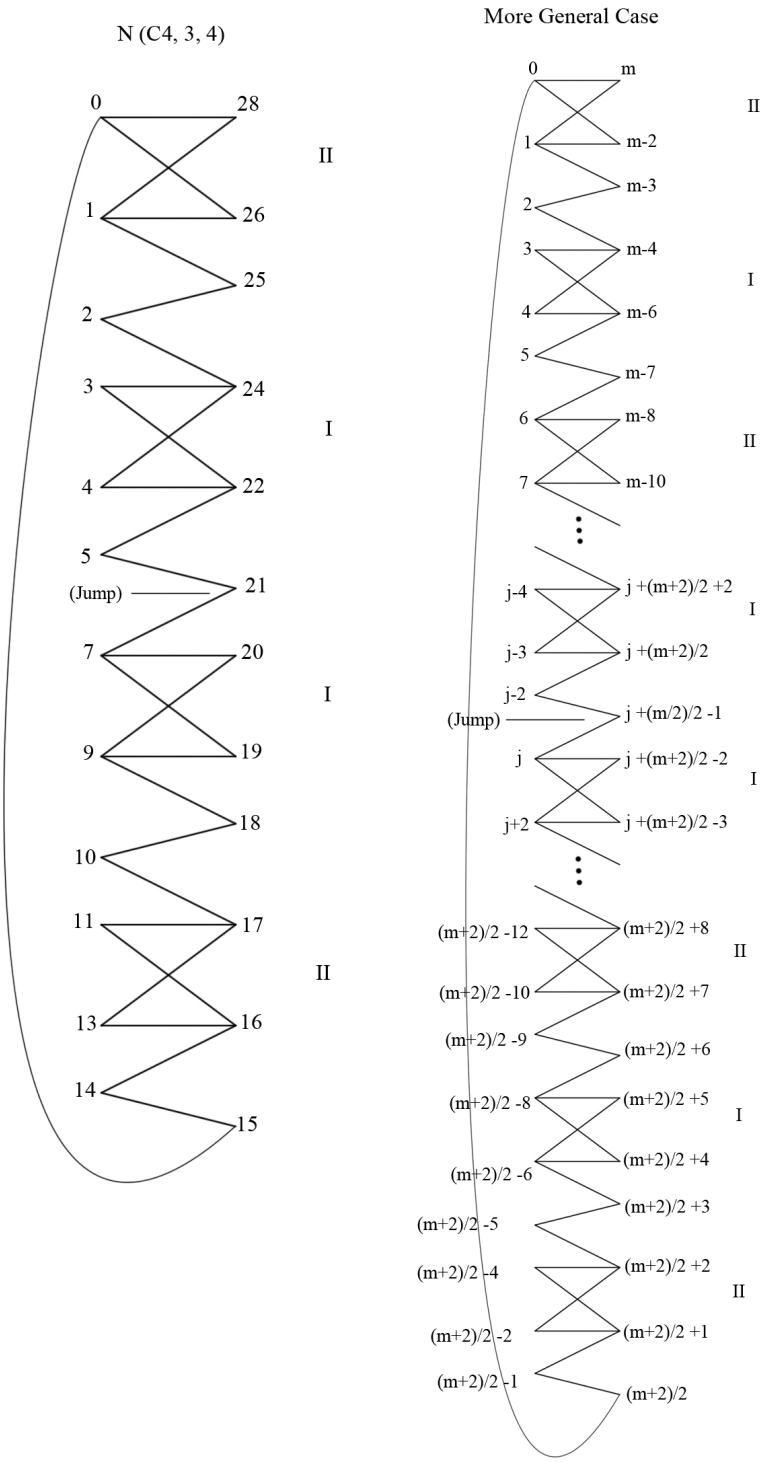

Figure 21: 6.1 Subfamily Illustrations

6.2 Subfamily: $r \equiv 1 \pmod 4$

This subfamily has all of the necklace graphs whose number of beads is congruent to 1 (mod 4); hence an odd number of beads so we will be using a near-bipartite representation. As always we begin by labeling $x_0 = 0$ and $y_0 = m$. The clasp edge length will be equal to $\frac{m-5}{2}$, causing the jump to occur in the $\frac{r+1}{2}^{th}$ bead. The third step is labeling the types of C_4 labelings we will use. The pattern for this subfamily is: II, I, I, I,..., II (jump), I,..., I, I, I. Notice that with our near-bipartite representation, only the first C_4 and the C_4 in the bead where the jump occurs is Type II. Every other C_4 is labeled with the Type I labeling. It is also interesting to note again the similarities between this subfamily and its Chapter 4 counterpart; that is, the subfamily from 4.2. The only difference here is that in 4.2 the last C_4 was also a Type II, whereas here it is still a Type I C_4. In any case, now that we have our initial labelings done we proceed as before. We label the remaining edge lengths, which are now forced by the types of four-cycles we have determined, and which consecutively decrease as we move down our near-bipartite graph. Followed by labeling the remaining vertex labels, which are forced by the determined edge lengths. As we have come to expect, the Y partition vertex labels are strictly decreasing and the X partition vertex labels are strictly increasing. Furthermore, as was the case in 4.2, we actually consecutively increasing vertex labels up to the jump, but not afterwards due to the types of four-cycles used and the flipping nature of the beads. With all this mind, we have now labeled yet another infinite subfamily of C_4 necklace graphs.

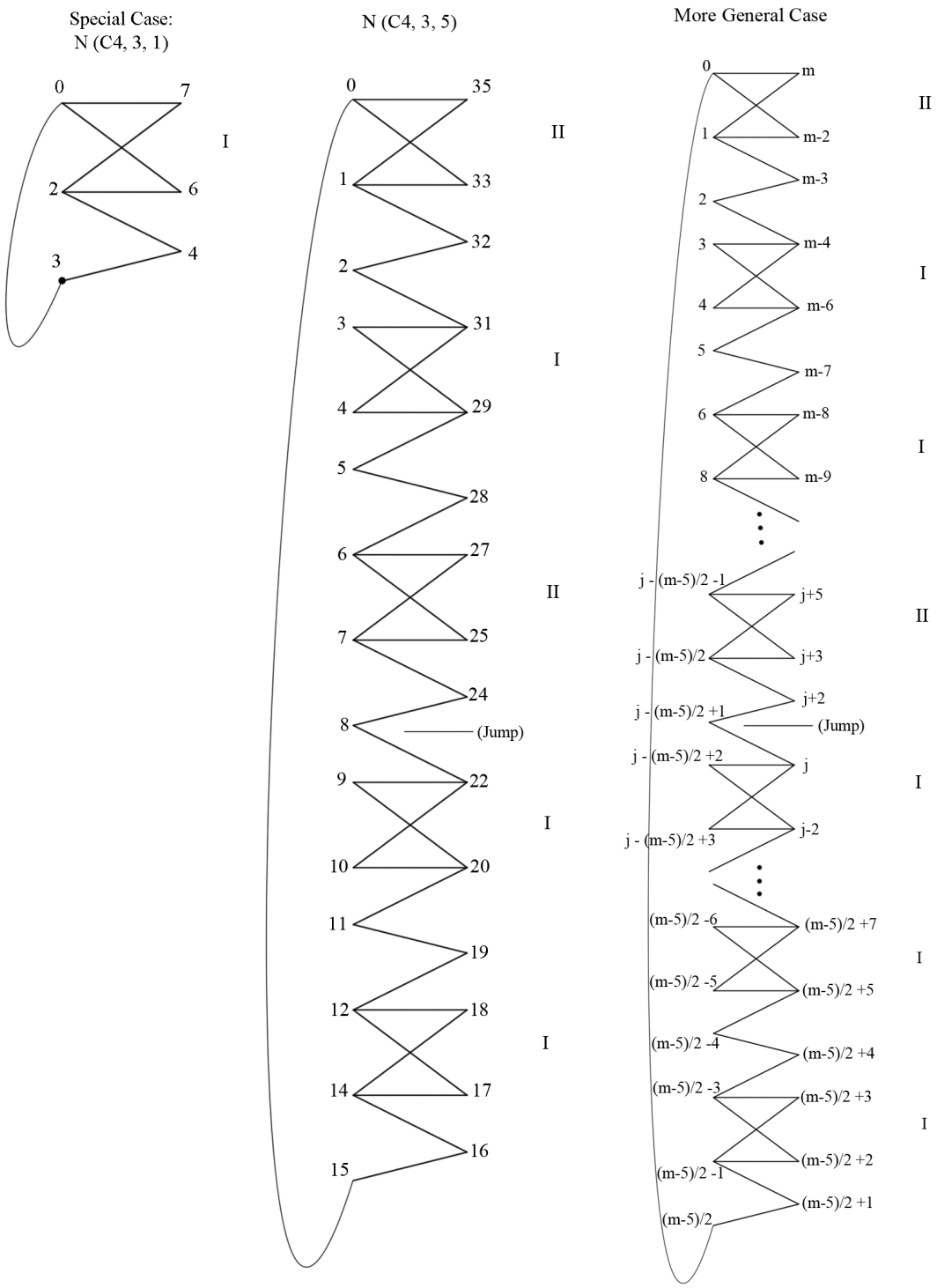

Figure 22: 6.2 Subfamily Illustrations

6.3 Subfamily: $r \equiv 2 \pmod 4$

In this subfamily we have an even number of beads, with an odd number of edges each, and so we will have a bipartite graph. We begin by labeling $x_0 = 0$ and $y_0 = m$, and the clasp edge length will be equal to $\frac{m}{2}$. The next step of labeling is determining the types of C_4 labelings to use. For this subfamily we follow the pattern: II, I, II, I, II, I,..., II (jump), I, I, II, I,..., II, I, II. What is happening here is that we start with a Type II C_4 and alternate until we reach the bead that will have the clasp edge in its path (that is, the $(\frac{r}{2})^{th}$ bead). After this bead, we have two Type I four-cycles and then continue alternating. It is important to note that, as with other subfamilies with odd path lengths, we will be drawing the near-bipartite representations of our graphs where the beads will be flipping. Furthermore, notice that this subfamily is also very much related to the subfamily in 4.3; both subfamilies use not only the same expression for the clasp edge, but they also use the same pattern for determining which type of four-cycle to use in each bead. We also see the same increasing/decreasing patterns arising with the edge lengths and vertex labels as in the previous subfamilies. Therefore, using our developed method at hand and the same logic as previous subfamilies, we now have yet one more infinite subfamily gracefully labeled.

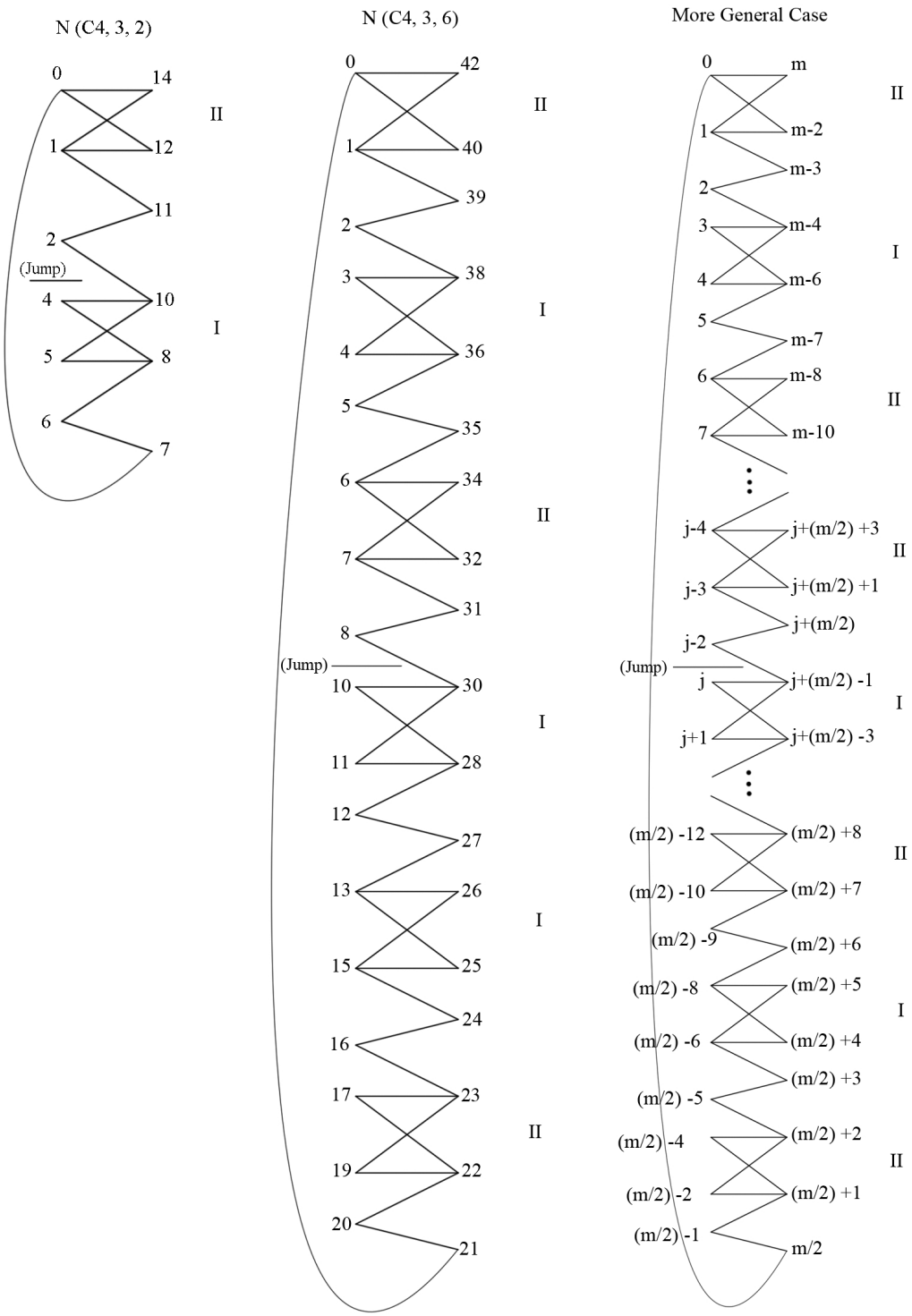

Figure 23: 6.3 Subfamily Illustrations

6.4 Subfamily: $r \equiv 3 \pmod 4$

With our last subfamily, we see that there is an odd number of beads, with an odd number of edges each. That is, we will yet again have a near-bipartite representation to work with. In any case, we start by labeling $x_0 = 0$ and $y_0 = m$ one more time, and calculate our clasp edge length equal to $\frac{m-3}{2}$. The next step in labeling our necklace graph is to label the types of four-cycles that we will use in each bead. The pattern for this subfamily is: II, I, I, I, I,..., II (jump), I,..., I, I, I, II. What is happening here is that we start with a Type II C_4 labeling,and use only Type I until we reach the $\frac{r+1}{2}^{th}$ bead whose path contains the jump, for which we use another Type II. After that bead, we continue using only Type I again until we reach the r^{th} bead, for which we use another Type II. Notice that this subfamily is also much like its Chapter 4 counterpart, that is the subfamily from 4.4. Both these of these two subfamilies use the same expression for calculating the clasp edge, and the patterns for labeling the types of four-cycles is also similar. The only difference in the four-cycle patterns is that in this subfamily when we reach the jump, it is only the bead that contains the jump that uses a Type II C_4; whereas, in the subfamily in 4.4 we also used a Type II C_4 for the bead directly after the bead containing the jump. This should not be surprising if we think about the reasoning. The difference between the 4.4 subfamily and this subfamily is that this subfamily has beads with paths length 3 (mod 4) instead of 1 (mod 4). Hence, for any necklace graph that belongs to the 4.4 subfamily, we need only add one vertex in each partition in each path to arrive at the corresponding necklace graph for this family. Therefore, by switching the type of four-cycle used immediately proceeding the bead with the jump in it, we change what partition jumps a vertex label at that point and wind up essentially reindexing the partitions from that point onward. This is necessary since our paths are two edges longer in this subfamily than the corresponding graphs from 4.4, and so due to the flipping nature of odd length path graphs we must adjust our vertex labels so that our clasp edge fits correctly with the appropriate vertex label. Thus, using similar reasoning and logic as before, we have gracefully labeled our last infinite subfamily.

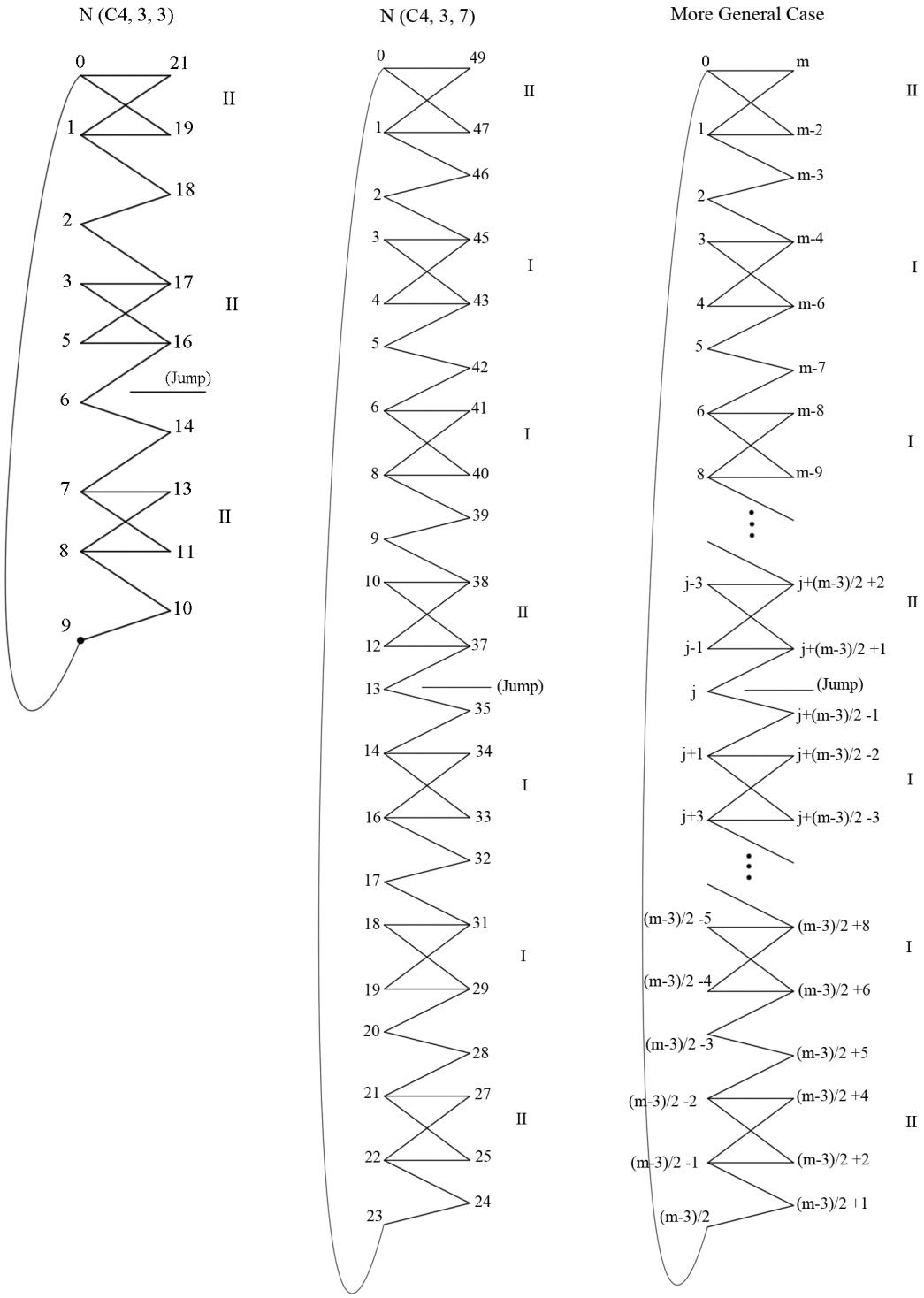

Figure 24: 6.4 Subfamily Illustrations

6.5 Summary of Family: $N(C_4, l, r)$, $l \equiv 2 \pmod 4$

These subfamilies were particularly interesting since they were very similar to the corresponding subfamilies with the same number of beads mod 4 from Chapter 4. Our subfamilies whose number of blocks were congruent to 0 or 2 (mod 4) actually worked exactly in the same way as those in Chapter 4. The "odd" subfamilies almost worked the same as their Chapter 4 counterparts, except for the type of four-cycle used in the bead immediately after the jump. In any case, from the above algorithms we have proved yet one more theorem.

Theorem 6.1: All necklace graphs $N(C_4, l, r)$, $l \equiv 3 \pmod 4$ are graceful.

7 Results

From the Theorems 3.1, 4.1, 5.1, and 6.1 we have now proved our next theorem.

Theorem 7.1: All necklace graphs $N(C_4, l, r)$ are graceful.

So now that we know that all C_4 necklace graphs may be gracefully labeled, let us summarize the algorithm that always allows us to gracefully label these graphs.

Step 1: Draw the necklace graph as a bipartite, or near-bipartite, graph as outlined earlier.
Step 2: Label the first two vertices in each partition. That is, label $x_0 = 0$ and $y_0 = m$, and then also the resulting edge length $\overline{x_0 y_0} = m$.
Step 3: Calculate the clasp edge length using the appropriate expression determined by the subfamily, and label this edge.
Step 4: Label the the four-cycles in each bead using the appropriate pattern determined by the subfamily.
Step 5: Label all remaining edge lengths that are now forced the by the indicated C_4 type in each bead, remembering that the edge lengths will be strictly decreasing as we move down our (near-) bipartite graph.
Step 6: Label all remaining vertices, whose labels are now forced by the edge lengths. Remember that the X partition vertex labels will be strictly increasing, while the Y partition vertex labels will be strictly decreasing.

Note: This algorithm merely provides a means to gracefully label $N(C_4, l, r)$ necklace graphs with a rather high level of regularity. There are multiple ways to label some necklace graphs, as well as entire subfamilies. For example, in our last subfamily where $l \equiv 3 \pmod 4$ and $r \equiv 3 \pmod 4$, it can also be shown that we could have also used only Type II C_4 labelings after the jump occurs in order to gracefully label this infinite subfamily.

The following tables outline the appropriate expressions and type patterns for each given subfamily.

Expressions for Clasp Edge Length

Path Length (l) (mod 4)

# of Beads (r) (mod 4)	0	1	2	3
0	$\frac{m}{2}$	$\frac{m+2}{2}$	$\frac{m}{2}$	$\frac{m+2}{2}$
1	$\frac{m-2}{2}$	$\frac{m-5}{2}$	$\frac{m-2}{2}$	$\frac{m-5}{2}$
2	$\frac{m}{2}$	$\frac{m}{2}$	$\frac{m}{2}$	$\frac{m}{2}$
3	$\frac{m-2}{2}$	$\frac{m-3}{2}$	$\frac{m-2}{2}$	$\frac{m-3}{2}$

Patterns for Labeling Four-Cycles

Path Length (l) (mod 4)

# of Beads (r) (mod 4)	0	1	2	3
0	I, I, …, I (jump), I, …, I, I	II, I, II, I, II,…, I (jump), I,…, II, I, II, I, II	I, I, …, I (jump), I, …, I, I	II, I, II, I, II,…, I (jump), I,…, II, I, II, I, II
1	II, I, …, I (jump), I, …, I, II	II, I, I, I,…, II (jump), I,…, I, I, II	II, I, …, I (jump), I, …, I, I	II, I, I, I,…, II (jump), I,…, I, I, I
2	I, I, …, I (jump), I, …, I, I	II, I, II, I, II, I,…, II (jump), I, I, II, I,…, II, I, II	I, I, …, I (jump), I, …, I, I	II, I, II, I, II, I,…, II (jump), I, I, II, I,…, II, I, II
3	II, I, …, I (jump), II, …, I, I	II, I, I, I, I,…, II (jump), II,…, I, I, I II	II, I, …, I (jump), I, …, I, I	II, I, I, I, I,…, II (jump), I,…, I, I, I, II

Figure 25: Algorithm Information Summary

Thinking back to Chapter 1 when we discussed various types of graph labelings, along with how we constructed and labeled all of our necklace graphs we notice something even more useful than a graceful labeling. Notice that in every subfamily, if we call the value of one less than the clasp edge length to be k, then it so happens that our graceful labeling f has the property that $f(x) \leq k$ for every $x \in X$ and $f(y) > k$ for every $y \in Y$. However, by definition this would be mean that for every subfamily that can be represented as bipartite, our graphs would actually admit an α-labeling by use of this algorithm. Subfamilies whose graphs may only be represented as near-bipartite we could therefore say are $near - alpha$. That is, by removing one edge our graph would admit an α-labeling. Thus we now have one more useful theorem.

Theorem 7.2: All necklace graphs $N(C_4, l, r)$ that do not have an odd path length in each bead and an odd total number of beads, admit an α-labeling.

Other interesting things one may notice when looking at the table summaries is that there is definitely some regularity to both the clasp edge expressions and the C_4 Type patterns. For example, path lengths seem to work similarly based on whether the path length in each bead is even or odd. We see that for even path length necklace graphs, the clasp edge length is either equal to $\frac{m}{2}$ or $\frac{m-2}{2}$, depending on whether we have an even or odd number of beads. We also notice that for odd path lengths, the clasp edge only depends on how many beads we have in our graph. That is, if we happen to have 5 beads in our graph, it does not matter how long our odd path length is in each bead, the clasp edge length is still $\frac{m-5}{2}$.

We also notice certain regularities within our C_4 Type patterns. In fact, we see the exact same relationships among subfamilies when looking at these patterns as well. That is, we see this separation of families with even path lengths acting similarly, as well as families with odd path lengths acting similarly. The only exception is when we have an odd number of beads. In this case, we have one bead that uses a different type of four-cycle than the corresponding bead in the "related" subfamily. For example, consider the subfamily where both path length and number of beads are congruent to 1 (mod 4). We would say that the related subfamily would be the one where $l \equiv 3 \pmod 4$ and $r \equiv 1 \pmod 4$. Notice that the only difference is in the pattern for C_4 types; the last bead is the only bead that must change type depending on the subfamily. Hence, we see once more that due to the highly structured nature of our C_4 necklace graphs, and all of the regularity within, we are able to see relationships and regularities among our subfamilies as well. From these patterns we are able to consider possible extensions on this work.

8 Further Work

We started this project knowing that all cycles that are of length congruent to 0 or 3 (mod 4) are graceful. Therefore, when looking at necklace graphs (a cyclic structure) the most natural subfamily to look at was the subfamily from 3.1, the subfamily whose path lengths are congruent to 0 (mod 4) and whose number of beads is congruent to 0 (mod 4). Sure enough this turned out to be a very regular and easy to label subfamily. With this in mind, we decided to try extending cycle lengths by using other cycles that have length congruent to 0 (mod 4). We thought that the natural place to start with this endeavor would be looking at $N(C_8, l, r)$ necklace graphs.

When dealing with $N(C_8, l, r)$ necklace graphs, we attempted to gracefully label them using the same techniques outlined in this paper. The difficulty that arose with this is that there are more than two ways to gracefully label an eight-cycle. Furthermore, since necklace graphs are constructed such that opposite vertices of a cycle are incident with the paths, $N(C_8, l, r)$ necklace graphs tend to look a little different. The biggest problem is that when looking at our paths, they no longer are incident to the "first" and "last" vertex in a given partition. This causes a little more difficulty in jumping edge lengths not only within beads, but in accounting for the clasp edge. Therefore, when we use more than two beads the problem is that we do not have enough vertex labels when jumping edge lengths. We were able to gracefully label $N(C_8, l, r)$ necklace graphs when $r = 2$ and $l \equiv 0 \pmod 4$, by choosing our clasp edge to be $\frac{m}{2}$ and using only one type of graceful bipartite representation of C_8. Namely, the representation of C_8 when the wrap-around edge length is four less than the edge length of $\overline{x_0 y_0}$. Therefore, we propose that one would be able to semi- or quasi-gracefully label $N(C_8, l, r)$ necklace graphs when $r > 2$. We propose that this is a great place to begin doing further work with necklace graphs; that is, trying to find an algorithm that allows one to semi- or quasi-gracefully label all C_8 necklace graphs. From here, we would expect that extensions into other cycle lengths would then come naturally.

References

[1] Bloom, Gary S., Golomb, Solomon W., *II Applications to Coding Theory*,Applications of Numbered Undirected Graphs, Proceedings of the IEEE, Vol. 65, No. 4, pp. 564.

[2] Bloom, Gary S., Golomb, Solomon W., *III B. Communication Network Labeling*,Applications of Numbered Undirected Graphs, Proceedings of the IEEE, Vol. 65, No. 4, pp. 567.

[3] Clark, J., Holton, D.A., *Chapter 1 - An Introduction to Graphs*, A First Look at Graph Theory, World Scientific, Singapore, 1991, pp. 2.

[4] El-Zanati, Saad I., Eynden, Charles Vanden, *3.1 ρ-Labelings*, On Rosa-Type Labelings and Cyclic Graph Decompositions, Illinois State University, Normal, IL, pp. 4.

[5] El-Zanati, Saad I., Eynden, Charles Vanden, *3.2 σ-Labelings*, On Rosa-Type Labelings and Cyclic Graph Decompositions, Illinois State University, Normal, IL, pp. 5-6.

[6] El-Zanati, Saad I., Eynden, Charles Vanden, *3.3 β-Labelings*, On Rosa-Type Labelings and Cyclic Graph Decompositions, Illinois State University, Normal, IL, pp. 6-8.

[7] El-Zanati, Saad I., Eynden, Charles Vanden, *3.4 α-Labelings*, On Rosa-Type Labelings and Cyclic Graph Decompositions, Illinois State University, Normal, IL, pp. 8.

[8] El-Zanati, Saad I., Eynden, Charles Vanden, *4.1 Ordered Labelings*, On Rosa-Type Labelings and Cyclic Graph Decompositions, Illinois State University, Normal, IL, pp. 9-10.

[9] Gallian, Joseph A., *1 Introduction*, A Dynamic Survey of Graph Labeling, Univeristy of Minnesota Duluth, Duluth, MN, pp. 6.

[10] Gallian, Joseph A., *2.2 Cycle-Related Graphs*, A Dynamic Survey of Graph Labeling, Univeristy of Minnesota Duluth, Duluth, MN, pp. 9.

[11] Truszczynski, Miroslaw, *Graceful Unicyclic Graphs*, Demonstratio Mathematica, Vol. 17, Number 2, pp. 382.