ON VARIATIONS OF DIFFUSION

by

Todd Mullen

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
March 2020

To my beautiful fiancée Bekah and her incredible patience!

Table of Contents

List of Tables

List of Figures

Abstract

This thesis will examine the Chip Firing variant, diffusion, in many of its different iterations. We will look at a previously studied version, Parallel Diffusion [9], along with four new variants: Quantum Parallel Diffusion, Two-One Diffusion, Pay it Backward, and Sequential Diffusion. The results discussed will center around the topics of regularity and periodicity.

Chip-firing processes move chips from vertex to vertex in a graph at discrete time increments. A specific distribution of chips on the vertices of a graph at a specific time is referred to as a *configuration*. We will show that most of these processes exhibit periodic behaviour, meaning that configurations recur as time increases. Also, in the instance of Pay it Backward, we see that not every variation guarantees periodic behaviour. It is with Pay it Backward, however, that we discover some of our most interesting results regarding the regularity of the movement of chips.

Of those variations which exhibit periodic behaviour, we will show examples of very large periods and very short periods, and we will also count the number of periodic configurations that exist in some processes. Specifically, we use Long and Narayanan's result [14] that every instance of Parallel Diffusion is eventually periodic with period 1 or 2 to determine the number of non-equivalent configurations that exist on paths, complete graphs, and star graphs.

List of Abbreviations and Symbols Used

$-x \sim y$	Edge label dictating that the negative stack size of x will compared to the stack size of y
$A \setminus B$	Set of all elements that are in A but not in B
$A \subseteq B$	A is a subset of B
A_n	Number of period configurations that exist on alternating arrow orientations of paths with n vertices
$C + k$	Configuration created by adding an integer k to every stack size in C
C_k	Configuration at time k of the $Seq(C)$
C_n	Cycle graph with n vertices
$E(G)$	Edge set of a graph
F_n	The number of p_2-configurations that exist on P_n
K_n	Complete graph with n vertices
$K_{1,n}$	Star graph with $n+1$ vertices
$N(v)$	Open neighbourhood a vertex v
P_n^{2-1}	P_n with the $2-1$ configuration
P_n	Path graph with n vertices
$QQ(G)$	Quantum quiescent number G
$QQ_2(G)$	2-quantum quiescent number G
$S(X)$	Set of all h-strips in board-pile polyomino X
S_k	k^{th} h-strip from the bottom in a board-pile polyomino
$Seq(C)$	Configuration sequence initiated by C
$V(G)$	Vertex set of a graph
Z_n	Number of 0-preorientations R on P_n such that R is not the fixed orientation
$Z_+^C(v)$	Set of all vertices in C which are richer than v
$Z_-^C(v)$	Set of all vertices in C which are poorer than v
$\overline{\boldsymbol{Seq(C_0)}}$	Singleton or ordered pair of configurations in $Seq(C_0)$
$\overline{L}(G)$	Set of all periods that exist on G
d_i	Difference between the greatest x-coordinate in S_i and the least x-coordinate in S_{i-1}.

$deg_t^+(v)$	Out-degree of v at time t
$m \rightarrow n$**,** $m, n \in \mathbb{Z}$	Firing rule dictating that vertices with stack size m will send to adjacent vertices with stack size n
n^+	P_n with the $2-1$ configuration
n^-	P_n with the $1-2$ configuration
s_n^k	The k^{th} stage of P_n
$sgn(m)$	$= 1$ if $m > 0$, $= 0$ if $m = 0$, and $= -1$ if $m = -1$
$u \rightarrow v$**,** $u, v \in V(G)$	Directed edge from vertex u to vertex v in graph G
uv	Edge with endpoints u and v
$x \sim -y$	Edge label dictating that the stack size of x will compared to the negative stack size of y
L(G)	Set of all configurations on G
CCD	Complementary component dominant

Acknowledgements

It truly takes a village to raise a mathematician.

Firstly, I would like to acknowledge my immediate and extended family for their continual support throughout my education. If education had not been so valued, I would never have gotten this far. Having parents that have encouraged me to further my education is why I am here today. Thank you mom and dad!

Secondly, I'd like to acknowledge all my friends who have remained by my side throughout my undergraduate and graduate studies. Huge thank you to my friends: Bill, Amber, Sarah, Kyle, Chris, and so many others. I am so thankful that I wasn't able to do this on my own, because otherwise I would not have made so many great friends at the Mount.

Thank you to Dalhousie University and Mount Saint Vincent University for their financial support from the start. I would like to also thank all of my professors who have played such an influential role in my education. Notably my supervisors Dr. Seager, Dr. Nowakowski, Dr. Messinger, Dr. Janssen, and Dr. Cox each deserve a huge shout-out for their efforts.

Finally, I would like to specifically thank Kyle MacKeigan and Chris Cawthra for all of their help. This thesis would be much different if it were not for the results that we discovered together while sitting around Chris' kitchen table.

To all of you who have helped me in achieving this academic accomplishment, I sincerely say thank you and God Bless!

To Him Be The Glory!

Chapter 1

Introduction

This thesis will examine diffusion in many of its different variations. Diffusion is a chip-firing process defined on a simple graph, G, in which each vertex is assigned an integral number which we refer to as its *stack size* (think of a positive number as the number of chips that reside on that vertex and a negative number as a level of debt that that vertex has fallen into). An assignment of stack sizes to the vertices of a graph is called a *configuration*. During each time step, a subset of these vertices "fire". When a vertex "fires", it sends a chip to a number of its neighbours dependent on the rules of the process. We refer to the original diffusion process from Duffy et al. [9] as *Parallel Diffusion* since each vertex fires at every time step. In Parallel Diffusion, at every time step, each vertex sends a chip to each of its poorer neighbours. An example of Parallel Diffusion is provided in Figure 1.1. It is in this original version that diffusion holds most truly to its name, as the chips diffuse simultaneously from areas of high concentration to areas of low concentration.

The first important question is "Is Parallel Diffusion periodic?" Let C be a configuration on a graph G, and let M be the sum of all of the stack sizes in C. Since stack sizes can be negative in diffusion processes, we cannot suppose that every stack size in every future configuration must be an integer from 0 to M. Thus, since vertices may have an infinite number of possible stack sizes, it is not immediately clear that diffusion processes necessarily exhibit periodic behaviour with some configuration eventually recurring. It is however true that Parallel Diffusion is always eventually periodic, as shown by Long and Narayanan [14], but this result is far from trivial. With Theorem 1.2.1, we extend Long and Narayanan's result to cover graphs with multi-edges.

Unlike some other chip-firing processes like Chip Firing [3], [15] and Brushing [11], [17], in Parallel Diffusion it is possible for a stack size to initially be positive but to become negative as time goes on. For example if some vertex v with a stack size of n, $n \in \mathbb{N}$, is adjacent to $n+1$ vertices, and each of which has a stack size of 0, then after firing, v would have at stack size of -1. However in [9], it was shown that Parallel Diffusion is such that an addition of some constant k, $k \in \mathbb{Z}$, to each stack size will have no effect on determining when and if a chip will move from one vertex to another. So if one wanted to view Parallel Diffusion as a process in which stack sizes are never negative, one would only need to add a sufficient constant k, $k \in \mathbb{N}$, to each stack size.

In Chapter 2, we discuss some previous work on processes similar to diffusion and look at some of the methods that others have used to prove results in this field. With only two previous publications on diffusion [9], [14], it is important to study the methods used by others in related problems and also the types of questions that others have sought to answer. None of the work in Chapter 2 is the work of the author.

Diffusion initially arose from a discrete math seminar at Dalhousie University given by Dr. Kolokolnikov in 2015. Dr. Nowakowski formulated the discrete version (now called Parallel Diffusion). At that time, approximately 500 simulations were run on various cartesian grids (with dimensions $a \times b$, $10 \leq a \leq 20$, $40 \leq b \leq 50$) and configurations with random stack sizes from 10 to 50. The periodic nature of these simulations gave rise to the conjecture that every configuration in Parallel Diffusion eventually leads to a period of length 1 or 2. This means that eventually either a configuration C arises such that every future configuration is C (period 1) or there exists a pair of configurations C and D such that as soon as either one arises, they will alternate in every future step (period 2). This conjecture was eventually published by Duffy et al. [9]. Later Dr. Kolokolnikov asked for the number of configurations possible on a path that existed in period 2. With Long and Narayanan's proof [14] of the conjecture from Duffy et al., we are now able to provide a solution for Dr. Kolokolnikov. In Chapter 3, we look at some counting problems relating to Parallel Diffusion, with our main results being the number of unique configurations that can exist on paths, complete graphs, and stars. With Theorem 3.3.8, we give a recurrence relation for counting the number of unique configurations on a labelled path. With Corollary 3.3.13, we see the asymptotic solution for the k^{th} value of this recurrence to be approximately 0.1564×3.6090^k. With Corollary 3.4.8, we give a recurrence relation for counting the number of unique configurations on an unlabelled complete graph. With Corollary 3.4.9, we see the asymptotic solution for k^{th} value of this recurrence is approximately 0.1809×3.2056^k. With Theorem 3.5.1, we give an explicit solution for the number of period configurations which exist on a labelled star graph.

Only configurations that contain a number of chips divisible by the number of vertices in the graph can possibly have a period length of 1 (See Lemma 3.1.16). Chapter 4 considers the cases which eventually lead to a period of length 1. Specifically, we take a graph in which every stack size is 0, and suppose some subset of vertices each send a chip to each of their respective neighbours. We refer to this process as Quantum Parallel Diffusion because it permits a vertex to send chips to its neighbours despite no discernible difference in their stack sizes. The word *quantum* evokes thoughts of randomness rather than the deterministic nature of Parallel Diffusion. Quantum Parallel Diffusion varies only slightly from Parallel Diffusion because after some subset of vertices initially sends chips despite having no poorer neighbours, the process continues exactly as Parallel Diffusion. The question that we ask is “When will this process result in every vertex having 0 chips?” Theorem 4.1.3 is a characterization of the graphs that return to the configuration in which every stack size is equal to 0 in two steps. The characterization is in terms of partitioning the graph into two dominating sets with special properties. The main conjecture from this chapter is that in Quantum Parallel Diffusion, if a graph returns to the configuration in which every stack size is 0, then it must happen within the first two firings of the vertices.

Parallel Diffusion is ultimately period 1 or 2 [14]. If the process is modified, can the period lengths change? Must there even exist a period at all? In Section 5.2, we show that the process *Pay it Backward*, which is created by slightly altering the rules of Parallel Diffusion, exhibits some regularity but is not, in general, periodic. Also, Section 5.1 gives examples of very long periods in the diffusion variant Two-One Diffusion. In Sequential Diffusion, the vertices still have integral stack sizes, but each time step only permits the firing of a single vertex. In Section 5.3, we look at Sequential Diffusion under a specific configuration and show an instance in which it exhibits periodic behaviour on trees with period length equal to the number of vertices in the tree. The number of ways that this single vertex can be chosen at each time step allows for much variation even in instances with the same initial integral assignments.

Any results not cited are the work of the author.

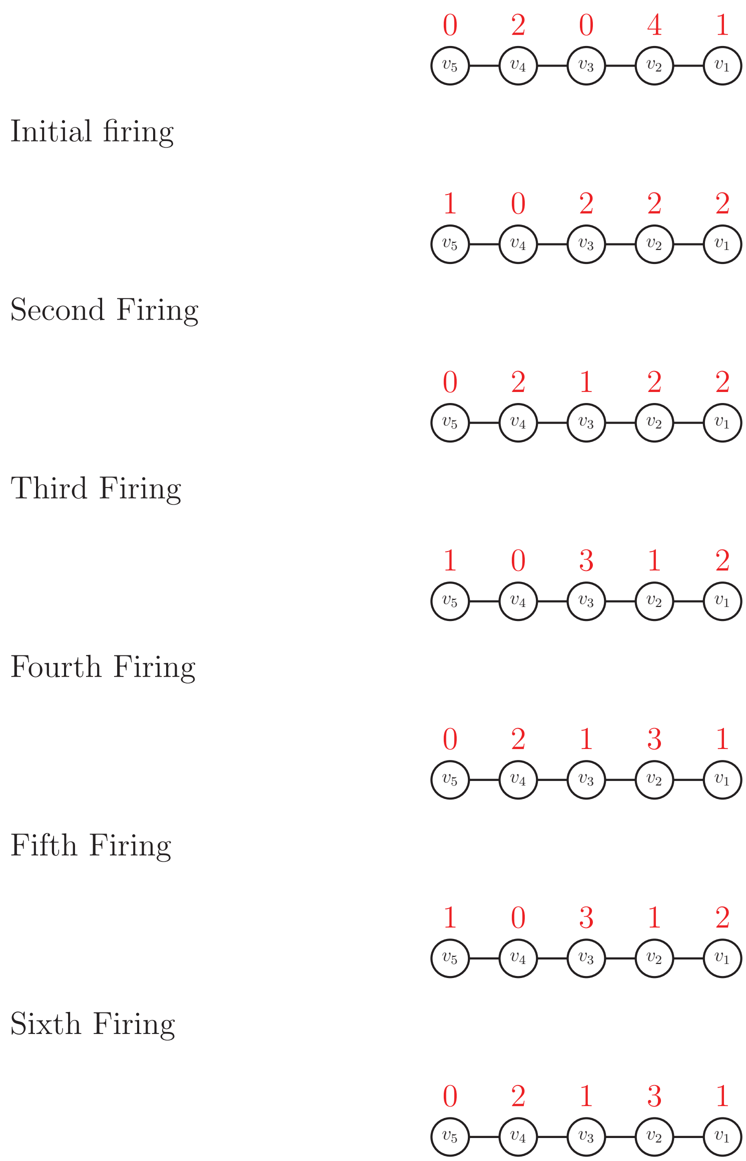


Figure 1.1: Several firings in Parallel Diffusion on P_5

1.1 Terminology

We begin with some basic graph theory definitions that can be found in [18].

Definition 1.1.1. *A* ***graph*** *is a triple consisting of a* ***vertex set*** $V(G)$*, an* ***edge set*** $E(G)$*, and a relation that associates with each edge two vertices (not necessarily distinct) called its* ***endpoints****.*

Definition 1.1.2. *When* u *and* v *are endpoints of an edge, they are* ***adjacent*** *and are said to be* ***neighbours****. When* $e, f \in E(G)$ *and* e *and* f *share an endpoint, they are* ***adjacent****. If vertex* v *is an endpoint of edge* e*, then* v *and* e *are* ***incident****.*

Definition 1.1.3. *A* ***loop*** *is an edge whose endpoints are the same vertex.* ***Multiple edges*** *are edges having the same pair of endpoints. A* ***simple graph*** *is a graph having no loops or multiple edges.*

A simple graph is specified by its vertex set and edge set. The edge set is treated as a set of unordered pairs of vertices. We write $e = uv$ or $e = vu$ for an edge e with endpoints u and v.

Definition 1.1.4. *A* ***subgraph*** *of a graph* G *is a graph* H *such that* $V(H) \subseteq V(G)$ *and* $E(H) \subseteq E(G)$ *and the assignment of endpoints to edges in* H *is the same as in* G*. We then write* $H \subseteq G$ *and say that* G ***contains*** H*. Given a set of vertices* $A \subseteq V(G)$*, the subgraph* ***induced*** *by* A*,* $G|_A$*, is the subgraph of* G *containing the vertices of* A *and every edge in* $E(G)$ *that has both endpoints in* A*.*

Definition 1.1.5. *A* ***path*** *is a simple graph whose vertices can be ordered so that two vertices are adjacent if and only if they are consecutive in the list. A path with* n *vertices will be written as* P_n*. A* ***cycle*** *is a graph with an equal number of vertices and edges whose vertices can be placed around a circle so that two vertices are adjacent if and only if they appear consecutively along the circle. A cycle with* n *vertices will be written as* C_n*. A* ***complete graph*** *is a simple graph whose vertices are pairwise adjacent. A complete graph with* n *vertices will be written as* K_n*. A graph* G *is* ***connected*** *if each pair of vertices in* G *belong to a subgraph of* G *which is a path.*

For the purposes of this thesis, we will restrict ourselves to simple, finite, connected graphs, unless otherwise stated.

Definition 1.1.6. *The* ***degree*** *of vertex v in a simple graph G, written $deg(v)$, is the number of edges incident with v. The* ***open neighbourhood*** *of v, written $N(v)$, is the set of vertices adjacent to v.*

Definition 1.1.7. *In a graph G,* ***contraction*** *of an edge e with endpoints u and v is the replacement of u and v with a single vertex whose incident edges are the edges other than e that were incident to u or v.*

Definition 1.1.8. *A* ***directed graph*** *G is a triple consisting of a* ***vertex set*** *$V(G)$, an* ***edge set*** *$E(G)$, and a function assigning each edge an ordered pair of vertices. The first vertex of the ordered pair is the* ***tail*** *of the edge, and the second is the* ***head****; together they are the* ***endpoints****. We say that an edge is an edge* ***from*** *its tail* ***to*** *or* ***toward*** *its head. The edges of a directed graph are called* ***directed edges*** *due to their assignment from this function. We use the notation $a \to b$ to represent a directed edge (a, b) with tail a and head b. An edge $a \to b$ is an* ***out-edge*** *of a and an* ***in-edge*** *of b.*

Definition 1.1.9. *A* ***mixed graph*** *is a graph that can contain both directed and undirected edges.*

Definition 1.1.10. *In a mixed graph or directed graph, a vertex v is a* ***source*** *if, for all $u \in V(G)$, every edge of the form uv is directed $v \to u$. A vertex y is a* ***sink*** *if, for all $x \in V(G)$, every edge of the form xy is directed $x \to y$.*

Definition 1.1.11. *In a mixed graph or directed graph, for all $v \in V(G)$, the* ***out-degree*** *of v is equal to the number of directed edges in which v is the tail. The* ***in-degree*** *of v is equal to the number of directed edges in which v is the head. Undirected edges play no part in calculating in-degree and out-degree.*

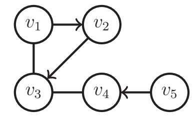
We now introduce some terminology specific to this thesis. We make a small change to the definition of graph orientation in [18], by allowing for undirected edges.

Definition 1.1.12. *A* ***graph orientation*** *of a graph G is a mixed graph obtained from G by choosing an orientation ($x \to y$ or $y \to x$) for each edge xy in some $A \subseteq E(G)$. We refer to the edges that are in $E(G) \backslash A$ as* ***flat****. We refer to the assignment of either $x \to y$, $y \to x$, or flat to an edge xy as xy's* ***edge orientation****.*

Definition 1.1.13. *Let R be a graph orientation of a graph G. A* ***suborientation*** *R' of R is a graph orientation of some induced subgraph G' of G such that every edge xy in G' is assigned the same edge orientation as in R.*

An example of a graph orientation and a suborientation is given in Figure 1.2.

R



R'

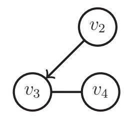


Figure 1.2: Graph orientation R and suborientation R'.

Definition 1.1.14. *A **configuration** is an assignment of integer values to the vertices of a graph.*

An example of a configuration is given in Figure 1.3.

C

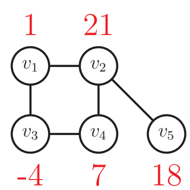


Figure 1.3: Configuration C

Definition 1.1.15. *The assigned value of a vertex v in configuration C is its **stack size in** C and is denoted $|v|^C$. We omit the superscript when the configuration is clear. A vertex v is said to be **richer** than another vertex u in configuration C if $|v|^C > |u|^C$. In this instance, u is said to be **poorer** than v in C. If $|v|^C < 0$, we say v is **in debt in** C.*

Definition 1.1.16. *In Parallel Diffusion, given a graph G and a configuration C on G, to **fire** C, or to **fire** the vertices of C, is to decrease the stack size of every vertex $v \in V(G)$ by the number of poorer neighbours v has and increase the stack size of v by the number of richer neighbours v has. More formally, for all v, let $Z_-^C(v) = \{u \in N(v) : |v|^C > |u|^C\}$ and let $Z_+^C(v) = \{u \in N(v) : |u|^C > |v|^C\}$. Then, firing results in every vertex v changing from a stack size of $|v|^C$ to a stack size of $|v|^C + |Z_+^C(v)| - |Z_-^C(v)|$.*

An example of a firing of a configuration in Parallel Diffusion is given in Figure 1.4.

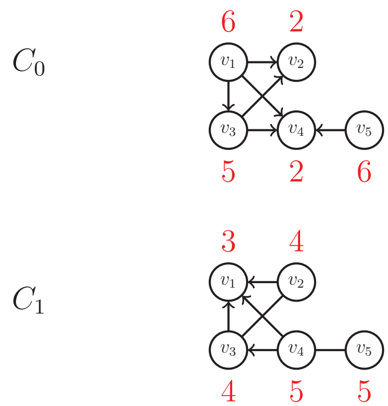


Figure 1.4: Configuration C_0 fires, yielding C_1. Directed edges depict the flow of chips from richer vertices to poorer vertices.

Definition 1.1.17. *We refer to the discrete time increments in diffusion processes as **steps**. The initial configuration of a graph G is referred to as the **configuration at step 0** or the **configuration at** $t = 0$ and the configuration of G at any subsequent step k is referred to as the **configuration at step** k or the **configuration at** $t = k$. At every step, a subset A of $V(G)$ is chosen to be fired. So a step k consists of both a configuration, C_k, and the firing of a subset A of $V(G)$ yielding the configuration at step $k+1$, C_{k+1}. The firing of vertices at step 0 yielding the configuration at step 1 is called the **initial firing**.*

In some variants of Parallel Diffusion, the subset A is chosen by the player and in others it is dictated by the rules of the process. Specifically, in Parallel Diffusion, $A = V(G)$. Note that by Definition 1.1.16, a vertex may fire without actually sending any chips, if it is not adjacent to any poorer vertices. This is an important distinction. In Parallel Diffusion every vertex fires at each step, but by Definition 1.1.16, the only vertices to send chips are those which have poorer neighbours.

Definition 1.1.18. *In diffusion processes, the assigned value of a vertex, v, at step t, is referred to as its **stack size at time** t. If the initial configuration is C, then the stack size at time t is denoted $|v|_t^C$. This implies that $|v|^C = |v|_0^C$. We omit the superscript when the configuration is clear.*

As an extension of Definition 1.1.15, a vertex v is said to be **richer** than another vertex u at time t if $|v|_t > |u|_t$. In this instance, u is said to be **poorer** than v at time t.

Definition 1.1.19. *In diffusion processes, given a graph G and an initial configuration C_0, then $C_n = \{(v, |v|_n^C) : v \in V(G)\}$. A **configuration sequence***

$Seq(C_0) = (C_0, C_1, C_2, \dots)$ is the sequence of configurations that arises as the time increases.

The configuration sequence clearly depends on both the initial configuration and the graph G. However, it will always be clear to which graph we are referring, so we omit any reference to G in our notation, $Seq(C_0)$. Note that given C_0, Parallel Diffusion will have a unique configuration sequence but in Sequential Diffusion (Section 5.3), many configuration sequences can possibly arise from the same initial configuration depending on when each vertex fires.

Definition 1.1.20. *Given two configurations, C and D, of a graph G, C and D are **equal** if $|v|^C = |v|^D$ for all $v \in V(G)$.*

Definition 1.1.21. *Let $Seq(C_0) = (C_0, C_1, C_2, \dots)$ be the configuration sequence on a graph G with initial configuration C_0. A positive integer p is a **period length** if $C_t = C_{t+p}$ for all $t \geq N$ for some nonnegative integer N. In this case, N is a **pre-period length**. For such a value, N, if $k \geq N$, then we say that the configuration, C_k, is **inside** the period.*

Comment 1.1.22. *For the purposes of this thesis, all references to period length will refer to the minimum period length p. Also, all references to pre-period length will refer to the minimum pre-period length that yields that minimum period length p in a given configuration sequence. If $Seq(C_0)$ has minimum period length p and minimum pre-period length N, then we will say $Seq(C_0)$ has period p and pre-period N.*

In Figure 1.1, the period is 2 and the pre-period is 3.

Observation 1.1.23. *In diffusion processes, every configuration induces a graph orientation.*

Proof. Let G be a graph and C_t a configuration on G. For all pairs of adjacent vertices u, v in G at step t, either u sends a chip to v, v sends a chip to u, or no chip is moved between u and v in C_t. Let uv be an edge. Assign directions as follows:

- If u sends a chip to v at time t, assign uv the edge orientation $u \to v$.
- If v sends a chip to u at time t, assign uv the edge orientation $v \to u$.
- If no chip is sent from u to v or from v to u at time t, do not direct the edge uv.

Thus, a graph orientation on G results.

□

We say that this graph orientation is **induced** by C_t, the configuration of G at time t. Note that the orientation induced depends on which vertices fire at a given step (Definition 1.1.17). So for instance, the orientation induced by C_t in Parallel Diffusion may differ from the orientation induced by C_t in Sequential Diffusion. We see an example of a graph orientation induced by a configuration in Parallel Diffusion in Figure 1.5.

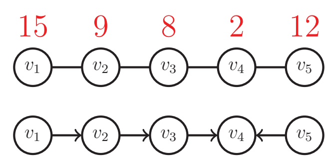


Figure 1.5: Configuration on P_5 and its induced graph orientation.

With diffusion processes, one must first decide on an initial configuration. The infinite number of ways that this can be done can make it difficult to study the process in a thorough or systematic way.

Three types of configurations in particular have been studied: the standard configurations, the Millpond configurations, and the fixed configurations. Defined in [9], the *standard configuration* assigns chips to each vertex equal to that vertex's degree. In the *Millpond configuration* [9], one vertex has stack size 1 and every other vertex has stack size zero. The name Millpond refers to the "ripple" effect that is created as the effect of that single chip diffuses throughout the graph. In Section 5.3, we analyze the Millpond configuration in Sequential Diffusion. *Fixed configurations* are those in which every stack size is equal. In Parallel Diffusion, this results in no movement of chips, so we instead look at the variant Quantum Parallel Diffusion. In Quantum Parallel Diffusion (Chapter 4), every vertex begins with a stack size of 0, and the process begins with a "quantum firing", where one or more vertices are initially chosen to send a chip to each of their neighbours, even though no vertex is poorer than another at this point. After the quantum firing, the process continues as Parallel Diffusion with no further quantum firings.

1.2 Previous Work on Parallel Diffusion

The study of Parallel Diffusion and its variants is a very young one, with the first paper on Parallel Diffusion by Duffy et al. published in 2018 [9], and the only other publication on the subject being Long and Narayanan's paper [14] which proves that Parallel Diffusion is always a periodic process with period lengths of 1 and 2.

The questions of periodicity and finding a minimum number of chips for a configuration such that no vertex has a negative stack size at any future step were first proposed by Duffy et al. The paper by Duffy et al. proves that a number of graph classes are necessarily periodic, and that any connected bipartite graph with the Millpond configuration will have a period of length 2. Also, they show that a configuration is periodic with period at most two if and only if each edge in the configuration can be assigned a single chip, which is the only chip that will ever be sent across that edge at any future step. Duffy et al. stated that Parallel Diffusion is such that an addition of k chips to each stack size in a configuration will have no effect on determining whether or not a vertex will fire in any future step. We prove this with Lemma 3.1.1.

Aside from these two papers, the author knows of two additional unpublished works: a master's thesis by Degaetani [8] which looks into Parallel Diffusion on infinite graphs, and a paper on ArXiv by Carlotti and Herrman [6] in which they determine that if a graph with n vertices is given a configuration in which every stack size is at least $n-2$, then every stack size will be nonnegative at every future step. Degaetani's work was in response to a question posed by Duffy et al., and Carlotti and Herrman's work answered a question posed by both Long and Narayanan, and Duffy et al.

Duffy et al. [9] conjectured that every configuration sequence in Parallel Diffusion has a period length of 1 or 2. This was proven by Long and Narayanan [14]. We now prove an extension of Long and Narayanan's result for a variation of Parallel Diffusion defined on graphs with multiple edges. Since Long and Narayanan's result trivially holds for graphs which are disconnected or contain loops, this multiple edge result is all that is needed to prove that for all graphs, whether simple or not, Parallel Diffusion is periodic with period 1 or 2. We present this proof to show how Long and Narayanan's methods can be used to solve similar problems. In Chapter 5, we will see examples of variants of diffusion for which Long and Narayanan's methods do not seem useful because these variants are not necessarily periodic with short periods.

Theorem 1.2.1. *In the variant of Parallel Diffusion in which, at every step, vertices send to their poorer neighbours a number of chips equal to the number of edges shared* *between the two vertices, every configuration sequence has period 1 or 2.*

We will closely follow Long and Narayanan's methods from [14], only varying at points to allow for the existence of multiple edges. Long and Narayanan's proof involved the use of a non-increasing potential function

$$P(t) = \sum_{v=1}^{n} |v|_t \times |v|_{t+1}$$

where n represents the number of vertices in the graph, v represents each individual vertex in the graph labelled from 1 through n, and $|v|_t$ represents the number of chips on vertex v at time t.

We begin by introducing some notation. Let G be a graph, and let $v \in V(G)$. For all neighbours u of v, let $E(uv)$ represent the number of edges with endpoints u and v. Let $A'_v(t)$ be the sum

$$\sum_{|u|_t > |v|_t} E(uv),$$

and $B'_v(t)$ be the sum

$$\sum_{|u|_t < |v|_t} E(uv).$$

The result of a firing on a vertex's stack size is $|v|_{t+1} = |v|_t + A'_v(t) - B'_v(t)$.

For the purposes of this proof, we will view vertices as being elements of the set $\{1, 2, \ldots, n\}$. So, if u is said to be *greater* than v, then u is a greater element in the set $\{1, 2, \ldots, n\}$ than v. Let $sgn(m)$ be a function that returns -1 if $m < 0$, 0 if $m = 0$, and 1 if $m > 0$. Supposing $u < v$, let $x'_{uv}(t) = E(uv) \times sgn(|u|_t - |v|_t)$ and $y'_{uv}(t) = E(uv) \times sgn(|u|_{t+1} - |v|_{t+1})$. Let $(x'_{uv}(t), y'_{uv}(t))$ be a label assigned to every edge uv at time t. Whenever the edge uv does not exist, we let $x'_{uv}(t) = 0$ and $y'_{uv}(t) = 0$. We now present the proof of Theorem 1.2.1.

Proof. Our method will involve first showing that this potential function, $P(t)$, is bounded below and then showing that it is monotonic decreasing. This will show the function to be eventually constant and we will show that this implies that every configuration sequence in Parallel Diffusion has period 1 or 2.

We begin by showing that this potential function is bounded below. We will use the fact that the stack size of a vertex can increase or decrease by no more than $|E(G)|$ in a single step.

Using this, we get that

$$\begin{aligned}
|(|v|_{t+1} - |v|_t)| &\leq |E(G)| \\
(|v|_t)^2 + (|v|_{t+1})^2 - 2(|v|_{t+1} \times |v|_t) &\leq |E(G)|^2 \\
-(|v|_t)^2 - (|v|_{t+1})^2 + 2(|v|_{t+1} \times |v|_t) &\geq -|E(G)|^2 \\
2|v|_{t+1} \times |v|_t &\geq -|E(G)|^2 \\
|v|_{t+1} \times |v|_t &\geq -\frac{1}{2}|E(G)|^2
\end{aligned}$$

So,

$$P(t) = \sum_{v=1}^{n} |v|_t \times |v|_{t+1} \geq -\frac{1}{2}n|E(G)|^2.$$

Next, we will show that the function is non-increasing with time. That is, we will show that

$$P(t+1) - P(t) = \sum_{v=1}^{n} |v|_{t+1} \times (|v|_{t+2} - |v|_t) \leq 0.$$

Now,

$$\begin{aligned}
|v|_{t+1} &= |v|_t + A'_v(t) - B'_v(t) \\
|v|_{t+2} &= |v|_{t+1} + A'_v(t+1) - B'_v(t+1) \\
|v|_{t+2} &= |v|_t + A'_v(t) + A'_v(t+1) - B'_v(t) - B'_v(t+1)
\end{aligned}$$

Supposing that $u < v$, we label each edge uv with the label $(x'_{uv}(t), y'_{uv}(t))$. With the convention that $(x'_{uv}(t), y'_{uv}(t)) = (0,0)$ whenever uv is not an edge of G, we get that the amount of change that occurs to a particular stack size $|v|_t$ over the course of two firings can be exhibited by

$$A'_v(t) + A'_v(t+1) - B'_v(t) - B'_v(t+1) = \sum_{u<v}(x'_{uv}(t) + y'_{uv}(t)) - \sum_{u>v}(x'_{vu}(t) + y'_{vu}(t)).$$

We break the sum into two pieces because the definitions of $x'_{uv}(t)$ and $y'_{uv}(t)$ represent the flow of chips to and from a vertex v differently depending on if v is the greater or lesser vertex in a particular edge. Thus, one sum represents the effects of two firings from the lesser vertices, and the other sum represents the effects of two firings from the greater vertices.

So,

$$|v|_{t+2} = |v|_t + \sum_{u<v}(x'_{uv}(t) + y'_{uv}(t)) - \sum_{u>v}(x'_{vu}(t) + y'_{vu}(t))$$

$$|v|_{t+2} - |v|_t = \sum_{u<v}(x'_{uv}(t) + y'_{uv}(t)) - \sum_{u>v}(x'_{vu}(t) + y'_{vu}(t)).$$

$$P(t+1) - P(t) = \sum_{v=1}^{n}\left(|v|_{t+1}\Big(\sum_{u<v}(x'_{uv}(t) + y'_{uv}(t)) - \sum_{u>v}(x'_{vu}(t) + y'_{vu}(t))\Big)\right).$$

The actions of these multiple sums can be grouped together by using a sum over all ordered pairs of vertices. The two sums with indices $u < v$ and $u > v$ for some v can be brought together under a single summation with all ordered pairs of vertices $u < v$ as the index.

$$P(t+1) - P(t) = \sum_{u<v}(|v|_{t+1} - |u|_{t+1})(x'_{uv}(t) + y'_{uv}(t)).$$

Next we will show that each term of this sum is at most 0 since each factor is positive if and only if the other is nonpositive. Let $u < v$. Remember that $x'_{uv}(t)$ is equal to $-E(uv)$, 0, or $E(uv)$ and that the same is true of $y'_{uv}(t)$. If $x'_{uv}(t) + y'_{uv}(t)$ is positive, then $y'_{uv}(t)$ is nonnegative and this implies that v does not have a greater stack size than u at time $t+1$, so $|v|_{t+1} - |u|_{t+1} \leq 0$. Conversely if $x'_{uv}(t) + y'_{uv}(t)$ is negative, then $y'_{uv}(t) \leq 0$, implying that $|v|_{t+1} - |u|_{t+1} \geq 0$.

Therefore, we can conclude that as t increases, $P(t)$ does not increase. Since $P(t)$ is bounded, there exists some T such that for all $t \geq T$, $P(t)$ is constant.

As long as our sum

$$\sum_{u<v}(x_{uv}(t) + y_{uv}(t))(|v|_{t+1} - |u|_{t+1})$$

has a nonzero term, then the function has not yet reached such a T value. This is because $P(t+1) - P(t)$ being negative implies that $P(t+1)$ is less than $P(t)$. When $P(t)$ has reached its minimum, $P(t{+}1)$ will be equal to $P(t)$. This will only occur when every summand is equal to zero. So, we seek to characterize the instances in which this sum has a negative summand. This will occur any time that both $x'_{uv}(t) + y'_{uv}(t) \neq 0$ and $y'_{uv}(t) \neq 0$. So, if the label $(deg(uv), deg(uv)), (0, deg(u, v)), (0, -deg(u, v))$, or $(-deg(u, v), -deg(u, v))$ appears on any edge, then $P(t)$ has not yet reached its minimum value.

So, if P has reached its minimum value, then every label fits into the set

$$\{(deg(u, v), -deg(u, v)), (-deg(u, v), deg(u, v)), (0, 0), (deg(u, v), 0), (-deg(u, v), 0)\}.$$

Suppose we have reached such a point. Note that the labels applied to each edge at a given time speak to the direction that chips will be travelling across that edge not just at the current step, but also at the next step. So a label of (i, j) at time t and a label of (k, ℓ) at time $t + 1$ implies that $j = k$. Thus, we get that every edge that has the label $(deg(u, v), 0)$ or $(-deg(u, v), 0)$ will at the next step have the label $(0, 0)$ and at no future step will have any other label. So there will reach a step T' in which every label belongs to the set

$$\{(deg(u, v), -deg(u, v)), (-deg(u, v), deg(u, v)), (0, 0)\}.$$

So from step T' onward, if chips are sent from u to v at step t, then chips must be sent from v to u at step $t+1$. This implies that it must be true for every edge u that $|u|_t = |u|_{t+2}$. Thus, the least period length of a configuration sequence must be at most 2.

□

In the proof of Theorem 1.2.1, it is shown that once inside the period, the only edge labels which can exist belong to the set

$$\{(deg(u, v), -deg(u, v)), (-deg(u, v), deg(u, v)), (0, 0)\}.$$

This implies that if chips are sent from a vertex u to another vertex v at step t, then chips must be sent from v to u at step $t + 1$. Also, this implies that if no chips are sent along the edge uv at time t, then no chips will be sent along the edge uv at time $t+1$. We will be using this result in future chapters, so we set it aside as the following corollary.

Corollary 1.2.2. *In Parallel Diffusion, let C_t be the configuration at time t, suppose C_t is inside the period, and suppose the vertices u and v are neighbours. If $|u|_t > |v|_t$, then $|v|_{t+1} > |u|_{t+1}$. Also, if $|u|_t = |v|_t$, then $|u|_{t+1} = |v|_{t+1}$.*

Chapter 2

Related Questions

In this chapter, we look at two previous processes which are similar to Parallel Diffusion. The two examples we explore are Chip Firing [3], [15] and Brushing [11], [17]. Since there are only two published papers on Parallel Diffusion and one pre-print [9],[14],[6], there are not yet any standard proof techniques. We present some of the theorems and proofs regarding Chip Firing and Brushing to illustrate techniques that others have used. We include a number of results and proofs to show the similarities and differences between these previous processes and the new ones found in this thesis. Diffusion, Chip Firing, and Brushing all share a number of qualities: a graph, integers representing quantities residing on the vertices of the graph, a rule dictating the way that quantities can change by sending or receiving, and periodic tendencies. One of the observations from all three processes is that once periodicity has been achieved, the chips are assigned to edges in Parallel Diffusion and Chip Firing, and to paths in Brushing. So the chips can be tracked as they move throughout the graphs; we do not simply lose track of them when they are added to a neighbour's stack size.

In Chip Firing, only one vertex fires at a time and when a vertex does fire, it sends a chip to each of its neighbours. Vertices are not permitted to go into debt, so only vertices with at least degree-many chips can fire. Both Parallel Diffusion and Chip Firing have the possibility of chips being sent for an infinite number of steps or for chips to at some point stop being sent. In Parallel Diffusion, if no chips are sent, then every vertex must have the same stack size. These configurations have period length of 1. In Chip Firing, firings will terminate if and only if every vertex has fewer than degree-many chips. The sequential nature of Chip Firing is another way that it differs from Parallel Diffusion. However, it is shown in [3] that in Chip Firing, the order of the firings will not determine whether or not the process will terminate. For Chip Firing, there are results pertaining to the total number of chips in the initial configuration that one can use to help determine whether or not the firings will terminate. The problem of determining which initial configurations in Parallel Diffusion will eventually lead to a configuration with an induced orientation of all flat edges is still very much open.

In Brushing, a number of brushes are initially placed on the vertices of a graph with the goal of eventually *cleaning* the graph. Every edge and vertex is initially *dirty* and at every step, a dirty vertex with at least degree-many brushes is chosen to send a brush to each of its neighbours, cleaning itself and all of its incident edges in the process. Much like in Chip Firing, the process terminates when no dirty vertex has at least degree-many brushes. Because of the sequential nature of the process, Sequential Diffusion is the closest analog to Brushing in this thesis. The difference between Sequential Diffusion and Parallel Diffusion is that in Sequential Diffusion, only one vertex is chosen at each step to send a chip to its poorer neighbours. The number of steps shown in this chapter to be required to clean a graph is $|V(G)|$. This is equal to the period length of a tree in Sequential Diffusion under the Millpond configuration shown in Section 5.3.

The results in this chapter are not the work of the author.

2.1 Chip Firing

In [3], Chip Firing is introduced. This one-player process is played on a connected graph G with each vertex having a stack containing a non-negative integral number of chips. A move is choosing a vertex that has at least degree-many chips and having it send one chip to each of its neighbours. The process ends when every vertex has fewer than degree-many chips. It is possible, depending on the number of chips in the initial configuration, that the process will never end.

One concern of Bjorner et al. [3] is to study the finiteness of the Chip Firing process. Bjorner et al. show that the order of firings does not matter. Whether or not the process is finite will depend entirely on the initial configuration. If a process is finite, then that implies that there exists some step t, after which no chips will be sent. This is analogous to a configuration in Parallel Diffusion in which every vertex has the same stack size, no chips will be sent. If a Chip Firing process is infinite and the chips are chosen to fire consistently in the same order, then that implies that there exists some period since the total number of chips in the process does not change as time increases and negative stack sizes are not possible. The following results on Chip Firing foreshadow the work on Sequential Diffusion under the Millpond configuration in Subsection 5.3.1. This is due to the sequential nature of both processes and the issue of determining whether or not the order of the firings matters. In Chip Firing, if the process is infinite and the chips are chosen to fire consistently in the same order, then the period is n, the number of vertices in the graph. This is implied by

Lemma 2.1.3. See [2] for a parallel version of Chip Firing. Bitar and Goles conjectured that Parallel Chip Firing also exhibited period n in every infinite case [2]. However, Kiwi et al. [12] show that there are configurations for which the period is much larger.

In Figure 2.1, we give an example of Chip Firing. In this example, when more than one vertex is capable of firing at a particular step, the choice is made arbitrarily.

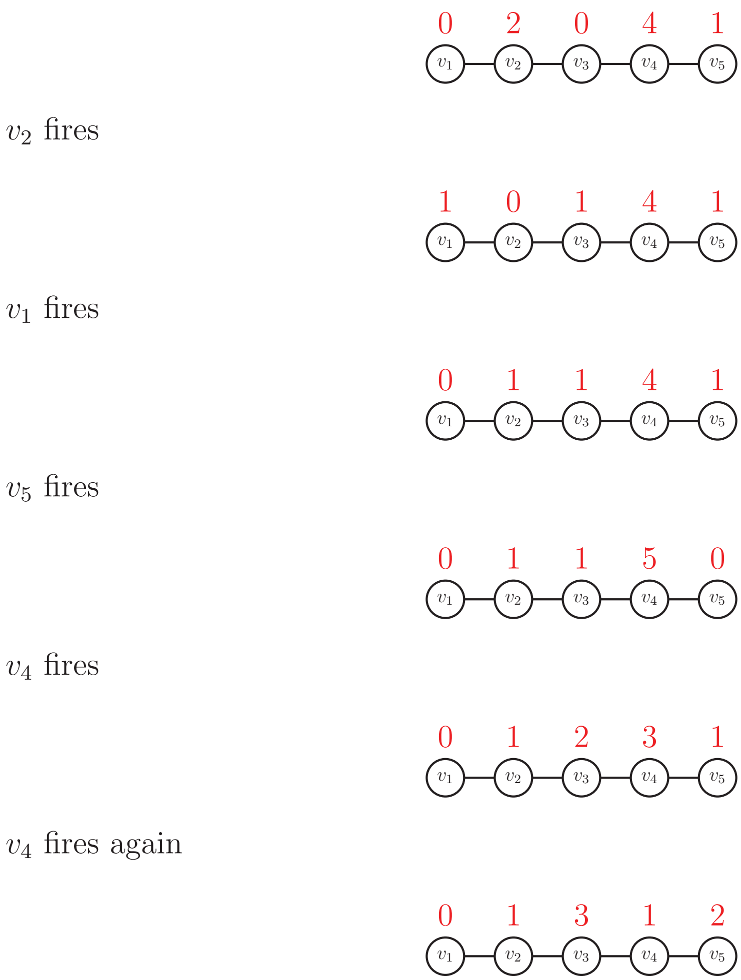


Figure 2.1: Some steps of a Chip Firing process on P_5.

We now go over some results regarding when a configuration yields a Chip Firing process of infinite length.

Lemma 2.1.1. *(Theorem 2.3(a) in [3]) Let C be a configuration on a graph G. If the sum of all integer value assignments in C is greater than $2 \cdot |E(G)| - |V(G)|$, then the Chip Firing process will never terminate.*

Proof. Let C be a configuration on a graph G such that the sum of all integer value assignments is greater than $2 \cdot |E(G)| - |V(G)|$. Suppose, by contradiction, that at some step t, the Chip Firing process has terminated. This means that for all $v \in V(G)$, $|v|_t < deg(v)$. By definition, the total degree of G is $2 \cdot |E(G)|$. At time t, each vertex v must have no more than $deg(v) - 1$ chips. Therefore, the total number of chips on the graph must be less than or equal to $2 \cdot |E(G)| - 1 \cdot |V(G)| = 2 \cdot |E(G)| - |V(G)|$. This is a contradiction. □

Lemma 2.1.2. *(Lemma 2.1 in [3]) If a Chip Firing process is infinite, then every vertex is fired infinitely often.*

Proof. A Chip Firing process can only be infinite if at least one vertex, v, is fired at infinite different steps. Suppose the initial configuration has a total of N chips. No vertex can ever have more than N chips. So, every vertex adjacent to v will have to fire infinitely often. Since we only view Chip Firing on connected graphs, this implies that every vertex is fired infinitely often. □

Lemma 2.1.3. *(Lemma 2.2 in [3]) If the Chip Firing process terminates, then there is a vertex which is not fired at all.*

Proof. We prove the contrapositive. So we suppose that every vertex has fired at least once by time t; we will reach that the process will not terminate. Let v be the vertex that last fired the greatest number of steps in the past. Then v must have received a chip from every one of its neighbours since the last time it fired and thus, must be ready to fire. So, if every vertex has fired, then the process will never terminate. □

Theorem 2.1.4. *(Theorem 2.3(c) in [3]) Let N be the total number of chips distributed on G. If $N < |E(G)|$, then the process is finite.*

Proof. Note that directions can be added to the edges of any graph such that no directed cycles exist by assigning real number values to every vertex and having edges always go from the greater value to the lesser value. This implies the existence of a source vertex. Let $deg_t^+(v)$ be the out-degree of v at time t. Let

$$T = \sum_{v \in V(G)} max\{0, |v|_t - deg_t^+(v)\}.$$

We say that a vertex u is **deficient at time** t if $|u|_t < deg_t^+(u)$. Since there are fewer chips than edges, the total number chips is less than the total out-degree of the directed graph, so at least one vertex is initially deficient. Note that a deficient vertex will return 0 in the T calculation. Let v be the vertex that fires at the first step. After v fires, change out-edges from v to in-edges. In the T summation, the value represented by v will:

- not be affected by the number of vertices adjacent to v with which v is the tail since the decrease of one in the stack size of v is counteracted by the edge changing direction so that v is now the head.
- decrease by one for every adjacent vertex with which v is the head since the stack size of v decreases but there is no edge changing direction to counteract it in the T calculation.

In the T summation, the value of any vertex u adjacent to v will:

- not be affected if u is the head of uv since the increase of one to its stack size is counteracted by the direction of uv changing.
- not be affected if u is the tail of uv and u is deficient since the increase of one to its stack size does not change its value in the T calculation.
- increase by one if u is the tail of uv and u is not deficient since the increase of one to its stack size is not counteracted by any edge changing direction.

So, T will be unchanged unless a deficient vertex received a chip. In this case, that deficient vertex will still return 0 in the T calculation but the vertex that sent the chip has decreased its value in the T calculation. So, we get that T will decrease at every step in which a deficient vertex receives a chip. However, T cannot decrease infinitely; its minimum is 0. Therefore, the firings must cease at some time step. Thus, the process is finite. □

In [15], Merino looks at a variant of the Chip Firing process and in particular, *critical configurations*. In Merino's version, a vertex q is chosen to have a nonpositive integral number of chips. Every other vertex begins with a non-negative integral number of chips such that the total number of chips is 0. The process runs as normal with every step involving a vertex with at least degree-many chips sending a chip to each neighbour, except in the event that no vertex has at least degree many chips, q sends a chip to each of its neighbours (thus further decreasing q's stack size). In this way, no matter what the initial configuration is, the process will never terminate. The following definitions and theorem are from Merino's paper [15]. Merino's work foreshadows the work done on Quantum Parallel Diffusion in Section 4 as both processes allow for a firing when the normal rules would not. Also, Merino uses the term recurrent to refer to a configuration that comes back to its starting point, like the work with Quantum Parallel Diffusion on restoring the fixed configuration.

Definition 2.1.5. *A configuration is* ***stable*** *if only q is able to fire in the next time step. A configuration C is* ***recurrent*** *if, after some sequence of firings, we arrive again at C. A* ***critical configuration*** *is a configuration that is both* ***stable*** *and* ***recurrent****.*

Definition 2.1.6. *The* ***weight*** *of a configuration, $w(C)$, at time t is the sum of all stack sizes except q. So, $w(C) = \sum_{v \neq q} |v|_t$. The* ***level*** *of a critical configuration, C, is given by $level(C) = w(C) - |E(G)| + deg(q)$.*

Theorem 2.1.7. *(Theorem 3.3 in [15]) If G is a graph and C a critical configuration on G, then $0 \leq level(C) \leq |E(G)| - |V(G)| + 1$.*

2.2 Brushing

In this section, we discuss the problem of "cleaning" a graph from a regenerating contaminant (like algae in a network of pipes [17]). In Brushing, each vertex of a graph is given a number of brushes and during each step a number of brushes travel along their incident edges "cleaning" them. We will look at both the sequential [17] and parallel [11] versions of Brushing.

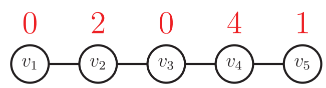
2.2.1 Sequential Brushing

In [17], a method of "cleaning" a graph is introduced. Instead of chips, brushes are placed on some of the vertices. Every edge and vertex is initially "dirty" and when a vertex fires, it is "cleaned" and it sends a brush down each incident dirty edge "cleaning" them as well. So, every time a brush is fired down an edge, one endpoint loses a brush and the other endpoint gains a brush. Note that a vertex can fire even if it is not adjacent to any dirty edges.

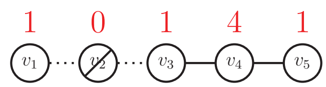
One goal with this process is to find a brush configuration that is capable of cleaning the entire graph (vertices and edges) and is such that once the graph has been cleaned, if it were to be labelled dirty again, the final configuration of brushes would be capable of once again cleaning the graph. Let $\omega_t(v)$ be the number of brushes on a vertex v at time t. Let $D_t(v)$ be the number of dirty edges incident to v at time t.

The cleaning algorithm labels all vertices dirty and then fires vertices in any order that follows the rule: a vertex v only fires at time t if $\omega_t(v) \geq D_t(v)$. When no such vertex exists, the algorithm completes and returns the set of remaining dirty vertices. This firing rule foreshadows the **Algorithm for Trees** used in Section 5.3 to show that trees can exhibit periodic behaviour in Sequential Diffusion under the Millpond configuration. Only one vertex fires during each step. The reversibility theorem, Theorem 2.2.2, foreshadows the work in Chapter 4 regarding trying to find quantum sets that return the fixed configuration.

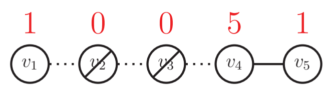
We begin with an example. In Figure 2.2, all edges and vertices will begin dirty, cleaned edges will be represented by dotted lines, and cleaned vertices will have a line through them.



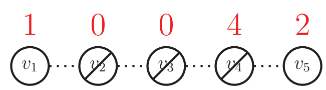
v_2 fires



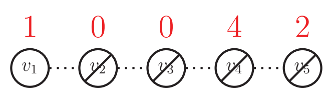
v_3 fires



v_4 fires



v_5 fires even though it has no incident dirty edge since v_5 must itself get cleaned



v_1 fires even though it has no incident dirty edges since v_1 itself must become clean

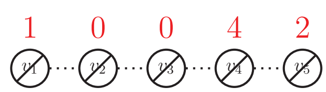


Figure 2.2: Sequential Brushing example on P_5 which will return no dirty vertices.

Theorem 2.2.1. *(Theorem 2.1 in [17]) Given a graph G and the initial configuration of brushes ω_0, the cleaning algorithm returns a unique final set of dirty vertices.*

Proof. We will suppose that we have two cleaning processes and reach that they return the same set of dirty vertices. Suppose we have two sequences of vertices, $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_m)$, that satisfy the cleaning algorithm. Since any vertex firing will result in every incident edge being cleaned, it suffices to prove that the sets $\{a_1, a_2, \ldots, a_n\}$ and $\{b_1, b_2, \ldots, b_m\}$ are equal, with the order of firings not mattering. Suppose, by contradiction and without loss of generality, that some vertex in B is not in A. Let b_ℓ, $1 \leq \ell \leq m$, be the first such vertex. Clearly, when sequence A completed at time n, b_ℓ had fewer brushes than dirty neighbours, since otherwise, this would have resulted in another firing occurring before the algorithm terminated. Also, we know that A contains the vertices $b_1, b_2, \ldots, b_{\ell-1}$. This implies that the number of dirty vertices adjacent to b_ℓ at the conclusion of firing sequence A is less than or equal to the number of dirty vertices adjacent to b_ℓ at time $\ell - 1$ in firing sequence B. This is because the additional firings contained in A may have served to clean additional vertices but could not have possibly caused more vertices to become dirty. If we let $D_t(v)$ be the number of dirty edges incident to v at time t under sequence A and $C_t(v)$ be the number of dirty edges incident to v at time t under sequence B, then this statement can be expressed as $D_n(b_\ell) \leq C_{\ell-1}(b_\ell)$. We now calculate the number of brushes on b_ℓ at the conclusion of firing sequence A. Since b_ℓ never fired, it still has all of its initial brushes. Also, b_ℓ has gained a brush from every one of its clean neighbours. If we let $\omega_t(v)$ be the number of brushes on v at time t under sequence A and $\tau_t(v)$ be the number of brushes on v at time t under sequence B (with $\tau_0(v) = \omega_0(v)$), then we get

$$
\begin{aligned}
\omega_n(b_\ell) &= \omega_0(b_\ell) + deg(b_\ell) - D_n(b_\ell) \\
&\geq \tau_0(b_\ell) + deg(b_\ell) - C_{\ell-1}(b_\ell) \\
&= \tau_{\ell-1}(b_\ell)
\end{aligned}
$$

Since b_ℓ was the ℓ^{th} firing in sequence B, $\tau_{\ell-1}(b_\ell) \geq C_{\ell-1}(b_\ell)$. So, $\omega_n(b_\ell) \geq \tau_{\ell-1}(b_\ell) \geq C_{\ell-1}(b_\ell) \geq D_n(b_\ell)$. This is a contradiction.

□

Theorem 2.2.2. *(Theorem 2.3 in [17]: The Reversibility Theorem) Given the initial configuration ω_0, suppose G can be cleaned yielding final configuration ω_n, where $n = |V(G)|$. Then, given initial configuration $\tau_0 = \omega_n$, G can be cleaned yielding the final configuration $\tau_n = \omega_0$.*

Proof. Let $A = (a_1, a_2, \ldots, a_n)$ be a firing sequence of vertices that will successfully clean G. That is, the cleaning algorithm will return an empty set of dirty vertices. Let $\omega_t(v)$ be the number of brushes on v at time t under sequence A and let $D_t(v)$ be the number of dirty edges incident to v at time t under sequence A. Let $N^-(a_t) = |a_t a_i \in E(G) : i < t|$ and $N^+(a_t) = |a_t a_i \in E(G) : i > t|$. Note that since A will clean the entire graph, A must contain every vertex in G. So, $deg(a_t) = N^-(a_t) + N^+(a_t)$. Note that after a vertex fires, its number of brushes will not change again before the algorithm completes. Also note that the number of brushes on a vertex before it has fired is equal to its initial number of brushes plus its number of clean neighbours. Using these facts, we get that

$$\begin{aligned}
\omega_n(a_t) &= \omega_t(a_t) \\
&= \omega_{t-1}(a_t) - D_{t-1}(a_t) \\
&= \omega_0(a_t) + deg(a_t) - D_{t-1}(a_t) - D_{t-1}(a_t) \\
&= \omega_0(a_t) + deg(a_t) - N^+(a_t) - N^+(a_t) \\
&= \omega_0(a_t) + N^-(a_t) - N^+(a_t)
\end{aligned} \tag{2.1}$$

Now, let $B = (a_n, a_{n-1}, \ldots, a_1)$. We will let $\tau_t(v)$ be the number of brushes on v at time t under sequence B (with $\tau_0 = \omega_n$) and we will let $C_t(v)$ be the number of dirty edges incident to v at time t. By induction on t, we will show that at each step t, the t^{th} vertex in B can fire. This will show that B will clean G. The tth vertex in B is a_{n-t+1}. For our base case we look at $t = 0$.

$$\tau_0(a_n) = \omega_n(a_n) = \omega_0(a_n) + N^-(a_n) - N^+(a_n) \geq N^-(a_n) = C_0(a_n)$$

Thus, the first firing does occur and the algorithm will continue since a_n has no fewer brushes than it has dirty neighbours.

For our induction step, we will assume that the vertices $a_n, a_{n-1}, \ldots, a_k$ $(n \geq k > 1)$ have already fired at the time $n - k + 1$. We check to see that a_{k-1} can fire in the next step.

$$\begin{aligned}\tau_{n-k+1}(a_{k-1}) &= \tau_0(a_{k-1}) + deg(a_{k-1}) - C_{n-k+1}(a_{k-1}) \\ &= \omega_n(a_{k-1}) + N^+(a_{k-1}) \\ &= \omega_0(a_{k-1}) + N^-(a_{k-1}) \\ &\geq N^-(a_{k-1}) \\ &= C_{n-k+1}(a_{k-1})\end{aligned}$$

To finish the proof we must show that $\tau_n = \omega_0$. By performing the calculations in Equation 2.1 again for τ_n instead of ω_n, we get that $\tau_n(a_t) = \tau_0(a_t)+N^+(a_t)-N^-(a_t)$. So, $\tau_n(a_t) = \omega_n(a_t)+N^+(a_t)-N^-(a_t)$ and substituting in our known value for $\omega_n(a_t)$, we get

$$\tau_n(a_t) = \omega_0(a_t) + N^-(a_t) - N^+(a_t) + N^+(a_t) - N^-(a_t) = \omega_0(a_t)$$

□

Theorem 2.2.3. *(Theorem 4.3 in [10]) It is an NP-complete problem to determine whether k brushes will clean a graph.*

2.2.2 Parallel Brushing

In [17], the parallel version is also discussed and it is expanded on in [11]. In parallel brushing, at every step, each vertex with at least as many brushes as incident dirty edges fires. As in the sequential version, once the algorithm terminates, because no vertices are capable of firing, it returns the set of dirty vertices. Since every vertex has the chance to fire at each step, and thus at no point does the player have to choose which vertex to fire, there is a unique firing sequence for each configuration.

Let $b(G)$ be the minimum number of brushes needed to clean every vertex of G before sequential algorithm terminates. Let $pb(G)$ be the minimum number of brushes needed to clean every vertex of G under the parallel algorithm. Unlike Sequential Brushing, there is no *Reversibility Theorem* for Parallel Brushing to imply that a graph can be continually cleaned, although some results exist for specific cases.

Theorem 2.2.4. *(Theorem 2.2 in [17]) For any graph G, $b(G) = pb(G)$.*

Proof. Suppose that G can be cleaned using the parallel brushing algorithm. So there exists a unique firing sequence (J_t) where J_i is a set containing every vertex that fired at time i. This can obviously serve also as a sequential brushing firing sequence in which, the ordering of the sets remains the same and some permutation is chosen of the elements in each J_i. So, $b(G) \leq pb(G)$.

Now, by contradiction, suppose that $b(G) < pb(G)$. Suppose also that we have a firing sequence $A = (a_1, a_2, a_3, \ldots, a_n)$ of vertices in G that satisfies the sequential brushing algorithm, uses exactly $b(G)$ brushes and returns 0 dirty vertices. Since $b(G) < pb(G)$, it must be true that the parallel brushing algorithm on the same configuration returns a non-empty set of dirty vertices. Let a_i be the vertex in A with the least index that returns dirty in the parallel brushing algorithm. The existence of this vertex shows that the parallel brushing algorithm should not have terminated when it did since it has received at least the same number of brushes by the end of the parallel brushing algorithm as it had at time i in the firing sequence provided for use in the sequential brushing algorithm. Thus, we have a contradiction. So, $b(G) = pb(G)$. □

Definition 2.2.5. *A graph G can be* ***continually cleaned in K steps*** *using the parallel cleaning process beginning from configuration ω_0 if every for every step, t, ω_t can serve as the initial configuration of a parallel cleaning process that will clean the graph in K steps.*

Definition 2.2.6. *The* ***continual parallel brush number****, $cpb(G)$, of a graph G is the minimum number of brushes needed to continually clean G using a parallel cleaning process.*

Theorem 2.2.7. *(Theorem 4.1 in [11]) For any cycle C_n with $n \geq 2$,*

$$cpb(C_n) = \begin{cases} 2 \text{ if } n \text{ is even} \\ 3 \text{ if } n = 3 \\ 4 \text{ otherwise} \end{cases}$$

Theorem 2.2.8. *(Theorem 4.4 in [11]) For any tree T, $cpb(T) = pb(T) = b(T)$.*

Chapter 3

Parallel Diffusion

In this chapter, we attempt to count configurations that exist inside the periods of their respective configuration sequences in Parallel Diffusion. This is however a problematic concept because in the original Parallel Diffusion paper by Duffy et al. [9], it is noted that a uniform addition of a constant k to the stack sizes of a configuration results in no change in the flow of chips (Lemma 3.1.1). This implies that there exist infinite configurations which exist inside the periods of their respective configuration sequences. Due to this fact, we will restrict our search by fixing a single vertex at 0 chips, understanding that the configurations we are now interested in counting are merely representatives from a set of infinitely many configurations which can be created by adding a constant to each stack size. We will show that this restriction is sufficient to reduce the number of configurations in which we are interested to a finite number, and additionally, we calculate the number of such representatives that exist on all stars, paths, and complete graphs. These results are stated in Theorem 3.3.8, Corollary 3.4.8, and Theorem 3.5.1, respectively. In addition to Lemma 3.1.1 from Duffy et al. [9], the work in this chapter relies on Long and Narayanan's result that every configuration eventually has period 1 or 2 [14]. With only these two publications on the subject, this thesis has the privilege of establishing not only the basic corollaries of Long and Narayanan's periodicity result, but also the much less obvious counting arguments regarding configurations which exist inside of their respective periods.

3.1 Basic Definitions and Lemmas

We begin with some basic definitions and lemmas regarding Parallel Diffusion. Recall the definition of graph orientation (Definition 1.1.12) as we will be counting graph orientations in this section. The following lemma is stated in [9], but is not proven. We present a proof here.

Lemma 3.1.1. *(from [9]) Let C and D be configurations on a graph G. Let k be an integer. Suppose that for all $v \in V(G)$, $|v|^C = |v|^D + k$. Then for all t, $|v|^C_t = |v|^D_t + k$.*

Proof. We will prove this by induction on t. Let C and D be configurations on a graph G. Let k be an integer. Suppose that for all $v \in V(G)$, $|v|^C = |v|^D + k$. So for all $u, v \in V(G)$, $|u|^C > |v|^C$ if and only if $|u|^C = |u|^D + k > |v|^D + k = |v|^C$. Thus after the first firing, we get that $|u|_1^C = |u|_1^D + k$ for all $u \in V(G)$. We will consider this as the base case of an induction. Our induction hypothesis is that $|v|_t^C = |v|_t^D + k$ for all $v \in V(G)$. So for all $u, v \in V(G)$, $|u|_t^C > |v|_t^C$ if and only if $|u|_t^C = |u|_t^D + k > |v|_t^D + k = |v|_t^C$. Thus, after the firing at step t, we get that $|u|_{t+1}^C = |u|_{t+1}^D + k$ for all $u \in V(G)$. Thus, we conclude that for all t, $|v|_t^C = |v|_t^D + k$. □

Definition 3.1.2. *Let* ***L(G)*** $= \{Seq(C) : C$ *is a configuration on* $G\}$.

Since the set of integers is infinite, on any graph G, $L(G)$ is an infinite set. Recall by Theorem 1.2.1, that the period length of any configuration sequence in Parallel Diffusion is 1 or 2.

Definition 3.1.3. *Let* $\boldsymbol{\overline{Seq(C_0)}}$ *be the singleton or ordered pair of configurations contained within the period of a configuration sequence* $Seq(C_0)$*. If* $Seq(C_0)$ *has period 2, define the first element of the ordered pair* $\overline{Seq(C_0)}$ *to be the one which occurs first in the configuration sequence.*

Definition 3.1.4. *Let* C *be a configuration on a graph* G*. Let* $\boldsymbol{C+k}$ *be the configuration created by adding an integer* k *to every stack size in the configuration* C*. Two configuration sequences,* $Seq(C)$ *and* $Seq(D)$ *in* $L(G)$*, are* ***equivalent*** *if* $\overline{Seq(C+k)} = \overline{Seq(D)}$ *for some integer* k*. For all configurations* C *and all integers* k*, we say that* C *and* $C+k$ *are* ***equivalent****. Let* $\overline{L}(G) = \{\overline{Seq(C)} : C$ *is a configuration on* $G\}$.

In Sections 3.3 and 3.5, we determine the sizes of the largest subsets of $\overline{L}(P_n)$ and $\overline{L}(K_{1,n})$ such that no two elements are equivalent.

We see an example of equivalent configuration sequences in Figure 3.1. We see an example of $\overline{L}(G)$ and a largest subset of $\overline{L}(G)$ such that no two elements are equivalent in Example 3.1.1.

Example 3.1.1. P_2

$\overline{L}(P_2) = \{\dots, \{(0,0)\}, \{1,1\}, \dots\} \bigcup \{\dots, \{(0,1),(1,0)\}, \{(1,0),(0,1)\}, \{(1,2),(2,1)\}, \dots\}$

A largest subset of $\overline{L}(P_2)$ *such that no two elements are equivalent:* $\{\{(0,0)\},\{(0,1),(1,0)\},\{(1,0),(0,1)\}\}$

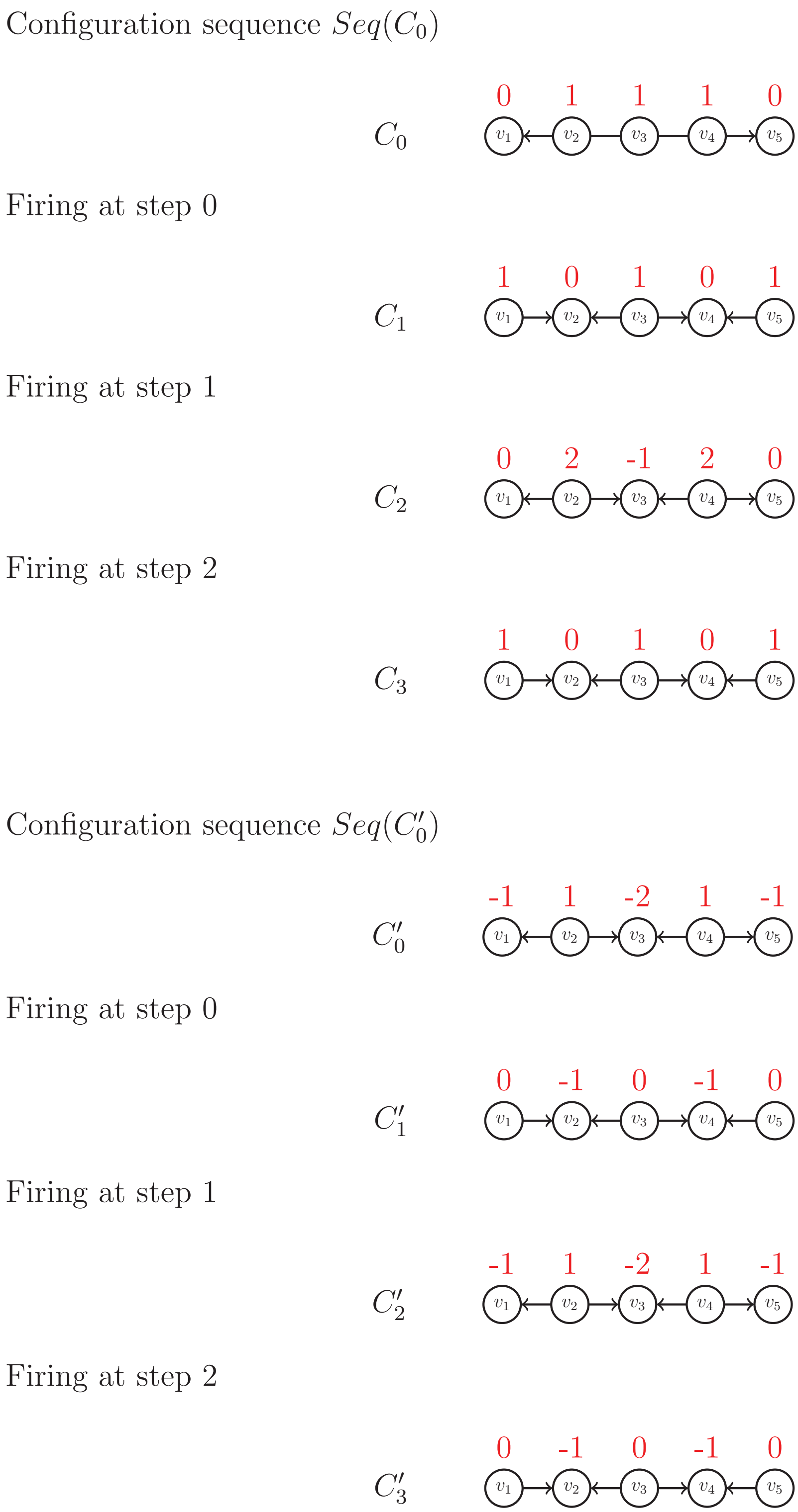


Figure 3.1: Two equivalent configuration sequences.

Lemma 3.1.5. *Equivalence among configurations is an equivalence relation.*

Proof. Reflexivity holds by setting $k = 0$. Now let C and $D = C + k$, $k \in \mathbb{Z}$, be two configurations. We know that $D = C + k$ and $C = D + (-k)$. Thus, symmetry holds. For transitivity, let C, D, and E be configurations. Suppose $D = C + k$ and $E = D + j$, where $j, k \in \mathbb{Z}$. Then, $E = C + k + j$. Thus, transitivity holds. □

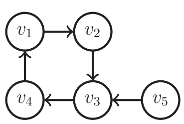
Lemma 3.1.6. *Equivalence among configuration sequences is an equivalence relation.*

Proof. Reflexivity holds by setting $k = 0$. Symmetry holds because if $\overline{Seq}(C + k) = \overline{Seq}(D)$ for some integer k, then $\overline{Seq}(D + (-k)) = \overline{Seq}(C)$. Transitivity holds because if $\overline{Seq}(C + k) = \overline{Seq}(D)$ for some integer k, and $\overline{Seq}(D + j) = \overline{Seq}(E)$ for some integer j, then $\overline{Seq}(C + k + j) = \overline{Seq}(E)$. □

Definition 3.1.7. *A graph orientation R on a graph G is **admissible** if there exists a configuration C on G such that R is induced by C. Conversely, a graph orientation R' on a graph G' is **inadmissible** if there is no configuration which induces R'.*

An example of an inadmissible graph orientation and an admissible graph orientation is given in Figure 3.2. The first graph orientation in Figure 3.2 is inadmissible because of the existence of a directed cycle. The directed cycle implies that any configuration inducing this orientation must have v_1 both richer and poorer than v_2.

Inadmissible



Admissible

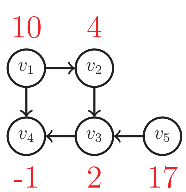


Figure 3.2: Inadmissible and admissible graph orientations with a configuration that induces the admissible graph orientation.

Definition 3.1.8. *A configuration D on a graph G is a **period configuration** if D is in $\overline{Seq}(C)$ for some configuration C. A configuration D on a graph G is a p_2-**configuration** if D is in $\overline{Seq}(C)$ for some configuration C and $\overline{Seq}(C)$ has exactly 2 elements. A configuration D on a graph G is a **fixed configuration** if D is in $\overline{Seq}(C)$ for some configuration C and $\overline{Seq}(C)$ has exactly 1 element.*

Definition 3.1.9. *A* ***period orientation*** *is an admissible graph orientation that is induced by a period configuration. A* p_2***-orientation*** *is an admissible graph orientation that is induced by a* p_2*-configuration. A* ***fixed orientation*** *is an admissible graph orientation that is induced by a fixed configuration.*

Note that the set of all period orientations on G is a subset of the set of all admissible graph orientations on G, and the set of all admissible graph orientations on G is a subset of the set of all graph orientations on G.

The orientation induced by a period configuration must be a period orientation, but not every configuration that induces a period orientation is itself a period configuration. In Figure 3.3, we see an example of a period orientation that can be induced by both a period configuration and a non-period configuration.

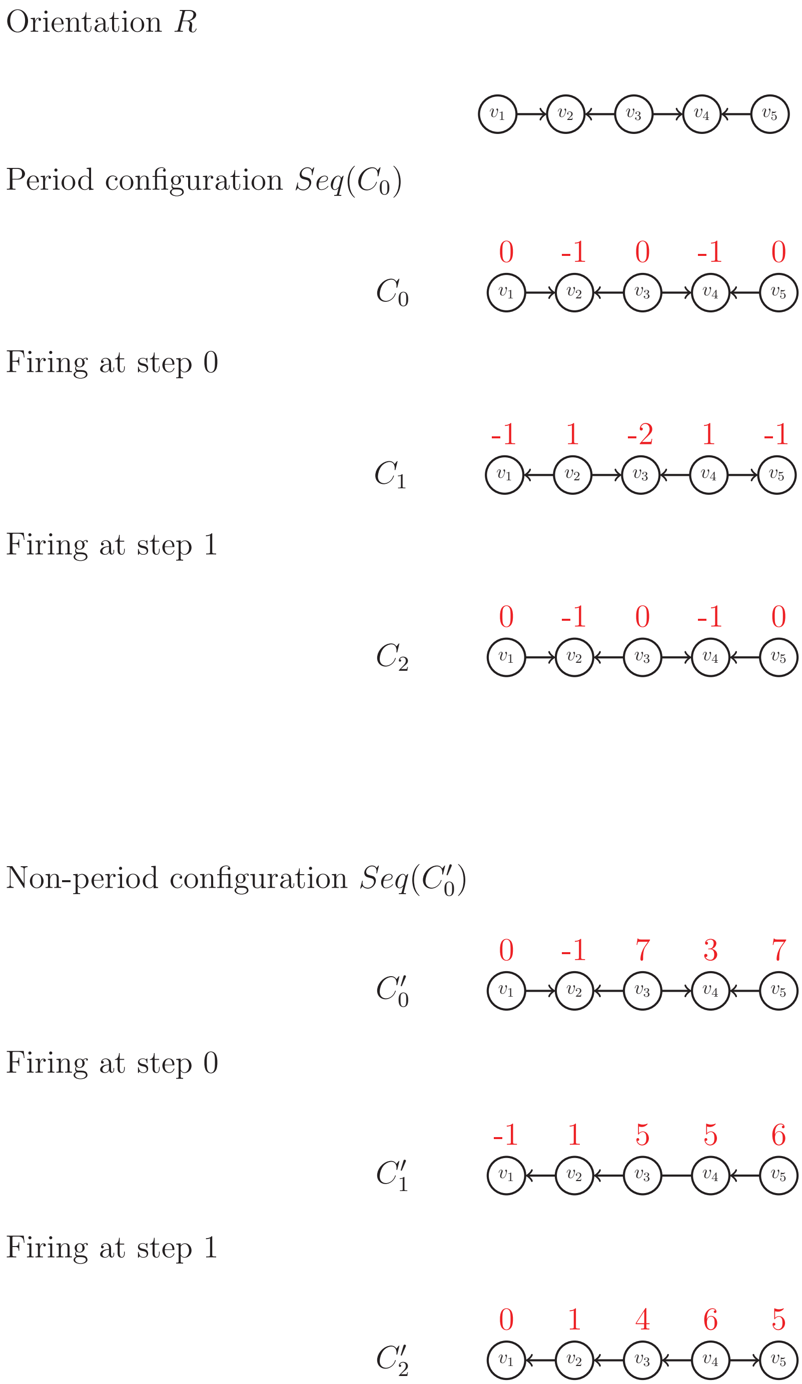


Figure 3.3: A period orientation R that can be induced by both a period configuration and a non-period configuration

Definition 3.1.10. *Let G be a graph and let R be a graph orientation on G. The set of vertices $S = \{v_0, v_1, \ldots, v_k\} \subseteq V(G)$ is a **flat-directed cycle** in R if for each i (indices taken modulo $k+1$), the edge v_iv_{i+1} exists and it is either directed toward*

v_{i+1} or is flat, and there is at least one directed edge.

Definition 3.1.11. *Let G be a graph and let R be a graph orientation on G. Let $x, y \in V(G)$. The **flat relation** $\approx$ relates x and y if and only if either $x = y$ or there exists a path P in G with endpoints x and y such that every edge in P is flat in R.*

This is the same as viewing all flat related vertices as having equal stack sizes in a given configuration on a connected graph.

For our next lemma, we will require a definition from West [18].

Definition 3.1.12. *A **walk** is a list $v_1, e_1, v_2, \ldots, e_k, v_k + 1$ of vertices and edges such that for $1 \leq i \leq k$, the edge e_i has endpoints v_i and v_{i+1}.*

Lemma 3.1.13. *The flat relation is an equivalence relation.*

Proof. We know reflexivity holds since every vertex is defined to be related to itself. We know symmetry holds since if x and y are connected by a path in which every edge is flat, then y and x are also connected by a path in which every edge is flat. We know transitivity holds since if x and y are connected by a path in which every edge is flat and so are y and z, then x and z are connected by a walk in which every edge is flat. Since a walk from x to z exists in which every edge is flat, there must exist some shortest walk from x to z in which every edge is flat. This shortest walk must be a path because otherwise, some edge or vertex is listed more than once, and thus it is not the shortest walk.

□

Lemma 3.1.14. *Let x, y be flat related vertices. If the edge xy exists in an admissible graph orientation, then xy is flat.*

Proof. Let G be a graph and let R be an admissible graph orientation on G. Suppose x and y are flat related in R. If $x = y$, then the edge xy does not exist (this thesis only looks at simple graphs unless otherwise stated). So there exists a path P in G with endpoints x and y such that every edge in P is flat in R. Thus, in any configuration which induces R, every vertex in P must have the same stack size. Therefore, if xy exists in G, then xy is flat in R. □

Lemma 3.1.15. *For all graphs G, a graph orientation of G contains a flat-directed cycle if and only if it is inadmissible.*

Proof. (⇒) Suppose a graph orientation R on a graph G contains a flat-directed cycle, $S = (v_0, v_1, ..., v_k)$. Since S being a flat-directed cycle implies the existence of at least one directed edge in S, any configuration C which induces R must be such that $|v_i|^C < |v_{i+1}|^C$ and $|v_{i+1}|^C < |v_i|^C$ for some integer i. Thus, R is inadmissible.

(⇐) We will prove the contrapositive. So suppose that R is a graph orientation on a graph G which does not contain a flat-directed cycle. We will reach that R is admissible. We will create a configuration which induces R to show that R is admissible. Since R does not contain a flat-directed cycle, there must exist some flat equivalence class X_1 such that no vertex in X is the tail of an edge in R. Assign 1 chip to each vertex in X_1. If we were to remove X_1 from the orientation, we would be left with a suborientation, R', in which no flat-directed cycles exist. So there must exist some flat equivalence class X_2 such that no vertex in X_2 is the tail of an edge in R. We continue assigning increasing large stack sizes to these equivalence classes until every vertex has been assigned a stack size. A configuration which induces R will result. □

Lemma 3.1.16. *Let G be a graph. Up to equivalence, the only fixed configuration on G is the one in which every vertex has 0 chips.*

Proof. Let C be a fixed configuration on a graph G. We know that, by convention, at least one vertex in C has exactly 0 chips. Let R be the graph orientation induced by C. Suppose, by way of contradiction, that an edge in R is directed. Without loss of generality, let that directed edge be $u \to v$.

Case 1: There exists a directed cycle D in R.

By Lemma 3.1.15, since all directed cycles are also flat-directed cycles, R is inadmissible. Thus there does not exist a configuration which induces R.

Case 2: There does not exist a directed cycle in R.

Let P be a maximal directed path in R, containing $u \to v$. Let x be the endpoint of this path which is a head of some directed edge but not the tail of any directed edge in P. Note that x may be v. This vertex, x, is receiving a chip in the initial firing, but it is not sending a chip. Therefore, the stack size of x will change. Thus, C is not a fixed configuration. This is a contradiction.

We can conclude that no directed edges exist in R and thus, all stack sizes in C are equal to 0.

□

3.2 Period Orientations

In this section, we take a look at period orientations on paths, stars and complete graphs. In each case, we develop a method to count the number of period orientations on arbitrarily large graphs. These results on period orientations will prove useful when we count the number of configurations which exist inside the period of paths, complete graphs, and stars, in Sections 3.3, 3.4, and 3.5, respectively. We begin this section with some results that will apply to all graphs.

Theorem 3.2.1. *If R is a graph orientation on a graph G, and R does not contain a vertex with in-degree 0, then R is inadmissible.*

Proof. Let R be a graph orientation on a graph G such that R does not contain a vertex with in-degree 0. We will create a directed cycle using the following algorithm:

1. $A = \emptyset$.
2. Choose a vertex, v_0 and add it to A. Set $i = 1$.
3. Select an edge in which v_{i-1} is the head, and call the tail v_i.
4. If $v_i \notin A$, add v_i to A, $i = i + 1$, back to 3. Else, end.

Since G only has finitely many vertices, this algorithm will terminate and thus, locate a directed cycle. Since R contains a directed cycle, by Lemma 3.1.15, R is inadmissible.

□

3.2.1 Complete Graphs

We now approach each of our graph families individually, beginning with the complete graphs, K_n. In Theorem 3.2.5, we show a method of counting all period orientations on K_n, $n \geq 1$.

Claim 3.2.2. *Let R be an orientation of a complete graph, K_n, $n \geq 1$. If no flat-directed cycles exist in R, then transitivity of directed edges holds. That is, $x \to y$ and $y \to z$ implies $x \to z$.*

Proof. This result is trivial for $n \leq 2$, since fewer than two edges exist. So suppose $n \geq 3$. Let R be an orientation of a complete graph, K_n. Suppose R contains no flat-directed cycles. Suppose that $x, y, z \in V(K_n)$ and that, in R, $x \to y$ and $y \to z$. The edge xz exists since R is an orientation of the complete graph. The edge xz cannot be flat or directed $z \to x$ because this would imply the existence of a flat-directed cycle. Thus, in R, $x \to z$. □

Claim 3.2.3. *Let R be an orientation of a complete graph K_n, $n \geq 1$. If no flat-directed cycles exist in R, then given two flat equivalence classes X and Y in R with $X \neq Y$, all edges of the form xy with $x \in X$ and $y \in Y$ are assigned the same direction.*

Proof. This result is trivial if $n \leq 2$, since fewer than two edges exist. So suppose $n \geq 3$. Let R be an orientation of a complete graph K_n. Suppose no flat-directed cycles exist in R. Let X and Y be two flat equivalence classes in R with $X \neq Y$.
Case 1: Both X and Y contain exactly one vertex. In this case, only one edge exists with an endpoint in X and an endpoint in Y. So trivially, all edges of the form xy with $x \in X$ and $y \in Y$ are assigned the same direction.
Case 2: Either X or Y contains at least two vertices. Without loss of generality, suppose X contains at least two vertices. Let $x_1, x_2 \in X$ and $y \in Y$. The edges x_1y, x_1x_2 and x_2y exist since R is an orientation of a complete graph. Since x_1 and x_2 are elements of the same flat equivalence class in R, the edge x_1x_2 is flat in R. Since $X \neq Y$, we know that the edges x_1y and x_2y are not flat. Suppose, without loss of generality, that $x_1 \to y$ in R. Now suppose, by contradiction, that $y \to x_2$ in R. This would create a flat-directed cycle x_1yx_2 as shown in Figure 3.4. This is a contradiction. Thus, $x_2 \to y$ in R. So, all edges of the form xy with $x \in X$ and $y \in Y$ are assigned the same direction.

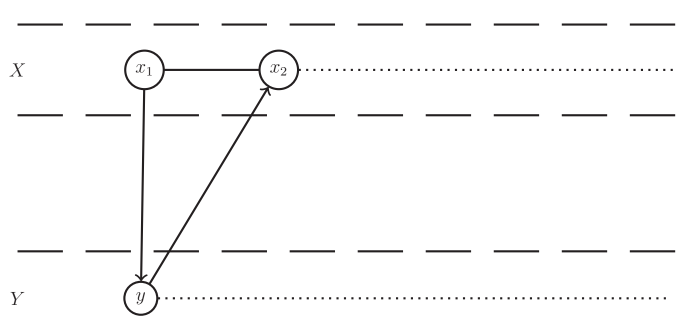


Figure 3.4: Flat-directed cycle x_1yx_2.

□

Lemma 3.2.4. *A graph orientation R of K_n, $n \geq 1$, is a period orientation if and only if R has no flat-directed cycles.*

Proof. ($\Rightarrow$) Suppose R is a period orientation. By definition, all period orientations are admissible. By Lemma 3.1.15, if R is admissible, then R has no flat-directed cycles.

($\Leftarrow$) Now suppose R has no flat-directed cycles. Trivially, our result holds for $n = 1$ and $n = 2$ since those complete graphs do not contain any cycles. So we will suppose $n \geq 3$. Let x, y, $z \in V(K_n)$.

By Claim 3.2.2, transitivity holds. By Claim 3.2.3, if X and Y are two different flat equivalence classes, each edge with endpoints in both X and Y must be assigned the same direction. This shows that the flat equivalence classes form a linear ordering based on their directed edges. We number the equivalence classes from 1 to j with X_1 containing vertices with in-degree 0 and X_j containing vertices with out-degree 0, where j is the number of equivalence classes.

Let C be the configuration in which n chips are assigned to each of the vertices of X_1, $n-1$ chips to the vertices of X_2, $\ldots$, and $n-j+1$ chips to the vertices of X_j.

We will show that for each pair of vertices u and v, if $|v|_0 < |u|_0$, then $|v|_1 > |u|_1$ and also if $|v|_0 = |u|_0$, then $|v|_1 = |u|_1$.

Let $u \in X_a$ and $v \in X_b$ with $a < b$. So, $|u|_0 - |v|_0 = b - a$. In the first firing, u gives a chip to every vertex that v gives a chip to, and v receives a chip from every vertex that u receives a chip from. However, u gives an additional chip to every vertex in X_b, X_{b-1}, $\ldots$, X_{a+1}. Note that this list of sets contains at least $b - a$ vertices. Thus, since u will lose at least $b - a$ chips to the vertices in $X_b \cup X_{b-1} \cup \cdots \cup X_{a+1}$ and v will gain at least $b - a$ chips from the vertices in $X_b \cup X_{b-1} \cup \cdots \cup X_{a+1}$, this results in $|v|_1 - |u|_1 \geq b - a$.

Since it is true that every vertex within the same equivalence class will have the same number of chips following the first firing (having given to and received from the same number of vertices), we can conclude that X_j will be the richest equivalence class following the first firing, X_1 will be the poorest and the others will follow in order. The next firing will revert the vertices back to their original stack size since they will send to every vertex they previously received from, they will receive from every vertex they previously sent to, and all flat edges have been maintained. Thus, we are inside the period. So, if R has no flat-directed cycles, then R is a period orientation.

□

Can this lemma be extended to all graphs? In Figure 3.5, we see an example of a graph for which this lemma does not hold.

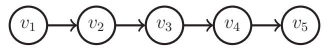


Figure 3.5: Graph orientation on P_5.

Obviously, no graph orientation on a path can have a flat-directed cycle. In this example, the initial configuration is such that the stack size of both v_3 and v_4 will remain the same following the initial firing. However, in order for this orientation R to exist inside the period, there must exist a configuration C which induces R and exists within the period of some configuration sequence. Since the edge v_3v_4 is directed from v_3 to v_4 in both step 0 and 1, we have a contradiction by Corollary 1.2.2.

In order to count the number of period orientations that exist on a complete graph, we must partition the n vertices into k non-empty distinct sets, where each set induces a clique of flat edges (a flat equivalence class). How many ways can we partition these vertices?

Theorem 3.2.5. *Let $R(K_n)$ be the number of period orientations that exist on K_n. For $n \geq 2$,*

$$R(K_n) = \sum_{i=1}^{n} \binom{n}{i} \cdot R(K_{n-i})$$

Proof. Let R be a period orientation on K_n. Let i be the number of vertices in the equivalence class of vertices with in-degree 0, X_1, in the orientation R. If we removed all of these vertices from our orientation, we must be left with a period orientation on $n-i$ vertices by Lemma 3.2.4. There are $\binom{n}{i}$ ways to choose the elements of X_1. So the number of ways for there to be i elements in X_1 in R is $\binom{n}{i} \cdot R(K_{n-i})$. There are n possible distinct cardinalities of X_1: the integers from 1 to n. This creates a list of n mutually exclusive and exhaustive cases based on the cardinality of X_1. Summing over each of these possible cardinalities of X_1, we get

$$R(K_n) = \sum_{i=1}^{n} \binom{n}{i} \cdot R(K_{n-i}).$$

□

In the OEIS [16], this sequence of numbers: 1, 3, 13, 75, 541, 4683, ... is A000670, the number of ways n competitors can rank in a competition, allowing for the possibility of ties (also known as Fubini numbers or ordered Bell numbers). This is an equivalent question to the one we are asking if we suppose that the higher ranking competitors are richer than the lower ranking competitors. The n^{th} term of this sequence is approximately equal to $\frac{n!}{2(\ln 2)^{n+1}}$ [1].

The number of ways to distribute the n vertices of a graph into k non-empty equivalence classes is given by the Stirling numbers of the second kind, $S(n, k)$ [5]. Thus, we get the following corollary.

Corollary 3.2.6. *The number of period orientations on K_n, $n \geq 2$, is given by*

$$R(K_n) = \sum_{k=1}^{n} S(n, k)k!$$

Here, k represents the number of equivalence classes. So, the first term is the one in which every edge is flat and the final term is the one in which no flat edges appear. The $k!$ term is a factor to account for the number of ways that these equivalence classes can be ordered.

3.2.2 Paths

We now approach the problem of counting all of the p_2-orientations on a path. We will draw our paths along a horizontal axis so as to allow for the terms "left" and "right" to have an obvious meaning. We will label the vertices from **right to left** with the rightmost vertex labelled v_1 and we will fix v_1 at zero chips. By Theorem 1.2.1, we know every configuration is eventually periodic with either period 1 or 2. Since we are fixing v_1 at zero chips, by Lemma 3.1.16, the only possible period 1 configuration on any path is the one in which every vertex has zero chips. So, we will restrict our view to only counting p_2-orientations as opposed to all period orientations. We will make frequent use of Corollary 1.2.2 in justifying the flow of chips inside the period.

Definition 3.2.7. *On a path drawn along a horizontal axis, a **left** edge is a directed edge in which the head is to the left of the tail. A **right** edge is a directed edge in which the head is to the right of the tail.*

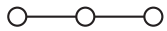
Definition 3.2.8. *When referring to edges contained within a path defined on a horizontal axis, two edges **agree** if they are either both right edges or both left edges. Two edges **disagree** if one is right and the other is left.*

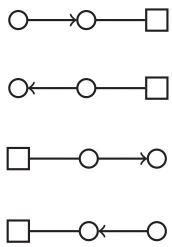
Definition 3.2.9. *An **alternating arrow orientation** is a path orientation in which every pair of adjacent edges disagree.*

Note that this means an alternating arrow orientation cannot contain any flat edges.

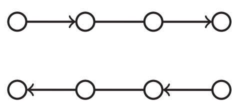
In order to find all of the p_2-orientations on P_n, we begin by characterizing when a path orientation is a p_2-orientation.

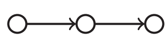
Theorem 3.2.10. *A path orientation R is a p_2-orientation if and only if (a), (b), (c), and (d) are true.*

(a)  *R does not contain this suborientation.*

(b)  *R does not contain any of these suborientations.*

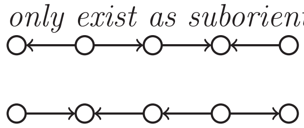
(let a square represent a leaf)

(c)  *R does not contain either of these suborientations.*

(d) 

and

only exist as suborientations of R within the larger suborientations



We express the proof with a series of lemmas. In particular, we prove the necessary condition with Lemmas 3.2.11–3.2.14, and we prove the sufficient condition with Lemma 3.2.15.

Lemma 3.2.11. *Case (a) in Theorem 3.2.10: No p_2-configuration on a path contains two incident flat edges.*

Proof. Suppose, by contradiction, that it were possible to have two flat edges adjacent to each other. Call these edges e_k and e_{k+1}, and call the vertices v_k, v_{k+1}, and v_{k+2}. So, the subpath in question is $X = v_{k+2}e_{k+1}v_{k+1}e_kv_k$. Since the period is two, we know that some edge in the graph must not be flat. Without loss of generality, suppose that the edge immediately to the right of X, e_{k-1}, is oriented left or right. This would result in v_{k+1} maintaining its number of chips in step 1, but in step 2, it is now adjacent to a vertex that either increased or decreased its stack size in the last step. Thus, the flat edge between v_{k+1} and v_k has not been maintained. So by Corollary 1.2.2, the orientation is not inside the period. This is a contradiction. □

Lemma 3.2.12. *Case (b) in Theorem 3.2.10: No p_2-configuration on a path induces an orientation in which a flat edge is incident with a leaf.*

Proof. Suppose, by contradiction, that it were possible for an edge incident with a leaf to be flat in an orientation induced by a p_2-configuration, C. Call the edge e_1 and its endpoints v_1 and v_2. Initially, v_1 and v_2 have equal stack sizes. By Lemma 3.2.11, we know that e_2, the edge adjacent with e_1, is not flat. So, v_2 either gains or loses a chip in the initial firing, while v_1's stack size remains unchanged. So, in the subsequent firing, a chip will move across e_1, indicating that it is not flat. This contradicts our supposition that C is a p_2-configuration. □

Lemma 3.2.13. *Case (c) in Theorem 3.2.10: Every flat edge in an orientation induced by a p_2-configuration, C, on a path is incident with two edges: one right and one left.*

Proof. Suppose, by contradiction, that it were possible for a flat edge to exist without being adjacent to one left and one right edge. Call this flat edge e_k and its endpoints v_k and v_{k+1}. In the initial firing, no chips will move across e_k since it is flat. In step 1, the same must be true by Corollary 1.2.2.

Case 1: e_k is incident with a leaf. By Lemma 3.2.12, we know that no flat edge can be incident with a leaf. This is a contradiction.

Case 2: At least one edge adjacent to e_k is flat. By Lemma 3.2.11, this is impossible. This is a contradiction.

Case 3: e_k is adjacent to two left edges. Then $|v_k|_1 = |v_k|_0 + 1$ and $|v_{k+1}|_1 = |v_k|_0 - 1$. So, the stack sizes of v_k and v_{k+1} are not equal at step 1. This is a contradiction.

Case 4: e_k is adjacent to two right edges. Then $|v_k|_1 = |v_k|_0 - 1$ and $|v_{k+1}|_1 = |v_k|_0 + 1$. So, the stack sizes of v_k and v_{k+1} are not equal at step 1. This is a contradiction.

Thus, every flat edge in a p_2-configuration on a path is incident with two edges: one right and one left. □

Lemma 3.2.14. *Case (d) in Theorem 3.2.10: Let R' be a suborientation of the orientation, R, induced by a p_2-configuration on a path, P_n, $n \geq 3$. If R' consists* *of three vertices and two right edges, then R' must be incident with two left edges in R. Conversely, if R' consists of three vertices and two left edges, then R' must be incident with two right edges in R.*

Proof. Let C be a p_2-configuration on P_n. Let R be the orientation induced by C. Let R' be the suborientation of R on the subpath $P'_n = v_{k+1}e_kv_ke_{k-1}v_{k-1}$ of P_n. Suppose, without loss of generality, that R' contains two right edges. In the initial firing, v_k gives and receives a single chip, maintaining its stack size. Initially, we have $|v_{k+1}|_0 > |v_k|_0 > |v_{k-1}|_0$. At step 1, these inequalities must reverse since we are already inside the period, by Corollary 1.2.2. So, since v_k maintains its stack size, v_{k+1} must lose at least 2 chips in step 0 so as to go from richer than v_k in step 0 to poorer than v_k in step 1. Also, v_{k-1} must gain at least 2 chips in step 0 so as to go from poorer than v_k in step 0 to richer than v_k in step 1. However, this is only possible if both edges in $E(R) \setminus E(R')$ that are incident with R' are left edges.

□

Lemma 3.2.15. *Any path orientation with no suborientations of the forms outlined in Theorem 3.2.10, is a p_2-orientation.*

Proof. We must now show that any orientation that does not contain one of these mixed graphs as a suborientation is a p_2-orientation. Our method will involve taking an arbitrary orientation R that does not contain any of the suborientations listed in Theorem 3.2.10, and proving that there exists a configuration that both induces R and exists within the period. There are 3 orientations that an edge may have: flat, left, and right. Let C be the configuration on P_n such that, moving from right to left, every vertex v_i has been assigned an initial stack size using the following rule:

$$|v_i|_0 = \begin{cases} |v_{i-1}|_0 + 1 \text{ if edge } v_{i-1}v_i \text{ is directed right.} \\ |v_{i-1}|_0 - 1 \text{ if edge } v_{i-1}v_i \text{ is directed left.} \\ |v_{i-1}|_0 \text{ if edge } v_{i-1}v_i \text{ is flat.} \end{cases}$$

and v_1 has been assigned 0 chips. See Figure 3.6 for an example.

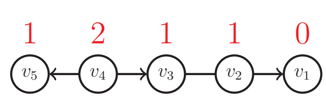


Figure 3.6: Configuration C on P_5.

We will show for every edge v_iv_{i+1}

- if $|v_i|_0 > |v_{i+1}|_0$, then $|v_{i+1}|_1 > |v_i|_1$

  and

- if $|v_i|_0 = |v_{i+1}|_0$, then $|v_i|_1 = |v_{i+1}|_1$.

This will imply that for all $v_i \in V(P_n)$, $|v_i|_0 = |v_i|_2$. Therefore, we can conclude that C exists inside the period and thus, the orientation R induced by C is a period orientation. Finally, since we have supposed that R does not have any pair of adjacent flat edges, we will know by Lemma 3.1.16, that R is a p_2-orientation.

We now inspect an edge $e_j = v_j v_{j+1}$ in R.

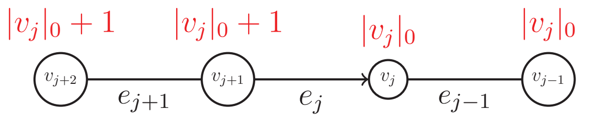
Case 1: The edge $e_j = v_j v_{j+1}$ is flat.

We know that neither v_j nor v_{j+1} is a leaf by Lemma 3.2.12. Our rule shows that v_j and v_{j+1} have both been initially assigned to have the same number of chips. However, in order for this configuration to be a p_2-configuration, we must also have that $|v_j|_1 = |v_{j+1}|_1$, by Corollary 1.2.2. In order to determine this, we must know the stack sizes of v_j and v_{j+1} at step 1. This will depend on the initial orientation of edges $e_{j-1} = v_{j-1}v_j$ and $e_{j+1} = v_{j+1}v_{j+2}$ (both of these edges are known to exist since e_j, being flat, cannot be adjacent to a leaf by Lemma 3.2.12). We know that since e_j is flat, no adjacent edge can be flat by Lemma 3.2.11. Also, e_{j-1} and e_{j+1} cannot be both right or both left by Lemma 3.2.13. So, e_{j-1} and e_{j+1} must disagree. Without loss of generality, suppose e_{j-1} is directed right and e_{j+1} is directed left. So, our rule dictates that $|v_{j-1}|_0 + 1 = |v_j|_0 = |v_{j+1}|_0 = |v_{j+2}|_0 + 1$. Thus, with both v_j and v_{j+1} sending one chip and receiving zero chips at step 0, we have that $|v_j|_1 = |v_{j+1}|_1$.

Case 2: The edge $e_j = v_j v_{j+1}$ is directed.

Suppose, without loss of generality, that e_j is directed right. Our rule shows that $|v_{j+1}|_0 = |v_j|_0 + 1$. In order for this configuration to be a p_2-configuration, we must have that $|v_{j+1}|_1 < |v_j|_1$, by Corollary 1.2.2. In order to determine this, we must know the stack sizes of v_j and v_{j+1} at step 1. This will depend on the initial orientation of edges $e_{j-1} = v_{j-1}v_j$ and $e_{j+1} = v_{j+1}v_{j+2}$. Note that either e_{j-1} or e_{j+1} may not exist depending on if either v_j or v_{j+1} is a leaf. However, the absence of either of these edges has the same effect on the stack size of the incident vertices as a flat edge would. We consider the possible orientations of e_{j-1} and e_{j+1}.

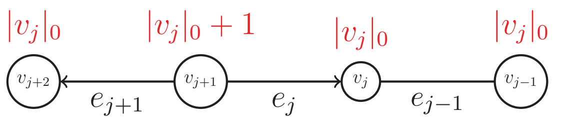
(i) *Both e_{j-1} and e_{j+1} are flat.*



So,

$$
\begin{aligned}
&|v_{j+1}|_1 = |v_{j+1}|_0 - 1 \text{ and } |v_j|_1 = |v_j|_0 + 1 \\
\text{Thus, } &|v_{j+1}|_1 = (|v_j|_0 + 1) - 1 = |v_j|_0 < |v_j|_0 + 1 = |v_j|_1 \\
\text{So } &|v_{j+1}|_1 < |v_j|_1.
\end{aligned}
$$

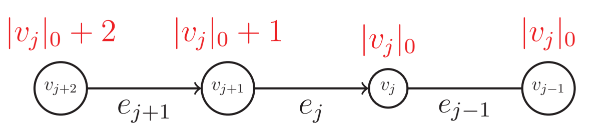
(ii) *e_{j-1} is flat and e_{j+1} is directed left.*



So

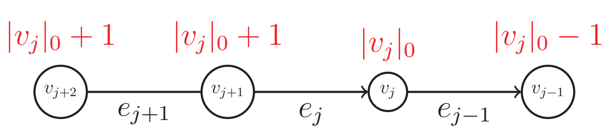
$$
\begin{aligned}
&|v_{j+1}|_1 = |v_{j+1}|_0 - 2 \text{ and } |v_j|_1 = |v_j|_0 + 1 \\
\text{Thus, } &|v_{j+1}|_1 = (|v_j|_0 + 1) - 2 = |v_j|_0 - 1 < |v_j|_0 + 1 = |v_j|_1 \\
\text{So } &|v_{j+1}|_1 < |v_j|_1.
\end{aligned}
$$

(iii) *e_{j-1} is flat and e_{j+1} is directed right.*



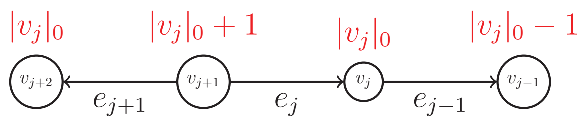
This suborientation cannot exist by assumption.

(iv) *e_{j-1} is directed right and e_{j+1} is flat.*



This suborientation cannot exist by assumption.

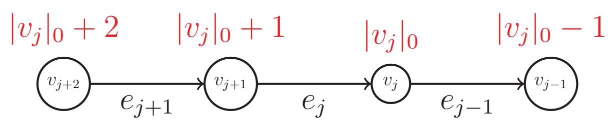
(v) *e_{j-1} is directed right and e_{j+1} is directed left.*



So

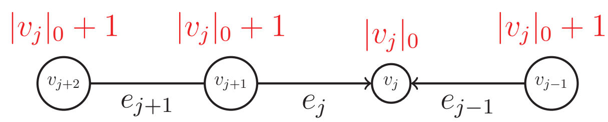
$$|v_{j+1}|_1 = |v_{j+1}|_0 - 2 \text{ and } |v_j|_1 = |v_j|_0$$
$$\text{Thus, } |v_{j+1}|_1 = (|v_j|_0 + 1) - 2 = |v_j|_0 - 1 < |v_j|_0 = |v_j|_1$$
$$\text{So } |v_{j+1}|_1 < |v_j|_1.$$

(vi) *Both e_{j-1} and e_{j+1} are directed right.*



This suborientation cannot exist within the period by Lemma 3.2.14.

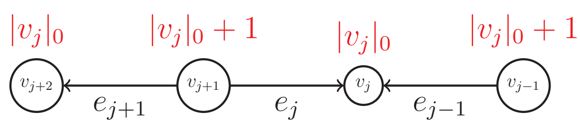
(vii) *e_{j-1} is directed left and e_{j+1} is flat.*



So

$$|v_{j+1}|_1 = |v_{j+1}|_0 - 1 \text{ and } |v_j|_1 = |v_j|_0 + 2$$
$$\text{Thus, } |v_{j+1}|_1 = (|v_j|_0 + 1) - 1 = |v_j|_0 < |v_j|_0 + 2 = |v_j|_1$$
$$\text{So } |v_{j+1}|_1 < |v_j|_1.$$

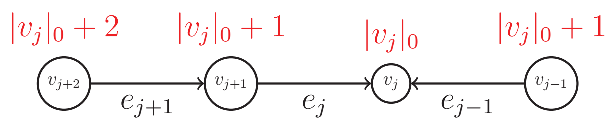
(viii) *Both e_{j-1} and e_{j+1} are directed left.*



So

$$|v_{j+1}|_1 = |v_{j+1}|_0 - 2 \text{ and } |v_j|_1 = |v_j|_0 + 2$$
$$\text{Thus, } |v_{j+1}|_1 = (|v_j|_0 + 1) - 2 = |v_j|_0 - 1 < |v_j|_0 + 2 = |v_j|_1$$
$$\text{So } |v_{j+1}|_1 < |v_j|_1.$$

(ix) *e_{j-1} is directed left and e_{j+1} is directed right.*



So

$$|v_{j+1}|_1 = |v_{j+1}|_0 \text{ and } |v_j|_1 = |v_j|_0 + 2$$
$$\text{Thus, } |v_{j+1}|_1 = |v_j|_0 + 1 < |v_j|_0 + 2 = |v_j|_1$$
$$\text{So } |v_{j+1}|_1 < |v_j|_1$$

So, for all possible graph orientations R, R is a p_2-orientation.

□

Let R_n be the number of p_2-orientations on P_n, $n \geq 1$. Quick calculations show that $R_1 = 0$, $R_2 = 2$, $R_3 = 2$, and $R_4 = 4$ (see Figure 3.7).

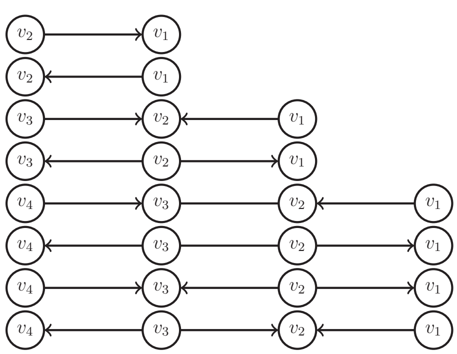


Figure 3.7: Every p_2-orientation of P_2, P_3, and P_4

Theorem 3.2.16. *The number of p_2-orientations, R_n, on a path P_n, $n \geq 4$, is given by the recurrence relation $R_n = R_{n-1} + 2R_{n-2} - R_{n-4}$ with initial values $R_1 = 0$, $R_2 = 2$, $R_3 = 2$, and $R_4 = 4$.*

Proof. Let R be a p_2-orientation on $P_n = v_1e_1v_2e_2v_3\ldots v_{n-1}e_{n-1}v_n$. There are three cases: e_{n-2} is flat, e_{n-2} agrees with e_{n-3}, or e_{n-2} is neither flat nor agreeing with e_{n-3}. For each of the three mutually exclusive and exhaustive cases, we will determine the number of p_2-orientations of that form which exist. We will then add together the three values to determine the recurrence relation for the number of p_2-orientations that exist on P_n.

Case 1: Suppose that in R, e_{n-2} is flat (see Figure 3.8). We will show that the number of p_2-orientations of P_n in which e_{n-2} is flat is equal to R_{n-2}. Let R' be the induced suborientation of R on $P_{n-2} = v_1e_1v_2\ldots v_{n-3}e_{n-3}v_{n-2}$. We must check that R' is a p_2-orientation by using our criteria from Lemmas 3.2.11 - 3.2.14.

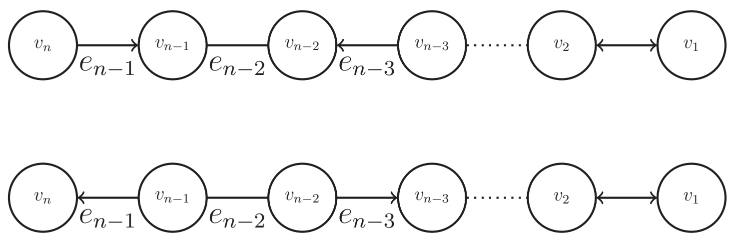


Figure 3.8: Two possible p_2-orientations of P_n with e_{n-2} flat. Note that the orientation of e_1, signified by a double sided arrow, is uncertain in both instances.

Lemma 3.2.11 states the non-existence of adjacent flat edges. Since R is a p_2-orientation, it does not contain adjacent flat edges. Therefore R', being an induced suborientation of R, also does not contain adjacent flat edges.

Lemma 3.2.12 states the non-existence of flat edges incident with a leaf. Since R is a p_2-orientation, it does not contain a flat edge incident with a leaf. The vertex v_{n-2} is a leaf in R' but not in R. However, we know, by Lemma 3.2.13, that e_{n-3} and e_{n-1} disagree. This implies that e_{n-3}, the only edge incident with v_{n-2} in R', is not flat. Therefore, R' does not contain a flat edge incident with a leaf.

Lemma 3.2.13 states that every flat edge must be adjacent to a pair of disagreeing edges. Since R is a p_2-orientation, each flat edge in R is adjacent to disagreeing edges. Since e_{n-3} is not flat, every flat edge in R' is adjacent to the same set of edges in R. Therefore, every flat edge in R' is adjacent to disagreeing edges.

Lemma 3.2.14 states that any edge e adjacent to an edge f with which it agrees must also be adjacent to an edge d with which it disagrees. We know by Lemma 3.2.14, that since R is a p_2-orientation and e_{n-2} is flat, e_{n-3} does not agree with e_{n-4}. So, every pair of adjacent agreeing edges in R' is adjacent to the same set of edges in R. Therefore, in R', every edge e adjacent to an edge f with which it agrees is also adjacent to an edge d with which it disagrees.

By Theorem 3.2.10, we can conclude that R' is a p_2-orientation. Also, R is uniquely determined by R'. That is, given R' and that e_{n-2} is flat, we know by Lemma 3.2.13 that in R, e_{n-1} disagrees with e_{n-3}. So, there is exactly one extension of R' to a p_2-orientation on P_n. Therefore, there exist exactly R_{n-2} different p_2-orientations on P_n in which e_{n-2} is flat.

Case 2: Suppose that in R, e_{n-2} agrees with e_{n-3} (see Figure 3.9). We will show that the number of p_2-orientations of P_n in which e_{n-2} agrees with e_{n-3} is equal to $R_{n-2} - R_{n-4}$. Let R' be the induced suborientation of R on $P_{n-2} = v_1e_1v_2 \dots v_{n-3}e_{n-3}v_{n-2}$. We must check that R' is a p_2-orientation by using our criteria from Lemmas 3.2.11 - 3.2.14.

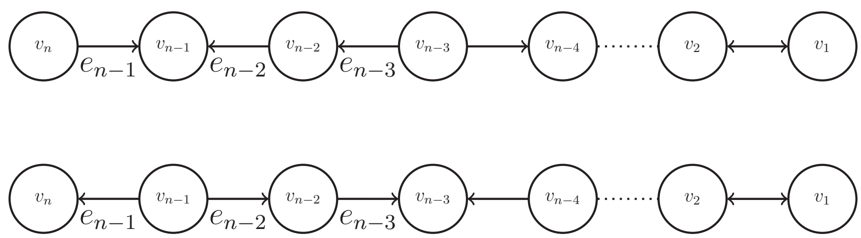


Figure 3.9: Two possible p_2-orientations of P_n in which e_{n-2} agrees with e_{n-3}. Note that the orientation of e_1, signified by a double sided arrow, is uncertain.

Lemma 3.2.11 states the non-existence of adjacent flat edges. Since R is a p_2-orientation, it does not contain adjacent flat edges. Therefore R', being an induced suborientation of R, also does not contain adjacent flat edges.

Lemma 3.2.12 states the non-existence of flat edges incident with a leaf. Since R is a p_2-orientation, it does not contain a flat edge incident with a leaf. The vertex v_{n-2} is a leaf in R' but not in R. However, we know that e_{n-3} and e_{n-2} agree. This implies that e_{n-3}, the only edge incident with v_{n-2} in R', is not flat. Therefore, R' does not contain a flat edge incident with a leaf.

Lemma 3.2.13 states that flat edges must be adjacent to disagreeing edges. Since R is a p_2-orientation, each flat edge in R is adjacent to disagreeing edges. Since e_{n-3} is not flat, every flat edge in R' is adjacent to the same set of edges in R. Therefore, every flat edge in R' is adjacent to disagreeing edges.

Lemma 3.2.14 states that any edge e adjacent to an edge f with which it agrees must also be adjacent to an edge d with which it disagrees. We know by Lemma 3.2.14, that since R is a p_2-orientation and e_{n-2} agrees with e_{n-3}, then e_{n-3} disagrees with e_{n-4}. So, every pair of adjacent agreeing edges in R' is adjacent to the same set of edges in R. Therefore, in R', every edge e adjacent to an edge f with which it agrees is also adjacent to an edge d with which it disagrees.

By Theorem 3.2.10, we can conclude that R' is a p_2-orientation. Also, R is uniquely determined by R'. That is, given R' and that e_{n-2} agrees with e_{n-3}, we know that in R, e_{n-1} disagrees with e_{n-2} by Lemmas 3.2.12 and 3.2.14. By Lemma 3.2.14, the number of p_2-orientations of P_n in which e_{n-2} agrees with e_{n-3} is equal to the number of p_2-orientations of R' in which e_{n-3} disagrees with e_{n-4}. We can determine this value recursively. From Case 1, we can see that the number of p_2-orientations of R' in which e_{n-3} disagrees with e_{n-4} is equal to R_{n-2} minus the number of p_2-orientations of R' in which $e_n - 2$ is flat. Thus we get that there exist $R_{n-2} - R_{n-4}$ p_2-orientations of P_n in which e_{n-2} agrees with e_{n-3}.

Case 3: Suppose that in R, e_{n-2} is neither flat nor agreeing with e_{n-3}. We will show that the number of p_2-orientations in which e_{n-2} is neither flat nor agreeing with e_{n-3} is equal to R_{n-1}. Let R' be the induced suborientation of R on $P_{n-1} = v_1e_1v_2 \dots v_{n-3}e_{n-3}v_{n-2}e_{n-2}v_{n-1}$. We must check that R' is a p_2-orientation by using our criteria from Lemmas 3.2.11 - 3.2.14.

Lemma 3.2.11 states the non-existence of adjacent flat edges. Since R is a p_2-orientation, it does not contain adjacent flat edges. Therefore R', being an induced suborientation of R, also does not contain adjacent flat edges.

Lemma 3.2.12 states the non-existence of flat edges incident with a leaf. Since R is a p_2-orientation, it does not contain a flat edge incident with a leaf. The vertex v_{n-1} is a leaf in R' but not in R. However, we know that e_{n-2}, the only edge incident with v_{n-1} in R', is not flat. Therefore, R' does not contain a flat edge incident with a leaf.

Lemma 3.2.13 states that flat edges must be adjacent to disagreeing edges. Since R is a p_2-orientation, each flat edge in R is adjacent to disagreeing edges. Since e_{n-2} is not flat, every flat edge in R' is adjacent to the same set of edges in R. Therefore, every flat edge in R' is adjacent to disagreeing edges.

Lemma 3.2.14 states that any edge e adjacent to an edge f with which it agrees must also be adjacent to an edge d with which it disagrees. Since e_{n-2} does not agree with e_{n-3}, every pair of adjacent agreeing edges in R' is adjacent to the same set of edges in R. Therefore, in R', every edge e adjacent to an edge f with which it agrees is also adjacent to an edge d with which it disagrees.

By Theorem 3.2.10, we can conclude that R' is a p_2-orientation. Also, R is uniquely determined by R'. That is, given R', we know that in R, e_{n-1} disagrees with e_{n-2}. So, there is exactly one extension of R' to a p_2-orientation on P_n. Therefore, there exist exactly R_{n-1} different p_2-orientations of this form on P_n. Adding together the values from our three cases, we get that $R_n = R_{n-1} + 2R_{n-2} - R_{n-4}$.

□

In the OEIS [16], this sequence: $0, 2, 2, 4, 8, 14, 28, 52, 100, 190, 362...$ is A052535, the sequence with generating sequence is $\frac{(1-x^2)}{(1-x-2x^2+x^4)}$ [16].

In order to find the explicit formula, we begin by finding the characteristic equation for this linear recurrence:

$$\begin{aligned}
R_n &= R_{n-1} + 2R_{n-2} - R_{n-4} \\
R_n - R_{n-1} - 2R_{n-2} + R_{n-4} &= 0 \qquad \text{Let } R_n = x^n \\
x^n - x^{n-1} - 2x^{n-2} + x^{n-4} &= 0 \\
x^{n-4}(x^4 - x^3 - 2x^2 + 1) &= 0 \\
x^4 - x^3 - 2x^2 + 1 &= 0
\end{aligned}$$

The roots of this equation are $\alpha_1 \approx 0.6710$, $\alpha_2 \approx 1.9052$, $\alpha_3 \approx -0.7881 - 0.4014i$, and $\alpha_4 \approx -0.7881 + 0.4014i$. Since these are all of the nonzero roots of the function, our general solution is a linear combination of all four values:

$$R_k = c_1(\alpha_1)^k + c_2(\alpha_2)^k + c_3(\alpha_3)^k + c_4(\alpha_4)^k.$$

In order to solve for c_1, c_2, c_3, and c_4, we must solve a system of four linear equations using the four initial values of our recurrence: $R_1 = 0$, $R_2 = 2$, $R_3 = 2$, and $R_4 = 4$. See [5] for more on solving recurrence relations. Our system of equations is:

$$\begin{aligned}
R_1 &= c_1(\alpha_1) + c_2(\alpha_2) + c_3(\alpha_3) + c_4(\alpha_4) = 0 \\
R_2 &= c_1(\alpha_1)^2 + c_2(\alpha_2)^2 + c_3(\alpha_3)^2 + c_4(\alpha_4)^2 = 2 \\
R_3 &= c_1(\alpha_1)^3 + c_2(\alpha_2)^3 + c_3(\alpha_3)^3 + c_4(\alpha_4)^3 = 2 \\
R_4 &= c_1(\alpha_1)^4 + c_2(\alpha_2)^4 + c_3(\alpha_3)^4 + c_4(\alpha_4)^4 = 4
\end{aligned}$$

Solving this system of equations, we get

$$\begin{aligned}
c_1 &= \frac{2(\alpha_2\alpha_3 + \alpha_2\alpha_4 + \alpha_3\alpha_4 - \alpha_2 - \alpha_3 - \alpha_4 + 2)}{\alpha_1(\alpha_1 - \alpha_2)(\alpha_1 - \alpha_3)(\alpha_1 - \alpha_4)} \\
c_2 &= \frac{2(\alpha_1\alpha_3 + \alpha_1\alpha_4 + \alpha_3\alpha_4 - \alpha_1 - \alpha_3 - \alpha_4 + 2)}{\alpha_2(\alpha_2 - \alpha_1)(\alpha_2 - \alpha_3)(\alpha_2 - \alpha_4)} \\
c_3 &= \frac{2(\alpha_1\alpha_2 + \alpha_2\alpha_4 + \alpha_1\alpha_4 - \alpha_1 - \alpha_2 - \alpha_4 + 2)}{\alpha_3(\alpha_3 - \alpha_1)(\alpha_3 - \alpha_2)(\alpha_3 - \alpha_4)} \\
c_4 &= \frac{2(\alpha_1\alpha_2 + \alpha_1\alpha_3 + \alpha_2\alpha_3 - \alpha_1 - \alpha_2 - \alpha_3 + 2)}{\alpha_4(\alpha_4 - \alpha_1)(\alpha_4 - \alpha_2)(\alpha_4 - \alpha_3)}
\end{aligned}$$

Plugging in our roots, we get

$$\begin{aligned}
c_1 &\approx 0.5797 \\
c_2 &\approx 0.3017 \\
c_3 &\approx 0.5593 - 0.1023i \\
c_4 &\approx 0.5593 + 0.1023i
\end{aligned}$$

So,

$$\begin{aligned}
R_k \approx\ & 0.5797(0.6710)^k \\
&+ 0.3017(1.9052)^k \\
&+ (0.5593 - 0.1023i)(-0.7881 - 0.4014i)^k \\
&+ (0.5593 + 0.1023i)(-0.7881 + 0.4014i)^k.
\end{aligned}$$

The dominating term, out of the four roots, is the one which has the greatest modulus. These moduli are roughly 0.6710, 1.9052, 0.8844, and 0.8844. Thus, the dominant term in the equation

$$\begin{aligned}
R_k \approx\ & 0.5797(0.6710)^k \\
&+ 0.3017(1.9052)^k \\
&+ (0.5593 - 0.1023i)(-0.7881 - 0.4014i)^k \\
&+ (0.5593 + 0.1023i)(-0.7881 + 0.4014i)^k
\end{aligned}$$

is $c_2(\alpha_2)^k \approx (0.3017)(1.9052)^k$.

Corollary 3.2.17. *R_k has an asymptotic value of approximately* $0.3017 \times (1.9052)^k$.

3.2.3 Stars

We now turn our attention to counting the number of period orientations that exist on the star graph $K_{1,n}$, $n \geq 2$.

Theorem 3.2.18. *There are 3 period orientations on a star graph, $K_{1,n}$, $n \geq 2$.*

Proof. **Case 1:** All edges flat.

This graph orientation can be satisfied by placing the same number of chips on each vertex. This is a fixed configuration.

Case 2: All edges directed toward the center.

This graph orientation can be satisfied by placing 0 chips in the center and 1 chip on every other vertex. This is a p_2-configuration.

Case 3: All edges directed from the center.

This graph orientation can be satisfied by placing 1 chip in the center and 0 chips on every other vertex. This is a p_2-configuration.

Case 4: There exists an edge directed toward the center and there exists an edge directed from the center.

In order for both of these edges to reverse at the next step, the center would need to both be richer and poorer than its initial stack size. Thus, there cannot exist both an edge directed toward the center and an edge directed away from the center.

Case 5: There exists both a flat edge and a directed edge.

In order for the directed edge to reverse direction at the next step, the center's stack size would need to either increase or decrease. However, for the flat edge to be maintained, the center's stack size must remain the same. Thus, there cannot exist both a flat edge and a directed edge.

Therefore, there are only 3 period orientations on $K_{1,n}$.

□

3.3 p_2-Configurations on Paths

We have already calculated the number of p_2-orientations that exist on a path. Now, we will calculate, given a p_2-orientation, the number of p_2-configurations that exist. For each vertex on a path, we determine the number of possible stack sizes that that vertex can have. We call this number the *multiplier* of that vertex.

Definition 3.3.1. *Given a graph orientation R on a path P_n, $n \geq 1$, the **multiplier** assigned to a vertex v represents the number of possibilities for the initial stack size of v in a p_2-configuration, supposing that an initial stack size has already been chosen for every vertex to the right of v, and the rightmost vertex v_1 has been chosen to have 0 chips.*

Given an assignment of stack sizes to the vertices $v_1, v_2, \cdots v_{i-1}$, the multiplier of v_i in R is the number of possibilities for the stack size of v_i in a p_2-configuration which induces R. Note that the multiplier of a vertex does not depend on the stack size of any vertex, only the graph orientation. Since we are assuming that we are already inside the period of a configuration with period 2, these calculations can be conducted locally as is shown in Example 3.3.1. That is, the multiplier of a vertex v_k depends only on the orientation of the edges incident to v_k and those incident to v_{k-1}.

We now look at an example:

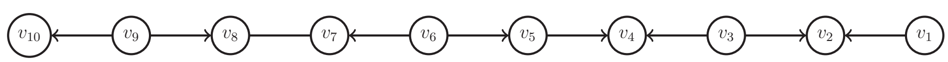


Figure 3.10: P_{10} under orientation R

Example 3.3.1. *We assign P_{10} to have induced orientation R pictured in Figure 3.10. Moving from right to left, we determine the number of possible stack sizes each vertex can take on:*

1. *We fix v_1 at 0 chips by convention. So, v_1 has a multiplier of 1.*

2. *We have that v_2 must have a negative stack size (so as to receive from v_1 in the initial firing) that is large enough to already be in the period. We know that the stack size of v_1 will decrease by 1 in the first step and the stack size of v_2 will increase by 2 in the first step. This means that $|v_2|_0 < |v_1|_0 = 0$ and $|v_2|_0 + 2 > |v_1|_0 - 1 = -1$. So, $0 > |v_2|_0 > -3$. Thus, the two possible values that $|v_2|_0$ can take on are -1 and -2. So, v_2 has a multiplier of 2.*

3. *Given a value for $|v_2|_0$, we calculate the number of possible initial stack sizes that v_3 can take on. We know $|v_3|_0 > |v_2|_0$ and $|v_3|_0 - 2 < |v_2|_0 + 2$. So, we have that $|v_2|_0 < |v_3|_0 < |v_2|_0 + 4$. Thus, the three possible initial stack sizes for v_3 are $|v_2|_0 + 1$, $|v_2|_0 + 2$, and $|v_2|_0 + 3$.*

4. *Given a value for* $|v_3|_0$*, we calculate the number of possible initial stack sizes that* v_4 *can take on. We know* $|v_4|_0 < |v_3|_0$ *and* $|v_4|_0 + 2 > |v_3|_0 - 2$*. So, we have that* $|v_3|_0 > |v_4|_0 > |v_3|_0 - 4$*. Thus, the three possible initial stack sizes for* v_4 *are* $|v_3|_0 - 1$*,* $|v_3|_0 - 2$*, and* $|v_3|_0 - 3$*.*

5. *Given a value for* $|v_4|_0$*, we calculate the number of possible initial stack sizes that* v_5 *can take on. We know* $|v_5|_0 > |v_4|_0$ *and* $|v_5|_0 + 1 - 1 < |v_4|_0 + 2$*. So, we have that* $|v_4|_0 < |v_5|_0 < |v_4|_0 + 2$*. Thus, the only possible initial stack size for* v_5 *is* $|v_4|_0 + 1$*.*

6. *Given a value for* $|v_5|_0$*, we calculate the number of possible initial stack sizes that* v_6 *can take on. We know* $|v_6|_0 > |v_5|_0$ *and* $|v_6|_0 - 2 < |v_5|_0 + 1 - 1$*. So, we have that* $|v_5|_0 + 2 > |v_6|_0 > |v_5|_0$*. Thus, the only possible initial stack size for* v_6 *is* $|v_5|_0 + 1$*.*

7. *Given a value for* $|v_6|_0$*, we calculate the number of possible initial stack sizes that* v_7 *can take on. We know that* $|v_7|_0 < |v_6|_0$ *and* $|v_7|_0 + 1 > |v_6|_0 - 2$*. So, we have that* $|v_6|_0 - 3 < |v_7|_0 < |v_6|_0$*. Thus, the two possible initial stack sizes for* v_7 *are* $|v_6|_0 - 2$ *and* $|v_6|_0 - 1$*.*

8. *Given a value for* $|v_7|_0$*, we calculate the number of possible initial stack sizes that* v_8 *can take on. We know* $|v_8|_0 = |v_7|_0$*. Thus, the only possible initial stack size for* v_8 *is* $|v_7|_0$*.*

9. *Given a value for* $|v_8|_0$*, we calculate the number of possible initial stack sizes that* v_9 *can take on. We know that* $|v_9|_0 > |v_8|_0$ *and* $|v_9| - 2 < |v_8| + 1$*. So, we have that* $|v_8|_0 < |v_9|_0 < |v_8| + 3$*. Thus, the two possible initial stack sizes for* v_9 *are* $|v_8|_0 + 1$ *and* $|v_8|_0 + 2$*.*

10. *Given a value for* $|v_9|_0$*, we calculate the number of possible initial stack sizes that* v_{10} *can take on. We know that* $|v_{10}|_0 < |v_9|_0$ *and* $|v_{10}|_0 + 1 > v_9 - 2$*. So, we have that* $|v_9|_0 - 3 < |v_{10}|_0 < |v_9|_0$*. Thus, the two possible initial stack sizes for* v_{10} *are* $|v_9|_0 - 2$ *and* $|v_9|_0 - 1$*. Thus, the multiplier for* v_{10} *is 2.*

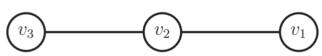
Multiplying all of these possibilities together we get $1\times2\times3\times3\times1\times1\times2\times1\times2\times2 = 144$ p_2*-configurations on this period orientation.*

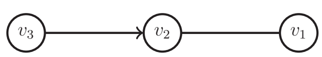
Our goal now is to determine the multiplier for any vertex v_k in any p_2-configuration on a path. In order to determine the multiplier of a given vertex v_k in a path P_n, we would like to be able to assume that v_k and v_{k-1} are each incident with two edges. We will begin with a smaller theorem that deals with calculating the multiplier for vertices in which this assumption fails.

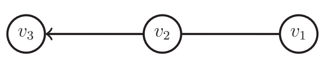
Theorem 3.3.2. *(Little Multiplier Theorem) Let* $P_n = v_1 e_1 v_2 e_2 \ldots e_{n-1} v_n$ *be a path on* $n \geq 3$ *vertices and let* R *be a* p_2*-orientation on* P_n*. Then*

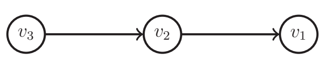
- v_1 *has a multiplier of 1.*
- *If* e_2 *is flat, then the multiplier of* v_2 *is 1.*
- *If* e_2 *is directed, then the multiplier of* v_2 *is 2.*
- *If* e_{n-2} *is flat, then the multiplier of* v_n *is 1.*
- *If* e_{n-2} *is directed, then the multiplier of* v_n *is 2.*

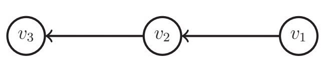
Proof. The multiplier for v_1 is always 1 because, by convention, we set v_1 at 0 chips. By Lemma 3.2.12, we know that e_1 cannot be flat. By Lemma 3.2.14, we know that e_2 does not agree with e_1. So, when calculating the multiplier for v_2, there are two cases. Either e_2 disagrees with e_1, or e_2 is flat. That is, we can exclude the following suborientations





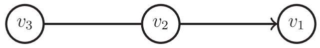






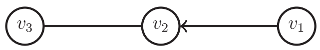
We will begin by supposing that e_2 is flat and reaching that v_2 has a multiplier of 1. There are two possibilities.

(i) e_1 *is directed right.*



The net effect of the first firing on v_2 is a decrease of one chip, and the net effect of the first firing on v_1 is an increase of one chip. This means that $|v_2|_0 > |v_1|_0$ and $|v_2|_0 - 1 < |v_1|_0 + 1$. So, $|v_1|_0 < |v_2|_0 < |v_1|_0 + 2$. Therefore, $|v_1|_0 + 1$ is the only possible initial stack size for v_2. Since there is only one possible initial stack size, v_2 has a multiplier of 1.

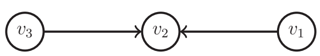
(ii) e_1 *is directed left.*



The net effect of the first firing on v_2 is an increase of one chip, and the net effect of the first firing on v_1 is a decrease of one chip. This means that $|v_2|_0 < |v_1|_0$ and $|v_2|_0 + 1 > |v_1|_0 - 1$. So, $|v_1|_0 > |v_2|_0 > |v_1|_0 - 2$. Therefore, $|v_1|_0 - 1$ is the only possible initial stack size for v_2. Since there is only one possible initial stack size, v_2 has a multiplier of 1.

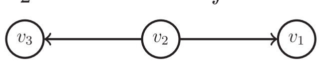
We will now suppose instead that e_2 disagrees with e_1 and reach that v_2 has a multiplier of 2. There are two possibilities.

(i) e_2 *is directed right.*



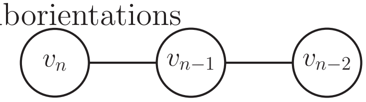
The net effect of the first firing on v_2 is an increase of two chips, and the net effect of the first firing on v_1 is a decrease of one chip. This means that $|v_2|_0 < |v_1|_0$ and $|v_2|_0 + 2 > |v_1|_0 - 1$. So, $|v_1|_0 > |v_2|_0 > |v_1|_0 - 3$. Therefore, $|v_1|_0 - 1$ and $|v_1|_0 - 2$ are the only possible initial stack size for v_2. Since there are only two possible initial stack sizes, v_2 has a multiplier of 2.

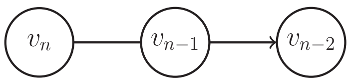
(ii) e_2 *is directed left.*

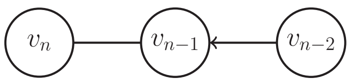


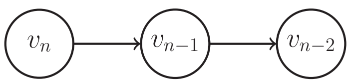
The net effect of the first firing on v_2 is a decrease of two chips, and the net effect of the first firing on v_1 is an increase of one chip. This means that $|v_2|_0 > |v_1|_0$ and $|v_2|_0 - 2 < |v_1|_0 + 1$. So, $|v_1|_0 < |v_2|_0 < |v_1|_0 + 3$. Therefore, $|v_1|_0 + 1$ and $|v_1|_0 + 2$ are the only possible initial stack size for v_2. Since there are only two possible initial stack sizes, v_2 has a multiplier of 2.

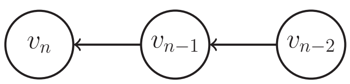
We now turn our attention to v_n. By Lemma 3.2.12, we know that the edge e_{n-1} is not flat. By Lemma 3.2.14, we know that the edge e_{n-2} does not agree with the edge e_{n-1}. So, when calculating the multiplier for v_n, there are two cases. Either e_{n-2} disagrees with e_{n-1} or e_{n-2} is flat. That is, we can exclude the following suborientations





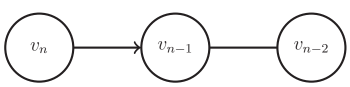






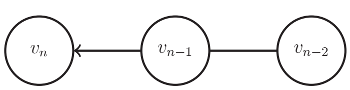
We will suppose first that e_{n-2} is flat and reach that v_n has a multiplier of 1. There are two possibilities.

(i) *e_{n-1} is directed right.*



The net effect of the first firing on v_n is a decrease of one chip, and the net effect of the first firing on v_{n-1} is an increase of one chip. This means that $|v_n|_0 > |v_{n-1}|_0$ and $|v_n|_0 - 1 < |v_{n-1}|_0 + 1$. So, $|v_{n-1}|_0 < |v_n|_0 < |v_{n-1}|_0 + 2$. Therefore, $|v_{n-1}|_0 + 1$ is the only possible initial stack sizes for v_n. Since there is only one possible initial stack size, v_n has a multiplier of 1.

(ii) *e_{n-1} is directed left.*

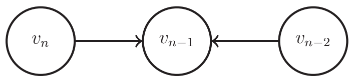


The net effect of the first firing on v_n is an increase of one chip, and the net effect of the first firing on v_{n-1} is a decrease of one chip. This means that $|v_n|_0 < |v_{n-1}|_0$ and $|v_n|_0 + 1 > |v_{n-1}|_0 - 1$. So, $|v_{n-1}|_0 > |v_n|_0 > |v_{n-1}|_0 - 2$.

Therefore, $|v_{n-1}|_0 - 1$ is the only possible initial stack sizes for v_n. Since there is only one possible initial stack size, v_n has a multiplier of 1.

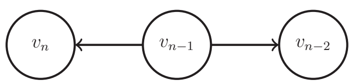
We will now suppose instead that e_n disagrees with e_{n-1} and reach that v_n has a multiplier of 2. There are two possibilities.

1. *e_n is directed right.*



The net effect of the first firing on v_n is a decrease of one chip, and the net effect of the first firing on v_{n-1} is an increase of two chips. This means that $|v_n|_0 > |v_{n-1}|_0$ and $|v_n|_0 - 1 < |v_{n-1}|_0 + 2$. So, $|v_{n-1}|_0 < |v_n|_0 < |v_{n-1}|_0 + 3$. Therefore, $|v_{n-1}|_0 + 1$ and $|v_1|_0 + 2$ are the only possible initial stack sizes for v_n. Since there are only two possible initial stack sizes, v_n has a multiplier of 2.

2. *e_n is directed left.*



The net effect of the first firing on v_n is an increase of one chip, and the net effect of the first firing on v_{n-1} is a decrease of two chips. This means that $|v_n|_0 < |v_{n-1}|_0$ and $|v_n|_0 + 1 > |v_{n-1}|_0 - 2$. So, $|v_{n-1}|_0 > |v_n|_0 > |v_1|_0 - 3$. Therefore, $|v_{n-1}|_0 - 1$ and $|v_{n-1}|_0 - 2$ are the only possible initial stack sizes for v_n. Since there are only two possible initial stack sizes, v_n has a multiplier of 2.

□

We will now look at the multipliers of the other vertices.

Theorem 3.3.3. *(The Multiplier Theorem) Let R be a p_2-orientation on a path $P_n = v_1 e_1 v_2 e_2 \ldots e_{n-1} v_n$ with $n \geq 4$. If a vertex v_k and its neighbour v_{k-1} each have exactly two neighbours, then the multiplier of v_k is 1, 2, or 3 depending on the suborientation within which it exists, as outlined in Table 3.1.*

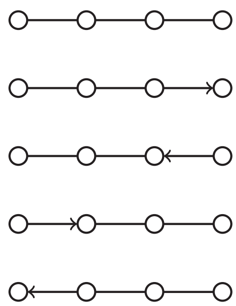
Multiplier				
3	$\bigcirc \leftarrow v_k \rightarrow \bigcirc \leftarrow \bigcirc$ $\bigcirc \rightarrow v_k \leftarrow \bigcirc \rightarrow \bigcirc$			
2	$\bigcirc \leftarrow v_k \rightarrow \bigcirc - \bigcirc$ $\bigcirc \rightarrow v_k \leftarrow \bigcirc - \bigcirc$	$\bigcirc - v_k \rightarrow \bigcirc \leftarrow \bigcirc$ $\bigcirc - v_k \leftarrow \bigcirc \rightarrow \bigcirc$		
1	$\bigcirc \leftarrow v_k \rightarrow \bigcirc \rightarrow \bigcirc$ $\bigcirc \rightarrow v_k \leftarrow \bigcirc \leftarrow \bigcirc$	$\bigcirc \leftarrow v_k \leftarrow \bigcirc \rightarrow \bigcirc$ $\bigcirc \rightarrow v_k \rightarrow \bigcirc \leftarrow \bigcirc$	$\bigcirc - v_k \rightarrow \bigcirc - \bigcirc$ $\bigcirc - v_k \leftarrow \bigcirc - \bigcirc$	$\bigcirc \leftarrow v_k - \bigcirc \rightarrow \bigcirc$ $\bigcirc \rightarrow v_k - \bigcirc \leftarrow \bigcirc$

Table 3.1: Multiplier based on neighbourhood

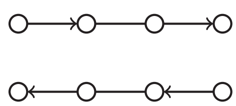
Proof. We will begin by proving that no suborientation omitted from Table 3.1 can be contained within a p_2-orientation.

Every edge has 3 possible orientations. Therefore, there exist $3^3 = 27$ graph orientations of P_4. However, we know several of these orientations cannot exist as suborientations within a p_2-orientation by Theorem 3.2.10. We now list these orientations which cannot exist within a p_2-orientation.

The following 5 suborientations cannot exist within a p_2-orientation by Lemma 3.2.11.



The following 2 suborientations cannot exist within a p_2-orientation by Lemma 3.2.13.



The following 6 suborientations cannot exist within a p_2-orientation by Lemma 3.2.14.

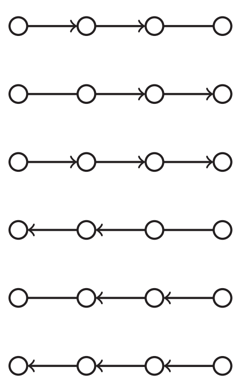
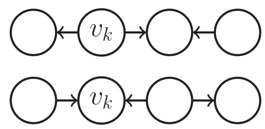


Figure 3.11: List of orientations which cannot exist as suborientations within a p_2-orientation

What remains are the $27 - 13 = 14$ suborientations listed in Table 3.1. We will break these 14 suborientations into 7 pairs of suborientations and show their multipliers using a case analysis. Each orientation will be paired with the orientation created by reversing the direction of every directed edge contained within. We will see that these pairs always have the same multiplier and can be proven using similar arguments. Note that by Corollary 1.2.2, every period that contains one of these orientations must also contain the one with which it is paired.

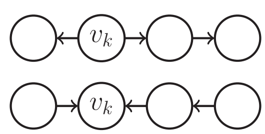
Case 1: Alternating arrow suborientation.



First assume that v_k is losing two chips in the first firing. The net effect of the first firing on v_k is a decrease of two chips, and the net effect of the first firing on v_{k-1} is an increase of two chips. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 - 2 < |v_{k-1}|_0 + 2$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 4$. Therefore, $|v_{k-1}|_0 + 1$, $|v_{k-1}|_0 + 2$, and $|v_{k-1}|_0 + 3$ are the only possible initial stack sizes for v_k. Since there are only three possible initial stack sizes, v_k has a multiplier of 3.

Now assume that instead, v_k is gaining two chips in the first firing. The net effect of the first firing on v_k is an increase of two chips, and the net effect of the first firing on v_{k-1} is a decrease of two chips. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 + 2 > |v_{k-1}|_0 - 2$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 4$. Therefore, $|v_{k-1}|_0 - 1$, $|v_{k-1}|_0 - 2$, and $|v_{k-1}|_0 - 3$ are the only possible initial stack sizes for v_k. Since there are only three possible initial stack sizes, v_k has a multiplier of 3.

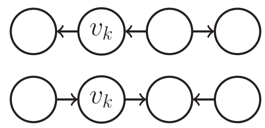
Case 2:



First assume that v_k is losing two chips in the first firing. The net effect of the first firing on v_k is a decrease of two chips, and the net effect of the first firing on v_{k-1} is no change in the number of chips. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 - 2 < |v_{k-1}|_0$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 2$. Therefore, $|v_{k-1}|_0 + 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Now assume that instead, v_k is gaining two chips in the first firing. The net effect of the first firing on v_k is an increase of two chips, and the net effect of the first firing on v_{k-1} is no change in the number of chips. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 + 2 > |v_{k-1}|_0$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 2$. Therefore, $|v_{k-1}|_0 - 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

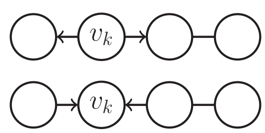
Case 3:



First assume that v_{k-1} is losing two chips in the first firing. The net effect of the first firing on v_k is no change in the number of chips, and the net effect of the first firing on v_{k-1} is a decrease of two chips. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 > |v_{k-1}|_0 - 2$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 2$. Therefore, $|v_{k-1}|_0 - 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Now assume that instead, v_{k-1} is gaining two chips in the first firing. The net effect of the first firing on v_k is no change in the number of chips, and the net effect of the first firing on v_{k-1} is an increase of two chips. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 < |v_{k-1}|_0 + 2$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 2$. Therefore, $|v_{k-1}|_0 + 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

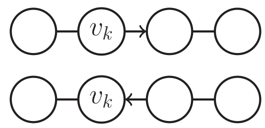
Case 4:



First assume that v_k is losing two chips in the first firing. The net effect of the first firing on v_k is a decrease of two chips, and the net effect of the first firing on v_{k-1} is an increase of one chip. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 - 2 < |v_{k-1}|_0 + 1$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 3$. Therefore, $|v_{k-1}|_0 + 1$ and $|v_{k-1} + 2$ are the only possible initial stack sizes for v_k. Since there are only two possible initial stack sizes, v_k has a multiplier of 2.

Now assume that instead, v_k is gaining two chips in the first firing. The net effect of the first firing on v_k is an increase of two chips, and the net effect of the first firing on v_{k-1} is a decrease of one chip. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 + 2 > |v_{k-1}|_0 - 1$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 3$. Therefore, $|v_{k-1}|_0 - 1$ and $|v_{k-1} - 2$ are the only possible initial stack sizes for v_k. Since there are only two possible initial stack sizes, v_k has a multiplier of 2.

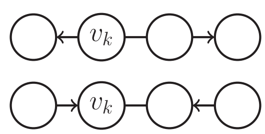
Case 5:



First assume that v_k is losing one chip in the first firing. The net effect of the first firing on v_k is a decrease of one chip, and the net effect of the first firing on v_{k-1} is an increase of one chip. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 - 1 < |v_{k-1}|_0 + 1$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 2$. Therefore, $|v_{k-1}|_0 + 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Now assume that instead, v_k is gaining one chip in the first firing. The net effect of the first firing on v_k is an increase of one chip, and the net effect of the first firing on v_{k-1} is a decrease of one chip. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 + 1 > |v_{k-1}|_0 - 1$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 2$. Therefore, $|v_{k-1}|_0 - 1$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Case 6:

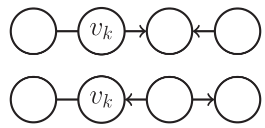


First assume that v_k is losing one chip in the first firing. The net effect of the first firing on v_k is a decrease of one chip, and the net effect of the first firing on v_{k-1} is a decrease of one chip. This means that $|v_k|_0 = |v_{k-1}|_0$ and $|v_k|_0 - 1 = |v_{k-1}|_0 - 1$. Therefore, $|v_{k-1}|_0$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Now assume that instead, v_k is gaining one chip in the first firing. The net effect of the first firing on v_k is an increase of one chip, and the net effect of the first firing on v_{k-1} is an increase of one chip. This means that $|v_k|_0 = |v_{k-1}|_0$ and

$|v_k|_0 + 1 = |v_{k-1}|_0 + 1$. Therefore, $|v_{k-1}|_0$ is the only possible initial stack size for v_k. Since there is only one possible initial stack size, v_k has a multiplier of 1.

Case 7:



First assume that v_k is losing one chip in the first firing. The net effect of the first firing on v_k is a decrease of one chip, and the net effect of the first firing on v_{k-1} is an increase of two chips. This means that $|v_k|_0 > |v_{k-1}|_0$ and $|v_k|_0 - 1 < |v_{k-1}|_0 + 2$. So, $|v_{k-1}|_0 < |v_k|_0 < |v_{k-1}|_0 + 3$. Therefore, $|v_{k-1}|_0 + 1$ and $|v_{k-1}|_0 + 2$ are the only possible initial stack sizes for v_k. Since there are only one two possible initial stack sizes, v_k has a multiplier of 2.

Now assume that instead, v_k is gaining one chip in the first firing. The net effect of the first firing on v_k is an increase of one chip, and the net effect of the first firing on v_{k-1} is a decrease of two chips. This means that $|v_k|_0 < |v_{k-1}|_0$ and $|v_k|_0 + 1 > |v_{k-1}|_0 - 2$. So, $|v_{k-1}|_0 > |v_k|_0 > |v_{k-1}|_0 - 3$. Therefore, $|v_{k-1}|_0 - 1$ and $|v_{k-1}|_0 - 2$ are the only possible initial stack sizes for v_k. Since there are only one two possible initial stack sizes, v_k has a multiplier of 2. $\square$

We now state a number of corollaries that come from the results regarding the multipliers of specific vertices found in Theorem 3.3.3.

Corollary 3.3.4. *The number of p_2-configurations that exist on alternating arrow orientations on P_n, $n \geq 3$, is $8 \times 3^{n-3}$.*

Proof. In an alternating arrow orientation, every edge e_i, $2 \leq i \leq n$, disagrees with the previous edge e_{i-1}. So, an alternating arrow orientation is unique based on the orientation of $e_1 = v_1v_2$. Therefore, there exist two alternating arrow orientations on a given path P_n, $n > 2$.

From Theorems 3.3.2 and 3.3.3, we get that the multiplier for v_1 is 1, the multiplier for both v_2 and v_n is 2, and every other multiplier is 3.

Thus, the number of p_2-configurations that exist on a particular alternating arrow orientation on P_n is $1 \times 2 \times 2 \times 3^{n-3}$. Multiplying by two different alternating arrow orientations depending on the orientation of the first edge, we get that the number of p_2-configurations that exist on alternating arrow orientations on P_n, $n \geq 3$, is $1 \times 2 \times 2 \times 3^{n-3} \times 2 = 8 \times 3^{n-3}$. $\square$

Define a sequence A_n to represent the number of p_2-configurations that exist on alternating arrow orientations on the path with n vertices.
So $A_n = 0, 2, 8, 24, 72, 216, 648, ..., A_k, 3A_k, 3 \times 3A_k, ...$

Corollary 3.3.5. *For all $n \geq 3$, $3A_n = A_{n+1}$.*

Corollary 3.3.6. *Let R be a p_2-orientation of P_n, $n \geq 2$. Let R_1, R_2, $\ldots$, R_k, be the suborientations of R on the k disjoint paths created by removing $k-1$ flat edges from P_n. The number of p_2-configurations that exist on R is equal to the product of the number of p_2-configurations that exist on the suborientations R_1, R_2, $\ldots$, R_k.*

Proof. Let R be a period orientation on P_n with at least one flat edge. Let R_1 and R_2 be the suborientations of R on the disjoint paths created by removing a single flat edge from P_n.

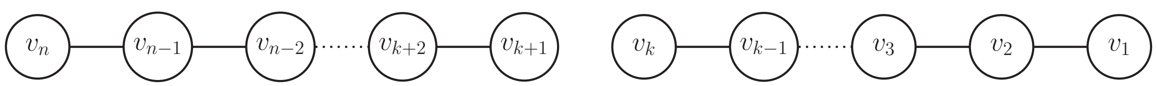


Figure 3.12: P_n with edge $v_k v_{k+1}$ removed.

Suppose we removed just one flat edge, $v_k v_{k+1} = e_k$. The only vertices that could have an altered multiplier are those which are endpoints of e_k or e_{k+1}. The vertices in question are v_k, v_{k+1}, and v_{k+2}. However, since what is measured in calculating the multiplier is the net effect of the firing, being incident to a flat edge is equivalent to not being incident to an edge at all. In particular, note that v_{k+1}, appearing to the left of the flat edge e_k, has only one possible initial stack size, that being $|v_k|_0$. This is equivalent to v_{k+1} having only one possible initial stack size, by convention, when viewed as the right leaf in R_2 (by Lemma 3.1.1, we can subtract $|v_k|_0$ from every stack size in R_2 without altering the movement of chips). So, it follows that any number of flat edge removals will still maintain this result.

$\square$

Next, we present a corollary of the multiplier theorem (Theorem 3.3.3) which will be useful in determining the number of p_2-configurations that exist which induce orientations with adjacent agreeing arrows. It will be shown that, given a p_2-orientation of P_n which contains some suborientation $v_{k+2}e_{k+1}v_{k+1}e_kv_k$ such that e_{k+1} agrees with e_k, the orientation of P_{n-2} created by contracting the edges e_{k+1} and e_k and reversing the direction of all directed edges e_i, $i > k+1$, is induced by the same number of p_2-configurations. We see an example of two such graph orientations in Figure 3.13.

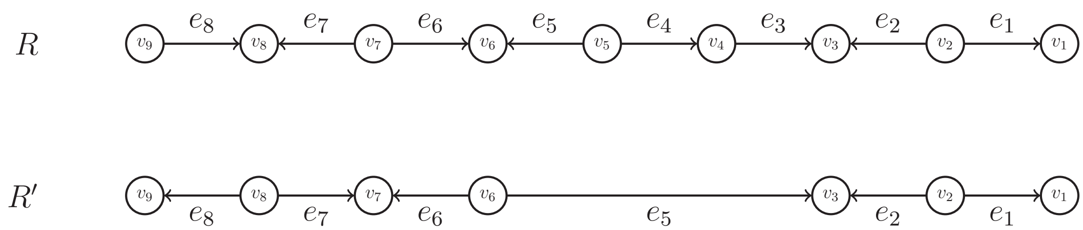


Figure 3.13: Graph orientations R and R', created by contracting two adjacent agreeing edges and reversing the direction of all subsequent directed edges

Note how the edges e_3 and e_4 have been contracted, removing v_4, and v_5, and every directed edge occurring to the left of the contraction has reversed direction.

Corollary 3.3.7. *Suppose there exist adjacent agreeing edges $e_k = v_k v_{k+1}$ and $e_{k+1} = v_{k+1}v_{k+2}$ in a period orientation, R, on a path, P_n, $n \geq 4$. Let R' be the graph orientation created by contracting e_k and e_{k+1}, reversing direction of every directed edge e_i, $i > k+1$, and maintaining every other edge orientation from R. The number of p_2-configurations on R is equal to the number of p_2-configurations on R'.*

Proof. By Theorem 3.3.3, given a suborientation $v_{k+2}e_{k+1}v_{k+1}e_k v_k$ of a p_2-orientation on a path P_n in which e_k and e_{k+1} agree, the multipliers of v_{k+2} and v_{k+1} are both equal to one. By contracting e_k and e_{k+1}, we are removing v_{k+2} and v_{k+1} from the orientation. By removing these vertices, assuming every other multiplier has been maintained, the number of p_2-configurations that exist inducing the resulting orientation is the same as the number of p_2-configurations that exist inducing the original orientation. Due to the reversing direction of every subsequent directed edge, v_k remains within the same P_4 suborientation from Theorem 3.3.3 and thus, maintains the same multiplier. Finally, every vertex appearing to the left of this contraction has had any incident directed edges reverse direction. However, by Theorem 3.3.3, such a flipping of directed edges does not change a vertex's multiplier. Therefore, the product of multipliers must only be divided by 1×1 (the product of the multipliers of the removed vertices) to accommodate the edge contraction, and thus, the number of p_2-configurations does not change.

So, we get that the number of configurations on R is equal to the number of configurations on R' and this process can be repeated until all pairs of adjacent agreeing edges have been removed.

□

Let F_n be the number of p_2-configurations that exist on P_n.

Theorem 3.3.8. *For all paths P_n, $n \geq 4$,*

$$F_{n+4} = 3F_{n+3} + 2F_{n+2} + F_{n+1} - F_n$$

with $F_1 = 0$, $F_2 = 2$, $F_3 = 8$, and $F_4 = 26$.

We will prove this theorem with the help of a number of claims.

In order to count the number of p_2-configurations on P_n, we will divide the set of all p_2-orientations into 3 cases. We have already solved for the number of p_2-configurations that exist on alternating arrow orientations on n vertices, A_n. Our other two cases will be the case in which, moving from right to left, a flat edge appears before the first pair of adjacent agreeing edges, and the case in which, moving from right to left, the first pair of adjacent agreeing edges appears before the first flat. We will then add up these three totals to determine the number of p_2-configurations that exist on P_n.

Claim 3.3.9. *The number of p_2-configurations on P_n, $n \geq 4$, in which, moving from right to left, a flat appears before the first pair of adjacent agreeing edges is*

$$\sum_{k=2}^{n-2} \frac{1}{2} A_k \times F_{n-k}$$

Proof. Suppose that, moving from right to left, the graph orientation R is alternating until the first flat appears. That is, the first flat appears before the first pair of adjacent agreeing edges appear. By Corollary 3.3.6, the number of p_2-configurations that exist on a graph with some flat edge e_k is equal to the product of the numbers of p_2-configurations that exist on the two suborientations created by removing e_k. Let $e_k = v_{k+1}v_k$ be the flat edge with the least index. We know that the suborientation $R_1 = v_k e_{k-1} v_{k-2} e_{k-2} \ldots e_1 v_1$ is an alternating arrow orientation by supposition. We know less about the suborientation $R_2 = v_n e_{n-1} v_{n-1} e_{n-2} \ldots e_{k+1} v_{k+1}$. By Lemma 3.2.13, we know that e_{k+1} disagrees with e_{k-1}. The number of configurations of P_{n-k} which induce an orientation in which the orientation of the edge with the least index is given (without loss of generality suppose it is right) is equal to $\frac{1}{2}F_{n-k}$ since half of the possibilities are excluded because the direction of the first edge is already known. So by Corollary 3.3.6, the number of p_2-configurations which induce R is $\frac{1}{2}F_{n-k} \times A_k$. Summing this value over all possible edges that could represent the first flat edge, we get

$$\sum_{k=2}^{n-2} \frac{1}{2} A_k \times F_{n-k}$$

□

Definition 3.3.10. *The k^{th}* ***stage*** *of a path is the total number of p_2-configurations that exist on that path in which either a pair of adjacent agreeing edges or a flat appears within the first $k+1$ edges, $e_1, \ldots, e_{k+1}$.*

For example, on P_8, the 1^{st} stage is the number of p_2-configurations that exist in which the second edge is flat. The 2^{nd} stage is the number of p_2-configurations that exist in which either the second or third edge is flat, or the third edge agrees with the second edge. And the 3^{rd} stage is the number of p_2-configurations that exist in which either the second, third, or fourth edge is flat, or either the third or fourth edge agrees with its previous edge. We denote the k^{th} stage of P_n by s_n^k.

Claim 3.3.11. *If $n \geq 3$, then $s_n^k = F_n$ for all $k \geq n-1$ and $s_n^k = F_n - A_n$ if $k = n-2$ or $k = n-3$.*

Proof. **Case 1:** $k = n-3$

From the definition of stage, we are counting the number of p_2-configurations that exist on P_n in which either a pair of adjacent agreeing edges or a flat appears within the first $n-2$ edges. The edge e_{n-1} cannot be flat or agree with e_{n-2} by Theorem 3.2.10. So s_n^{n-3} counts every p_2-configuration except for those that induce an alternating arrow orientation. Thus $s_n^{n-3} = F_n - A_n$.

Case 2: $k = n-2$

We are counting every configuration from Case 1, but also including the possibility of e_{n-1} being flat and the possibility of e_{n-1} agreeing with e_{n-2}. However, by Theorem 3.2.10, there are no p_2-orientations in which either of these situations arise. So, $s_n^{n-2} = s_n^{n-3} = F_n - A_n$.

Case 3: $k \geq n-1$

We are counting every configuration from Case 2, but also including the possibility that there does not exist a flat edge or pair of adjacent agreeing edges within the $n-1$ edges. So, every p_2-configuration must be counted. So, $s_n^k = F_n$ for all $k \geq n-1$.

□

Claim 3.3.12. *The number of p_2-configurations on P_n, $n \geq 5$, in which, moving from right to left, a pair of adjacent agreeing edges appear before a flat is*

$$\sum_{k=3}^{n-3}(F_{n-2} - s_{n-2}^{k-2}).$$

Proof. We say $n \geq 5$ since, by Theorem 3.2.10, no pair of adjacent agreeing edges can exist in a p_2-configuration on a path with fewer than 5 vertices. We are assuming that, moving from right to left, the graph orientation is entirely alternating until the first pair of adjacent agreeing edges appears. That is, the first pair of adjacent agreeing edges appears before the first flat appears. When adjacent agreeing edges appear, the number of p_2-configurations is equal to the number of p_2-configurations on the path with two fewer vertices in which the agreeing edges are removed and subsequent directed edges are reversed as outlined in Corollary 3.3.7. So every graph orientation of this form on P_n can be viewed as a similar graph orientation on P_{n-2} without changing the multipliers of any vertices. This allows for a recurrence, helping us to evaluate F_n using F_{n-2}. However we have supposed that up to some edge e_k, the graph orientation is alternating. So we must subtract the proper stage of the path on $n-2$ vertices. This will remove the possibility of agreeing edges and flat edges appearing to the right of e_k. Taking this sum over all possible edges that could represent, moving from right to left, the first edge that agrees with its immediate predecessor, we get

$$\sum_{k=3}^{n-3}(F_{n-2} - s_{n-2}^{k-2})$$

□

We now calculate F_n based on F_{n-1}, F_{n-2}, F_{n-3}, and F_{n-4}. Given a p_2-orientation on P_n, there are 4 mutually exclusive and exhaustive cases: e_{n-2} is flat, e_{n-3} is flat, e_{n-2} and e_{n-3} agree, e_{n-2} and e_{n-3} disagree. We know that these are the only possibilities by Theorem 3.2.10.

For each of these four cases, we will determine the number of p_2-configurations that exist in that case. We then add up these four totals to calculate F_n.

Case 1: e_{n-2} is flat.

This calculation is equivalent to the first flat edge being e_2. We know that this is $A_2 \times \frac{1}{2}F_{n-2} = F_{n-2}$. This is the $k = 2$ summand from Claim 3.3.9.

Case 2: e_{n-3} is flat.

This calculation is equivalent to the first flat edge being e_3. $A_3 \times \frac{1}{2}F_{n-3} = 4F_{n-3}$. This is the $k = 3$ summand from Claim 3.3.9.

Case 3: e_{n-2} and e_{n-3} agree.

We use our rule from Corollary 3.3.7 for contracting agreeing arrows. What we get is every solution on P_{n-2} that begins with two disagreeing arrows. This is equivalent to just subtracting the possibility that the first edge is flat. When e_2 is flat, we get $A_2 \times F_{n-4} = F_{n-4}$. So, we get $F_{n-2} - F_{n-4}$.

Case 4: e_{n-2} and e_{n-3} disagree.

This can be viewed as adding a new leftmost vertex to P_{n-1}. This vertex adds a multiplier of 3 (being amongst an alternating arrow suborientation) unless e_{n-3} is flat. However, since we know e_{n-3} to not be flat, we can remove that possibility from our calculation. If e_{n-3} is flat in P_{n-1}, then there are $A_2 \times \frac{1}{2}F_{n-3} = F_{n-3}$ p_2-configurations. So, we get $3(F_{n-1} - F_{n-3})$.

The total sum is thus,

$$\begin{aligned}
F_n &= F_{n-2} + 4F_{n-3} + F_{n-2} - F_{n-4} + 3(F_{n-1} - F_{n-3}) \\
&= 3F_{n-1} + 2F_{n-2} + F_{n-3} - F_{n-4}.
\end{aligned}$$

In order to find the explicit formula, we begin by finding the characteristic equation for this linear recurrence:

$$\begin{aligned}
F_n &= 3F_{n-1} + 2F_{n-2} + F_{n-3} - F_{n-4} \\
F_n - 3F_{n-1} - 2F_{n-2} - F_{n-3} + F_{n-4} &= 0 \qquad \text{Let } F_n = x^n \\
x^n - 3x^{n-1} - 2x^{n-2} - x^{n-3} + x^{n-4} &= 0 \\
x^{n-4}(x^4 - 3x^3 - 2x^2 - x + 1) &= 0 \\
x^4 - 3x^3 - 2x^2 - x + 1 &= 0
\end{aligned}$$

The roots of this equation are $\alpha_1 \approx 3.6096$, $\alpha_2 \approx 0.4290$, $\alpha_3 \approx -0.5193 - 0.6133i$, and $\alpha_4 \approx -0.5193 + 0.6133i$. Since these are all of the nonzero roots of the function, our general solution is a linear combination of all four values:

$$F_k = c_1(\alpha_1)^k + c_2(\alpha_2)^k + c_3(\alpha_3)^k + c_4(\alpha_4)^k.$$

In order to solve for c_1, c_2, c_3, and c_4, we must solve a system of four linear equations using the four initial values of our recurrence: $F_1 = 0$, $F_2 = 2$, $F_3 = 8$, and $F_4 = 26$. The approximate solutions are

$$\begin{aligned} c_1 &\approx 0.1564 \\ c_2 &\approx 0.4449 \\ c_3 &\approx 0.6995 - 0.0234i \\ c_4 &\approx 0.6995 + 0.0234i \end{aligned}$$

So,

$$\begin{aligned} F_k \approx\ & 0.1564(3.6096)^{\,k} \\ & + 0.4449(0.4290)^{\,k} \\ & + (0.6695 - 0.0234i)(-0.5193 - 0.6133i)^{\,k} \\ & + (0.6695 + 0.0234i)(-0.5193 + 0.6133i)^{\,k}. \end{aligned}$$

The dominating term, out of the four roots, is the one which has the greatest modulus. These moduli are roughly 3.6096, 0.4290, 0.8036, and 0.8036. Thus, the dominant term in the equation

$$\begin{aligned} F_k \approx\ & 0.1564(3.6096)^{\,k} \\ & + 0.4449(0.4290)^{\,k} \\ & + (0.6695 - 0.0234i)(-0.5193 - 0.6133i)^{\,k} \\ & + (0.6695 + 0.0234i)(-0.5193 + 0.6133i)^{\,k} \end{aligned}$$

is $c_1(\alpha_1)^{\,k} \approx (0.1564)(3.6096)^{\,k}$.

Corollary 3.3.13. *F_k has an asymptotic value of approximately* 0.1564×3.6096^k.

Suppose now that a graph G_k is composed of some graph G_0 connected to a path P_k, $k \geq 4$, with a bridge (an edge which, upon removal, would disconnect the graph). Due to the fact that multiplier calculations are localized for each vertex in a path, we conjecture that if a new vertex v were added to the end of this path then the new vertex will be such a distance away from G_0 that this recurrence relation will hold. That is, if we know the number of p_2-configurations that exist on the four graphs G_i, $i = 0, 1, 2, 3$, then this same recurrence relation can calculate the number of p_2-configurations that exist on G_4. In this way, we conjecture that our recurrence relation solution extends to any graph connected to a path of length at least 4.

Conjecture 3.3.14. *Let G_k be a graph composed of some graph G_0 connected to a path P_k, $k \geq 3$, with a bridge. Then the number of p_2-configurations on G_k, $F(G_k)$, can be determined using the recurrence*

$$F(G_k) = 3F(G_{k-1}) + 2F(G_{k-2}) + F(G_{k-3}) - F(G_{k-4}).$$

3.4 Period Configurations on Complete Graphs

In this section, we count the number of period configurations that exist on complete graphs. In contrast with the results on paths, we will be counting the number of period configurations that exist on *unlabelled* complete graphs. We do not label the vertices in order to best exhibit the unexpected relationship between the number of period configurations on the complete graph of size n and the number of board-pile polyominoes containing n unit squares. We will show that these two totals are in fact, equal. The number of board-pile polyominoes containing exactly n unit squares comes from the sequence A001169 in the OEIS [16], and the values are exhibited in Table 3.2. The generating function is $\frac{x(1-x)^3}{(1-5x+7x^2-4x^3)}$ [16]. This sequence of numbers follows the recurrence relation $a_n = 5a_{n-1} - 7a_{n-2} + 4a_{n-3}$ for $n \geq 5$, with initial values $a_1 = 1$, $a_2 = 2$, $a_3 = 6$, and $a_4 = 19$.[13]

In order to find the explicit formula, we begin by finding the characteristic equation for this linear recurrence:

$$\begin{aligned} a_n &= 5a_{n-1} - 7a_{n-2} + 4a_{n-3} \\ a_n - 5a_{n-1} + 7a_{n-2} - 4a_{n-3} &= 0 \qquad \text{Let } a_n = x^n \\ x^n - 5x^{n-1} + 7x^{n-2} - 4x^{n-3} &= 0 \\ x^{n-3}(x^3 - 5x^2 + 7x - 4) &= 0 \\ x^3 - 5x^2 + 7x - 4 &= 0 \end{aligned}$$

The roots of this equation are $\alpha_1 \approx 3.2056$, $\alpha_2 \approx 0.8972 - 0.6655i$, and $\alpha_3 \approx 0.8972 + 0.6655i$. Since these are all of the nonzero roots of the function, our general solution is a linear combination of all three values:

$$a_k = c_1(\alpha_1)^k + c_2(\alpha_2)^k + c_3(\alpha_3)^k.$$

In order to solve for c_1, c_2, and c_3, we must solve a system of three linear equations using three initial values of our recurrence: $a_2 = 2$, $a_3 = 6$, and $a_4 = 19$. The approximate solutions are

$$\begin{aligned} c_1 &\approx 0.1809 \\ c_2 &\approx 0.0658 + 0.0391i \\ c_3 &\approx 0.0658 - 0.0391i \end{aligned}$$

So,

$$\begin{aligned} a_k \approx\ & 0.1809(3.2056)^k \\ & + (0.0658 + 0.0391i)(0.8972 - 0.6655i)^k \\ & + (0.0658 - 0.0391i)(0.8972 + 0.6655i)^k. \end{aligned}$$

The dominating term, out of the three roots, is the one which has the greatest modulus. These moduli are roughly 3.2056, 1.1171, and 1.1171. Thus, the dominant term in the equation

$$\begin{aligned} a_k \approx\ & 0.1809(3.2056)^k \\ & + (0.0658 + 0.0391i)(0.8972 - 0.6655i)^k \\ & + (0.0658 - 0.0391i)(0.8972 + 0.6655i)^k \end{aligned}$$

is $c_1(\alpha_1)^k \approx (0.1809)(3.2056)^k$.

Corollary 3.4.1. *a_k has an asymptotic value of approximately* 0.1809×3.2056^k.

n	# of board-pile polyominoes
1	1
2	2
3	6
4	19
5	61
6	196
7	629
8	2017
9	6466
10	20727
11	66441

Table 3.2: Number of board-pile polyominoes containing n unit squares for $1 \leq n \leq 11$.

We will begin by introducing the concept of polyominoes. These definitions are from David Klarner's paper, reworded slightly to make our proofs easier to follow. [13]

Definition 3.4.2. *[13] A* ***polyomino*** *is a plane figure composed of a number of connected unit squares joined edge on edge. A polyomino with exactly n unit squares is called an* ***n-omino****.*

Definition 3.4.3. *[13] In a polyomino X, a* ***horizontal strip****, or* ***h-strip****, is a maximal rectangle of height one.*

By convention, we will set each h-strip in the plane so that its height spans from an integer k to $k + 1$.

Definition 3.4.4. *[13] The infinite area enclosed by the lines $y = k$ and $y = k + 1$ is called a* ***row****.*

Definition 3.4.5. *[13] A* ***board-pile polyomino*** *is a polyomino which has at most one h-strip per row. A board-pile polyomino with n unit squares is called a* ***board-pile n-omino****.*

Figure 3.14 provides examples of some plane figures that are not board-pile polyominoes and one which is a board-pile polyomino.

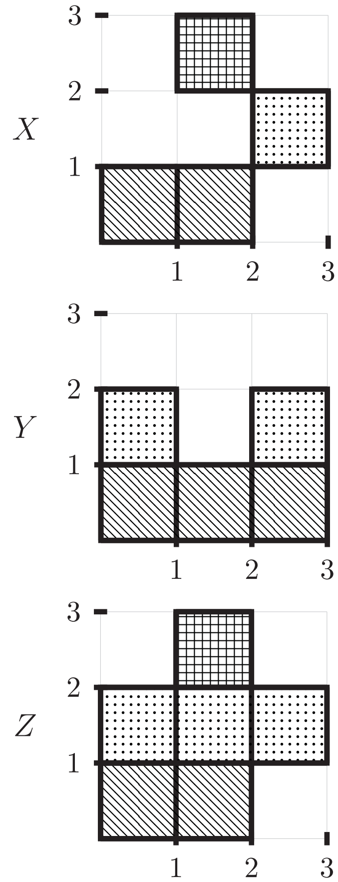


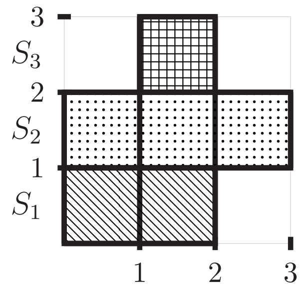
Figure 3.14: Three plane-figures: X, Y, and Z are shown with their h-strips differentiated by shading. X is not a board-pile polyomino because the strips are not connected edge on edge. Y is not a board-pile polyomino because there exists a row with multiple h-strips. Z is a board-pile polyomino since each row contains at most one h-strip.

We will now develop a notation for polyominoes that will eliminate the necessity of a pictorial representation, define a mapping from the set of all polyominoes on n unit squares to the set of all period configurations of (an unlabelled) K_n up to equivalence, and then show that mapping to be a bijection.

We will use the convention of labelling the first h-strip from the bottom as S_1, the next one up as S_2, and so on. Let $S(X)$ be the set of all h-strips in a board-pile polyomino X and let N be the number of h-strips in X.

A board-pile polyomino X, can be represented as a sequence of ordered pairs of the form $X = \{(d_1, |S_1|), (d_2, |S_2|), (d_3, |S_3|), \ldots (d_N, |S_N|)\}$, where $|S_i|$ is the number of unit squares in the h-strip S_i, and d_i is the difference between the greatest x-coordinate in S_i and the least x-coordinate in S_{i-1}. By convention, $d_1 = 0$. See

Figure 3.15 for an example.

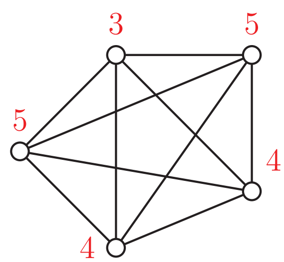


$$= \{(0,2),(3,3),(2,1)\}$$

$$S(X) = \{S_1, S_2, S_3\}$$

Figure 3.15: Board-pile 6-omino X with shading differentiating between S_1, S_2, and S_3.

A configuration of an unlabelled K_n can be represented by a multiset of cardinality n with the stack sizes as elements. We will use the notation a^k to represent k instances of stack size a in the configuration. An example is shown in Figure 3.16.



$$C = \{3, 4^2, 5^2\}$$

Figure 3.16: Configuration on an unlabelled complete graph

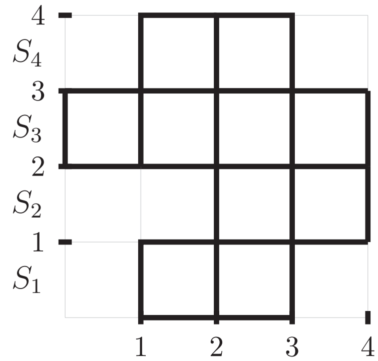
Given n, define a map f from the set of all board-pile n-ominoes to the set of all complete graph configurations on K_n.

For a board-pile n-omino $X = \{(d_1, |S_1|), (d_2, |S_2|), (d_3, |S_3|), \ldots (d_N, |S_N|)\}$, let

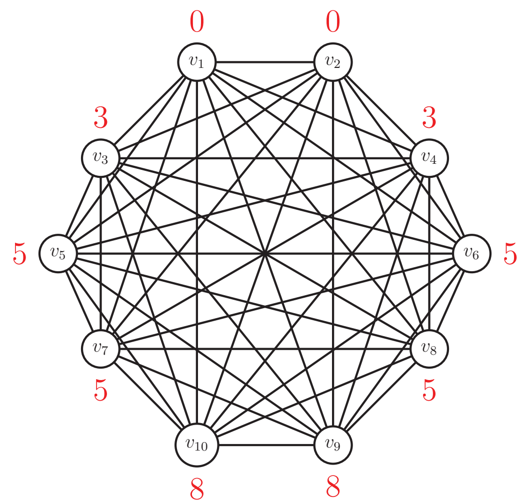
$$\begin{aligned} f(X) &= f(\{(d_1, |S_1|), (d_2, |S_2|), (d_3, |S_3|), \dots (d_n, |S_N|)\}) \\ &= \left\{0^{|S_1|}, \left(\sum_{i=1}^{2} d_i\right)^{|S_2|}, \left(\sum_{i=1}^{3} d_i\right)^{|S_3|}, \dots \left(\sum_{i=1}^{N} d_i\right)^{|S_N|}\right\}. \end{aligned}$$

So, for all $S_k \in S(X)$, the complete graph configuration $f(X)$ has $|S_k|$ vertices, each of which contain $\sum_{i=1}^{k} d_i$ chips.

We denote the set of vertices in $f(X)$ corresponding to the strip S_k to be V_k for all $k \leq N$. An example of this mapping is shown in Figure 3.17.



$$X = \{(0,2),(3,2),(2,4),(3,2)\}$$



$$\begin{aligned} f(X) &= \{0^2, (0+3)^2, (0+3+2)^4, (0+3+2+3)^2\} \\ &= \{0^2, 3^2, 5^4, 8^2\} \end{aligned}$$

Figure 3.17: Mapping a board-pile 10-omino to its corresponding configuration of K_{10}.

We will show that only period configurations of K_n exist in the range of f and that f is a bijection.

Claim 3.4.6. *Let* $X = \{(d_1, |S_1|), (d_2, |S_2|), (d_3, |S_3|), \ldots (d_N, |S_N|)\}$ *be a board-pile polyomino with exactly* N *h-strips. If* $1 \leq i \leq N-1$, *then* $1 \leq d_{i+1} \leq |S_i| + |S_{i+1}| - 1$.

Proof. Since S_i and S_{i+1} are adjacent h-strips, and polyominoes, by definition, are joined edge on edge, the distance from the least x-coordinate of S_i to the greatest

x-coordinate of S_{i+1} must be less than the sum of the two lengths $(|S_i| + |S_{i+1}|)$. So, d_{i+1} must be less than $|S_i| + |S_{i+1}|$. Since d_{i+1} is equal to the difference between the greatest x-coordinate in S_{i+1} and the least x-coordinate in S_i, and since S_{i+1} and S_i are connected edge on edge, $d_{i+1} \geq 1$. Thus, we conclude $1 \leq d_{i+1} \leq |S_i| + |S_{i+1}| - 1$. □

Theorem 3.4.7. *For any board-pile polyomino, X, on n unit squares, $n \geq 1$, $f(X)$ is a period configuration of K_n and for any period configuration, C, of K_n, $n \geq 1$, there is some board-pile polyomino X on n unit squares such that $C = f(X)$.*

Proof. ($\Rightarrow$) A board-pile polyomino with only a single h-strip maps trivially to a period configuration of K_n. In this case, every stack size is equal to 0. Therefore, the configuration is a period configuration with a period length of 1. We will now show that any board-pile polyomino with only two h-strips maps to a period configuration. Let X be such a board-pile polyomino. These two h-strips, S_1 and S_2, map to two sets of vertices, V_1 and V_2, with distinct stack sizes, $d_1 = 0$ and d_2 respectively. We know from Claim 3.4.6 that in $f(X)$, the vertices of V_2 must have between 1 and $|S_1| + |S_2| - 1$ chips. Hence, the vertices of V_2 have d_2 chips with $1 \leq d_2 \leq |S_1| + |S_2| - 1$. After the initial firing, the vertices of V_1 will each have $|S_2|$ chips, having just received from $|S_2|$ richer neighbours, and the vertices of V_2 will each have $d_2 - |S_1|$ chips, having just sent to $|S_1|$ poorer neighbours. By Lemma 3.1.1, we can normalize this by subtracting $d_2 - |S_1|$ from both totals, leaving $|S_1| + |S_2| - d_2$ chips on each of the vertices in V_1 and leaving 0 chips on each of the vertices in V_2. Note that since the x-distance from the least x-coordinate of S_1 to the greatest x-coordinate of S_2 is d_2, then the x-distance from the least x-coordinate of S_2 to the greatest x-coordinate of S_1 must, when added to d_2, equal the sum of the two strip lengths. So, the x-distance from the least x-coordinate of S_2 to the greatest x-coordinate of S_1 is $|S_1| + |S_2| - d_2$. To show that the relative sizes have changed and that in the second firing, the vertices of V_1 will send chips to the vertices of V_2, we must show that $|S_1| + |S_2| - d_2 > 0$. We know that the maximum value that d_2 can take on is $|S_1| + |S_2| - 1$. So,

$$
\begin{aligned}
|S_1| + |S_2| - 1 &\geq d_2 \\
|S_1| + |S_2| - d_2 &\geq 1 \\
|S_1| + |S_2| - d_2 &> 0
\end{aligned}
$$

Thus, we can conclude that the vertices of V_1 are now richer than the vertices of V_2.

This gives us that the configuration following the initial firing, call it $[f(X)]'$, is itself equal to $f(X')$ for some board-pile n-omino X'. In fact, X' is the board-pile n-omino created by reflecting X about the horizontal axis since $[f(X)]'$ represents an interchange of the relative stack sizes of the two sets of vertices, V_1 and V_2, corresponding to the two h-strips in X. So, after the second firing, the vertices of V_2 will have $|S_1|$ chips and the vertices of V_1 will have $|S_1| - d_2$ chips. By adding $d_2 - |S_1|$ to both totals (to counteract our subtracting of $d_2 - |S_1|$ chips previously), we get back where we started with the vertices of V_1 having 0 chips and the vertices of V_2 having d_2 chips. So, we have that $f(X)$ is a period configuration.

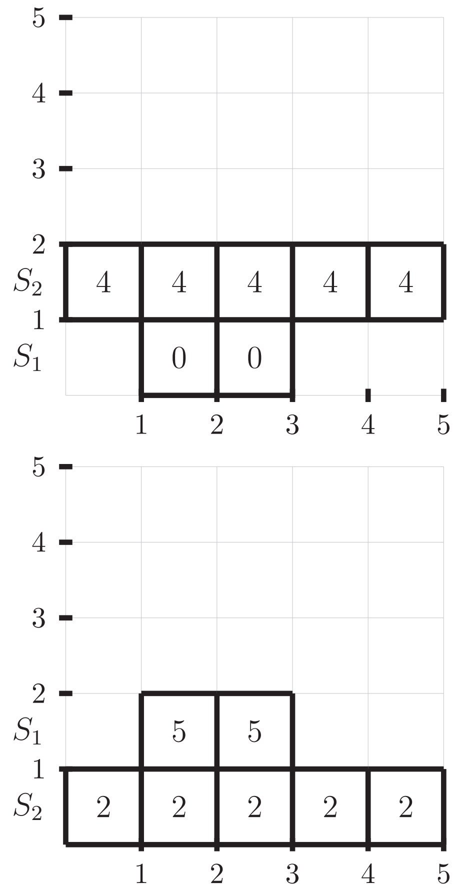


Figure 3.18: Board-pile with two strips "flipping"

We will use this case as the basis of an induction. We will induct on the number of h-strips to show that for all board-pile n-ominoes, X, the initial firing of $f(X)$ yields $f(X')$ (up to an addition of a constant to each of the stack sizes) where X' is the board-pile n-omino created by reflecting X about the horizontal axis. This will imply that $f(X)$ is a period configuration of some complete graph because two firings will return to the original configuration.

Our inductive hypothesis is that for all board-pile polyominoes X with at most k h-strips, the initial firing of $f(X)$ yields $f(X')$ (up to an addition of a constant to each of the stack sizes) where X' is the board-pile polyomino created by reflecting X about the horizontal axis.

Now suppose we have a board-pile polyomino, Y, with exactly $k+1$ h-strips. Then, $f(Y)$ is some configuration of some complete graph. Thus, by Claim 3.4.6, d_{k+1} is bounded so that $1 \leq d_{k+1} \leq |S_k| + |S_{k+1}| - 1$. We know by our inductive hypothesis that if S_{k+1} were removed from Y, that the resulting polyomino would map to a period configuration of some complete graph. Also, the vertices of V_{k+1} in $f(Y)$ will act as source vertices enriching every other vertex upon firing. The addition of such a source strip cannot affect the relative stack sizes of the other vertices within the graph. We know from our base case that the board-pile polyomino Z composed of just the squares of S_k and S_{k+1} is such that $f(Z') = [f(Z)]'$. Therefore, since the effect of the vertices in $V(f(Y)) \setminus V(f(Z))$ will enrich the remaining stack sizes of the graph equally upon firing, we have that Y satisfies our criteria with $f(Y') = [f(Y)]'$.

($\Leftarrow$) Now let C be a period configuration of K_n. We seek to show that there exists a board-pile polyomino X such that $f(X) = C$.

Suppose the vertices of C have N distinct stack sizes

$$\{0^{a_1}, (\sum_{i=1}^{2} d_i)^{a_2}, (\sum_{i=1}^{3} d_i)^{a_3}, \dots (\sum_{i=1}^{N} d_i)^{a_N}\}$$

in ascending order for some $a_1, a_2, \dots, a_N \in \mathbb{N}$ and some $d_1 = 0, d_2, d_3, \dots, d_N$ with $d_2, \dots, d_N \in \mathbb{N}$. Note that by Lemma 3.1.1, we can alter the stack sizes of any configuration so that the minimum stack size becomes zero. Thus, we have not lost generality by assuming that our least stack size is zero. Let V_j be the set of all vertices in C with stack size $(\sum_{i=1}^{j} d_i)^{a_j}$ for all $j \leq N$. Let X be a collection of N h-strips on a plane with exactly one h-strip per row for $y = 1, 2, \dots, N$. Call these h-strips $S_1, S_2, \dots, S_N$, containing $|V_1|, |V_2|, \dots, |V_N|$ unit squares, respectively, with S_i spanning y-coordinates $i-1$ to i for all $i \leq N$. Arrange these strips so that d_j is equal to the x-distance from the leftmost coordinate of S_{j-1} to the rightmost coordinate of S_j. X is a board-pile polyomino if and only if it is a connected plane figure.

In particular, we must prove that $d_{j+1} \leq |V_j| + |V_{j+1}| - 1$ for all j. This will imply that every pair of strips $|S_j|$ and $|S_{j+1}|$ are connected edge on edge, proving that a single board-pile polyomino will result rather than a number of disconnected board-pile polyominoes in the plane.

By contradiction, suppose $d_{j+1} > |V_j| + |V_{j+1}| - 1$ for some j. Then, following the initial firing, the vertices of V_j would each have $(\sum_{i=1}^{j} d_i) + |V_{j+1}| + r$ chips, where r represents the difference between the number of vertices richer than those in V_{j+1} and the number of vertices poorer than those in V_j. Also, following the initial firing, the vertices of V_{j+1} would each have $(\sum_{i=1}^{j+1} d_i) - |V_j| + r$ chips. Since C is a period configuration, we know that $(\sum_{i=1}^{j} d_i) + |V_{j+1}| + r > (\sum_{i=1}^{j+1} d_i) - |V_i| + r$. But, this implies that $d_{j+1} < |V_{j+1}| + |V_j|$ which implies that $d_{j+1} \leq |V_{j+1}| + |V_j| - 1$ which contradicts our assumption. So, for all period configurations, C, $C = f(X)$, for some board-pile polyomino X. □

With it now proven that the number of unlabelled period configurations of K_n is equal to the number of board-pile n-ominoes, we know the following two results to be true of unlabelled complete graph period configurations.

Corollary 3.4.8. *The number of period configurations of an unlabelled complete graph on n vertices follows the recurrence relation $a_n = 5a_{n-1} - 7a_{n-2} + 4a_{n-3}$ for $n \geq 5$ with initial values $a_1 = 1$, $a_2 = 2$, $a_3 = 6$, and $a_4 = 19$ (the same recurrence shown to calculate the number of board-pile polyominoes on n vertices [13]).*

Corollary 3.4.9. *The number of period configurations of an unlabelled K_k has an asymptotic value of approximately 0.1809×3.2056^k.*

How many period configurations exist on *labelled* complete graphs? We know that a configuration C is a period configuration if and only if $f^{-1}(C)$ is a board-pile polyomino.

Our method must involve finding all labelled ordered partitions of the vertices. Then, given a labelled ordered partition, we must determine the number of ways that the i^{th} part can be connected edge on edge with parts $i-1$ and $i+1$.

Let K_n be a complete graph. The number of labelled ordered partitions of a complete graph is just the ordered Bell numbers. We can view this as all of the ways that a set of labelled unit squares can be grouped together into ordered strips but without yet affixing the strips together to form a polyomino.

For every possible ordered partition, we must determine all of the possible stack sizes which could be shared by the vertices in each respective part. Recall that this is equivalent to determining all of the ways that an ordered set of strips can be oriented with respect to each other so as to create a polyomino.

For each period orientation of K_n, assign all possible stack sizes by using the product $\prod_{i=1}^{N}(|S_{i-1}+|S_i|-1)$ (the product of the number of ways each strip S_i with length $|S_i|$ can be adjacent to its neighbours). Give the period orientations of K_n some ordering from 1 to $R(K_n)$. Let $|S_i^j|$ be the length of the i^{th} strip in the j^{th} period orientation of K_n. Then we reach that the number of labelled period configurations that exist on K_n is

$$\sum_{j=1}^{R(K_n)}\prod_{i=1}^{N}(|S_{i-1}^j|+|S_i^j|-1).$$

3.5 p_2-Configurations on Stars

Unlike paths and complete graphs, the number of period configurations on a star, $K_{1,n}$, $n \geq 2$, can be proven quickly and can be calculated with an explicit formula.

Theorem 3.5.1. *There exist $2n^n+1$ different period configurations on $K_{1,n}$, $n \geq 2$.*

Proof. By Theorem 3.2.18, there are only 3 period orientations on any star $K_{1,n}$, $n \geq 2$. Those being the orientation with all flat edges, the orientation with all edges directed toward the center vertex, and the orientation with all edges directed away from the center.

Case 1: *All edges are flat.*

In this case, each vertex must have zero chips. So, there is only one period configuration which induces this orientation on any star $K_{1,n}$, $n \geq 2$.

Case 2: *All edges are directed toward the center vertex.*

Suppose that the center vertex has 0 chips. Then every other stack size must be positive. Also, we know that following the initial firing, every arrow in the induced orientation must change direction. In $K_{1,n}$, the center vertex will come out of the initial firing with exactly n chips. The leaves must have between 1 and n chips initially, so as to be richer than the center vertex initially yet poorer than the center vertex following the initial firing. There exist n^n period configurations which induce this orientation.

Case 3: *All edges are directed away from the center vertex.*

By Corollary 1.2.2, every period of length 2 must contain one configuration which induces an orientation from Case 2 and one configuration which induces an orientation from Case 3. Thus, there must exist exactly one period configuration in which the induced orientation has all edges pointing away from the center vertex for every period configuration in which the induced orientation has all edges pointing toward the center vertex. Thus, this case must have n^n possible period configurations as well.

Thus, $K_{1,n}$ has $2n^n + 1$ different period configurations.

□

Chapter 4

Quantum Parallel Diffusion

We define Quantum Parallel Diffusion on a graph G as the variant of Parallel Diffusion in which the initial configuration is the configuration in which every stack size is zero, and the firing at step 0 is such that for some $H \subseteq V(G)$, for each vertex $v \in H$, v sends a chip to each of its neighbours. We call H a *quantum set.* In every subsequent step, Quantum Parallel Diffusion is identical to Parallel Diffusion. Quantum Parallel Diffusion allows for chips to be sent in instances where chips would not normally be sent in Parallel Diffusion. In Parallel Diffusion, chips will be sent at every step unless every stack size is equal. Quantum Parallel Diffusion, on the other hand, is only defined on initial configurations in which every stack size is equal.

By Lemma 3.1.16, up to equivalence, the configuration in which every stack size is zero is the only fixed configuration. We will refer to this configuration as *the* fixed configuration. Our study of Quantum Parallel Diffusion will focus on finding quantum sets that will eventually, after some number of steps, return the fixed configuration.

We wanted to study Quantum Diffusion because it shows us how the model could conceivably continue after reaching the fixed configuration, when normally no further chips would be sent. What we ended up finding was that the results in this chapter involved more topics in graph theory than perhaps any other chapter, from dominance to independence to colouring.

After a basic lemma and definition, we present our results that apply to all graphs before moving on to a detailed analysis and counting argument on paths. In Theorem 4.1.3, we show that a quantum set will return the fixed configuration in two steps if and only if it follows a particular domination criterion, *complementary component dominance.* There is an associated concept called *quantum quiescence* and in Theorem 4.1.13, we count the number of QQ_2 (or 2-Quantum Quiescent) sets that exist on paths of any length. In Theorem 4.2.1, we show that the number of pre-positions of the fixed configuration that exist on P_n, Z_n, follows the recurrence $Z_n = Z_{n-1} + 2Z_{n-2} + Z_{n-3}$. In Corollary 4.2.2, we show that the asymptotic value of Z_k is approximately 0.3885×2.1479^k.

In this chapter, much like Chapter 3, we have very few previous results to lean on. Long and Narayanan's result regarding periodicity was very useful throughout Chapter 3, but it will be much less useful in this chapter. In this chapter we will be studying configurations that can arise after the initial firing in Quantum Parallel Diffusion and configurations which eventually lead to the fixed configuration. These are questions about pre-period configurations, a topic which Long and Narayanan did not touch on.

Definition 4.0.1. *Let G be a graph with the fixed configuration and H be a subset of $V(G)$. A **quantum firing** of H is when the vertices of H each send a chip to each of their respective neighbours in G. We call H a **quantum set**.*

Definition 4.0.2. *Given a configuration C, a **pre-position** of C is a configuration D such that if D is the configuration at step t, then C is the configuration at step $t+1$.*

An example of a quantum firing is shown in Figure 4.1.

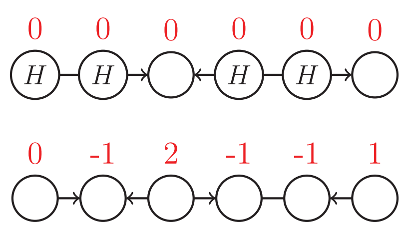


Figure 4.1: Quantum firing of the quantum set H of $V(P_6)$ with directed edges depicting the flow of chips from elements of H to elements excluded from H.

Due to the similarities between the rules of Parallel Diffusion and Quantum Parallel Diffusion, the following lemma applies to both processes.

Lemma 4.0.3. *Given a graph, G, and an orientation R, if there exists a configuration which induces R and is a pre-position of the fixed configuration (all stack sizes equal to zero), then it is unique and can be determined based on R.*

Proof. Let G be a graph and R an orientation of G. The orientation R dictates the number of chips that each vertex will send and receive at the next firing. Thus, for each vertex v_k in G, the stack size of v_k following the next firing is equal to the current stack size of v_k plus the number of edges directed toward v_k, A_{v_k}, minus the number of edges directed away from v_k, B_{v_k}. So, if we have that $|v_k| + A_{v_k} - B_{v_k} = 0$, then the stack size of v_k can be determined because it is the only unknown in the equation. $\square$

Definition 4.0.4. *A configuration at step t in a configuration sequence is a* 0***-pre-position*** *if the configuration at step $t+1$ is the fixed configuration. An orientation, R, is a* ***0-preorientation*** *if there exists a* 0*-pre-position which has R as its induced orientation.*

4.1 0_2-invoking subsets

We establish the concept of 0_2-invoking subsets and apply it to all graphs before focusing on paths in particular. In Theorem 4.1.13, we determine the number of 0_2-invoking subsets that exist on any given path. The following definitions will help us to answer our primary question about Quantum Parallel Diffusion: Which quantum sets eventually lead us back to the fixed configuration?

Definition 4.1.1. *Let G be a graph and let H be a subset of $V(G)$.*

- *We say H is* 0***-invoking*** *if a quantum firing of H in G results in a configuration C such that $Seq(C)$ has a period length of 1.*
- *H is* 0_2***-invoking*** *if a quantum firing of H results in a configuration C such that the configuration at the next time step, call it C_1, is the fixed configuration.*
- *The* ***quantum quiescent number*** *or* ***QQ number*** *of a graph G, denoted* ***QQ***(G)*, is the size of a smallest nontrivial* 0*-invoking subset of $V(G)$. So, $QQ(G) = min\{|H| : H \neq \emptyset$ is 0-invoking$\}$.*
- *The* ***2-quantum quiescent number*** *or* ***QQ***$_2(G)$ *is the size of a smallest nontrivial* 0_2*-invoking subset of $V(G)$. So, $QQ_2(G) = min\{|H| : H \neq \emptyset$ is 0_2-invoking$\}$.*

In other words, a 0-invoking set is one which eventually, after some number of steps, yields the fixed configuration (recall from Lemma 3.1.16 that the fixed configuration is the only configuration that can exist inside a period of length one, up to equivalence). Note that QQ(G) and QQ$_2(G)$ are well-defined because $V(G)$ is itself both a 0-invoking subset and a 0_2-invoking subset of $V(G)$.

Definition 4.1.2. *Let G be a graph. A subset H of $V(G)$ is Complementary Component Dominant (or* ***CCD****) if*

- *For all adjacent pairs of vertices, $x, y \in H$, the number of neighbours of x in $V(G) \setminus H$ is equal to the number of neighbours of y in $V(G) \setminus H$*

  and

- *For all adjacent pairs of vertices, $u, v \in V(G) \setminus H$, the number of neighbours of u in H is equal to the number of neighbours of v in H.*

Note that this definition implies that if H is complementary component dominant in G, then so is $V(G) \setminus H$. An example of a complementary component dominant subset is given in Figure 4.2.

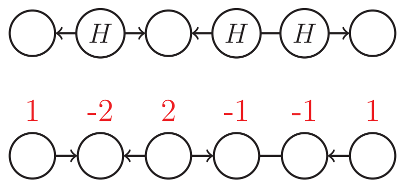


Figure 4.2: Graph P_6 with complementary component dominant vertex subset, H

Theorem 4.1.3. *Let G be a graph with the fixed configuration. A subset H of $V(G)$ is 0_2-invoking in G if and only if H is CCD.*

Proof. ($\Leftarrow$) Let a graph G have the fixed configuration. Suppose $H \subseteq V(G)$ is CCD. In Figure 4.3, we see $G|_H$ and $G|_{V(G)\setminus H}$ separated into their respective connected components.

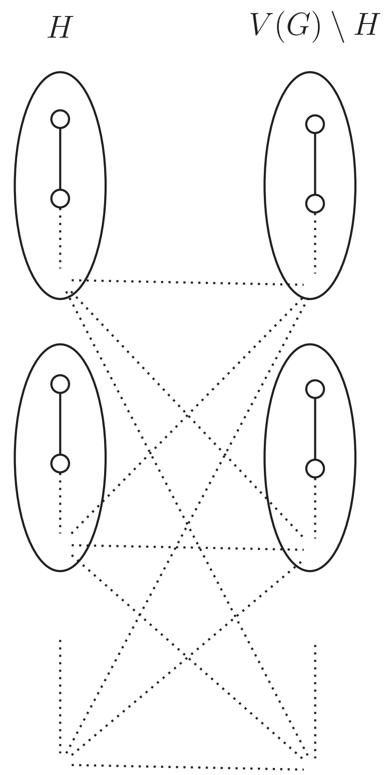


Figure 4.3: Graph, G, with CCD subset, H, of $V(G)$

Remember that when every vertex in H quantum fires, the edges that have both endpoints in H will have chips travelling along them both ways. So, we can equivalently see these as flat edges. For all vertices h in H, let $deg_{V(G)\setminus H}(h)$ be the number of vertices in $V(G) \setminus H$ that are adjacent to h, and for all vertices g in $V(G) \setminus H$, let $deg_H(g)$ be the number of vertices in H that are adjacent to g. Thus, if every vertex in H were to fire a chip to each of its neighbours, the resulting configuration would leave every vertex, h, in H with a number of chips equal to $0 - deg_{V(G)\setminus H}(h)$. Every vertex, g, in $V(G) \setminus H$ would be left with $0 + deg_H(g)$ chips. We know from the definition of CCD that every pair of adjacent vertices in H must be adjacent to the same number of vertices in $V(G) \setminus H$. By transitivity, this will extend to entire connected components within $G|_H$.

Since the definition of CCD also dictates that the vertices of $V(G) \setminus H$ follow the same rule with every adjacent pair of vertices being adjacent to the same number of vertices in the complement, we get, by transitivity, that this extends to entire connected components in $G|_{V(G)\setminus H}$. This will leave each connected component within $G|_H$ with a common stack size and each connected component within $G|_{V(G)\setminus H}$ with a common stack size.

Every vertex in H has a negative stack size and each vertex in the complement has a positive stack size. So, when the vertices fire at the next step, every vertex in H will receive from each of its neighbours in $V(G) \setminus H$ and will not send to or receive from any vertices in H. Likewise, every vertex in $V(G) \setminus H$ will send to each of its neighbours in H and will not send to or receive from any vertices in $V(G) \setminus H$. So for each $h \in H$, we get that

$$\begin{aligned} |h|_2 &= |h|_1 + deg_{V(G)\setminus H}(h) \\ &= -deg_{V(G)\setminus H}(h) + deg_{V(G)\setminus H}(h) \\ &= 0 \end{aligned}$$

and for all $g \in V(G) \setminus H$,

$$\begin{aligned} |g|_2 &= |g|_1 - deg_H(g) \\ &= deg_H(g) - deg_H(g) \\ &= 0 \end{aligned}$$

Thus, the fixed configuration is restored in two steps.

($\Rightarrow$) Let H be a quantum set in $V(G)$. Suppose H is 0_2-invoking. This means that if the configuration at $t = 0$ is the fixed configuration, then so is the configuration at $t = 2$. This implies that the net effect of two steps of firings on each vertex is $+0$. Note that every vertex in H will necessarily send a chip to each of its neighbours in $V(G) \setminus H$ in the initial firing and receive from those same vertices in the firing at step 1. So, for all vertices h in H, if h receives a chip from a vertex in H during the firing at step 1, then h must also send a chip to a vertex in H at step 1 as well. However following the initial firing, for each connected component H_i in H, there must exist some vertex in H_i that has no poorer neighbours in H_i. So if any chip is sent from a vertex in H to another vertex in H during the firing at step 1, then there will exist at least one vertex h_i that received a chip from a neighbour in H, but did not send a chip to a neighbour in H. This implies that h_i will have a positive stack size at step 2, having received more chips in the firing at step 1 than it sent in the initial firing. This, however, contradicts our assumption that H is 0_2-invoking. Thus, we can conclude that every vertex in a connected component in $G|_H$ has a common stack size after the initial firing. This implies that each vertex belonging to the same connected component in $G|_H$ shares the same number of neighbours in $V(G) \setminus H$. A similar argument will show the result for vertices in $V(G) \setminus H$. Thus, we can conclude that all 0_2-invoking subgraphs are CCD.

□

Corollary 4.1.4. *If H is 0_2-invoking in G, then so is $V(G) \setminus (H)$.*

Note that not all graphs have a proper non-trivial 0_2-invoking subset. In Figure 4.4, we see such a graph.

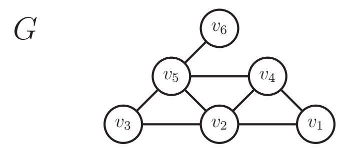


Figure 4.4: Graph G with no proper nontrivial 0_2-invoking subsets

Claim 4.1.5. *Graph G from Figure 4.4 has no proper non-trivial 0_2-invoking subsets.*

Proof. Suppose, by way of contradiction, that H is a 0_2-invoking subset of $V(G)$. By Corollary 4.1.4, we know that if H is 0_2-invoking, then so is $V(G) \setminus (H)$. So we can suppose without loss of generality that v_2 is in H.

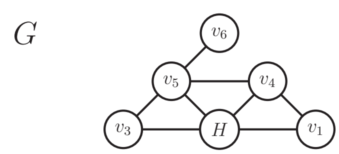


Figure 4.5: Graph G with v_2 in H

Since H is 0_2-invoking, proper, and non-trivial, and since v_6 is a leaf, we know that exactly one of v_5 and v_6 is in H by Theorem 4.1.3.

Case 1: Suppose v_6 is in H.

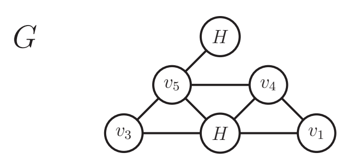


Figure 4.6: Graph G with v_2 and v_6 in H

If v_3 is in H, then every vertex adjacent to v_3 which is in H must be adjacent to exactly one vertex from $V(G) \setminus H$. This implies that both v_1 and v_4 must be in H. This is a contradiction since v_1 is not adjacent to any vertices from $V(G) \setminus H$.

Otherwise if v_3 is not in H, then v_5 must be adjacent to exactly one vertex in H. This is a contradiction.

Case 2: Suppose v_5 is in H.

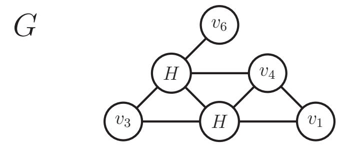


Figure 4.7: Graph G with v_2 and v_5 in H

We know that neither v_3 nor v_6 can be in H since they are not adjacent to any vertices in $V(G) \setminus H$. Thus, since v_5 is adjacent to both of these vertices, every edge in H must be adjacent to at least two vertices in $V(G) \setminus H$. So neither v_1 nor v_4 can be in H. Since the set $H = \{v_2, v_5\}$ is not CCD, we get a contradiction.

□

Definition 4.1.6. *[18] In a graph G, $S \subseteq V(G)$ is a* ***dominating set*** *if every vertex not in S has a neighbour in S. The* ***domination number*** *$\gamma(G)$ is the minimum size of a dominating set in G.*

Corollary 4.1.7. *Given a graph G, all non-trivial 0_2-invoking subsets of $V(G)$ are also dominating sets.*

Proof. Let G be a graph and let $H \subseteq V(G)$ be a 0_2-invoking subset. By Theorem 4.1.3, H is CCD. By the definition of CCD, every vertex in the complement of H must be adjacent to at least one vertex in H unless H is empty. So if H is non-trivial, then H is dominating. □

From [4], an *efficient dominating set*, or *perfect code*, is an independent subset, A, of the vertex set of a graph, G, such that every vertex in $V(G) \setminus A$ is adjacent to exactly one vertex in A.

Corollary 4.1.8. *Efficient dominating sets (or perfect codes) are CCD and thus 0_2-invoking.*

Lemma 4.1.9. *Every minimal dominating set on P_n, $n \geq 2$, is CCD and thus, 0_2-invoking.*

Proof. Let H be a minimal dominating set on P_n. We will show that H is CCD. Since H is a dominating set, every vertex in $V(P_n) \setminus H$ that is adjacent to another vertex in $V(P_n) \setminus H$ must also be adjacent to exactly one vertex in H. Since H is a minimal dominating set, every vertex in H that is adjacent to another vertex in H must also be adjacent to exactly one vertex in $V(P_n) \setminus H$. Thus, H is CCD. □

Question 4.1.10. *Is there a characterization of minimal dominating sets that are also* 0_2*-invoking subsets?*

If $\gamma(G) = 1$, then there must be a dominating vertex. This vertex is itself a 0_2-invoking set. If $\gamma(G) = 2$, with dominating set $\{x, y\}$, then the solution is not so simple. We will break the problem into two cases: x not adjacent to y, and x adjacent to y. Suppose first that x and y are not adjacent to each other. For this pair of vertices to also be a 0_2-invoking set, it must be true that the set $\{x, y\}$ is also complementary component dominant.

So, every vertex in a given connected component in $G \setminus \{x, y\}$ must be adjacent to the same number of vertices in $\{x, y\}$ (either 1 or 2). Consider the subset of vertices adjacent to x and not adjacent to y, call it V_x, and the subset of vertices adjacent to y and not adjacent to x, call it V_y, and the subset of vertices adjacent to both x and y, call it V_{xy}. In order for $\{x, y\}$ to be complementary component dominant, it must be true that no edges exist between V_{xy} and $V_x \cup V_y$.

Now, if x and y are adjacent,we must also have an additional rule since $\{x, y\}$ is CCD. If x is adjacent to y, then we have the additional rule that $|V_x| = |V_y|$ since both x and y must be adjacent to the same number of vertices. Moving to dominating sets of size 3 or greater appears to be much more difficult.

In complete multi-partite graphs, minimal dominating sets come in two forms: either one vertex from two different parts, or an entire part. The former is not necessarily CCD, while the latter is necessarily CCD. In a multi-partite graph $K_{n,n,n,\ldots,n}$, a set composed of one vertex from each part is not minimally dominating, but is 0_2-invoking.

Question 4.1.11. *Is there a graph such that some subset of its vertex set is 0-invoking but not* 0_2*-invoking.*

4.1.1 0_2-invoking subsets on Paths

Let J_n represent the number of 0_2-invoking sets that exist on P_n. We will now look at the issue of counting all 0_2-invoking subsets on a path with n vertices. With Theorem 4.1.13, we determine a recurrence relation for calculating J_n for all $n \geq 3$.

Lemma 4.1.12. *Let $H \subset V(P_n) = \{v_1, v_2, \ldots, v_{n-1}, v_n\}$ be proper, non-trivial, and 0_2-invoking. Then $v_n \in H$ if and only if $v_{n-1} \in V(G) \setminus H$.*

Proof. Let $H \subset V(P_n) = \{v_1, v_2, \ldots v_{n-1}, v_n\}$ be 0_2-invoking, proper and nontrivial. ($\Rightarrow$) Suppose first that $v_n \in H$. We know that v_{n-1} is the only neighbour of v_n in P_n. If $v_{n-1} \in H$, then v_n would be adjacent to 0 vertices in $V(P_n) \setminus V(H)$ and thus, since H is 0_2-invoking, every vertex in the same connected component as v_n in $G|_H$ would be adjacent to 0 vertices in $V(G) \setminus H$. Since P_n is connected, this implies that H is not a proper subset of $V(P_n)$ which is a contradiction. Thus, if $v_n \in H$, then $v_{n-1} \in V(G) \setminus H$.
($\Leftarrow$) Suppose now that $v_{n-1} \in V(G) \setminus H$. Then if $v_n \in V(G) \setminus H$, it would be adjacent to 0 vertices in H and thus, since H is 0_2-invoking, every vertex in the same connected component as v_n in $G|_H$ would be adjacent to 0 vertices in H. Since P_n is connected, this implies that H is the trivial subset of $V(G)$ which is a contradiction. Thus, if $v_{n-1} \in V(G) \setminus H$, then $v_n \in H$. □

Theorem 4.1.13. *$J_n = J_{n-1} + J_{n-2} - 2$, for $n \geq 3$, with $J_1 = 2$ and $J_2 = 4$.*

Proof. Note first that we are including the trivial and improper cases, so as to count every 0_2-invoking set on P_n. We begin with the initial values. The path with only one vertex cannot send chips because it has no edges. Thus, whether the lone vertex quantum fires or not, the chosen set is 0_2-invoking. So, P_1 has two 0_2-invoking subsets: $\emptyset$ and $V(P_1)$. On P_2, a quantum firing of any subgraph will return to the fixed configuration after another step. Thus, P_2 has four 0_2-invoking subsets.

Trivially, the empty subset and the entire vertex set are 0_2-invoking in P_n. We will take note of this and move forward counting the 0_2-invoking subgraphs that are both nonempty and have nonempty complement.

We will view the problem of partitioning the vertices of a path into H and $V(G) \setminus H$ as a colouring problem, colouring the vertices of P_n, $n \geq 2$, red if they are in H and blue if they are in $V(G) \setminus H$. Suppose we have P_n coloured in such a way that H (and thus, also $V(G) \setminus H$) is a 0_2-invoking subset. Suppose also that at least one vertex is red and at least one vertex is blue. We will now count all such possible colourings and we will refer to these as **0_2-invoking colourings**.

By Lemma 4.1.12, we know that v_n and v_{n-1} must be different colours, see Figure 4.8.

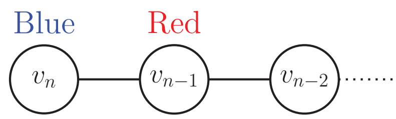


Figure 4.8: The two vertices, v_n and v_{n-1} must have different colours since they are adjacent to different numbers of blue vertices.

Suppose P_n has a 0_2-invoking colouring. Look at the colours assigned to the final three vertices: v_n, v_{n-1}, and v_{n-2}. By Lemma 4.1.12, we are able to exclude some possible colourings of these final three vertices. See Table 4.1.

Colouring of $v_nv_{n-1}v_{n-2}$	# of 0_2-invoking colourings
$RRR\cdots$	0
$RRB\cdots$	0
$RBR\cdots$	?
$RBB\cdots$	?
$BRR\cdots$	?
$BRB\cdots$	?
$BBR\cdots$	0
$BBB\cdots$	0

Table 4.1: Colourings of the last three vertices of a path

For the four remaining possible colourings of v_n, v_{n-1}, and v_{n-2}, we will develop a recurrence relation, building on values from smaller paths.

Case 1: Suppose P_n has a 0_2-invoking colouring in which v_n is red, v_{n-1} is blue, and v_{n-2} is red. By Theorem 4.1.3, a colouring is 0_2-invoking if and only if the two colour classes are CCD. Clearly if we were to remove v_n from this colouring, yielding a colouring on P_{n-1}, the resulting colouring would be CCD since v_{n-1}, the only vertex which was adjacent to v_n, is not adjacent to any other blue vertices. Thus, for every 0_2-invoking colouring of P_n in which v_n is red, v_{n-1} is blue, and v_{n-2} is red, there exists exactly one 0_2-invoking colouring of P_{n-1} in which v_{n-1} is blue and v_{n-2} is red. Since there is no fundamental difference between the colours red and blue, and the final two vertices must have opposing colours by Lemma 4.1.12, the number of 0_2-invoking colourings of P_{n-1} in which v_{n-1} is blue and v_{n-2} is red is equal to half of the total number of 0_2-invoking colourings of P_{n-1}. The number of 0_2-invoking colourings of P_{n-1} is equal to $J_{n-1}-2$ (remembering to account for the improper and trivial cases which are 0_2-invoking but are not defined to be 0_2-invoking colourings). Thus, the number of 0_2-invoking colourings of P_n in which v_n is red, v_{n-1} is blue and v_{n-2} is red is equal to $\frac{1}{2}(J_{n-1}-2) = \frac{1}{2}J_{n-1} - 1$.

Case 2: Suppose P_n has a 0_2-invoking colouring in which v_n is blue, v_{n-1} is red, and v_{n-2} is blue. Since there is no fundamental difference between the colours red and blue, we know that there are also $\frac{1}{2}J_{n-1}-1$ 0_2-invoking colourings of this form on P_n.

Colouring of $v_n v_{n-1} v_{n-2} \ldots$	# of 0_2-invoking colourings
$RRR\cdots$	0
$RRB\cdots$	0
$RBR\cdots$	$\frac{1}{2}J_{n-1}-1$
$RBB\cdots$	
$BRR\cdots$	
$BRB\cdots$	$\frac{1}{2}J_{n-1}-1$
$BBR\cdots$	0
$BBB\cdots$	0

Table 4.2: Colourings of the last three vertices of a path

Case 3: Suppose P_n has a 0_2-invoking colouring in which v_n is red, and both v_{n-1} and v_{n-2} are blue. By Theorem 4.1.3, we know that both colour sets are CCD. So we know that v_{n-2} must be adjacent to a red vertex. Thus, v_{n-3} is red. However, we do not know whether v_{n-4} is red or blue. We have no knowledge of the remainder of the colours except that both colour sets are CCD. If we were to remove v_n and v_{n-1}, the resulting colouring of P_{n-2} would be 0_2-invoking because the only vertex adjacent to either of these vertices is v_{n-2}, and in the resulting colouring of P_{n-2}, v_{n-2} is not adjacent to any other blue vertices. Thus, the number of 0_2-invoking colourings of P_n in which v_n is red, and both v_{n-1} and v_{n-2} are blue is equal to the number of 0_2-invoking colourings of P_{n-2} in which v_{n-2} is blue and v_{n-3} is red. Since there is no fundamental difference between the colours red and blue, and the final two vertices must have opposing colours by Lemma 4.1.12, the number of 0_2-invoking colourings of P_{n-2} in which v_{n-2} is blue and v_{n-3} is red is equal to half of the total number of 0_2-invoking colourings of P_{n-2}. The number of 0_2-invoking colourings of P_{n-2} is equal to $J_{n-2}-2$ (remembering to account for the improper and trivial cases which are 0_2-invoking but are not defined to be 0_2-invoking colourings). Thus, the number of 0_2-invoking colourings of P_n in which v_n is red, and both v_{n-1} and v_{n-2} are blue is equal to $\frac{1}{2}(J_{n-2}-2)=\frac{1}{2}J_{n-2}-1$.

Case 4: Suppose P_n has a 0_2-invoking colouring in which v_n is blue, and both v_{n-1} and v_{n-2} are blue. Since there is no fundamental difference between the colours red and blue, we know that there are also $\frac{1}{2}J_{n-2}-1$ 0_2-invoking colourings of this form on P_n.

Colouring of $v_n v_{n-1} v_{n-2} \ldots$	# of 0_2-invoking colourings
$RRR \cdots$	0
$RRB \cdots$	0
$RBR \cdots$	$\frac{1}{2}J_{n-1} - 1$
$RBB \cdots$	$\frac{1}{2}J_{n-2} - 1$
$BRR \cdots$	$\frac{1}{2}J_{n-2} - 1$
$BRB \cdots$	$\frac{1}{2}J_{n-1} - 1$
$BBR \cdots$	0
$BBB \cdots$	0

Table 4.3: Colourings of the last three vertices of a path

So $J_n - 2 = J_{n-1} - 2 + J_{n-2} - 2$. Therefore $J_n = J_{n-1} + J_{n-2} - 2$. □

Corollary 4.1.14. *Let F_i be the i^{th} Fibonacci number with $F_0 = 0$, $F_1 = 1$, and $F_i = F_{i-1} + F_{i-2}$. Then $J_{k+1} = 2(F_k + 1)$.*

Proof. Note that

$J_1 = 2(F_0 + 1) = 2;$

$J_2 = 2(F_1 + 1) = 4;$

$J_3 = 2(F_2 + 1) = 4.$

Assume for $2 \leq i \leq k$ that $J_i = 2(F_{i-1} + 1)$. Then

$$\begin{aligned} J_{k+1} &= J_k + J_{k-1} - 2 \\ &= 2(F_{k-1} + 1 + F_{k-2} + 1) - 2 \\ &= 2(F_{k-1} + F_{k-2} + 1) \\ &= 2(F_k + 1). \end{aligned}$$

□

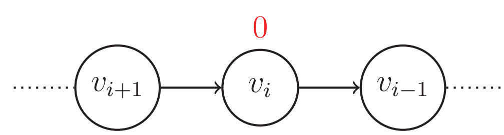
With Theorem 4.1.3 we determined what 0_2-invoking subsets look like and with Theorem 4.1.13 we determined the number of 0_2-invoking subsets that exist on a path of any length. With the following lemma, we combine these ideas to determine what properties are known about 0_2-invoking subsets on paths.

Lemma 4.1.15. *Let C be a 0-pre-position, which is not itself the fixed configuration, on a path, P_n. Let R be the orientation induced by C. If there exists some 0_2-invoking subset H of $V(P_n)$ such that when H is quantum fired from the fixed configuration, it results in the configuration C. Then the following three statements are all true.*

- *R has no agreeing pair of adjacent arrows (recall Definition 3.2.8);*
- *$P_n|_H$ has no P_3 subgraph (recall Definition 1.1.4);*
- *C has no stack sizes equal to 0.*

Proof. Let C be a 0-pre-position, which is not itself the fixed configuration, on a path, P_n. Let R be the orientation induced by C, and let H be the set of vertices that would need to fire from the fixed configuration to create C.

First, we will prove that R has no agreeing pair of adjacent arrows. Suppose, by way of contradiction, that R contains a pair of adjacent agreeing arrows. Then, since C is a 0-pre-position, we know that the vertex adjacent to both of these agreeing arrows, call it v_i, must have a stack size of 0 in C. Without loss of generality, we suppose these are both right arrows. Since v_{i-1} is receiving from v_i, we know that $|v_{i-1}|^C < 0$ and since v_{i+1} is sending to v_i, we know that $|v_{i+1}|^C > 0$.



Since v_i is adjacent to a vertex with fewer than 0 chips in C, call it v_{i-1}, we know that both v_i and v_{i-1} are in H. Also, since $|v_{i+1}|^C > 0$, we know that $v_{i+1} \notin H$. However, this implies that the quantum firing left v_i with a stack size of -1 which is a contradiction.

We now prove that C has no stack sizes equal to 0. Suppose, by way of contradiction, that C contains a vertex with stack size 0, call it v. Then either v is incident to a pair of agreeing arrows, or v is incident to only flat edges. The case in which v is incident with a pair of agreeing arrows has already been contradicted. If v is incident to only flat edges, then all of v's neighbours must have a stack size of 0 and thus, be incident to only flat edges. This implies that every vertex in C has a stack size of 0. So, C is the fixed configuration. This is a contradiction.

We now prove H has no P_3 subpath. Suppose, by way of contradiction, that H contains a P_3 subpath. Then C contains a vertex with a stack size of 0. This has already been contradicted.

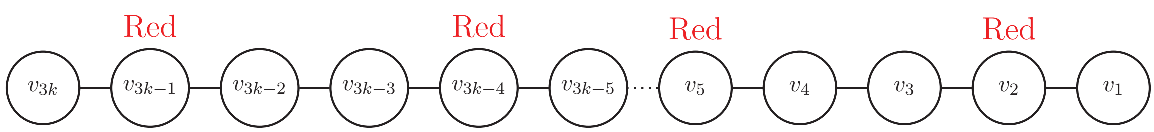
□

Theorem 4.1.16. *$QQ_2(P_n) = \lceil \frac{n}{3} \rceil$ (where $QQ_2(P_n)$ is as in Definition 4.1.1).*

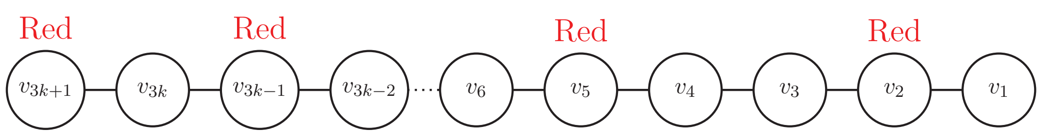
Proof. We will first prove that $QQ_2(P_n) \geq \lceil \frac{n}{3} \rceil$, and then prove that $QQ_2(P_n) \leq \lceil \frac{n}{3} \rceil$. ($\Rightarrow$) The domination number of a path P_n is $\lceil \frac{n}{3} \rceil$, see [7]. Let H be a nontrivial 0_2-invoking subset of $V(P_n)$. By the definition of 0_2-invoking, if a vertex v in $V(P_n) \setminus H$ does not have any neighbours in H, then neither do the neighbours of v. This implies that either H is trivial or every vertex in $V(P_n) \setminus H$ is adjacent to a vertex in H. Since we have supposed that H is nontrivial, we conclude that H is a dominating set. So $QQ_2(P_n) \geq \lceil \frac{n}{3} \rceil$.

($\Leftarrow$) We will partition the possible numbers of vertices into three cases: paths of length $3k$, $3k+1$, and $3k+2$, where k is an integer. We will represent the 0_2-invoking subsets as red vertices.

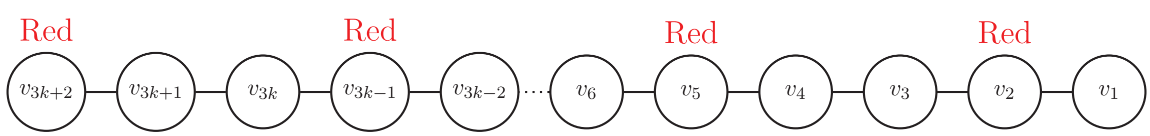
Case 1:



Case 2:



Case 3:



In both Case 1 and Case 3, the coloured vertices are simply the vertices with index equal to 2 (mod 3). In Case 2, since v_{3k+2} is excluded, we instead colour the leaf, v_{3k+1}. Note that in each case, $\lceil \frac{n}{3} \rceil$ vertices are shown to be in H. So $QQ_2(P_n) \leq \lceil \frac{n}{3} \rceil$. □

We do not yet have an example of a configuration on any graph that can arise from a quantum firing that will eventually lead to the fixed configuration in any more than

1 step. That is, we have not yet found a graph for which any subset of its vertices is 0-invoking but not 0_2-invoking. For a thorough example, we will show that every subset of the vertices in $P_4 = v_1v_2v_3v_4$ is either 0_2-invoking or yields a configuration sequence with period 2. The 0_2-invoking subsets are $\{\emptyset\}$, $\{v_1, v_4\}$, $\{v_2, v_3\}$, $\{v_1, v_3\}$, $\{v_2, v_4\}$, and $\{v_1, v_2, v_3, v_4\}$. This means that the remaining subsets that must be tested to determine if they are 0-invoking are:

- $\{v_1\}$ (which is equivalent to $\{v_4\}$)
- $\{v_2\}$ (which is equivalent to $\{v_3\}$)
- $\{v_1, v_2\}$ (which is equivalent to $\{v_3, v_4\}$)
- $\{v_1, v_3\}$ (which is equivalent to $\{v_2, v_4\}$)
- $\{v_1, v_2, v_3\}$ (which is equivalent to $\{v_2, v_3, v_4\}$)
- $\{v_1, v_2, v_4\}$ (which is equivalent to $\{v_1, v_3, v_4\}$)

We will show that on P_4 no other quantum set yields a period of length 1.

	v_4	v_3	v_2	v_1
Step 0	0	0	0	0
Step 1	0	0	1	-1
Step 2	0	1	-1	0
Step 3	1	-1	1	-1
Step 4	0	1	-1	0

	v_4	v_3	v_2	v_1
Step 0	0	0	0	0
Step 1	0	1	-2	1
Step 2	1	-1	0	0
Step 3	0	1	-1	0
Step 4	1	-1	1	-1
Step 5	0	1	-1	0

	v_4	v_3	v_2	v_1
Step 0	0	0	0	0
Step 1	0	1	-1	0
Step 2	1	-1	1	-1
Step 3	0	1	-1	0

Figure 4.9: Quantum sets on P_4 that yield a period of length 2.

	v_4	v_3	v_2	v_1
Step 0	0	0	0	0
Step 1	1	-2	2	-1
Step 2	0	0	0	0

	v_4	v_3	v_2	v_1
Step 0	0	0	0	0
Step 1	1	-1	0	0
Step 2	0	1	-1	0
Step 3	1	-1	1	-1
Step 4	0	1	-1	0

	v_4	v_3	v_2	v_1
Step 0	-1	2	-1	0
Step 1	0	0	1	-1
Step 2	0	1	-1	0
Step 3	1	-1	1	-1
Step 4	0	1	-1	0

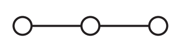
Figure 4.10: More quantum sets on P_4 that yield a period of length 2.

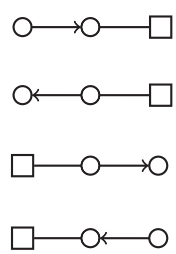
4.2 Counting 0-Preorientations

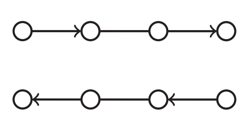
Previously, when counting configurations, we used a method of first counting the orientations that could possibly be induced (Section 3.2). However, given a path, P_n, and an orientation, R, by Lemma 4.0.3, we know that there is a maximum of one configuration that both induces R and is a 0-pre-position. Thus, we can equivalently count all of the orientations of P_n and then subtract those that cannot exist as the induced orientation of a 0-pre-position. Very little is known about pre-periods in Parallel Diffusion. In this section we look at counting all of the pre-positions of just the fixed configuration. This is the first step in solving for which pre-period configurations will eventually lead to the fixed configuration and which will lead to a p_2-configuration, a problem which remains open.

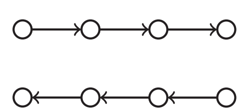
In order to count all 0-pre-positions that exist on paths, we are now going to do a forbidden subgraph characterization to exclude every suborientation which cannot exist within a graph orientation induced by a 0-pre-position.

Theorem 4.2.1. *Let Z_n be the number of orientations R that exist on P_n such that R is a 0-preorientation and not itself the fixed orientation. Then, $Z_n = Z_{n-1} + 2Z_{n-2} + Z_{n-3}$, with initial values $Z_1 = 0$, $Z_2 = 2$, and $Z_3 = 4$. Additionally, a path orientation, which is not itself the fixed orientation, is a 0-preorientation if and only if none of the following mixed graphs exist as a suborientation (A square will represent a vertex that must be a leaf. A circle will represent a vertex that may or may not be a leaf.).*

(a) 

(b) 

(c) 

(d) 

The proof of this theorem is composed of several steps. After a few base cases, we use a case analysis to show that there does not exist an orientation which both contains a forbidden suborientation and is induced by a 0-pre-position. Then, moving to the converse, we begin by finding a way to partition every orientation on a path of length n

into four cases. We then use a case analysis to show that every orientation which does not contain one of the four forbidden suborientations must be a 0-preorientation. This case analysis also implies the recursive relation from the statement of this theorem.

Proof. We begin with the base cases. The path on 1 vertex clearly has no orientations since there are no edges. The path on 2 vertices has one edge and thus, three possible orientations. One of these three orientations is the fixed configuration. However, the other two are 0-pre-positions as evidenced by Figure 4.11.

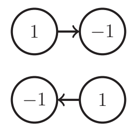


Figure 4.11: Two 0-pre-positions on K_2 with different induced graph orientations

The path on 3 vertices has three edges and thus, nine possible orientations.

Orientation
$\rightarrow\rightarrow$
$\rightarrow -$
$\rightarrow\leftarrow$
$- \rightarrow$
$- -$
$- \leftarrow$
$\leftarrow\rightarrow$
$\leftarrow -$
$\leftarrow\leftarrow$

Table 4.4: Orientations of P_3

Note that one of these orientations is the fixed orientation. On P_3, no orientation with a flat edge can be a 0-preorientation except for the fixed orientation itself. This is because, in each of the four cases, one stack size incident with the flat edge will change in the initial firing and the other will not. This implies that these two values will not be equal at the next step. Thus, no configuration which induces any of these four orientations can be a 0-pre-position, and thus, none of these four orientations is a 0-preorientation.

Orientation	
$\rightarrow\rightarrow$	
$\rightarrow -$	X
$\rightarrow\leftarrow$	
$- \rightarrow$	X
$- -$	
$- \leftarrow$	X
$\leftarrow\rightarrow$	
$\leftarrow -$	X
$\leftarrow\leftarrow$	

Table 4.5: Orientations of P_3 with X's signifying those that are not 0-preorientations

The other four orientations, however, are 0-preorientations, as evidenced by Figure 4.12.

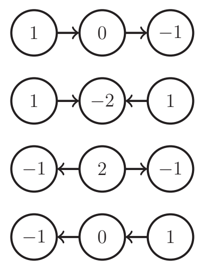


Figure 4.12: Four 0-pre-positions on P_3 with different induced graph orientations

Let R be a path orientation that contains a forbidden suborientation of form $(a), (b), (c)$, or (d), from the statement of this theorem. We will show that R is not a 0-preorientation.

(a) Suppose that in R, two flat edges are incident to one another. The vertex that is incident to both of these edges must initially have 0 chips since its stack size will go unchanged in the initial firing. This implies that both adjacent vertices must also initially have 0 chips. Any vertex with 0 chips that is known to be incident with at least one flat edge cannot be adjacent to any directed edges. Thus, every edge is flat. Therefore, every vertex must initially have 0 chips. This configuration is the fixed configuration.

(b) Suppose that in R, some leaf is incident to a flat edge. Then that leaf must initially have 0 chips since its stack size will go unchanged in the initial firing. This implies that the vertex adjacent to this leaf must also initially have 0 chips. Thus, every vertex must initially have 0 chips. This configuration is the fixed configuration.

(c) Suppose that in R, some flat edge is adjacent to two agreeing arrows. The two vertices incident with this flat edge must initially have equal stack sizes, but in the initial firing one of them will increase and the other will decrease. Therefore, that flat edge will not be maintained. Thus, no configuration inducing R can be a 0-pre-position.

(d) Suppose that in R, some edge is adjacent to two arrows that agree with it. That is, either all three edges are right edges or all three edges are left edges. Then, the two vertices incident with this edge will not have their stack sizes change in the initial firing. Therefore, the arrow will be maintained, not becoming a flat edge. Thus, no configuration inducing R can be a 0-pre-position.

Thus, if R contains one of the four forbidden suborientations from the statement of this theorem, then R is not a 0-preorientation.

For the converse, suppose we have an orientation, R, of a path, P_n, such that R does not contain any of the forbidden suborientations. We seek to show that R is a 0-preorientation.

We will look at every orientation that does not contain one of the four forbidden suborientations and show that they are all 0-preorientations.

We must first generate every orientation of P_n that does not contain a forbidden suborientation. Our method will involve looking only at the final three edges, e_{n-1}, e_{n-2}, and e_{n-3}, and supposing that the remainder of the path does not contain any forbidden suborientations. This will allow us to construct a recursive way of calculating the number of 0-preorientations, Z_n, that exist on P_n, based on Z_{n-1}, Z_{n-2}, and Z_{n-3}.

Begin by noting that if we are to construct every orientation of P_n that does not contain any of our four forbidden subgraphs, then by suborientations (a) and (b), e_{n-1} cannot be flat.

We now construct Tables 4.6 and 4.7 to make clear which orientations can and cannot exist. We will mark any illegal suborientations with their corresponding forbidden suborientation (a,b,c, or d). We will let a "–" represent a flat edge.

Firstly, Table 4.6 in which we suppose that e_{n-1} is a left arrow.

e_{n-2} \ e_{n-3}	$-$	$\rightarrow$	$\leftarrow$
$-$	a		c
$\rightarrow$			
$\leftarrow$			d

Table 4.6: Illegal suborientations given that e_{n-1} is directed $\leftarrow$

Secondly, Table 4.7 in which we suppose that e_{n-1} is a right arrow.

		e_{n-3}		
		$-$	$\rightarrow$	$\leftarrow$
	$-$	a	c	
	$\rightarrow$		d	
e_{n-2}	$\leftarrow$			

Table 4.7: Illegal suborientations given that e_{n-1} is directed $\rightarrow$

There are 12 empty cells between Tables 4.6 and 4.7. These 12 empty cells represent 12 combinations of orientations that e_{n-1}, e_{n-2}, and e_{n-3} can have such that they do not contain any forbidden subgraphs. This implies 12 cases that need to be checked. However, we can group some of these cases together. The arrow combinations in these 12 cases can be expressed as the following four mutually exclusive and exhaustive cases:

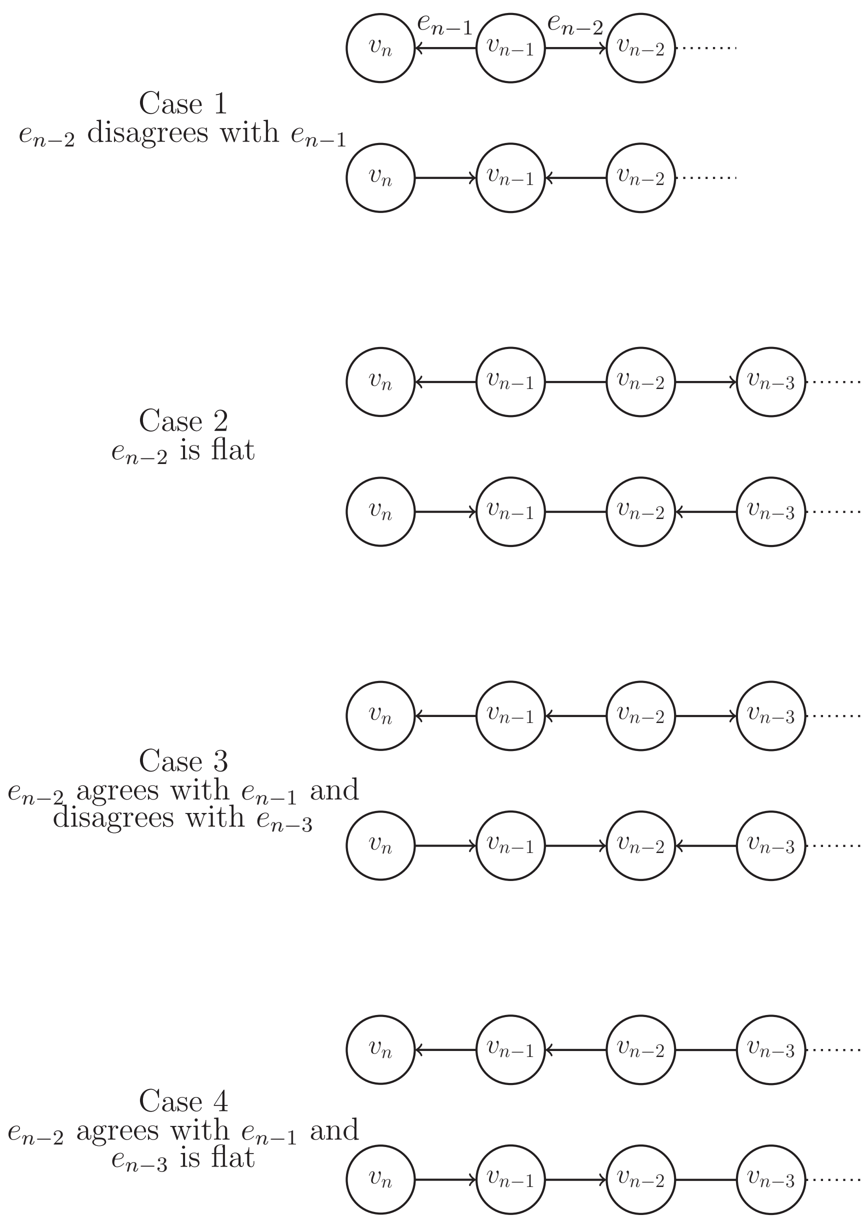


Figure 4.13: All four possible cases represented with forbidden suborientations excluded

e_{n-2} \ e_{n-3}	$-$	$\rightarrow$	$\leftarrow$
$-$	a	2	c
$\rightarrow$	1	1	1
$\leftarrow$	4	3	d

Table 4.8: Given that e_{n-1} is directed $\leftarrow$, illegal orientations are labelled with their corresponding illegal suborientation, and legal orientations are labelled with their corresponding case number

e_{n-2} \ e_{n-3}	$-$	$\rightarrow$	$\leftarrow$
$-$	a	c	2
$\rightarrow$	4	d	3
$\leftarrow$	1	1	1

Table 4.9: Given that e_{n-1} is directed $\rightarrow$, illegal orientations are labelled with their corresponding illegal suborientation, and legal orientations are labelled with their corresponding case number

We now have four cases that account for every orientation which is not forbidden. We will proceed by induction, using the cases of P_1, P_2, and P_3 as base cases. For our induction hypothesis, we will suppose that for all paths P_k, $k \leq n-1$, $n \in \mathbb{N}$, that every orientation of P_k which does not contain a forbidden suborientation is a 0-preorientation. This will conclude the converse portion of our proof and we will be able to conclude that an orientation is a 0-preorientation if and only if it does not contain any forbidden suborientation. We will also show that, Z_n, the number of 0-preorientations on P_n, is equal to $Z_{n-1} + 2Z_{n-2} + Z_{n-3}$. From Lemma 4.0.3, we will also be able to conclude that the number of 0-pre-positions on a path is equal to $Z_{n-1} + 2Z_{n-2} + Z_{n-3}$.

We will run through these four cases, proving not only that each one is a 0-preorientation, but also determining how many 0-pre-positions exist with that particular induced orientation. Remember, Lemma 4.0.3 states that for every 0-preorientation, R, there exists a unique 0-pre-position that induces R. However, our 4 cases do not involve the entire orientation, but rather the orientation of a subgraph. This is why each of our four cases can be induced, not by a single unique 0-pre-position, but by multiple.

Let C be a 0-pre-position on P_n with induced orientation R having the form in Case 1. That is, we know that e_{n-1} and e_{n-2} disagree in R. Note that, from forbidden suborientations a and b, we know that every 0-preorientation on P_{n-1} must have e_{n-2} directed right or left. So by removing v_n and its incident edge e_{n-1} from C, what remains is a configuration C' on P_{n-1} which induces an orientation R' that does not contain any forbidden suborientations. By our induction hypothesis, there exist Z_{n-1} such configurations. Given a 0-preorientation on P_{n-1}, there exists a unique 0-preorientation on P_n that can be created by adding a disagreeing edge. Thus, the number of 0-pre-positions on P_n with an induced orientation of the form in Case 1 is Z_{n-1}. These orientations with associated stack sizes are shown in Figure 4.14.

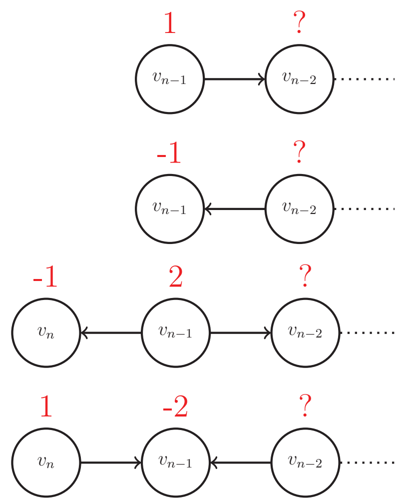


Figure 4.14: The orientations of Case 1 on P_n shown as extensions of non-forbidden orientations on P_{n-1}.

Let C be a 0-pre-position on P_n with induced orientation R having the form in Case 2. That is, we know that e_{n-1} is flat. Note that, from forbidden suborientations a and b, we know that every 0-pre-position on P_{n-2} must have e_{n-3} directed right or left. So by removing v_n and v_{n-1} along with their incident edges e_{n-1} and e_{n-2} from C, what remains is a configuration C' on P_{n-2} which induces an orientation R' that does not contain any forbidden suborientations. By our induction hypothesis, there exist Z_{n-2} such configurations. Given a 0-preorientation on P_{n-2}, there exists a unique 0-preorientation on P_n having the form in Case 2 (created by adding two vertices v_n and v_{n-1} along with a flat edge e_{n-2} with endpoints v_{n-1} and v_{n-2}, and a directed edge e_{n-1} with endpoints v_n and v_{n-1} which disagrees with e_{n-3}). Thus, the number of 0-pre-positions on P_n with an induced orientation of the form in Case 2 is Z_{n-2}. These orientations with associated stack sizes are shown in Figure 4.15.

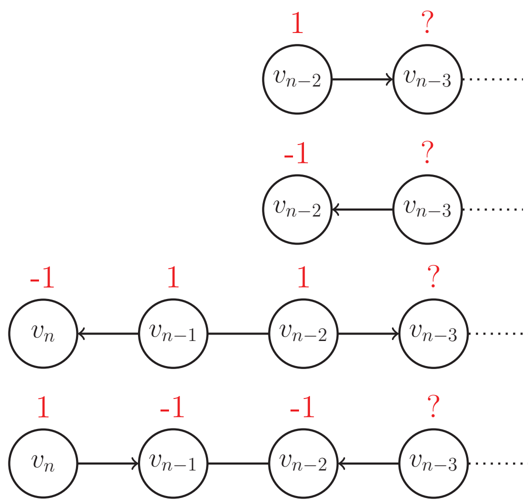


Figure 4.15: The orientations of Case 2 on P_n shown as extensions of non-forbidden orientations on P_{n-2}.

Let C be a 0-pre-position on P_n with induced orientation R having the form in Case 3. That is, we know that e_{n-2} agrees with e_{n-1} and disagrees with e_{n-3}. Note that, from forbidden suborientations a and b, we know that every 0-pre-position on P_{n-2} must have e_{n-3} directed right or left. So by removing v_n and v_{n-1} along with their incident edges e_{n-1} and e_{n-2} from C, what remains is a configuration C' on P_{n-2} which induces an orientation R' that does not contain any forbidden suborientations. By our induction hypothesis, there exist Z_{n-2} such configurations. Given a 0-preorientation on P_{n-2}, there exists a unique 0-preorientation on P_n having the form in Case 3 (created by adding two vertices v_n and v_{n-1} along with a directed edge e_{n-2} with endpoints v_{n-1} and v_{n-2} which disagrees with e_{n-3}, and a directed edge e_{n-1} with endpoints v_n and v_{n-1} which also disagrees with e_{n-3}). Thus, the number of 0-pre-positions on P_n with an induced orientation of the form in Case 3 is Z_{n-2}. These orientations with associated stack sizes are shown in Figure 4.16.

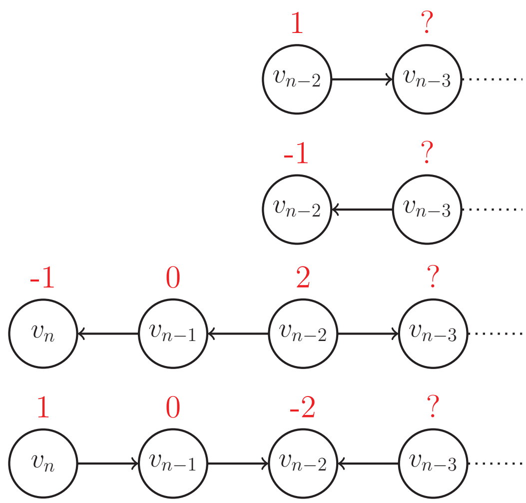


Figure 4.16: The orientations of Case 3 on P_n shown as extensions of non-forbidden orientations on P_{n-2}.

Let C be a 0-pre-position on P_n with induced orientation R having the form in Case 4. That is, we know that e_{n-2} agrees with e_{n-1} and e_{n-3} is flat. Note that, from forbidden suborientations a and b, we know that every 0-pre-position on P_{n-3} must have e_{n-4} directed right or left. Also note that since R does not contain any forbidden subgraphs, we know that e_{n-4} disagrees with e_{n-2}. So by removing v_n, v_{n-1}, and v_{n-2} along with their incident edges e_{n-1}, e_{n-2} and e_{n-3} from C, what remains is a configuration C' on P_{n-3} which induces an orientation R' that does not contain any forbidden suborientations. By our induction hypothesis, there exist Z_{n-3} such configurations. Given a 0-preorientation on P_{n-3}, there exists a unique 0-preorientation on P_n having the form in Case 4 (created by adding three vertices v_n, v_{n-1}, and v_{n-2} along with three edges: a flat edge e_{n-3} with endpoints v_{n-3} and v_{n-2}, a directed edge e_{n-2} with endpoints v_{n-2} and v_{n-1} which disagrees with e_{n-4}, and a directed edge e_{n-1} with endpoints v_n and v_{n-1} which also disagrees with e_{n-4}). Thus, the number of 0-pre-positions on P_n with an induced orientation of the form in Case 4 is Z_{n-3}. These orientations with associated stacks sizes are shown in Figure 4.17.

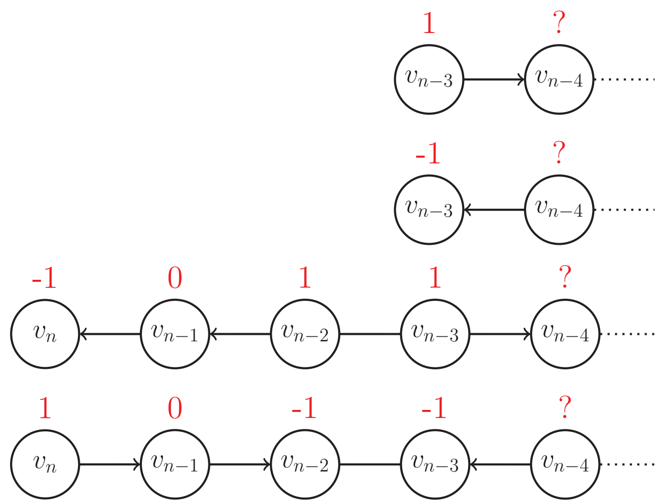


Figure 4.17: The orientations of Case 4 on P_n shown as extensions of non-forbidden orientations on P_{n-3}.

Thus, we conclude that every orientation that does not contain any of the four forbidden suborientations is a 0-preorientation. In addition, we conclude that Z_n is equal to the sum of our four cases. So, $Z_n = Z_{n-1} + 2Z_{n-2} + Z_{n-3}$.

□

This sequence is A218078 in OEIS [16] and its generating function is $\frac{2x(1+x)}{1-x-2x^2-x^3}$ [16]. In order to find the explicit formula, we begin by finding the characteristic equation for this linear recurrence:

$$\begin{aligned} Z_n &= Z_{n-1} + 2Z_{n-2} + Z_{n-3} \\ Z_n - Z_{n-1} - 2Z_{n-2} - Z_{n-3} &= 0 \qquad \text{Let } Z_n = x^n \\ x^n - x^{n-1} - 2x^{n-2} - x^{n-3} &= 0 \\ x^{n-3}(x^3 - x^2 - 2x - 1) &= 0 \\ x^3 - x^2 - 2x - 1 &= 0 \end{aligned}$$

The roots of this equation are $\alpha_1 \approx 2.1479$, $\alpha_2 \approx -0.5740 - 0.3690i$, and $\alpha_3 \approx -0.5740 + 0.3690i$. Since these are all of the nonzero roots of the function, our general solution is a linear combination of all three values:

$$Z_k = c_1(\alpha_1)^k + c_2(\alpha_2)^k + c_3(\alpha_3)^k.$$

In order to solve for c_1, c_2, and c_3, we must solve a system of three linear equations using three initial values of our recurrence: $Z_1 = 0$, $Z_2 = 2$, and $Z_3 = 4$. The approximate solutions are

$$\begin{aligned} c_1 &\approx 0.3885 \\ c_2 &\approx 0.8057 + 0.1226i \\ c_3 &\approx 0.8057 - 0.1226i \end{aligned}$$

So,

$$\begin{aligned} Z_k \approx\ & 0.3885(2.1479)^{\,k} \\ & + (0.8057 + 0.1226i)(-0.5740 - 0.3690i)^{\,k} \\ & + (0.8057 - 0.1226i)(-0.5740 + 0.3690i)^{\,k}. \end{aligned}$$

The dominating term, out of the three roots, is the one which has the greatest modulus. These moduli are roughly 2.1479, 0.6823, and 0.6823. Thus, the dominant term in the equation

$$\begin{aligned} Z_k \approx\ & 0.3885(2.1479)^{\,k} \\ & + (0.8057 + 0.1226i)(-0.5740 - 0.3690i)^{\,k} \\ & + (0.8057 - 0.1226i)(-0.5740 + 0.3690i)^{\,k} \end{aligned}$$

is $c_1(\alpha_1)^{\,k} \approx (0.3885)(2.1479)^{\,k}$.

Corollary 4.2.2. *Z_k has an asymptotic value of approximately* 0.3885×2.1479^k.

Chapter 5

Variants

In Parallel Diffusion, firing results in richer vertices giving to poorer vertices. However, this idea of firing can be abstracted to obtain variants of Parallel Diffusion. In this chapter we will define some variants of Parallel Diffusion in part by changing the way in which different stack sizes interact with each other. By changing the *firing rules*, we change the process by having, for example, stack sizes of 7 only send chips to their neighbours who have less than 2 chips, or by having stack sizes of 10 only send chips to neighbours that have exactly 12 chips, etc. These variants show us how periodicity changes when the process changes. In this chapter, we will see an example of a variant, Two-One Diffusion, with very large periods (Section 5.1) and an example of a variant, Pay it Backward, which does not generally exhibit periodic behaviour (Section 5.2). In Subsection 5.2.1, we show our results for a specific Pay it Backward example on P_3, concluding that it is not generally periodic (Lemmas 5.2.2 and 5.2.3), its pre-positions are unique (Lemma 5.2.5), and that its configuration sequence, when plotted into 3-space with its stack sizes as coordinates, generates a spiral outward from its initial configuration, revealing regularities about the underlying orientations (Figure 5.16). In Subsection 5.2.2, we characterize all configurations with period 1 in Pay it Backward (Theorem 5.2.11).

Definition 5.0.1. *A **firing rule** is an ordered pair (a, b) written as $a \to b$, where $a, b \in \mathbb{Z}$, $a \neq b$. Let $S = \{a \to b : a, b \in \mathbb{Z}\}$ be the set of all possible firing rules. In any particular diffusion process, the flow of chips is determined by a subset of S, call it S'. Let v be a vertex in G and let C be a configuration on G. Let $Z(S', C, v) = \{u \in N(v) : |v|^C \to |u|^C \in S'\}$ (Note how this set $Z(S', C, v)$ corresponds to the set $Z_-^C(v)$ in Definition 1.1.16). To **fire** v is to add one to the stack size of every vertex in $Z(S', C, v)$ and reduce the stack size of v by $|Z(S', C, v)|$. If $u \in Z(S', C, v)$, we say that u **receives** a chip from v and that v **sends** a chip to u.*

Note that if $(a, b) \in S'$ and $(b, a) \in S'$, then if a vertex with stack size a is adjacent to a vertex with stack size b, neither stack size will change as a result of chips being sent across their shared edge. An example of a process with only two firing rules is given in Figure 5.1.

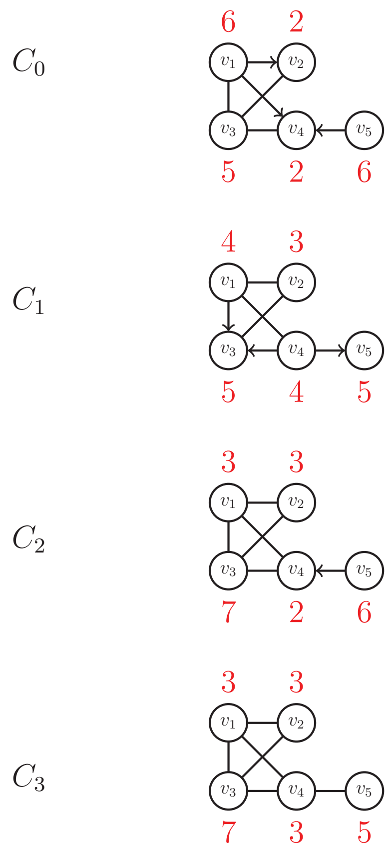


Figure 5.1: Three steps in a process with firing rules $4 \to 5$ and $6 \to 2$. Directed edges depict the flow of chips.

Note how this definition allows for the firing of only a single vertex and also broadens the definition of how chips can move in a variant of Parallel Diffusion, allowing for the possibility of only certain vertices sending chips to poorer vertices, or even having poorer vertices send chips to richer vertices. In Section 5.1, we discuss a diffusion variant with different firing rules.

Note also that Parallel Diffusion has infinitely many firing rules with

$$S' = \{a \to b : a, b \in \mathbb{Z}, a > b\}.$$

No matter what stack size a vertex u has, there exist infinitely many lesser possible integral stack sizes which could be adjacent to u.

In the most general terms, diffusion is a process in which we compare stack sizes of neighbouring vertices and fire according to our firing rules (Definition 5.0.1). If we change the way we compare the stack sizes of neighbouring vertices or change the firing rules, we obtain a new variant of diffusion. In this chapter, we will analyze three diffusion variants: Two-One Diffusion, Pay it Backward, and Sequential Diffusion.

In *Two-One Diffusion*, the variation lies in a change to the firing rules. Instead of chips being sent along every edge from richer to poorer, we restrict the rules so that chips only move from stack sizes of two to stack sizes of one, and from stack sizes of three to stack sizes of zero. So, rather than the infinitely many firing rules in Parallel Diffusion, we have only two in Two-One Diffusion: $2 \to 1$ and $3 \to 0$. We discuss Two-One Diffusion in Section 5.1 and with Conjecture 5.1.1 we propose that the period lengths in Two-One Diffusion may be arbitrarily large.

Instead of changing the firing rules, another way that a diffusion variant can be defined is by changing the way in which adjacent stack sizes are compared. That is, rather than comparing the stack sizes of two adjacent vertices, $|a|^C$ and $|b|^C$, and having the greater fire to the lesser, we instead compare $f(|a|^C)$ with $g(|b|^C)$ where f and g are functions, and have the greater of these two values fire to the lesser. In such cases, a labelling of the edges is required so we know which endpoint will be assigned function f and which endpoint will be assigned function g. In *Pay it Backward*, we use this sort of variation with $f(x) = x$ and $g(x) = -x$. In Pay it Backward, each edge v_1v_2 is assigned a label $v_1 \sim -v_2$ or $-v_1 \sim v_2$ so it is clear, for each endpoint, which function f or g is applied. Such an edge labelling could rightly be viewed as a graph orientation, but we will avoid this terminology so as to prevent any confusion with the graph orientation induced by the configuration (Lemma 1.1.23). In Section 5.2, we show that Pay it Backward exhibits very regular behaviour but is not, in general, periodic.

In Section 5.3, we look at a variant, Sequential Diffusion, in which the vertices do not all fire at the same time. In *Sequential Diffusion*, the firing rules are equivalent to those of Parallel Diffusion, but only a single vertex fires at each step. We examine this process with the Millpond configuration and in Theorem 5.3.8, we show an instance in which it displays periodic behaviour.

5.1 Two-One Diffusion

In this section, we analyze the diffusion variant, Two-One Diffusion. With Long and Narayanan's result that Parallel Diffusion always has period 1 or 2, we sought to determine if every diffusion process with some subset of the infinite firing rules in Parallel Diffusion also necessarily exhibits period 1 or 2. We show multiple examples in which Two-One Diffusion has a period larger than 2 and then conjecture that for any $n \in \mathbb{N}$, there exists a configuration on the path of length 2^n which exists inside a period of length 2^n.

Two-One Diffusion varies from Parallel Diffusion in its firing rules. Instead of the infinitely many firing rules in Parallel Diffusion, in Two-One Diffusion, there are only two: $2 \to 1$ and $3 \to 0$. As in Parallel Diffusion, every vertex will fire at each step. Note that this implies that any vertex with a stack size greater than three or less than zero will never increase or decrease in any future step. The only situations in which chips are sent are when a vertex with two chips sends a chip to a vertex with one chip, and when a vertex with three chips sends a chip to a vertex with zero chips. The reason for the name Two-One Diffusion comes just as much from the useful initial configuration on which we have decided to focus as it does from the firing rule.

We will work exclusively with paths and we will draw them on a horizontal axis so that terms like 'rightmost' and 'leftmost' have a clear meaning. We will be particularly interested in the configuration on a path, P_{2^n}, in which the stack size of the leftmost vertex is 2 and the stack sizes alternate between 1 and 2 as shown in Figure 5.2. We will refer to this configuration on a path as the 2-1 configuration, and a path on n vertices with the 2-1 configuration will be referred to as P_n^{2-1}.

Figure 5.2: P_{2^n} with the 2-1 configuration

We will show the first few configurations from each of P_2^{2-1}, P_4^{2-1}, and P_8^{2-1} in Figures 5.3, 5.4, and 5.5, respectively.

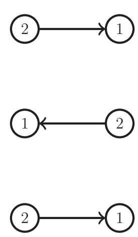


Figure 5.3: Period of P_2^{2-1}

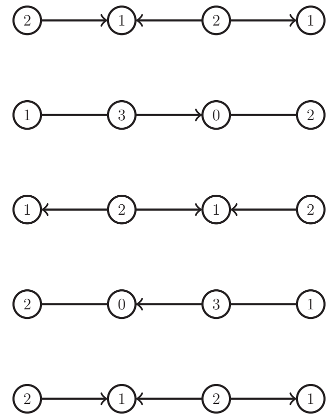


Figure 5.4: Period of P_4^{2-1}

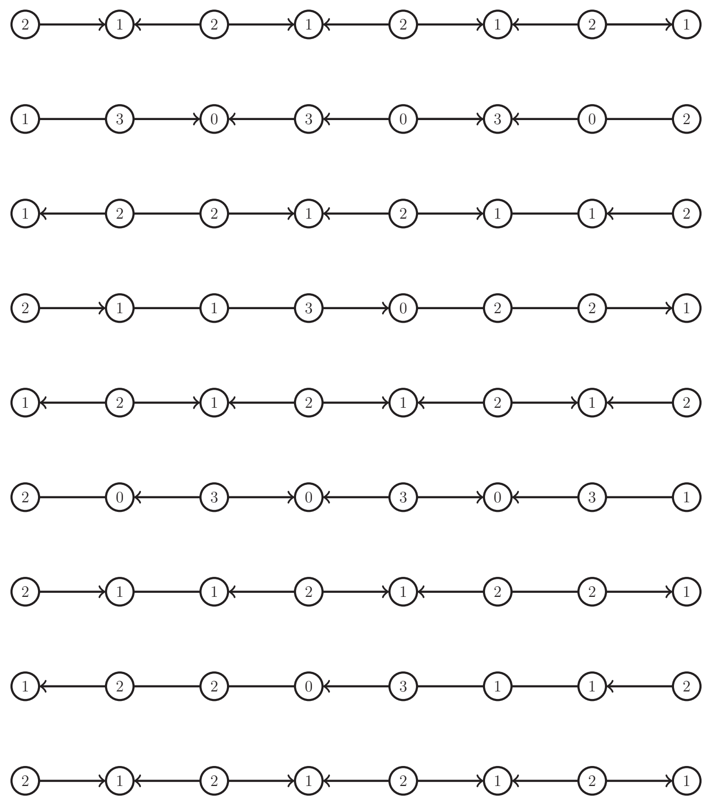


Figure 5.5: Period of P_8^{2-1}

Each of these configuration sequences can be seen to be equal to the number of vertices in the path. This leads us to our conjecture.

Conjecture 5.1.1. *Let G be a path with the 2-1 configuration. In Two-One Diffusion, if G has 2^n vertices, then the configuration sequence will have period 2^n.*

We will make use of the path on n vertices in which stack sizes of 1 and 2 alternate, and the leftmost vertex has stack size 1. We will refer to this path as having the 1-2 configuration.

Finally, we conclude with three more examples which provide evidence for our conjecture. Since these examples have many vertices, we will adopt a shorthand notation. A path P_n with the 2-1 configuration will be signified by n^+ and a path P_n with the 1-2 configuration will be signified by n^-. For instance, the initial configuration in Figure **??** could be written as 8^+.

What follows are the *even numbered steps* from the configuration sequences initiated by 16^+, 32^+, and 64^+ (the longest period which we have calculated). Also, for the purposes of these examples, we will adopt the notation of concatenating configurations. So we could write for instance, that $P_n^{2-1} = P_2^{2-1} P_{n-4}^{2-1} P_2^{2-1}$. We develop these notations and exhibit the configuration sequences in this way to make it more clear to the reader the similarities shared between the configuration sequences of different paths with the 2-1 configuration. We show for each of these examples that it takes 2^{n-1} steps to turn n^+ into n^-. It will clearly then take 2^{n-1} steps to turn n^- into n^+. So we can see that the period of each example is equal to the number of vertices in the path.

$$16^+$$
$$2^- 12^+ 2^-$$
$$4^- 8^+ 4^-$$
$$4^+ 2^- 4^+ 2^- 4^+$$
$$16^-$$

Figure 5.6: First 8 steps on 16^+

$$32^+$$
$$2^-28^+2^-$$
$$4^-24^+4^-$$
$$4^+2^-20^+2^-4^+$$
$$8^-16^+8^-$$
$$2^+4^-2^+2^-12^+2^-2^+4^-2^+$$
$$8^+4^-8^+4^-8^+$$
$$2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-$$
$$32^-$$

Figure 5.7: First 16 steps on 32^+

$$64^+$$
$$2^-60^+2^-$$
$$4^-56^+4^-$$
$$4^+2^-52^+2^-4^+$$
$$8^-48^+8^-$$
$$2^+4^-2^+2^-44^+2^-2^+4^-2^+$$
$$8^+4^-40^+4^-8^+$$
$$2^-4^+2^-4^+2^-36^+2^-4^+2^-4^+2^-$$
$$16^-32^+16^-$$
$$2^+12^-2^+2^-28^+2^-2^+12^-2^+$$
$$4^+8^-4^+4^-24^+4^-4^+8^-4^+$$
$$4^-2^+4^-2^+4^-4^+2^-20^+2^-4^+4^-2^+4^-2^+4^-$$
$$16^+8^-16^+8^-16^+$$
$$2^-12^+2^-2^+4^-2^+2^-12^+2^-2^+4^-2^+2^-12^+2^-$$
$$4^-8^+4^-8^+4^-8^+4^-8^+4^-8^+4^-$$
$$4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+2^-4^+$$
$$64^-$$

Figure 5.8: First 32 steps on 64^+

5.2 Pay It Backward

In this section, we define and study Pay it Backward. We begin with a thorough analysis of Pay it Backward on P_3 with a specific labelling of the edges (Figure 5.9) in Subsection 5.2.1, before moving on to more general results in Subsection 5.2.2.

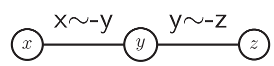


Figure 5.9: P_3 with $x \sim -y$ and $y \sim -z$

Our analysis of Pay it Backward will focus on periodicity and regularity. We will analyze some instances in which Pay it Backward yields periodic configuration sequences, but also (in contrast to Parallel Diffusion) some instances in which Pay it Backward yields aperiodic configuration sequences. These aperiodic configuration sequences, however, exhibit noteworthy regularity in their induced orientations.

In Subsection 5.2.1, we characterize configuration sequences which, given a specific labelling of the edges, exhibit periodic behaviour and those which do not. Also, in Subsection 5.2.1, we show that the pre-position of a given configuration is well-defined (at least on some graphs). In Subsection 5.2.2, we define a useful type of auxiliary graph. In Theorem 5.2.11, we use this auxiliary graph to show the necessary conditions for a cycle to have a p_1-configuration in Pay it Backward.

Given a graph G, suppose that each edge ab has been assigned either the label $a \sim -b$ or $-a \sim b$. Given a configuration C, when the vertices fire at each step, rather than vertices sending chips to their poorer neighbours, every edge is evaluated separately according to its label. An edge with label $a \sim -b$ will compare $|a|^C$ to $-|b|^C$, and the vertex with the greater value will send a chip to the vertex with the lesser value. An edge with label $-a \sim b$ will compare $-|a|^C$ to $|b|^C$, and the vertex with the greater value will send a chip to the vertex with the lesser value. Notably, the firing rules in Pay it Backward are identical to those of Parallel Diffusion, with greater always firing to lesser. The variation lies in the edge labelling that may change which value is greater and which one is lesser. An example of a firing in Pay it Backward is given in Figure 5.10.

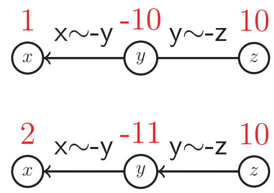


Figure 5.10: One firing in Pay it Backward on P_3.

5.2.1 P_3 analysis of Pay It Backward

We will now focus our attention on a particular edge labelling on P_3, shown in Figure 5.11.

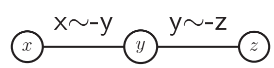


Figure 5.11: P_3 with $x \sim -y$ and $y \sim -z$

We will explore the configurations that can exist in Pay it Backward on P_3 under this labelling of the edges, and the orientations that those configurations induce. Through a number of lemmas, we will show that as time increases, a very regular behaviour is exhibited, but the configuration sequence is, however, not generally periodic.

The configuration sequences generated by configurations on P_3 under this labelling of the edges, are shown to be generally aperiodic in Lemma 5.2.3. In Lemma 5.2.5, it is shown that every position has a unique pre-position (if a pre-position exists at all). This is another way that Pay it Backward, under this labelling, differs from Parallel Diffusion (recall Chapter 4, where we determined the number of pre-positions of the fixed configuration in Quantum Parallel Diffusion and Parallel Diffusion).

We begin by determining, given a configuration C that induces an orientation R, which of the possible induced orientations could exist at the next step. This result will be referenced throughout this subsection. Recall that $|x|^C$ represents the stack size of vertex x in configuration C (which may be negative).

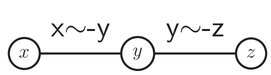
Lemma 5.2.1. *Given P_3 with $x \sim -y$, $y \sim -z$, and an initial configuration C_0, the orientation induced by C_1 is related to the orientation induced by C_0 as shown in Table 5.1.*

	Initial Orientation ($t = 0$)	Possible Orientations at $t = 1$	
R_1	$x - y - z$	$x - y - z$	R_1
R_2	$x \to y - z$	$x \to y \to z$	R_6
R_3	$x \leftarrow y - z$	$x \leftarrow y \leftarrow z$	R_7
R_4	$x - y \to z$	$x \leftarrow y \to z$	R_8
R_5	$x - y \leftarrow z$	$x \to y \leftarrow z$	R_9
R_6	$x \to y \to z$	$x - y \to z$	R_4
		$x \to y \to z$	R_6
R_7	$x \leftarrow y \leftarrow z$	$x - y \leftarrow z$	R_5
		$x \leftarrow y \leftarrow z$	R_7
R_8	$x \leftarrow y \to z$	$x \leftarrow y - z$	R_3
		$x \leftarrow y \to z$	R_8
R_9	$x \to y \leftarrow z$	$x \to y - z$	R_2
		$x \to y \leftarrow z$	R_9

Table 5.1: On the left, each possible initial orientation of P_3 is shown with a corresponding name to make future referencing simpler. Given a configuration that induces an orientation on the left, the next column represents the list of all possible orientations that may be induced by the resulting configuration at time $t = 1$, and the rightmost column contains their corresponding names.

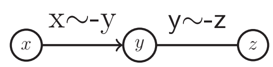
Proof. With two edges, and three possible orientations per edge, there are 9 possible initial orientations of this path. We now go through each of these 9 cases to show that the possible orientations that can arise following the initial firing, given that we know the orientation induced by the initial configuration, are those outlined in Table 5.1.

Case 1: The configuration at Step 0 is such that, in the initial firing, no chips will be sent (R_1).



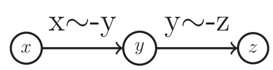
If both edges are flat, then the stack sizes will not change in the initial firing or any subsequent firing. So, this orientation will always yield itself. We know that this configuration sequence has a period of length 1 and pre-period of length 0. Any configuration C which induces this orientation must be of the form $|x|^C = -|y|^C = |z|^C$.

Case 2: The configuration at Step 0 is such that, in the initial firing, x will send a chip to y, and no chip will move across yz (R_2).

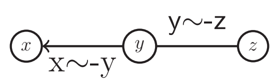


We know $|x|_0 > -|y|_0$ and $|y|_0 = -|z|_0$. Thus, $|x|_1 = |x|_0 - 1$, $|y|_1 = |y|_0 + 1$, and $|z|_1 = |z|_0$.
So $|y|_1 = |y|_0 + 1 = -|z|_0 + 1 = -|z|_1 + 1$, which implies $|y|_1 > -|z|_1$. Also, $|x|_1 + 1 = |x|_0 > -|y|_0 = -|y|_1 + 1$ which implies $|x|_1 > -|y|_1$.

Therefore, the only orientation that could arise as a result of a firing of Case 2 is $x \to y$, $y \to z$.

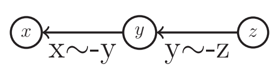


Case 3: The configuration at Step 0 is such that, in the initial firing, y will send a chip to x, and no chip will move across yz (R_3).

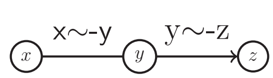


We know $|x|_0 < -|y|_0$ and $|y|_0 = -|z|_0$. Thus, $|x|_1 = |x|_0 + 1$, $|y|_1 = |y|_0 - 1$, and $|z|_1 = |z|_0$.
So $|y|_1 = |y|_0 - 1 = -|z|_0 - 1 = -|z|_1 - 1$ which implies $|y|_1 < -|z|_1$. Also, $|x|_1 - 1 = |x|_0 < -|y|_0 = -|y|_1 - 1$ which implies $|x|_1 < -|y|_1$.

Therefore, the only orientation that could arise as a result of a firing of Case 3 is $x \leftarrow y$, $y \leftarrow z$.

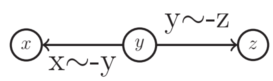


Case 4: The configuration at Step 0 is such that, in the initial firing, no chip will move across xy, and y will send a chip to z (R_4).

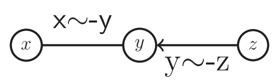


We know $|x|_0 = -|y|_0$ and $|y|_0 > -|z|_0$. Thus, $|x|_1 = |x|_0$, $|y|_1 = |y|_0 - 1$, and $|z|_1 = |z|_0 + 1$.
So $|y|_1 = |y|_0 - 1 = -|x|_0 - 1 = -|x|_1 - 1$ which implies $|x|_1 < -|y|_1$. Also, $|y|_1 + 1 = |y|_0 > -|z|_0 = -|z|_1 + 1$ which implies $|y|_1 > -|z|_1$.

Therefore, the only orientation that could arise as a result of a firing of Case 4 is $x \leftarrow y$, $y \rightarrow z$.

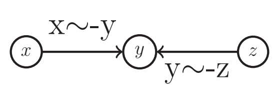


Case 5: The configuration at Step 0 is such that, in the initial firing, no chip will move across xy, and y will send a chip to z (R_5).

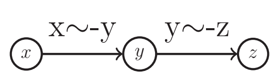


We know $|x|_0 = -|y|_0$ and $|y|_0 < -|z|_0$. Thus, $|x|_1 = |x|_0$, $|y|_1 = |y|_0 + 1$, and $|z|_1 = |z|_0 - 1$.
So $|y|_1 = |y|_0 + 1 = -|x|_0 + 1 = -|x|_1 + 1$ which implies $|x|_1 > -|y|_1$. Also, $|y|_1 - 1 = |y|_0 < -|z|_0 = -|z|_1 - 1$ which implies $|y|_1 < -|z|_1$.

Therefore, the only orientation that could arise as a result of a firing of Case 5 is $x \rightarrow y$, $y \leftarrow z$.



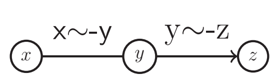
Case 6: The configuration at Step 0 is such that, in the initial firing, x will send a chip to y, and y will send a chip to z (R_6).



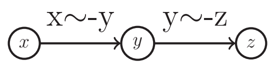
We know $|x|_0 > -|y|_0$ and $|y|_0 > -|z|_0$. Thus, $|x|_1 = |x|_0 - 1$, $|y|_1 = |y|_0$, and $|z|_1 = |z|_0 + 1$.
So $|x|_1 + 1 = |x|_0 > -|y|_0 = -|y|_1$ which implies $|x|_1 + 1 > -|y|_1$. Also, $|y|_1 = |y|_0 > -|z|_0 = -|z|_1 + 1$ which implies $|y|_1 > -|z|_1$.

Therefore, there exist two possible orientations that could arise as a result of a firing of Case 6.

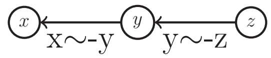
If $|x|_0 = -|y|_0 + 1$, then we get xy flat, $y \rightarrow z$.



Otherwise, we get $x \rightarrow y$, $y \rightarrow z$.



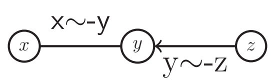
Case 7: The configuration at Step 0 is such that, in the initial firing, y will send a chip to x, and z will send a chip to y (R_7).



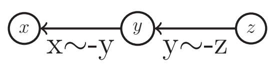
We know $|x|_0 < -|y|_0$ and $|y|_0 < -|z|_0$. Thus, $|x|_1 = |x|_0 + 1$, $|y|_1 = |y|_0$, and $|z|_1 = |z|_0 - 1$.
So $|x|_1 - 1 = |x|_0 < -|y|_0 = -|y|_1$, which implies $|x|_1 - 1 < -|y|_1$. Also, $|y|_1 = |y|_0 < -|z|_0 = -|z|_1 - 1$ which implies $|y|_1 < -|z|_1$.

Therefore, there exist two possible orientations that could arise as a result of a firing of Case 7.

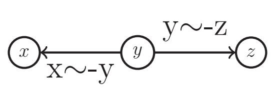
If $|x|_0 + 1 = -|y|_0$, then we get xy flat, $y \leftarrow z$.



Otherwise, we get $x \leftarrow y$, $y \leftarrow z$.



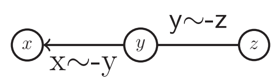
Case 8: The configuration at Step 0 is such that, in the initial firing, y will send a chip to x, and y will send a chip to z (R_8).



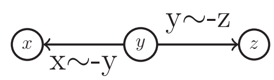
We know $|x|_0 < -|y|_0$ and $|y|_0 > -|z|_0$. Thus, $|x|_1 = |x|_0 + 1$, $|y|_1 = |y|_0 - 2$, and $|z|_1 = |z|_0 + 1$.
So, $|x|_1 - 1 = |x|_0 < -|y|_0 = -|y|_1 - 2$ which implies $|x|_1 < -|y|_1$. Also, $|y|_1 + 2 = |y|_0 > -|z|_0 = -|z|_1 + 1$ which implies $|y|_1 + 1 > -|z|_1$.

Therefore, there exist two possible orientations that could arise as a result of a firing of Case 8.

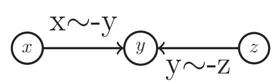
If $|y|_0 - 1 = -|z|_0$, then we get $x \leftarrow y$, yz flat.



Otherwise, we get $x \leftarrow y$, $y \rightarrow z$.



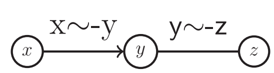
Case 9: The configuration at Step 0 is such that, in the initial firing, x will send a chip to y, and z sends a chip to y (R_9).



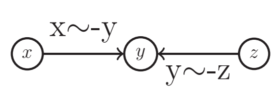
We know $|x|_0 > -|y|_0$ and $|y|_0 < -|z|_0$. Thus, $|x|_1 = |x|_0 - 1$, $|y|_1 = |y|_0 + 2$, and $|z|_1 = |z|_0 - 1$.
So $|x|_1 + 1 = |x|_0 > -|y|_0 = -|y|_1 + 2$ which implies $|x|_1 > -|y|_1$. Also, $|y|_1 - 2 = |y|_0 < -|z|_0 = -|z|_1 - 1$ which implies $|y|_1 - 1 < -|z|_1$.

Therefore, there exist two possible orientations that could arise as a result of a firing of Case 9.

If $|y|_0 + 1 = -|z|_0$, then we get $x \rightarrow y$, yz flat.



Otherwise, we get $x \rightarrow y$, $y \leftarrow z$.



Therefore, the data in Table 5.1 are accurate.

□

We will show that the only periodic configuration sequence under the labelling $x \sim -y$, $y \sim -z$ on P_3 is the one with period 1 and only contains configurations that induce R_1. This will happen whenever $|x|_0 = -|y|_0 = |z|_0$. Table 5.1 can be represented by a directed graph that provides insight into how these orientations are related. In Figure 5.12, we see such a graph. Every directed edge represents a possible pair of induced orientations at Steps 0 and 1, with the tail representing the induced orientation at Step 0 and the head representing the induced orientation at Step 1.

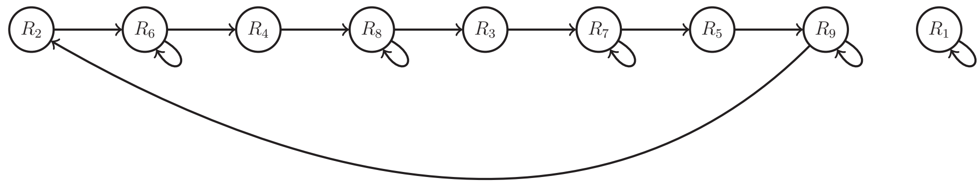


Figure 5.12: Directed graph showing the possible resulting orientations from a firing on a given orientation

In Figure 5.12, the edges are directed toward the possible orientations that can arise in the next step. So, if we suppose that a specific configuration exists inside the period and that its induced orientation is not R_1, then either that period contains only one orientation or it contains each of the other orientations, except for R_1.

Lemma 5.2.2. *Given P_3 with $x \sim -y$, $y \sim -z$, and an initial configuration C_0, suppose $Seq(C_0)$ exhibits periodic behaviour. That is, $C_t = C_{t+k}$ for some non-negative integers t and k. A configuration which induces R_1 is in $Seq(C_0)$ if and only if every configuration in the period of $Seq(C_0)$ induces the same orientation.*

Proof. Suppose every configuration in the period of $Seq(C_0)$ induces the same orientation and that orientation is not R_1. Then some vertex v will change its stack size at each step. Since the orientation never changes, this change will be the same increase or decrease at every step. Therefore, v will have a distinct stack size at every step (either increasing or decreasing by the same constant at every step). Therefore, there is no period.

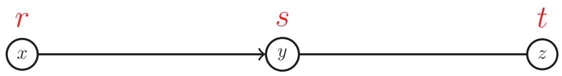
Conversely, if a configuration induces R_1 in $Seq(C)$, there will be no sending of chips during any subsequent firing. Therefore, every configuration in the period of $Seq(C)$ will be equal and thus, will induce R_1.

□

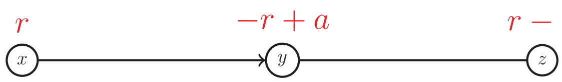
We will now show that every configuration sequence which contains each orientation R_i, for $2 \leq i \leq 9$, is aperiodic with Lemma 5.2.3. With Lemmas 5.2.2 and 5.2.3, we can conclude that the only periodic configuration sequences are those which only contain configurations which induce R_1.

Lemma 5.2.3. *Given P_3 with $x \sim -y$, $y \sim -z$, and an initial configuration C_0, if $Seq(C_0)$ is periodic, then $Seq(C_0)$ does not contain configurations which induce each orientation R_i, for $2 \leq i \leq 9$.*

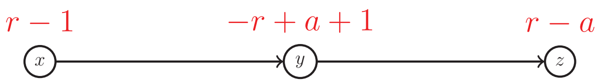
Proof. Suppose, by way of contradiction, that $Seq(C_0)$ is periodic and contains configurations which induce each orientation R_i, for $2 \leq i \leq 9$. Without loss of generality, we will choose initial configuration C_0 which induces R_2, assigning r, s, and t chips to the vertices x, y, and z, respectively. We will show that as time increases, every time we repeat orientation R_2 it corresponds to a distinct configuration.



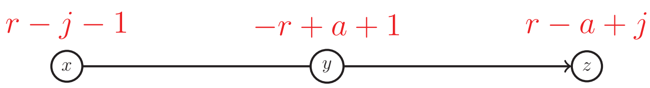
However, since we know how the edges are oriented, we can let $s = -r + a$ and $t = r - a$ for some positive integer a.



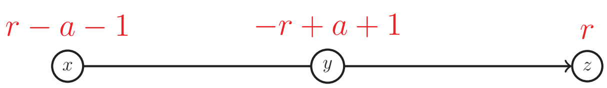
From Table 5.1, we know that at Step 1, the induced orientation must be R_6. So, at Step 1, the configuration is



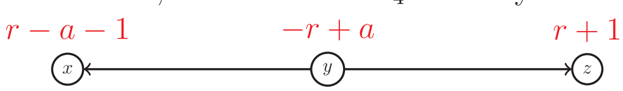
From Table 5.1, we know that R_6 either leads to itself or R_4. So, we will suppose that R_6 is maintained for j steps before it yields R_4, $j \geq 0$.



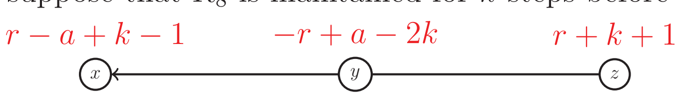
However, we know from this orientation that $r - j - 1 = -(-r + a + 1)$. So $j = a$.



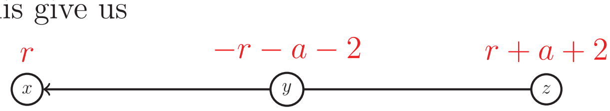
From Table 5.1, we know that R_4 can only lead to R_8. So, at the next step, we have



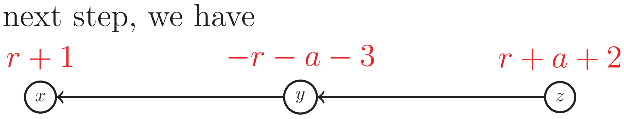
From Table 5.1, we know that orientation R_8 can either lead to itself or R_3. So, we will suppose that R_8 is maintained for k steps before it yields R_3, $k \geq 0$.



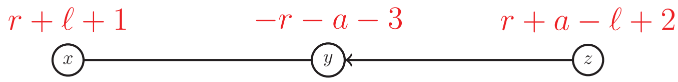
However, we know from this orientation that $-r+a-2k = -(r+1+k)$. So, $k = a+1$. This give us



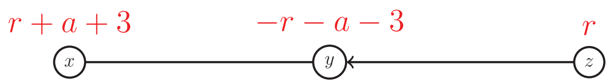
From Table 5.1, we know that orientation R_3 can only lead to orientation R_7. So, at the next step, we have



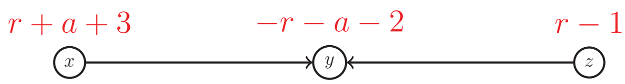
From Table 5.1, we know that R_7 can either lead to itself or R_5. So, we will suppose that R_7 is maintained for ℓ steps before it yields R_5, $\ell \geq 0$.



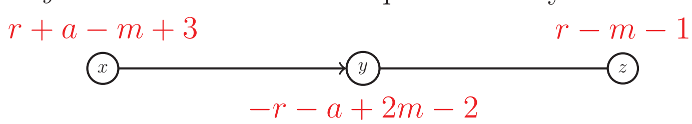
However, we know from this orientation that $r+\ell+1 = -(-r-a-3)$. So, $\ell = a+2$, yielding



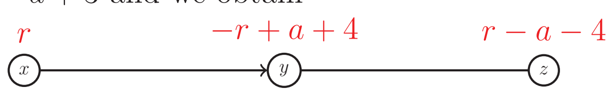
From Table 5.1, we know that R_5 can only lead to R_9. So, at the next step, we have



From Table 5.1, we know that R_9 can either lead to itself or R_2. So, we will suppose that R_9 is maintained for m steps before it yields orientation R_2, $m \geq 0$.



However, we know from this orientation that $-r - a + 2m - 2 = -(r - m - 1)$. So, $m = a + 3$ and we obtain



Thus, we have that every time the orientation R_2 has returned, a distinct configuration with induced orientation R_2 will be the result. Thus, no period can contain configurations which induce all 8 of these orientations. □

An example is given in Figure 5.13. Also, we have shown through the proof of this lemma that the number of repetitions of the orientations containing no flat edges increases by one consistently as time increases (from a to $a+1$ to $a+2$, etc.). So, if we were to run the process in reverse, and watch that number slowly decrease to 0, would we reach a terminal configuration? Is the process even well-defined in reverse? We ask this question because the existence of a very few terminal configurations may make a comprehensive analysis much more straight-forward.

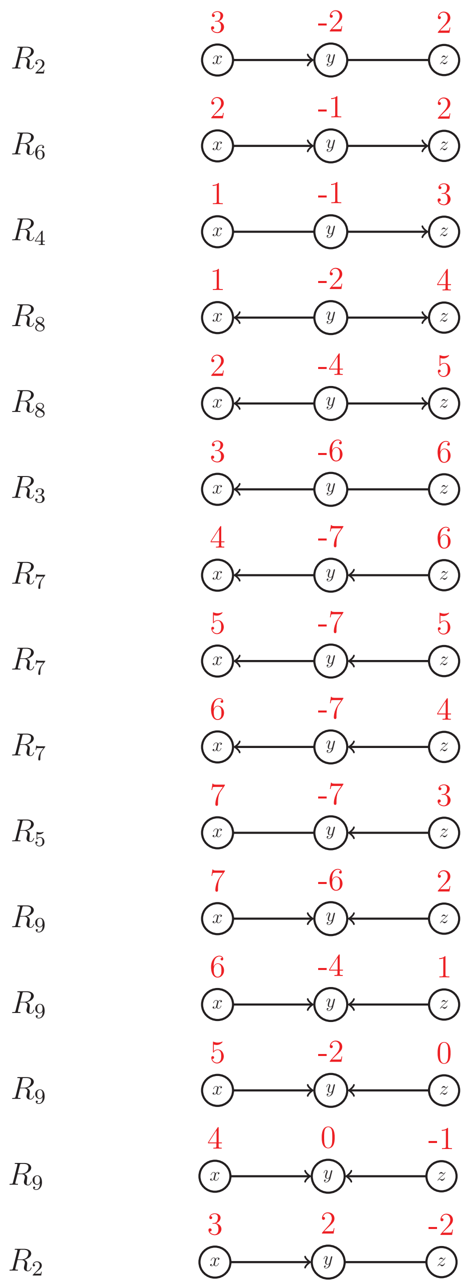


Figure 5.13: Sample firings in Pay it Backward with corresponding orientations labelled

In order to represent the sequence of orientations induced by the configurations within a configuration sequence succinctly, we will use superscripts to represent multiple concurrent instances of the same induced orientation. So, the induced orientations from Figure 5.13 can be written as $R_2R_6R_4R_8^2R_3R_7^3R_5R_9^4R_2$.

Corollary 5.2.4. *Given P_3 with $x \sim -y$, $y \sim -z$, and a configuration C which does* *not induce orientation R_1, the sequence of orientations induced by the configurations of Seq(C) follows one of the following forms depending on the orientation induced by C:*

- $R_2R_6^aR_4R_8^{a+1}R_3R_7^{a+2}R_5R_9^{a+3}R_2R_6^{a+4}\dots$
- $R_6^bR_4R_8^aR_3R_7^{a+1}R_5R_9^{a+2}R_2R_6^{a+3}R_4\dots$
- $R_4R_8^aR_3R_7^{a+1}R_5R_9^{a+2}R_2R_6^{a+3}R_4R_8^{a+4}\dots$
- $R_8^bR_3R_7^aR_5R_9^{a+1}R_2R_6^{a+2}R_4R_8^{a+3}R_3\dots$
- $R_3R_7^aR_5R_9^{a+1}R_2R_6^{a+2}R_4R_8^{a+3}R_3R_7^{a+4}\dots$
- $R_7^bR_5R_9^aR_2R_6^{a+1}R_4R_8^{a+2}R_3R_7^{a+3}R_5\dots$
- $R_5R_9^aR_2R_6^{a+1}R_4R_8^{a+2}R_3R_7^{a+3}R_5R_9^{a+4}\dots$
- $R_9^bR_2R_6^aR_4R_8^{a+1}R_3R_7^{a+2}R_5R_9^{a+3}R_2\dots$

where $0 \leq b < a$.

If we look at the three stack sizes in this example as coordinates, we get that as time increases, the resulting points will all exist on the same plane since the sum of the three stack sizes will never change (no chips are ever added or removed from the system). This allows us to view the changes from one orientation to the next in another manner by graphing these coordinates into 3-space. In Figure 5.15, the 15 configurations from Figure 5.13 are shown as points in 3-space beginning with the point (3,-2,2). From this first point, the points rotate outwards forming a spiral shape with every 90 degree turn representing a change from one orientation to another. In Figure 5.16, we see that the spiral effect becomes much more clear with 100 points shown. In Figure 5.14, we have added a third edge to form a triangle. This edge is labelled $-x \sim z$. Figure 5.17 contains the first 100 points from the initial point (3,-2,2) and shows the hexagonal shape that the spiral forms. This is a different graph from the one have been studying, but we include it to show how the spirals may be different for different graphs.

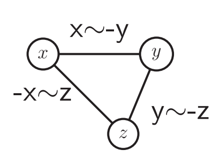


Figure 5.14: Triangle created by adding a third edge labelled $-x \sim z$

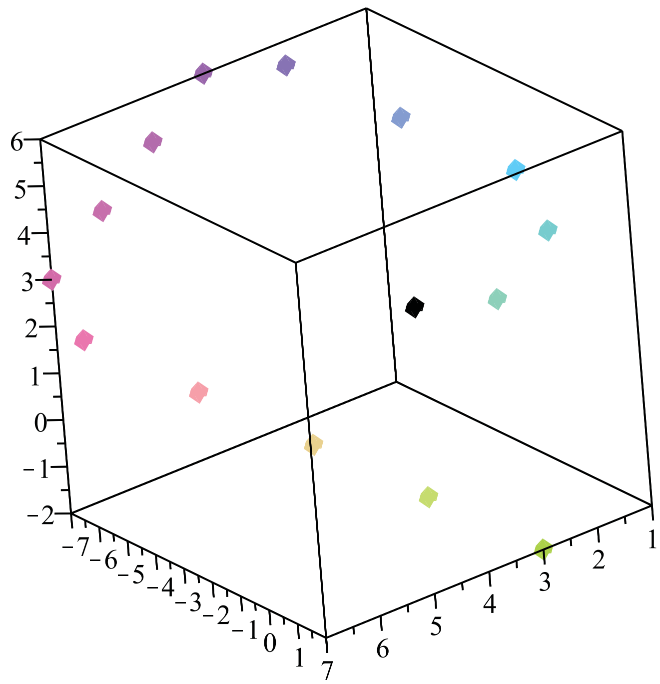


Figure 5.15: The configurations from Figure 5.13 shown in 3-space.

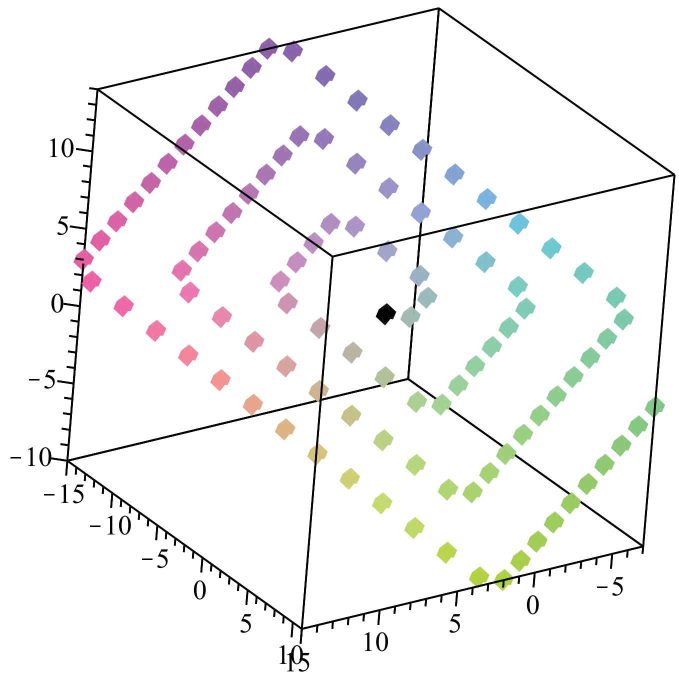


Figure 5.16: 100 configurations arising from the initial configuration $|x|^C = 3$, $|y|^C = -2$, $|z|^C = 2$.

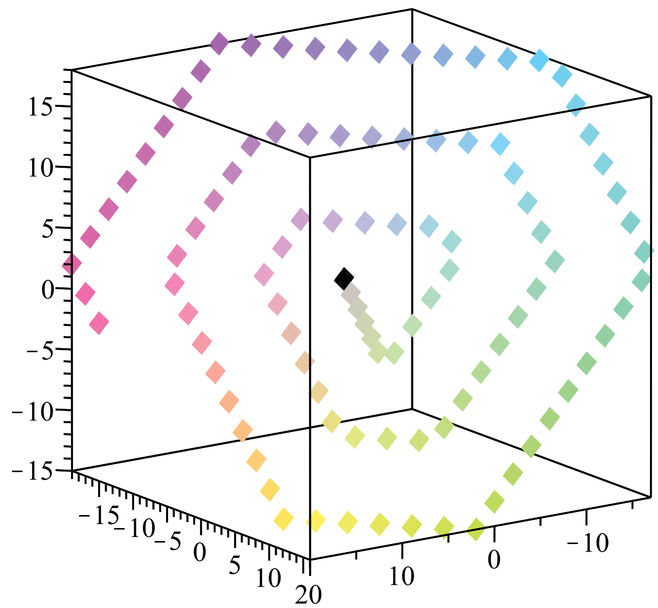


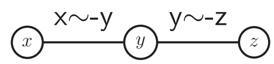
Figure 5.17: 100 configurations arising from the initial configuration $|x|^C = 3$, $|y|^C = -2$, $|z|^C = 2$ with an additional edge, labelled $-x \sim z$, added forming a triangle.

Returning to our example on P_3 with $x \sim -y$ and $y \sim -z$, we now show that no configuration has more than one possible pre-position. If the time in Pay it Backward moved in reverse, and we instead looked at all pre-positions of a given configuration, then Figure 5.12 (with all of the directed edges flipped) would still show the possible orientations that can arise from a given orientation. We must now prove that given a configuration, its pre-position is well-defined. This result is particularly noteworthy because it is not true of Parallel Diffusion. Recall for instance, Theorem 4.2.1, which states the number of pre-positions of the fixed configuration in Quantum Parallel Diffusion and Parallel Diffusion.

Lemma 5.2.5. *Given P_3 with $x \sim -y$, $y \sim -z$, and a configuration C, the pre-position of C is either unique or does not exist.*

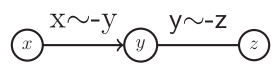
Proof. We will look at each case individually. Note that, if we are given a configuration C_t and the orientation induced by the configuration at the previous step, C_{t-1}, then we can solve for C_{t-1} since its stack sizes are uniquely determined by its induced orientation and C_t.

Case 1: Orientation R_1.



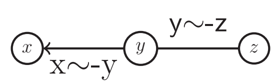
From Table 5.1, the pre-position must induce R_1. Therefore, the pre-position is unique with $|x|_t = |x|_{t-1}$, $|y|_t = |y|_{t-1}$, and $|z|_t = |z|_{t-1}$.

Case 2: Orientation R_2.



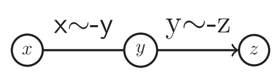
From Table 5.1, the pre-position must induce R_9. Therefore, the pre-position is unique with $|x|_t = |x|_{t-1} - 1$, $|y|_t = |y|_{t-1} + 2$, and $|z|_t = |z|_{t-1} - 1$.

Case 3: Orientation R_3.



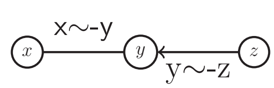
From Table 5.1, the pre-position must induced R_8. Therefore, the pre-position is unique with $|x|_t = |x|_{t-1} + 1$, $|y|_t = |y|_{t-1} - 2$, and $|z|_t = |z|_{t-1} + 1$.

Case 4: Orientation R_4.



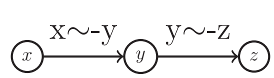
From Table 5.1, the pre-position must induce R_6. Therefore, the pre-position is unique with $|x|_t = |x|_{t-1} - 1$, $|y|_t = |y|_{t-1}$, and $|z|_t = |z|_{t-1} + 1$.

Case 5: Orientation R_5.



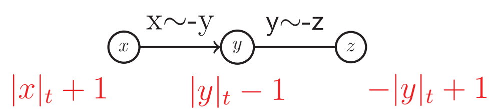
From Table 5.1, the pre-position must induce R_7. Therefore, the pre-position is unique with $|x|_t = |x|_{t-1} + 1$, $|y|_t = |y|_{t-1}$, and $|z|_t = |z|_{t-1} - 1$.

Case 6: Orientation R_6.



From Table 5.1, the pre-position has two possible orientations: R_2 and R_6. We know that $|y|_t > -|z|_t$.

(i) Suppose first that $|y|_t = -(|z|_t - 1)$. It is then possible for the pre-position to induce orientation R_2 with $|x|_t = |x|_{t-1} - 1$, $|y|_t = |y|_{t-1} + 1$, and $|z|_t = |z|_{t-1}$.

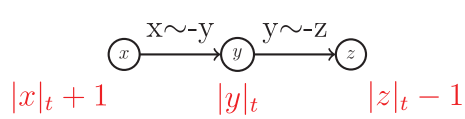


Would it be possible for the pre-position to have induced orientation R_6?

$\|y\|_{t-1} = \|y\|_t > -\|z\|_{t-1} = -(\|z\|_t - 1)$	if the pre-position induced orientation R_6
$\|y\|_t = -\|z\|_t + 1$	by supposition
$\|y\|_{t-1} = -(\|z\|_{t-1} + 1) + 1 = -\|z\|_{t-1}$	by substitution

However, if $|y|_{t-1} = -|z|_{t-1}$, then the pre-position does not have induced orientation R_6. Thus, if $|y|_t = -(|z|_t - 1)$, then the pre-position must have induced orientation R_2.

(ii) If we instead suppose that $|y|_t > -(|z|_t - 1)$, we get that R_6 is a possible induced orientation for the pre-position with $|x|_t = |x|_{t-1} - 1$, $|y|_t = |y|_{t-1}$, and $|z|_t = |z|_{t-1} + 1$.

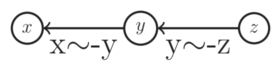


Would it be possible for the pre-position to have induced orientation R_2?

$\lvert y\rvert_{t-1} + 1 = \lvert y\rvert_t > -\lvert z\rvert_t = -\lvert z\rvert_{t-1}$	if the pre-position induced orientation R_2
$\lvert y\rvert_t > -\lvert z\rvert_t + 1$	by supposition
$\lvert y\rvert_{t-1} + 1 > -(\lvert z\rvert_{t-1}) + 1 = -\lvert z\rvert_{t-1} + 1$	by substitution

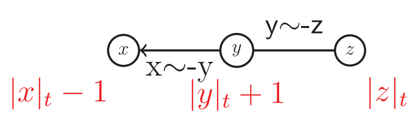
However, if $|y|_{t-1} = -|z|_{t-1}$, then the pre-position does not have induced orientation R_2. Thus, if $|y|_t > -(|z|_t - 1)$, then the pre-position must have induced orientation R_6.

Case 7: Orientation R_7.



From Table 5.1, the pre-position has two possible orientations: R_3 and R_7. We know that $|y|_t < -|z|_t$.

(i) Suppose first that $|y|_t = -(|z|_t + 1)$. It is then possible for the pre-position to induce orientation R_3 with $|x|_t = |x|_{t-1} + 1$, $|y|_t = |y|_{t-1} - 1$, and $|z|_t = |z|_{t-1}$.

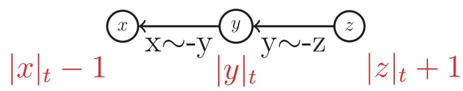


Would it be possible for the pre-position to induce orientation R_7?

$\lvert y\rvert_{t-1} = \lvert y\rvert_t < -\lvert z\rvert_{t-1} = -\lvert z\rvert_t - 1$	if the pre-position induced orientation R_7
$\lvert y\rvert_t = -\lvert z\rvert_t - 1$	by supposition
$\lvert y\rvert_{t-1} = (-\lvert z\rvert_{t-1} + 1) - 1 = -\lvert z\rvert_{t-1}$	by substitution

However, if $|y|_{t-1} = -|z|_{t-1}$, then the pre-position does not have induced orientation R_7. Thus, if $|y|_t = -|z|_t - 1$, then the pre-position must have induced orientation R_3.

(ii) If we instead suppose that $|y|_t > -(|z|_t + 1)$, we get that R_7 is a possible orientation for the pre-position with $|x|_t = |x|_{t-1} + 1$, $|y|_t = |y|_{t-1}$, and $|z|_t = |z|_{t-1} - 1$.

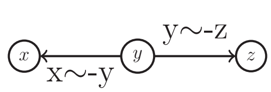


Would it be possible for the pre-position to induce orientation R_3?

$\lvert y\rvert_{t-1} = \lvert y\rvert_t + 1 < -\lvert z\rvert_t = -\lvert z\rvert_{t-1}$	if the pre-position induced orientation R_3
$\lvert y\rvert_t > -\lvert z\rvert_t - 1$	by supposition
$\lvert y\rvert_{t-1} - 1 > -\lvert z\rvert_{t-1} - 1$	by substitution

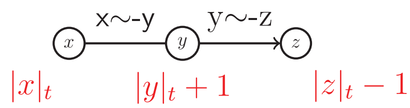
However, if $|y|_{t-1} > -|z|_{t-1}$, then the pre-position does not have induced orientation R_3. Thus, if $|y|_t > -|z|_t - 1$, then the pre-position must have induced orientation R_7.

Case 8: Orientation R_8.



From Table 5.1, the pre-position has two possible orientations: R_4 and R_8. We know that $x < -y$.

(i) Suppose first that $|x|_t = -(|y|_t + 1)$. It is then possible for the pre-position to induce R_4 with $|x|_t = |x|_{t-1}$, $|y|_t = |y|_{t-1} - 1$, and $|z|_t = |z|_{t-1} + 1$.



Would it be possible for the pre-position to induce R_8?

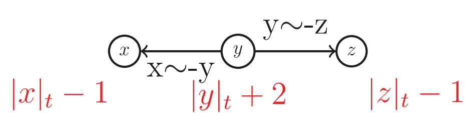
$|x|_{t-1} + 1 = |x|_t < -|y|_t = -|y|_{t-1} + 2$ if the pre-position induced orientation R_8

$|x|_t = -|y|_t - 1$ by supposition

$|x|_{t-1} + 1 = -|y|_{t-1} + 1$ by substitution

However, if $|x|_{t-1} = -|y|_{t-1}$, then the pre-position does not have induced orientation R_8. Thus, if $|x|_t = -|y|_t - 1$, then the pre-position must induce R_4.

(ii) If we instead suppose that $|x|_t < -(|y|_t + 1)$, we get that R_8 is a possible orientation for the pre-position with $|x|_t = |x|_{t-1} + 1$, $|y|_t = |y|_{t-1} - 2$, and $|z|_t = |z|_{t-1} + 1$.



Would it be possible for the pre-position to induce R_4?

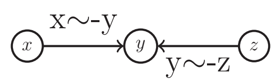
$|x|_{t-1} = |x|_t < -|y|_t = -|y|_{t-1} + 1$ if the pre-position induced orientation R_4

$|x|_t < -|y|_t - 1$ by supposition

$|x|_{t-1} < -|y|_{t-1} + 1 - 1 = -|y|_{t-1}$ by substitution

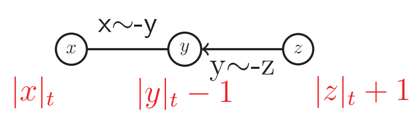
However, if $|x|_{t-1} < -|y|_{t-1}$, then the pre-position does not have induced orientation R_4. Thus, if $|x|_t < -|y|_t - 1$, then the pre-position must induce R_8

Case 9: Orientation R_9.



From Table 5.1, the pre-position has two possible orientations: R_5 and R_9. We know that $x > -y$.

(i) Suppose first that $|x|_t = -(|y|_t - 1)$. It is then possible for the pre-position to induce R_5 with $|x|_t = |x|_{t-1}$, $|y|_t = |y|_{t-1} + 1$, and $|z|_t = |z|_{t-1} - 1$.



Would it be possible for the pre-position to induce R_9?

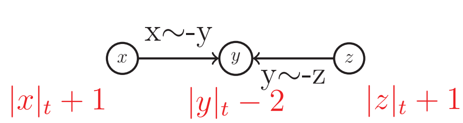
$|x|_{t-1} - 1 = |x|_t > -|y|_t = -|y|_{t-1} - 2$ if the pre-position induced orientation R_9

$|x|_t = -|y|_t + 1$ by supposition

$|x|_{t-1} - 1 = -|y|_{t-1} - 2 + 1 = -|y|_{t-1} - 1$ by substitution

However, if $|x|_{t-1} = -|y|_{t-1}$, then the pre-position does not have induced orientation R_9. Thus, if $|x|_t = -|y|_t + 1$, then the pre-position must induce R_5.

(ii) If we instead suppose that $|x|_t > -(|y|_t - 1)$, we get that I is a possible orientation for the pre-position with $|x|_t = |x|_{t-1} - 1$, $|y|_t = |y|_{t-1} + 2$, and $|z|_t = |z|_{t-1} - 1$.



Would it be possible for the pre-position to induce R_5?

$\|x\|_{t-1} = \|x\|_t > -\|y\|_t = -\|y\|_{t-1} - 1$	if the pre-position induced orientation R_5
$\|x\|_t > -\|y\|_t + 1$	by supposition
$\|x\|_{t-1} > -\|y\|_{t-1} - 1 + 1 = -\|y\|_{t-1}$	by substitution

However, if $|x|_{t-1} > -|y|_{t-1}$, then the pre-position does not have induced orientation R_5. Thus, if $|x|_{t-1} > -|y|_{t-1}$, then the pre-position must induce R_9.

So, we can conclude that every configuration has at most one possible pre-position.

□

We have shown that every configuration has at most one possible pre-position. With our next lemma, we will show that not every configuration has a pre-position and that with the exception of a trivial example, every configuration exists in a configuration sequence generated by a configuration which has no pre-position.

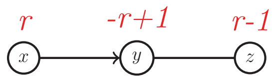
Lemma 5.2.6. *Given P_3 with $x \sim -y$, and $y \sim -z$, if C is a configuration on P_3 and does not induce R_1, then $C \in Seq(C_0)$ for some configuration C_0 which has no pre-position.*

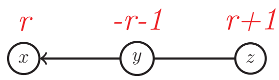
Proof. By Corollary 5.2.4, we see that the number of concurrent instances of orientations R_6, R_7, R_8, and R_9 increases by one every time. That is, if R_6 occurred exactly m times concurrently in a configuration sequence, it would be followed by R_4 and then $m+1$ concurrent instances of R_8. Therefore, in reverse it will decrease by one each time until we reach 0. This implies that there exists some set of configurations with no pre-position and that every configuration shares a configuration sequence with one such configuration (with the exception of sequences which induce R_1 which is the only induced orientation that can exist in a periodic configuration sequence by Lemmas 5.2.2 and 5.2.3).

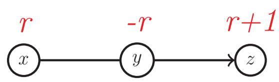
□

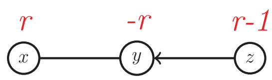
In general, what does a configuration with no pre-position look like?

Lemma 5.2.7. *Given P_3 and $x \sim -y$, and $y \sim -z$, the set of all configurations with no pre-positions consists of four infinite families:*

- 

- 

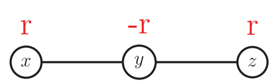
- 

- 

where r is an integer.

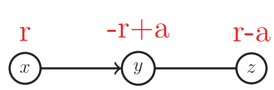
Proof. We will perform a case analysis based on the nine different possible induced orientations and conclude that only configurations of the four families shown have no pre-position.

Case 1:

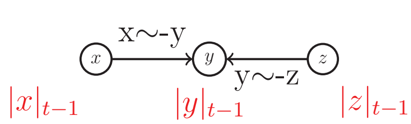


Any configuration that induces R_1 is maintained at every step because no chips will be sent. So, any configuration that induces Orientation R_1 is such that its pre-position exists and induces Orientation R_1 as well.

Case 2:



This configuration, call it C_t, with $a \geq 1$, induces orientation R_2. Suppose that some pre-position of C_t exists. By Lemma 5.2.1, the induced orientation of a pre-position of C_t must induce orientation R_9.



Thus,

$r = |x|_t = |x|_{t-1} - 1, \; -r + a = |y|_t = |y|_{t-1} + 2$, and $r - a = |z|_t = |z|_{t-1} - 1$.

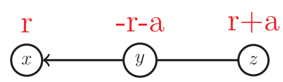
So, $|x|_{t-1} = r + 1$, $|y|_{t-1} = -r + a - 2$, and $|z|_{t-1} = r - a + 1$.

Since $|x|_{t-1} > -|y|_{t-1}$, we get that $r + 1 > r - a + 2$ which implies $r > r - a + 1$.

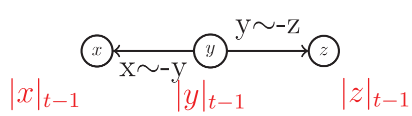
Also, since $|y|_{t-1} < -|z|_{t-1}$, we get that $-r + a - 2 < -r + a - 1$.

The only possible value of a for which this pair of inequalities does not hold is $a = 1$. Thus, the configuration $|x|_t = r$, $|y|_t = -r + 1$, $|z|_t = r - 1$ is the only configuration which induces R_2 and has no pre-position.

Case 3:



This configuration, call it C_t, with $a \geq 1$, induces orientation R_3. Suppose that some pre-position of C_t exists. By Lemma 5.2.1, the induced orientation of a pre-position of C_t must induce orientation R_8.



Thus,

$r = |x|_t = |x|_{t-1} + 1, \; -r - a = |y|_t = |y|_{t-1} - 2$, and $r + a = |z|_t = |z|_{t-1} + 1$

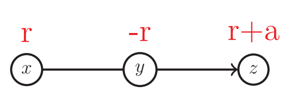
So, $|x|_{t-1} = r - 1$, $|y|_{t-1} = -r - a + 2$, and $|z|_{t-1} = r + a - 1$.

Since $|x|_{t-1} < -|y|_{t-1}$, we get that $r - 1 < r + a - 2$ which implies $r < r + a - 1$.

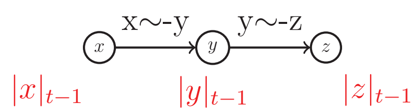
Also, since $|y|_{t-1} > |z|_{t-1}$, we get that $-r - a + 2 > -r - a + 1$.

The only possible value of a for which this pair of inequalities does not hold is $a = 1$. Thus, the configuration $|x|_t = r$, $|y|_t = -r - 1$, $|z|_t = r + 1$ is the only configuration which induces orientation R_3 and has no pre-position.

Case 4:



This configuration, call it C_t, with $a \geq 1$, induces orientation R_4. Suppose that some pre-position of C_t exists. By Lemma 5.2.1, the induced orientation of a pre-position of C_t must induce orientation R_6.



Thus,

$$r = |x|_t = |x|_{t-1} - 1, \ -r = |y|_t = |y|_{t-1}, \text{ and } r + a = |z|_t = |z|_{t-1} + 1.$$

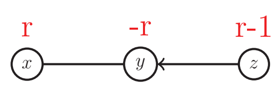
So, $|x|_{t-1} = r + 1$, $|y|_{t-1} = -r$, and $|z|_{t-1} = r + a - 1$.

Since $|x|_{t-1} > -|y|_{t-1}$, we get that $r + 1 > r$.

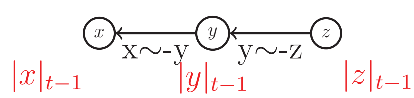
Also, since $|y|_{t-1} > |z|_{t-1}$, we get that $-r > -r - a + 1$.

The only possible value of a for which this pair of inequalities does not hold is $a = 1$. Thus, the configuration $|x|_t = r$, $|y|_t = -r$, $|z|_t = r+1$ is the only configuration which induces orientation R_4 and has no pre-position.

Case 5:



This configuration, call it C_t, with $a \geq 1$, induces orientation R_5. Suppose that some pre-position of C_t exists. By Lemma 5.2.1, the induced orientation of a pre-position of C_t must induce orientation R_7.



Thus,

$$r = |x|_t = |x|_{t-1} + 1, \ -r = |y|_t = |y|_{t-1}, \text{ and } r - a = |z|_t = |z|_{t-1} - 1.$$

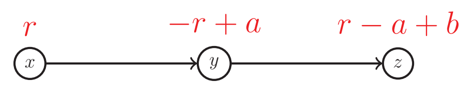
So, $|x|_{t-1} = r - 1$, $|y|_{t-1} = -r$, and $|z|_{t-1} = r - a + 1$.

Since $|x|_{t-1} < -|y|_{t-1}$, we get that $r - 1 < r$.

Also, since $|y|_{t-1} < -|z|_{t-1}$, we get that $-r < -r + a - 1$.

The only possible value of a for which this pair of inequalities does not hold is $a = 1$. Thus, the configuration $|x|_t = r$, $|y|_t = -r$, $|z|_t = r-1$ is the only configuration which induces orientation R_5 and has no pre-position.

Case 6:



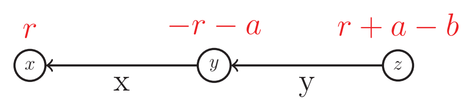
This configuration, call it $\mathscr{C}_t$, with $a \geq 1$ and $b \geq 1$, induces orientation R_6.

If $b \neq 1$, then a pre-position which induces orientation R_6 exists: $|x|_{t-1} = r + 1$, $|y|_{t-1} = -r + a$, $|z|_{t-1} = r - a + b - 1$.

If $b = 1$, then a pre-position which induces orientation R_2 exists: $|x|_{t-1} = r + 1$, $|y|_{t-1} = -r + a - 1$, $|z|_{t-1} = r - a + 1$.

Thus, every configuration which induces orientation R_6 has a pre-position.

Case 7:



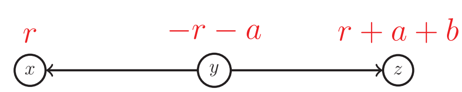
This configuration, call it C_t, with $a \geq 1$ and $b \geq 1$, induces orientation R_7.

If $b \neq 1$, then a pre-position which induces orientation R_7 exists: $|x|_{t-1} = r - 1$, $|y|_{t-1} = -r - a$, $|z|_{t-1} = r + a - b + 1$.

If $b = 1$, then a pre-position which induces orientation C exists: $|x|_{t-1} = r - 1$, $|y|_{t-1} = -r - a + 1$, $|z|_{t-1} = r + a - 1$.

Thus, every configuration which induces orientation R_7 has a pre-position.

Case 8:



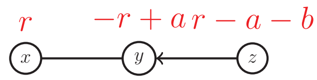
This configuration, call it C_t, with $a \geq 1$ and $b \geq 1$, induces orientation R_8.

If $a > 1$, then a pre-position which induces orientation R_8 exists: $|x|_{t-1} = r - 1$, $|y|_{t-1} = -r - a + 2$, $|z|_{t-1} = r + a + b - 1$.

If $a = 1$, then a pre-position which induces orientation R_4 exists: $|x|_{t-1} = r$, $|y|_{t-1} = -r$, $|z|_{t-1} = r + b$.

Thus, every configuration which induces orientation R_8 has a pre-position.

Case 9:



This configuration, call it C_t, with $a \geq 1$ and $b \geq 1$, induces orientation R_9.

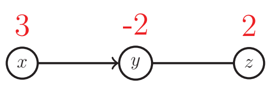
If $a > 1$, then a pre-position which induces orientation R_9 exists: $|x|_{t-1} = r + 1$, $|y|_{t-1} = -r + a - 2$, $|z|_{t-1} = r - a - b + 1$.

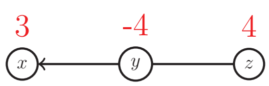
If $a = 1$, then a pre-position which induces orientation R_5 exists: $|x|_{t-1} = r$, $|y|_{t-1} = -r + a - 1$, $|z|_{t-1} = r - a - b + 1$.

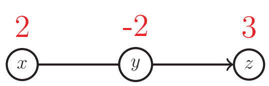
Thus, every configuration which induces orientation R_9 has a pre-position.

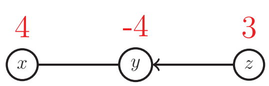
$\square$

Can multiple different configuration sequences exist on the same plane? Yes, by Lemma 5.2.7, on the plane $x + y + z = 3$, there exist four configurations which have no pre-position:

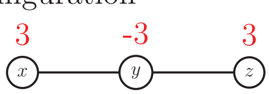
- 

- 

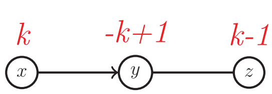
- 

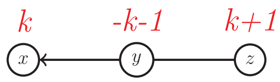
- 

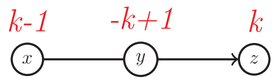
So, there exist 5 configuration sequences on the plane $x + y + z = 3$, in which no configuration appears in multiple. These are the four initiated by the four configurations just listed and also the configuration sequence which only contains the configuration

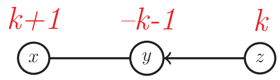


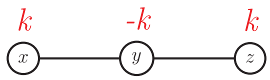
Corollary 5.2.8. *For a fixed sum k, given an ordered integer triple (a, b, c) such that $a + b + c = k$, the configuration of P_3 defined by these three stack sizes exists in the configuration sequence initiated by one of the following configurations:*

- 

- 

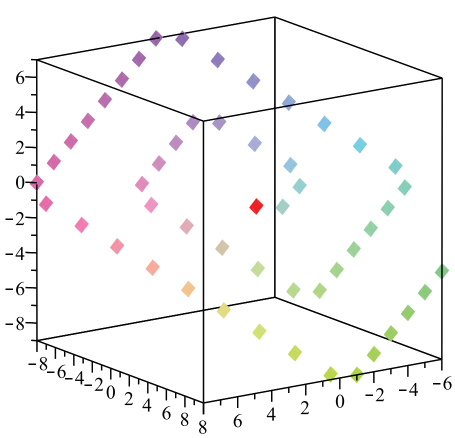
- 

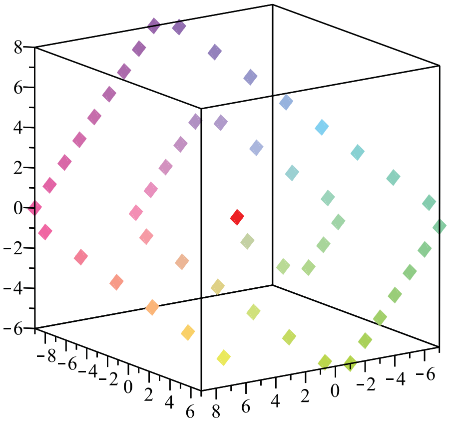
- 

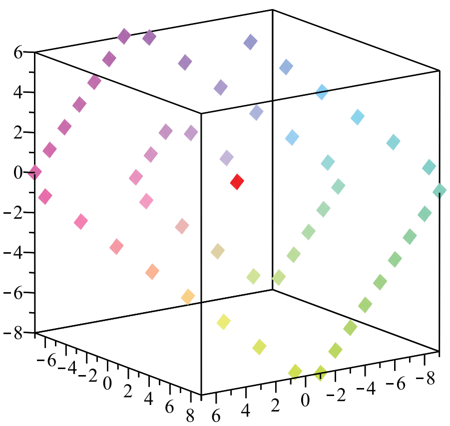
- 

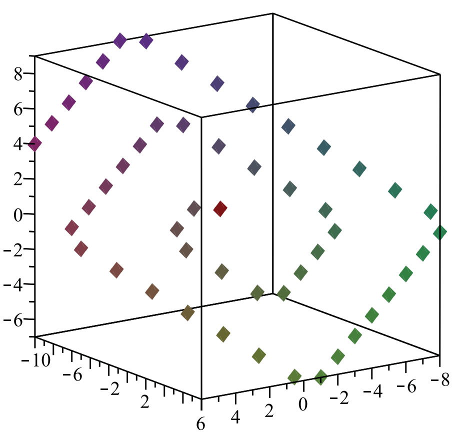
Proof. Since (a, b, c) is an ordered triple of integers, $|x|_t = a$, $|y|_t = b$, $|z|_t = c$ is a configuration of P_3 at some time t. By Lemmas 5.2.6 and 5.2.7, every configuration of P_3 belongs to exactly one configuration sequence initiated by a configuration with no pre-position (or a configuration which induces R_1). By Lemma 5.2.7, these are the five configurations that together generate every configuration with stack sizes adding to k. □

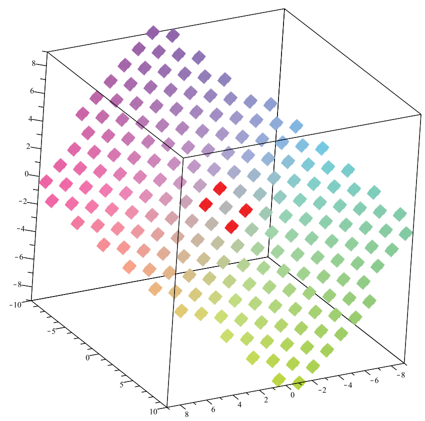
We now present images in 3-space of the four configuration sequences generated by the four configurations from Lemma 5.2.7 with zero as the sum of the stack sizes. Additionally, we include a fifth image of all of these configurations sequences overlaying each other and the point (0,0,0) (the periodic configuration sequence with sum 0).




8
6
4
2
0
-2
-4
-6
-8 -6 -4 -2 0 2 4 6
8 6 4 2 0 -2 -4 -6


6
4
2
0
-2
-4
-6
-8
-6 -4 -2 0 2 4 6 8
6 4 2 0 -2 -4 -6 -8


8
6
4
2
0
-2
-4
-6
-10 -6 -2 2 6
4 2 0 -2 -4 -6 -8



Given a plane $x+y+z=k$, every integer triple (x, y, z) on that plane exists in the configuration sequence generated by exactly one of the five initial configurations in Corollary 5.2.8. But how can we map from the configurations on the plane $x+y+z=k$ to the configurations on the plane $x+y+z=k+r$?

Theorem 5.2.9. *Suppose $a+b+c=k$ and that the configuration at time t on P_3 in some configuration sequence $Seq(C)$ is $C_t = \{|x|_t = a,\ |y|_t = b,\ |z|_t = c\}$. We know C_t can be mapped into 3-space on the plane $x+y+z=k$. Suppose the configuration at time 0, C, is one of the five configurations from Corollary 5.2.8. Then, the configuration $|x|^D = a+r$, $|y|^D = b-r$, $|z|^D = c+r$ exists at time t in its respective configuration sequence initiated by one of the five configurations from Corollary 5.2.8 (with $k+r$ substituted in for k).*

Proof. The five configurations from Corollary 5.2.8 (viewed as coordinates) are

- $(k,\ -k+1,\ k-1)$
- $(k,\ -k-1,\ k+1)$
- $(k-1,\ -k+1,\ k)$
- $(k+1,\ -k-1,\ k)$
- $(k,\ -k,\ k)$

Adding $(r, -r, r)$ to each one, we get

- $((k+r),\ -(k+r)+1,\ (k+r)-1)$
- $((k+r),\ -(k+r)-1,\ (k+r)+1)$

- $((k + r) - 1, \ -(k + r) + 1, \ (k + r))$
- $((k + r) + 1, \ -(k + r) - 1, \ (k + r))$
- $((k + r), \ -(k + r), \ (k + r))$

So, each of these five initial configurations on $x+y+z = k$ map to their respective corresponding initial configurations on $x + y + z = k + r$ by adding $(r, \ -r, \ r)$.

Every configuration that lies on the plane $x + y + z = k$ exists in a configuration sequence initiated by one of these five configurations.

Let L and M be two configurations with no pre-positions such that, when represented as triples, $L = M + (r, -r, r)$ for some integer r. Then, when represented as triples, $L_t = M_t + (r, -r, r)$ because the induced graph orientations at every step i , $0 \leq i \leq t$, are equal in the two configuration sequences.

□

5.2.2 Pay it Backward on Other Graphs

We now look to other graphs. First, is it possible to have a configuration inside a period of length one in Pay it Backward in which there exists an edge labelled $a \sim -b$ with $|a| \neq -|b|$? Yes, this is a difference between Pay it Backward and Parallel Diffusion and it is evidenced in Figure 5.18. Recall Lemma 3.1.16 which states that, up to equivalence, the only configuration that exists in a period of length one in Parallel Diffusion is the one in which every stack size is 0.

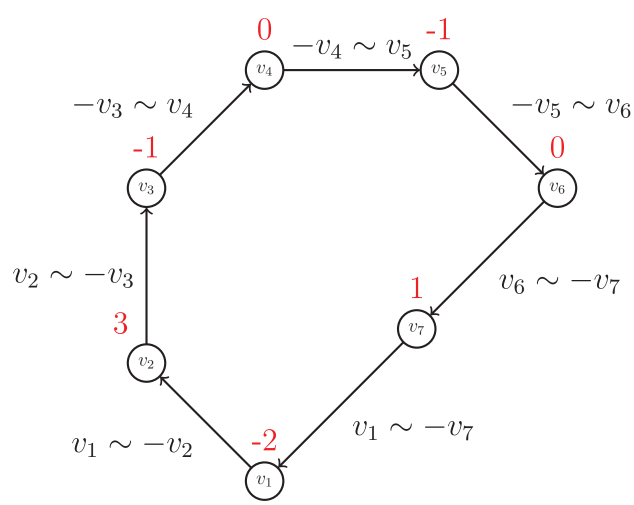


Figure 5.18: Period 1 example

To list all of the ways that a given cycle can yield a period of 1 in Pay it Backward, we begin by defining a noteworthy type of auxiliary graph.

Definition 5.2.10. *Given a graph G with n vertices, labelled v_1 to v_n, let G' be the graph with $2n$ vertices, labelled $\pm v_1$ to $\pm v_n$, such that a directed edge exists from v_i to $-v_j$ in G' if there exists an edge between v_i and v_j in G, that edge is labelled $-v_j \sim v_i$, and v_i will send a chip to v_j in the initial firing. We call G' the* ***Pay It Backward Auxiliary Graph****, shortened to* ***PIB Auxiliary Graph****.*

Clearly, G' is a bipartite graph, since for all v_i and v_j, there cannot exist an edge between v_i and v_j and there cannot exist an edge between $-v_i$ and $-v_j$.

For example, given the graph in Figure 5.19, the auxiliary graph is shown in Figure 5.20.

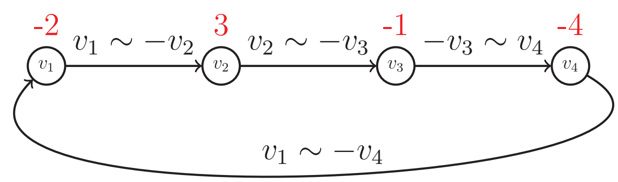


Figure 5.19: Pay it Backward period 1 example

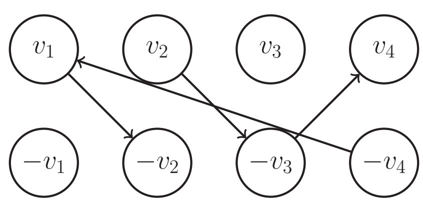


Figure 5.20: PIB-auxiliary graph from Figure 5.19

Note how the PIB-auxiliary graph is a graphical representation of a system of inequalities. In the case of Figure 5.20, the system of inequalities is

$$|v_2| > -|v_3| > |v_4|$$

$$-|v_4| > |v_1| > -|v_2|$$

Theorem 5.2.11. *Let G be a cycle and let G' be a PIB-auxiliary graph of G with orientation R such that R does not contain any flat edges, and some labelling of the edges. G' has no directed cycle and in-degree(v) + in-degree$(-v)$ = out-degree(v) +*

out-degree($-v$) = 1 for all $v \in V(G')$ if and only if there exists a configuration C satisfying G' such that $Seq(C)$ has period 1 and pre-period 0.

Proof. ($\Leftarrow$) Suppose that there exists some configuration C of G satisfying G' such that $Seq(C)$ has period 1 and pre-period 0. Every vertex $v \in V(G)$ has in-degree = out-degree = 1. So it must be true in G' that in-degree(v) + in-degree($-v$) = out-degree(v) + out-degree($-v$) = 1 for all $v \in V(G')$. Suppose now, by contradiction, that there exists a directed cycle in G'. We have previously noted that the PIB-auxiliary graph is equivalent to a system of inequalities. For some $v_i \in V(G')$, a directed cycle implies either $|v_i|_0 > -|v_{i+1}|_0 > |v_{i+2}|_0 > \cdots > |v_i|_0$ or $|v_i|_0 < -|v_{i+1}|_0 < |v_{i+2}|_0 < \cdots < |v_i|_0$. This is a contradiction since this system of inequalities cannot be satisfied.

($\Rightarrow$) Suppose now that G' has no directed cycle and that in-degree(v_i) + in-degree($-v_i$) = out-degree(v_i) + out-degree($-v_i$) = 1 for all $v_i \in V(G')$. Let $G = v_1v_2v_3 \ldots v_nv_1$. Since there are no directed cycles in G', there must exist a $v_i \in V(G')$ such that either in-degree(v_i) = out-degree($-v_i$) = 1 or in-degree($-v_i$) = out-degree(v_i) = 1. Choose such a v_i and call it v_1. As noted before, a PIB-auxiliary graph is equivalent to a system of inequalities. Without loss of generality, suppose the directed edge $v_1 \to -v_2$ exists in G'. This implies that the edge $v_n \to -v_1$ exists as well in G'. Thus, $|v_1|_0 > -|v_2|_0$ and $|v_1|_0 > -|v_n|_0$.

We will assign stack sizes to the vertices using the following algorithm to create a configuration which satisfies G'.

1. Let $|v_1|_0 = n$. Set $i = 1$. Set $k = n - 1$.

2. If $-v_{i+1} \to v_{((i+2) \bmod n)}$ in G', then let $|v_{i+1}|_0 = -k$. Otherwise, if $v_{i+1} \to -v_{((i+2) \bmod n)}$ in G', then let $|v_{i+1}|_0 = k$. $i = (i + 1)$. $k = k - 1$.

3. If $i = n$, end. Else, back to 2.

Every time the algorithm repeats, an initial stack size is chosen for v_{i+1} such that it will satisfy both of its directed edges. We can be certain that the final edge $v_n \to -v_1$ is also satisfied since the v_n necessarily receives a stack size of 1 by our algorithm and $1 > -n$. Thus, since each vertex receives exactly one chip and sends exactly one chip in the initial firing of C, we can conclude that C is a configuration of G satisfying G' such that $Seq(C)$ has period 1 and pre-period 0.

□

For completeness, we note that the orientation of a cycle in which every edge is flat can be induced by the configuration in which every stack size is 0. No orientation of a cycle in which some edges are flat and others are not can be induced by a configuration C such that $Seq(C)$ has period 1 and pre-period 0. This is because there exists at least one vertex v for which in-degree(v) + in-degree($-v$) $\neq$ out-degree(v) + out-degree($-v$) in G'.

On cycles, the existence of a directed cycle in the PIB-auxiliary graph is necessary and sufficient to identify an orientation that cannot exist with a period 1 configuration. However, when there exist vertices with degree larger than 2, the PIB-auxiliary graph becomes less useful in identifying orientations that cannot exist with a period 1 configuration because directed cycles are no longer the only contradiction. In Figure 5.21, we see a graph, G, that implies a system of inequalities

$$|v_1| > -|v_2| > |v_3| > -|v_4| > |v_1|$$

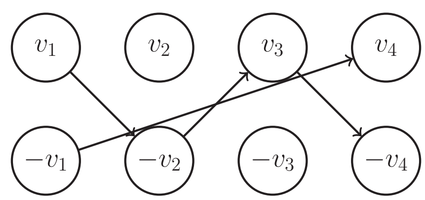


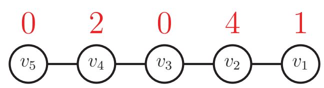
Figure 5.21: PIB-auxiliary graph which cannot exist with period 1 configuration

which cannot be satisfied with real numbers. This implies that if we are looking for period 1 orientations, any PIB-auxiliary graph that contains G as a subgraph fails. This is problematic as G is not as easily classified from a Graph Theory perspective as the directed cycles were from the previous theorem. However, a PIB-auxiliary graph is just a system of inequalities, no matter how many edges are involved. There exists a solution to the system of inequalities using integers if and only if there exists a configuration with a period of 1 satisfying the PIB-auxiliary graph.

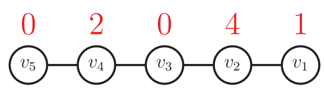
5.3 Sequential Diffusion

In this section, we introduce Sequential Diffusion. In Sequential Diffusion, at each step, only a single vertex is chosen to fire. It will then send a chip to each of its poorer neighbours. In this process, determining a rule for which vertex will fire at a given step can be difficult, so in Subsection 5.3.1, we introduce the notion of a vertex ordering to determine which vertex will fire at a given step. These processes are defined broadly so that an ordering is not necessary and any vertex can be chosen to fire at a given step, but by determining an ordering of firings at the onset, we can more easily prove results. In Subsection 5.3.1, we examine Sequential Diffusion specifically on the Millpond configuration. We clarify the notion of periodicity in Sequential Diffusion and then in Theorem 5.3.8, we prove that trees can exhibit periodic configuration sequences if their firing order is chosen using a certain algorithm. An example of Sequential Diffusion is provided in Figure 5.22.

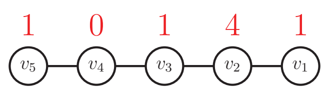
This is joint work with Dr. Margaret-Ellen Messinger of Mount Allison University.



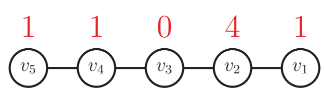
v_5 fires but has no poorer neighbour



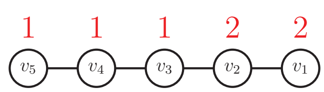
v_4 fires



v_3 fires



v_2 fires



v_1 fires but has no poorer neighbour

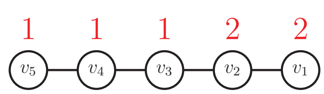


Figure 5.22: Several steps in a Sequential Diffusion process on P_5.

5.3.1 Millpond

We now examine Sequential Diffusion with the Millpond configuration. Recall that this means that one vertex has a chip and every other vertex has 0 chips. In this subsection, we look at the idea of a period in Sequential Diffusion and conclude with Theorem 5.3.8, which shows that trees can have periodic configuration sequences if a particular ordering is followed. In [9], Duffy et al. explore Millpond in Parallel Diffusion. They conclude that given a graph G, the pre-period can have length no larger than the greatest distance between any two vertices in G.

Definition 5.3.1. *In Millpond, the **initial vertex** is the vertex that begins with a chip.*

In Sequential Diffusion, at each step, only one vertex will fire. For the purposes of this section, we will suppose that the vertices fire based on a given ordering. That is, if a vertex v fired at step t, then v will fire at step k if and only if $k = t + ni$ where i is an integer and n is the number of vertices in G.

In Sequential Diffusion, vertices need not follow such an ordering. However, for the remainder of this section, we will suppose that they do.

Definition 5.3.2. *In Sequential Diffusion on a graph G with n vertices, when the vertices fire according to an ordering, every n steps signifies a **round**. So, Round 1 consists of steps 0 to $n-1$, Round 2 consists of steps n to $2n-1$, and Round k consists of steps $(k-1)n$ to $kn-1$.*

Lemma 5.3.3. *Let M be the maximum stack size of an initial configuration C. In Sequential Diffusion with initial configuration C, no stack size can ever exceed M.*

Proof. Suppose, by contradiction, that the first time a vertex has a stack size greater than M occurs at step t. In Sequential Diffusion, since vertices fire one at a time, no vertex can gain more than one chip at a single step. So, at step $t-1$, our vertex in question, v, must have gained a single chip. So, $|v|_{t-1} = M$ and for some vertex u adjacent to v, $|u|_{t-1} > M$. This, however, is a contradiction. □

Lemma 5.3.4. *Let G be a graph and $x \in V(G)$. In Sequential Diffusion on the Millpond configuration, if t is the first step in which x sends a chip to a poorer neighbour, then $|x|_t > 0$.*

Proof. Suppose, by way of contradiction, that step t is the first step in which a vertex x sends a chip to a poorer neighbour, and suppose that $|x|_t \leq 0$. Suppose y is adjacent to x and $|y|_t < |x|_t$. Since $|y|_t < 0$, y must have sent a chip to one of its neighbours during a previous step $j < t$. When y fired at step j, it must have had one chip since t is the first step in which a vertex with 0 or fewer chips sends a chip to a poorer neighbour, and, by Lemma 5.3.3, no vertex can ever have a stack size greater than 1. Before y fired at step j, x must have had at least 0 chips (having not yet sent a chip to a poorer neighbour). When y fired at step j, either x had 1 chip or x had 0 chips and was thus, poorer than y. Therefore, after y fired, x must have had exactly 1 chip. But this contradicts that x had less than one chip when it first sent a chip to a poorer neighbour. Thus, if t is the first step in which x sends a chip to a poorer neighbour, then $|x|_t > 0$. □

Lemma 5.3.5. *Let T be a tree on n vertices. In Sequential Diffusion on the Millpond configuration, for any step $t \leq n-1$, let U_t be the set of vertices that have sent a chip to a poorer neighbour during any step i where $i \leq t$. Let v_0 be the initial vertex. For any $v \in U_t$, there is a path $P_{v_0 v}$ connecting v_0 to v such that every vertex in $P_{v_0 v}$ is also in U_t.*

Proof. Since T is a tree, there is a unique path connecting v_0 to v for all $v \in V(T)$. We will induct on the number of steps that have occurred in which the firing vertex had at least one poorer neighbour. The first such firing occurs when v_0 sends a chip to all of its neighbours. The second such firing occurs when one of the vertices adjacent to v_0 sends a chip to each of its poorer neighbours, including v_0. Each of these vertices has a unique path connecting it to v_0 (consisting of just those two vertices). Thus, our base case is satisfied. Suppose now that k firings have occurred that satisfy our criteria and that the $(k+1)$-th such firing comes from v_{k+1} at step $t \leq n-1$.

Case 1: $|v_{k+1}|_t \leq 0$

Since step t occurs within the first round, we know that v_{k+1} did not fire in any previous step. By Lemma 5.3.4, v_{k+1} does not have any poorer neighbours at time t. This contradicts our assumption that v_{k+1} will fire at step t and have at least one poorer neighbour. Thus, v_{k+1} has more than 0 chips at step t.

Case 2: $|v_{k+1}|_t = 1$

Initially, v_{k+1} had 0 chips, and v_{k+1} has not yet lost any of its chips. Thus, on exactly one previous step, v_{k+1} received a chip from a neighbour. So, v_{k+1} is adjacent to a vertex that has already fired, call it v_j, and we know that v_j had a poorer neighbour when it fired. We know that there exists a unique path connecting v_0 to v_j containing only vertices which have previously sent a chip to a poorer neighbour. Thus, by adding the edge $v_j v_{k+1}$ and the vertex v_{k+1} to the end of this path, we get a path connecting v_0 to v_{k+1} containing only vertices that have previously sent a chip to a poorer neighbour. □

Corollary 5.3.6. *Let T be a tree. In Sequential Diffusion on the Millpond configuration, suppose $x \in V(T)$ and x is not the initial vertex. Let step $t \leq n-1$ be the first step in which x sends a chip to a poorer neighbour. Then at step t, x is adjacent to exactly one vertex that has fired previously.*

This corollary follows because, if there were more than one vertex adjacent to x that had previously sent a chip to a poorer neighbour, then they could not both have a unique path connecting them to the initial vertex without including x itself.

Lemma 5.3.7. *Let T be tree, $x \in V(T)$. In Sequential Diffusion on the Millpond configuration, let i be the first step in which x sends a chip to a poorer neighbour. Suppose that x is not the initial vertex and that $i < n-1$. Then $|x|_t < 1$ for all $i < t \leq n-1$.*

Proof. Let $i < n-1$ be the first step in which a vertex, call it v, other than the initial vertex both fires and has a poorer neighbour. After this firing, v must have less than 1 chip. This first step in which a vertex other than the initial vertex sends will serve as a base case. Let D_k be the set of all vertices that sent a chip to a poorer neighbour at any time $j \leq k$. For our induction hypothesis, we suppose that every vertex in D_t, with $i \leq t < n-1$, has less than 1 chip. Suppose x fires at step $t+1$ (still in the first round). By Corollary 5.3.6, at step $t+1$, x is adjacent to exactly one vertex, call it y, that sent a chip at a previous step. By Lemma 5.3.4, at step $t+1$, x has exactly one chip. By our induction, we get that $|y|_t < 1$. So, when x fires, every one of its neighbours has exactly 0 chips except for y which may have less. Thus, the stack size of x will reduce from 1 to $-deg(x)+1$. As the round progresses, the stack size of x may increase as its neighbours fire (the size of the increase depending on the ordering of the vertices). However, the stack size of x will only increase by at most one chip per step and x has at most $deg(x)-1$ neighbours that have yet to fire. Thus, by the end of the round, the stack size of x can reach a maximum of $(-deg(x)+1)+(deg(x)-1)=0$. □

Algorithm for Trees: Let T be a tree on n vertices. Consider any sequence derived from the following rule: At each step, fire any vertex that both has a poorer neighbour and has not yet fired, if one exists. After every vertex has fired, continue firing each round using the same ordering.

Theorem 5.3.8. *Orderings derived from the* ***Algorithm for Trees*** *have pre-period* 0 *and period* n*, where* n *is the number of vertices in the tree.*

Proof. By the **Algorithm for Trees**, the first vertex in our sequence must be the initial vertex. By Lemma 5.3.4, the second vertex must be one that is adjacent to the initial vertex since these are the only vertices which have a chip at this point. Suppose that we have reached some step t by using the **Algorithm for Trees** and have not yet at any step failed to find an unfired vertex with a poorer neighbour. Look now at the subtree within T consisting of all vertices that have fired thus far (We know that it is a subtree as opposed to a subforest by Lemma 5.3.5). Call it T'. Note that every vertex in $T \setminus T'$ that is adjacent to a vertex in T' must have exactly 1 chip. This is because these vertices began at 0 chips, have yet to fire, and must have received a chip from their unique neighbour's firing in T'. Thus, at step t, if there exists a vertex that has not yet fired, then there exists a vertex with exactly 1 chip that has a poorer neighbour (by Lemma 5.3.7). So, our algorithm will complete only after every vertex has fired. That is, the **Algorithm for Trees** never fails to find a vertex that both has a poorer neighbour and has not yet fired, until every vertex has fired.

Because of this, we know that after a vertex x (other than the initial vertex) fires in the first round, going from 1 chip to $-deg(x) + 1$ chips, there exist $deg(x) - 1$ vertices adjacent to x yet to fire. So, x must finish the first round with 0 chips. Since the sum of chips in the tree can never change, this implies that the initial vertex must finish the first round with one chip. Finally we note that this configuration of chips could not occur at any point between the first and last steps of the first round since at every step after the first, there exists a vertex (other than the initial vertex) with exactly 1 chip. Thus, orderings derived from the **Algorithm for Trees** have pre-period 0 and period n, where n is the number of vertices in the tree. □

Chapter 6

Discussion

In this chapter, we look at the questions that remain open and may guide the future study of these topics.

Question 6.0.1. *Do similar recurrence relations exist to explain the number of Parallel Diffusion configurations that exist on cycles, trees, k-regular graphs, bipartite graphs, etc.?*

Thus far, recurrence relations have been discovered for complete graphs and paths, with a simple explicit solution for stars. Finding a way to count period configurations on other families of graphs appears to be the next logical step. However, much effort has been put into solving for such recurrence relations on cycles, trees, and complete bipartite graphs, with no results.

Question 6.0.2. *In Parallel Diffusion, how can one quickly determine from an initial configuration whether the period length of the configuration sequence will be 1 or 2?*

For Chip Firing [3], there exist results which can help sort initial configurations into those that will and those that will not eventually terminate. However, for Parallel Diffusion, we know very little about mapping pre-period configurations to the period configurations with which they share a configuration sequence.

Question 6.0.3. *In Sequential Diffusion, under the Millpond layout, must the process be eventually periodic when the same firing order is followed every round?*

Our Algorithm for Trees can be used to yield a periodic configuration sequence. However, we have no results regarding any other ways of determining an ordering. Choosing another ordering appears to increase the pre-period, but we do not yet know if the Millpond configuration always has a periodic configuration sequence, or what periods it may exhibit.

Define Sign Diffusion to be the diffusion variant in which the firing rules are the following subset of those in Parallel Diffusion:

$$S' = \bigcup_{a>0>b} \{a \to b\}, a, b \in \mathbb{Z}$$

Question 6.0.4. *What period lengths are possible in Sign Diffusion?*

Long and Narayanan's result regarding periodicity [14] shows that Parallel Diffusion is periodic with period 1 or 2. We already know that changing the rules too much will create a process that is either not periodic or periodic with large periods like Pay it Backward and Two-One Diffusion, respectively. With this question, we ask if the same argument or a similar argument to the one used by Long and Narayanan can be used to find period lengths on a process with similar rules, like Sign Diffusion.

Question 6.0.5. *Beyond P_3, do there exist more graphs for which pre-positions are unique in Pay it Backward?*

A lot of time and effort went into reaching the results for Pay it Backward on P_3 in Subsection 5.2.1. Preliminary work on K_3 and P_4 under specific edge labellings have shown that it will be a much more difficult problem to prove or disprove unique pre-positions for these graphs than it was with P_3.

Question 6.0.6. *In Sequential Diffusion, what firing orders allow for the player to force as many vertices as possible into debt? What firing orders allow for the player to keep as many vertices as possible out of debt?*

The biggest driving force in studying Sequential Diffusion is the desire to turn the wealth sharing process of Parallel Diffusion into a one or two player game. There are many possibilities as to what the goals of the respective players may be in these games. Ideas have included one player games where the player is attempting to force as many vertices as possible into debt, or attempting to keep as many stack sizes as possible positive. Alternatively, we could define a two player game in which players alternate choosing vertices to fire in which one player wants to divide wealth and the other wants consolidate wealth. Answering these types of questions would make analyzing such games much easier.

Bibliography

[1] R. W. Bailey, The number of weak orderings of a finite set, *Social Choice and Welfare*, 1998, 15, 559-562.

[2] J. Bitar, E. Goles, 1992, Parallel chip firing games on graphs, *Theoretical Computer Science*, 92, 291-300.

[3] A. Bjorner, L. Lovász, P. W. Shor, 1991, Chip-firing games on graphs, *European Journal of Combinatorics*, 12, 283-291.

[4] A. Brandstädt, A. Leitert, D. Rautenbach, 2012, Efficient dominating and edge dominating sets and hypergraphs, *Algorithms and Computation*, *Lecture Notes in Comput. Sci.*, Springer and Heidelberg, 7676, 267-277.

[5] R. Brualdi, Introductory Combinatorics, Prentice Hall, 2009.

[6] A. Carlotti, R. Herrman, 2018, Uniform bounds for non-negativity of the diffusion game, arXiv:1805.05932v1.

[7] G. Chartrand and P. Zhang, Introduction to Graph Theory, McGraw Hill, 2005.

[8] J. Degaetani, Candy sharing and chip firing games on graphs, Masters Thesis, Montclair State University, 2018.

[9] C. Duffy, T. F. Lidbetter, M. E. Messinger, R. J. Nowakowski, 2018, A Variation on Chip-Firing: the diffusion game, *Discrete Mathematics & Theoretical Computer Science*, 20, #4.

[10] S. Gaspers, M. E. Messinger, R. J. Nowakowski, P. Pralat, 2009, Clean the graph before you draw it! *Inform Process Lett*, 109(10):463-467.

[11] S. Gaspers, M. E. Messinger, R.J. Nowakowski, P. Pralat, 2010, Parallel cleaning of a network with brushes, *Discrete Applied Mathematics*, 158, 467-478.

[12] M. A. Kiwi, R. Ndoundam, M. Tchuente, E. Goles, No polynomial bound for the period of the parallel chip firing game on graphs, Theoretical Computer Science, 1994, vol. 136, pp. 527-532.

[13] D. A. Klarner, 1965, Some results concerning polyominoes, *Fibonacci Quarterly*, 3, 9-20.

[14] J. Long and B. Narayanan, 2019, Diffusion On Graphs Is Eventually Periodic, *Journal of Combinatorics*, 10, no.2, 235-241.

[15] C. Merino, The chip firing game and matroid complexes, Discrete Mathematics and Theoretical Computer Science Proceedings, Paris, 2001, 245-256.

[16] OEIS Foundation Inc. (2019), The On-Line Encyclopedia of Integer Sequences, http://oeis.org.

[17] M. E. Messinger, R. J. Nowakowski, P. Pralat, Cleaning a network with brushes, *Theoretical Computer Science*, 2008, 399, 191-205.

[18] D. West, Introduction to graph theory, Prentice Hall, 2001.