Technische Universiteit Delft
Faculteit Elektrotechniek, Wiskunde en Informatica
Delft Institute of Applied Mathematics

The Chip firing game

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

BACHELOR OF SCIENCE
in
TECHNISCHE WISKUNDE

door

HELEEN SCHUTTE

Delft, Nederland
Juni 2015

BSc verslag TECHNISCHE WISKUNDE

"The Chip firing game"

HELEEN SCHUTTE

Technische Universiteit Delft

Begeleider

Dr. D. Gijswijt

Overige commissieleden

Dr. F. van der Meulen

Dr. B van den Dries

Juni, 2015

Delft

Abstract

We consider the following solitary game. Each node of a graph contains a pile of chips. A move consists of selecting a node with at least as many chips as the number of outgoing lines. After we selected a node we have to fire the node. This means that one chips is transported from the selected node to all the neighboring nodes over the outgoing lines. The game terminates if there are no more nodes left which can be fired. We analyze the game on directed graphs and we investigate the finiteness of the game. We introduce some properties of the chip firing game by associating it with a left-hereditary, permutable and locally free language. We use these properties to calculate, given an initial position and a final position, how many times we have to fire each node to finish the game or we can draw the conclusion that the game will never terminate. We introduce a linear system in terms of the Laplacian to decide finiteness of the game. In order to set up this system we use some linear algebra to calculate the Hermite normal form of the linear system. With this Hermite form we can find an integer solution for the number of steps needed to finish the game. We also introduce a Java applet in which you can make your own chip firing game and play it.

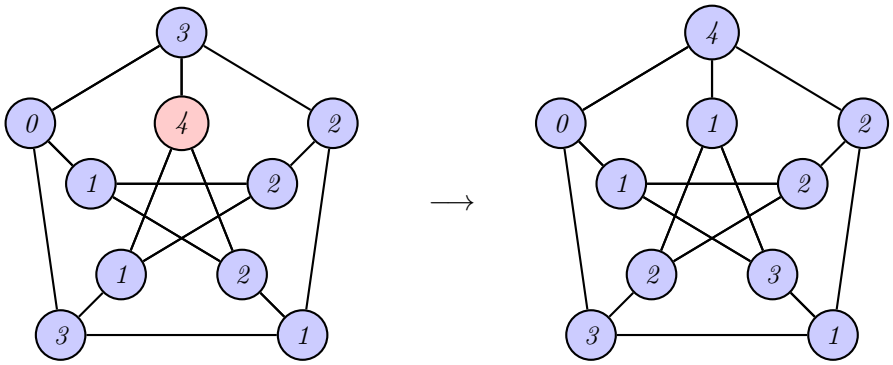
Contents

Chapter 1

Introduction

The process called the chip firing game has been around for more than 30 years now. It has become an important and interesting object of study for structural combinatorics. In Spencer introduced the Balancing game [1] on graphs which look like long paths. This game corresponds to the discrepancy problem of matrices [2]. Later, in [3], the game was defined on a general graph and the Chip firing game was born. The rules of the Chip firing game are as follows. Given a graph $G = (V, E)$ we place an integer number of chips on each node. A move consists of selecting a node with at least as many chips as its outdegree. After selecting this node, we fire the node, which means that one chip is transported over each outgoing line to a neighbor node. This way, chips are moved around the graph but the total number of chips is unchanged.

Example 1.1. *We give an example of a move in the chip firing game on a undirected Petersen graph. The number on a node represents the number of chips placed on that node. We fire the red node in the first figure and we see the result in the second figure.*

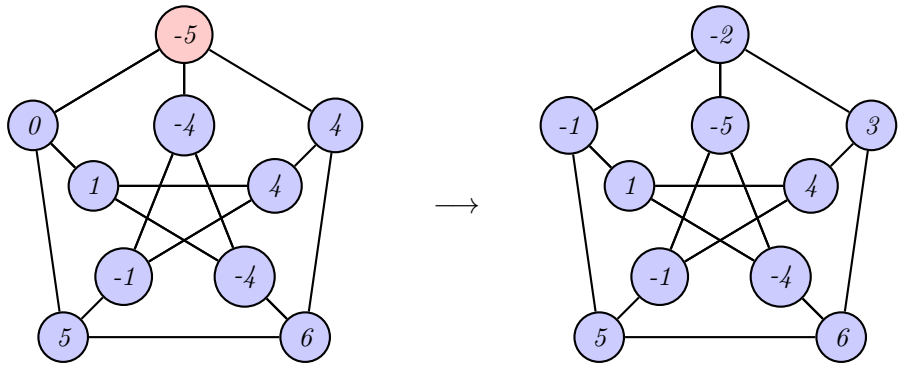


After the introduction of the Chip firing game, Bak,Tang and Wiesenfeld introduced a related process that is known as the Abelian sandpile. This dynamic process was the first one discovered with self-organized criticality. The idea of self-organised criticality can be illustrated by a pile of sand. Grains of sand are dropped onto a pile one by one. The pile ultimately reaches a stationary 'critical' state in which the slope of the pile fluctuates about a constant angle of repose. The angle of repose is the steepest angle to which a material can be piled without slumping. The game that corresponds to the Abelian sandpile is similar to the Chip firing game but slightly different. Merino [6] described it in the following way: the rules are the same for every vertex except for q, q has a debit of chips equal to the total number of chips on the graph and can only be fired when every other vertex cannot be fired, then q is fired until some vertex has enough chips to get fired again. The last rule ensures an infinite game. This game is called the dollar

game, and dollars are used instead of chips.

In a more sophisticated version of the Dollar game, depths are allowed on more vertices, and the firing means getting one dollar from all the neighbors, not sending. Here we show an example of a move in the Dollar game:

Example 1.2. *We give an example of a move in the Dollar game on a undirected Petersen graph. The number on a node represents the number of chips placed on that node. We fire the red node in the first figure and we see the result in the second figure.*



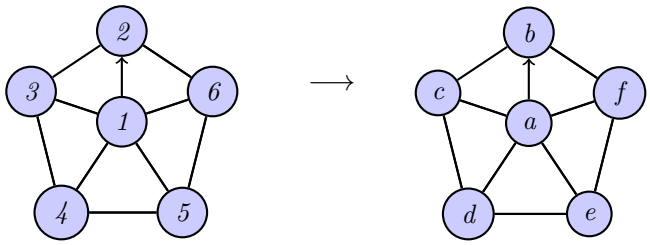
The chip firing game will be investigated here. It is clear that not every random graph with a random number of chips on each node gives a terminating game. How can we, given a graph and an initial position of chips, decide if the configuration is winning/terminating? And can we find a solution in polynomial time? I will investigate some properties of the chip firing game related to these questions and in particular the finiteness of the game.

Chapter 2

Language to describe properties of the chip firing game

To give a good analysis of the chip firing game we need give a 'formulation' of the chip firing game in terms of a language L. This idea was introduces by Bj owner and lovás [5]. Let V be a finite set. A *language* over V is any set of finite words formed by elements of V. We describe the sequence of firing nodes as a word, where each letter represents a node.

Example 2.1. *If we give each node a number (not the number of chips, just the name of the node) than we can replace this number by a letter:*



If we fire node a two times and then fire node b one time we get the word aab. So a combination of letters is a combination of firing nodes. A *subword* of a word α is any string obtained by deleting letters from α arbitrarily. So aba is a subword of $ababbc$. We also introduce the *score* of a word. The *score* of a word is a vector in which the ith entry represents the number of times letter i occurs in the word. We denote the *score* of word α as $[\alpha]$. We call a word *basic* if it is not the beginning section of any other word.

We can write the legal sequences of moves from a given initial chip configuration in a chip firing game as a language, this language L has three properties:

- The language is *left-hereditary*
  If a word is legal (i.e. corresponds to a legal sequence of moves), then all the beginning sections of this word are legal. For example let $abcda \in L$ than $abc \in L$ and $abcd \in L$.

- The language is *permutable*
  If α and β are to words with the same score: $[\alpha] = [\beta]$ and αx is a legal word in L, then βx is also a legal word in L. This means that the order in which we fire nodes is not relevant for the result of the final position after these moves, only the number of times that we fire

each node is important. Let $\alpha = abbccdad$ and $\beta = aabcddbc$, if $abbccdadd \in L$ than also $aabcddbcd \in L$.

- The language is *locally free*
  If $\alpha x \in L$ and $\alpha y \in L$ with $x, y \in V$ then also $\alpha xy \in L$ and $\alpha yx \in L$. So if we can fire the nodes corresponding to letter x and letter y after a sequence of moves corresponding to word α, then we can choose which node we fire first and after this move we can still fire the other node. Let $\alpha = abba \in L$. If $abbab \in L$ and $abbac \in L$ then also $abbacb \in L$.

2.1 The strong exchange property

We introduce a basic fact about the finiteness of the chip-firing game established by Björner and Lovász:

Theorem 2.2 ([5])**.** *Given a directed graph G, and an initial position of chips, either every legal game can be continued indefinitely, or every legal game terminates after the same number of moves with the same final position. The number of times a given node is fired is the same in every legal game.*

To prove this theorem we use the following proposition:

Proposition 2.3. *Let L be a left-hereditary, permutable and locally free language. Then the following hold.*

(i) *If $\alpha, \beta \in L$ then there exists a subword α' of α such that $\beta\alpha' \in L$ and $[\beta\alpha']$ it the entry-wise maximum of $[\alpha]$ and $[\beta]$. (The strong exchange property)*

(ii) *If there is a basic word then the language is finite.*

(iii) *All basic words have the same score (in particular, the same length)*

1. *If L is finite then two words $\alpha, \beta \in L$ have $[\alpha] = [\beta]$ if and only if*

$$\{\gamma : \alpha\gamma \in L\} = \{\gamma : \beta\gamma \in L\}$$

Proof. We proof (ii) and (iii) using (i).

1. -

2. We have to prove that if L has a *basic* word α then there are a finite number of words in L. Since a *basic* word is not the beginning section of any other word (i.e. we cannot add any letters to α). We can on the other hand delete letters from α because L contains al the subwords of α. We are left with words γ for which $[\gamma]_i > [\alpha]_i \forall i \in V$. This set of words cannot be in L. Because of (1) we can add the letters to α which occur more times in β and γ, but this contradicts that α is a *basic* word. So if a language L has a basic word α then L contains only subwords of α and since α has a finite length, the number of subwords of α is also finite which implies that the language is finite.

3. Assume that we have two *basic* words α and β with different scores $[\alpha]$ and $[\beta]$. Then because of the *strong exchange property* there exists a *sub word* α' of α such that $\beta\alpha' \in L$ and $[\beta\alpha']$ is the entry-wise maximum of $[\alpha]$ and $[\beta]$. Then β is the beginning section of a longer word $\beta\alpha'$, this contradicts that β is a *basic* word.

4. -

□

Proof of Theorem 1. We see that the score of a word in the language L determines the position reached by the corresponding game, because of the *permutable* property. The order is not of importance here, only the number of times we fire a node: the *score*. A word is *basic* if and only if that position is terminal. Otherwise we could fire another node, which means that we can add another letter to our *basic* word, but this contradicts that the word is *basic*. So theorem 1 follows directly form the *left-hereditary*, *permutable*, *locally free* and the *strong exchange property*.

□

Chapter 3

The finiteness of the game

3.1 No polynomial bound for the directed chip firing game

We can ask ourselves how we construct a legal finite game and if we found such a game how long the game will last before it terminates? It is easy to understand that the game will never end if we draw a network of nodes connected by lines (graph) and we add more chips on each node than the total number of nodes. If we add more chips on each node than the total number of lines leaving each node, than we also have a infinite game.

So what is the maximum number of chips that allows a finite game and what is the minimum number of chips that allows an infinite game? Let G be a connected grap and let V be the set of nodes and E be the set of lines, n represents the number of nodes and m the number of (undirected)lines. Fewer than m chips guarantees that the game is finite.[7]

It was shown [5] that more than $2m - n$ chips guarantees that the games is infinite. Since

$$m = |E| = \frac{1}{2} \sum_{v \in V} d(u)$$

we get

$$2m - n = \sum_{v \in V} d(v) - n = \sum_{v \in V} d(v) - |V|$$

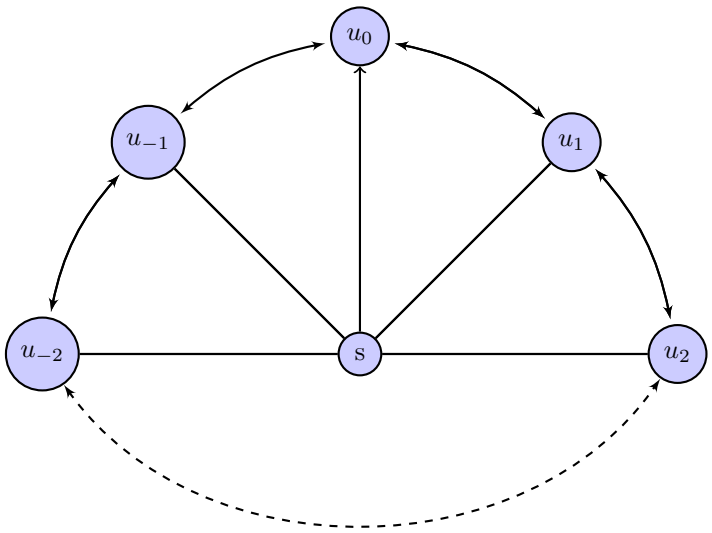
Hence if we add all the degrees of every $v \in V$ and then subtract the number of nodes, this every node can have as many chips as his degree minus 1, and as soon as we add one more chip than there will always be one node that has the same number of chips or more than the number of outgoing lines.

Theorem 3.1. *[7] For every number N of chips on graph G with $m \leq N \leq 2m - n$, there are initial positions that lead to an finite game and initial positions that lead to a infinite game.*

For directed graphs, we can say that the game doesn't terminate if we have more than $m - n$ chips on the graph, because each line goes one way, we do not have to take in account both ways so we we don't have to double the number of lines. The maximum number of chips that guarantee that the game will terminate is unknown. We also don't know how to determine the minimum number of chips allowing an infinite game on a general digraph. This is not a function of just the number of nodes and edges.

Eriksson [8] shows that no polynomial bound can be found for the chip firing game on directed graphs. He sets up an example of a bidirected graph for which the final position will be

reached, but this will be achieved in a exponential number of moves. In this example he uses a graph which is set up in the following way. A game is defined for every even positive number n. Take a bidirected circuit of $n-1$ nodes and add a center node with edges to all other nodes, all but one bidirected, the last one directed from the center node to the top node. The initial position is $3n-5$ chips on the center node and no chips on the circuit nodes. The final position is reached when we have $n-2$ chips on the center node, 1 chip on the top node and 2 chips on all other nodes. The game will always end because Eriksson set up this special graph, if the final position isn't reached there will always be a node that can be fired.
Eriksson showed that this game will always terminate. Consider the directed graph G and follow Eriksson's proof, we add some more detailed arguments.



All edges are bidirected except su_0. Let x_k be the number of chips on node k, the chips distribution on G will be denoted by a sequence of type:

$$(...x_{-2}, x_{-1}, x_0, x_1, x_2, ...)$$

The subscripts are modulo $(n-1)$.Note that $d^+(s) = n-1$ (center node), $d^+(0) = 2$ (top node) and $d^+(i) = 3$ (all other nodes). We start with $3n-5$ chips on the center node and 0 chips on all the other nodes. We can fire this node once, which results in 1 chip on every node except for the center node. Then we can perform the following sequence:

1. fire the center node
2. fire the top node
3. fire all the other node on the circuit clockwise
4. fire the center node
5. fire the top node
6. fire all the other nodes on the circuit clockwise
7. fire the top node

Each node is fired twice except for the top node which is fired three times. So this operation consists of $2n+1$ moves and results in a netto flow of two chips from the center node to one of each of the neighbors of the top node. This operation can be done over and over again with a few extra steps in between. Starting from the initial position the sequence will look as follows:

fire the center node:	...1111111...	P_0
$2n+1$ moves	...1121211...	P_1
$2n+1$ moves	...1131311...	
fire the top node's neighbors and the top node	...1211121...	
$2n+1$ moves	...1221221...	P_2

Let now P_i denote the position with i twos on each side of the top node. Assume that for $i = 1, 2, ..., k-1$ we know sequences of a_i moves leading from P_{i-1} to P_i. So the game from position P_{k-1} to P_k is as follows:

we start with	...112...212...211...	P_{k-1}
a_{k-1} moves	...132...212...231...	
fire all nodes from left 3 to right 3 $(2k-1)$ moves		
rebuild inner twos $\sum_1^{k-1} a_i$	...222...212...222...	P_k

Let a_k be the number of moves from P_{k-1} to P_k then we get

$$a_k = a_{k-1} + (2k-1) + \sum_{i=1}^{k-1} a_i \text{ for } k \geq 2, a_1 = 2n+1$$

Let s_k be the total number of moves from initial position to P_k then we have to add the first time we fire the center node to get: ...111...111...111.... We get the following formula:

$$s_k = 1 + \sum_{i=1}^{k} a_i$$

$$s_k - 3s_{k-1} + s_{k-2} = 2k - 2$$

This recurrence has the following solution:

$$s_k = \left(\frac{2n}{\sqrt{5}} + \frac{1+\sqrt{5}}{2}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{2k} + \left(\frac{-2n}{\sqrt{5}} - \frac{2}{1+\sqrt{5}}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{-2k} - 2k$$

So the number of moves necessary to reach this final position grows exponentially with the number of nodes:

$$(2n/\sqrt{5} + (1+\sqrt{5})/2) \cdot ((1+\sqrt{5})/2)^{n-2}$$

Therefore there is no polynomial bound for the chip firing game on a directed graph.
To make the game more visible and check Erikssons algorithm I made a Java applet where you can play the Chip firing game (see appendix).

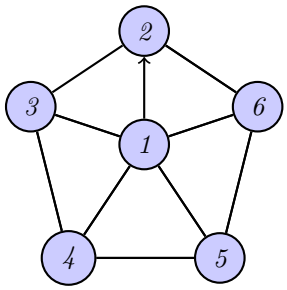
3.2 Linear algebra for the chip firing game

Firing a node in the chip firing game means that we decrease the number of chips on the node by the out degree of the node. At the same time we increase the number of chips on al the outneighbor nodes by 1. We denote the out degree of a node i by $d^+(i)$ and we denote the in degree of node i by $d^-(i)$.

We can now define the Laplacian operator L where we denote the number of edges from node i to node j by $d_{i,j}$.

$$L_{ij} = \begin{cases} d_{j,i} & \text{if } i \neq j; \\ -d^+(i) + d_{i,i} & \text{if } i = j. \end{cases}$$

Example 3.2. *We take the following graph as an example:*



For this graph the Laplacian will look as follows:

$$L = \begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix}$$

Each column in our Laplacian represents what happens when we fire the node corresponding to this column.

Theorem 3.3. *Given a graph $G = (V, E)$ with initial position $B \in \mathbb{Z}^V$, if we fire node i x_i times then we find the configuration: $B + Lx$.*

So if we add a linear combination of the columns of the Laplacian, then this represents a sequence of moves in the game. The first element of vector x represents the number of times node 1 gets fired, the second element of x represents the number of times node 2 gets fired and so on. Because adding is a commutative property, we see that the order in which we fire the nodes is not relevant for the resulting position in the game, only the number of times each node is fired determines the position.
We can only fire a node a positive integer number of times, so solution x of $B + Lx$ must be a positive integer solution.

If we find an integer solution x of $B+Lx=E$with a negative component then we can transform this solution into a positive integer solution which still satisfies $B+Lx=E$ with the same E. We use the theorem of Perron-Frobrenius:

Definition 3.4. *a matrix is reducible if and only if it can be placed into block upper-triangular form by simultaneous row/column permutations.*

Theorem 3.5 ([9])**.** *Let A be an nonnegative irreducible n times n matrix ($a_{ij} \geq 0$ for $1 \leq i,j \leq n$) the following statements hold:*

1. *There is a positive real number r, such that r is an eigenvalue of A and for every other eigenvalue λ: $|\lambda| < r$*
2. *r is a simple root of the characteristic polynomial of A. Consequently, the eigenspace associated to r is one-dimensional.*
3. *There exists an eigenvector $v = (v_1, ..., v_n)$ of A for eigenvalue r such that $Av = rv$ and $v_i > 0$ for $1 \leq i \leq n$. Respectively, there exists a positive left eigenvector w for which $w^T A = rw^T$ with $w_i > 0$*
4. *There are no other positive eigenvectors except positive multiples of v, all other eigenvectors must have at least one negative or non-real component.*

Definition 3.6. *A graph is strongly connected if every node is reachable from every other node.*

Theorem 3.7. *[5] Assume G is a strongly connected graph and let L be the Laplacian of G, then the space of all $v \in \mathbb{R}^V$ such that $Lv = 0$ has a basis a non-negative vector.*

Proof. If D is the maximum out degree of G, then $L + DI$ is a nonnegative irreducible matrix, which has an all-positive left eigenvector 1 with eigenvalue D. Hence, because the Perron-Frobenius Theorem tells us that every real square matrix with positive entries has a unique largest real eigenvalue with corresponding eigenvector which has strictly positive components, D is the largest eigenvalue of $L + DI$. Again by the Perron-Frobenius theorem, the right eigensubspace of $L+DI$ belonging to D is one-dimensional an spanned by an all-positive eigenvector. But this eigensubspace is just the null space of L. Hence L has rank $n-1$ and its Null space is spanned by an all-positive vector, which after scaling can be assumed to have integer components. So $\{v|Lv = 0\}$ not only has a basis of nonnegative vectors in case G is strongly connected, it is spanned by one such vector. □

Remark 3.8. *If G is strongly connected it follows that the dimension of the* Null *space is* 1 *and that the Laplacian of G is irreducible.*

With Theorem 5 we can proof the following Theorem 6:

Theorem 3.9. *Let G be a connected graph, let L be the Laplacian of G, let $B \in \mathbb{Z}^n$ be an initial position on G and let $E \in \mathbb{Z}^n$ be a final position on G . If we find an integer solution to $B + Lx = E$ then we can always transform this solution into a positive integer solution.*

Proof. Suppose u and v are both solutions of $B + Lx = E$. Then:

$$L(u - v) = Lu - Lv = E - B - (E - B) = 0$$

Hence $u - v \in$ Null space L. It follows that $u - v = c$ for some vector c in the null space of L. If we rewrite this we get: if u and v are two solutions of $Lx = E - B$ then $u = v + c$ with $c \in$

null space of L.
On the other hand, suppose that z satisfies $Lz = E - B$ and w satisfies $Lw = 0$ then:

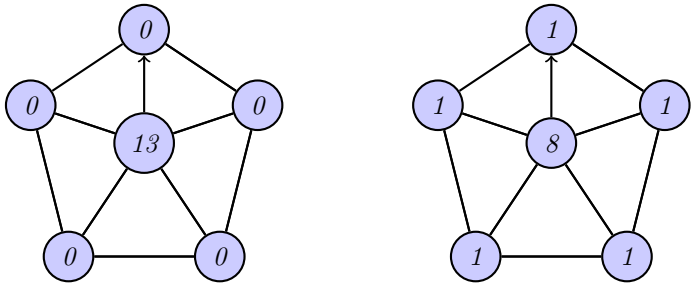
$$L(z + w) = Lz + Lw = E - B + 0 = E - B$$

It follows that if x is a solution of $Lx = E - B$ and y is a vector in the null space of L than $x + y$ forms another solution of $Lx = E - B$. Since the null space forms a linear subspace, the general solution of $Lx = E - B$ is of the form: $x + \lambda y$ with $\lambda \in \mathbb{R}$ where x is a fixed solution of $Lx = E - B$ and y is a fixed solution of $Ly = 0$. We want $x, y \in \mathbb{Q}$ so that we can easily obtain if an integer point that lies on the line $x + \lambda y$.

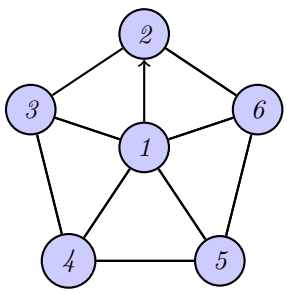
By Theorem 5 we can always transform $x + \lambda y$ to a positive solution of $B + Lx = y$ because $y > 0$. □

Example 3.10. *In this example we show how a move is the game gives a linear equation. In our initial position of the game we put 13 chips on node* 1 *and zero chips on the other nodes. If we only fire node 1 we get:*

$$\begin{bmatrix} 13 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 13 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -5 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 8 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$



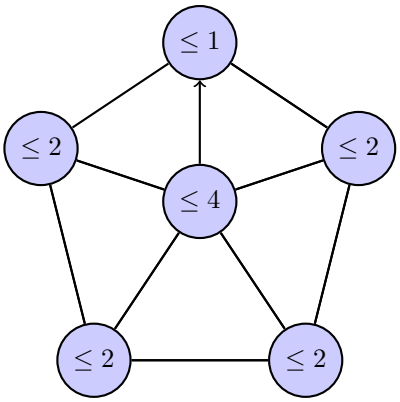
Example 3.11. *We use the Erikssons graph to illustrate the necessary condition for a finite game. This because there exists only one final position for the initial position of* $3n - 5$ *chips on the center node and because the game will always end. We also know the bound for the number of moves. Take* $n = 6$*, our Erikkson graph will look like our previous example for the Laplacian.*



We used the same graph in Example 1.1 *so our Laplacian is:*

$$L = \begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix}$$

We can easily see that there is only one possible final position. Node 1 cannot be fired if the amount of chips is less or equal to 4, node 2 cannot be fired if the amount is less of equal to 1 and for nodes 3 up to and including 6 the amount must be less of equal to 2. All the upper limits of the nodes add up to 13. We get the following final position:



We can write this final position as a vector, where the first element corresponds with node number 1 etc.

$$E = \begin{bmatrix} 4 & 1 & 2 & 2 & 2 & 2 \end{bmatrix}^T$$

We can also write the initial position as a vector:

$$B = \begin{bmatrix} 13 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

To determine if we can reach final position E given the initial position B, we solve the linear equation:

$$B + Lx = E \rightarrow Lx = E - B$$

3.3 A necessary condition for a finite game

Theorem 3.12. *Let G be a connected graph, let L be the Laplacian of G, let $B \in \mathbb{Z}^n$ be the initial position on G and let $E \in \mathbb{Z}^n$ be a final position on G. If we find a integer solution to the matrix equation $B + Lx = E$, then the game terminates.*

Proof. If we find a integer solution for $B + Lx = E$ then we have three possible scenario's:

1. We can only fire node i, which we are not allowed to fire according to x_i (i.e. with $x_i = 0$).
2. The game ends before we fired each node i x_i times.
3. We get to the final position by firing each node i x_i times.

1. Let P be the set of sequences p that are not allowed according to x. We call our chip distribution C on the moment that we cannot meet with x anymore. At this moment we can only transform distribution C by firing a node $p \in V$. So we know that:

   $$C + Lp \geq 0$$

   But then also:

   $$E + Lp \geq 0$$

   Because $E \geq 0$ and $Lp \geq 0$.Which implies that if we reach the final position we can still fire node p, but this contradicts that E is a final position.

2. The game is still finite but ends earlier than integer solution x prescribes. This means that there is another possible final position which gives a strictly smaller integer solution x. We can only reach the final position which gives the smallest integer solution x to $B+Lx = E$. If we find a negative integer solution or semi negative solution x of $B + Lx = E$ then we can transform this solution into a positive integer solution which still satisfies $B+Lx = E$ with the same E because of Theorem 5.

□

If we can't find a integer solution then the game is infinite because a *basic* word gives an integer solution. Now we can ask ourselves if a graph gives opportunity to several final positions which give us several integer solutions, can we then always find a smallest integer solution? Can it be that we find two smallest integer solutions for which one solution is not strictly greater or smaller than the other solution?

Theorem 3.13. *Let G be a connected graph and let L be the Laplacian of G. If G gives opportunity to several final positions E^k, and if we can find integer solutions x^k to the equation $B+Lx_k = E^k$ for final positions E^k, then there will always be one smallest element x^j for which $x_i^j < x_i^k$.*

Proof. If we find several integer solutions x^1, x^2 then we can only reach the smallest solution, otherwise we could add the letters from the greater solution to the smallest, which implies that the smallest solution wasn't reaching a final position. But if we find two smallest integer solutions x^3, x^4 in the partial ordered set of solutions, then we can form two scores of two words associated with this solutions x^3, x^4 $[\alpha]$ and $[\beta]$. Because they are both a smallest solution, we

can say that for some i and j: $[\alpha_i] < [\beta_i]$ and $[\alpha_j] > [\beta_j]$, than because of the strong exchange property there exists a subword β' of β such that $[\alpha\beta']$ is the entry-wise maximum of $[\alpha]$ and $[\beta]$. We can add the nodes that are fired more in β to α and vise versa. This gives us a integer solution γ with: $[\alpha], [\beta] < [\gamma]$. But this means that if we fired each node as many times as prescribe by $[\alpha]$ than we can fire the nodes that are fired more times in $[\beta]$, but this contradicts that $[\alpha]$ gives a final position, we can still fire nodes, so we can never find two integer solutions $[\alpha]$ and $[\beta]$ for which $[\alpha] \not\leq [\beta]$ and $[\alpha] \not\geq [\beta]$. □

3.4 Cramer's rule

In order to get the solution x of $B + Lx = E$ and solution y of $Ly = 0$ to form $x + \lambda y$ with $x, y \in \mathbb{Q}$, we can use Cramer's rule: an explicit manner to find a solution of a system of linear equations, with as many equations as unknowns. This method is valid whenever the system has an unique solution. Our system doesn't have a unique solution, because we have one free variable. If we set the free variable (we choose the first entry of x) equal to 1. Then we can leave out the first row and the first column of L and form matrix L'. We also leave out the first row of $E - B$. In this way we find a linear equation that has a unique solution.

Example 3.14. *We give an example of a linear system that doesn't give a unique solution. We use our Eriksson graph with $n = 6$ to do this. We try to find an integer solution of the equation $B + Lx = E$. The linear equation $B + Lx = E$ doesn't have a unique solution because we have one free variable: x_6. We can see this in the reduced matrix L in echelon form:*

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -0.750 \\ 0 & 1 & 0 & 0 & 0 & -1.375 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & -0.875 \\ 0 & 0 & 0 & 0 & 1 & -0.875 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

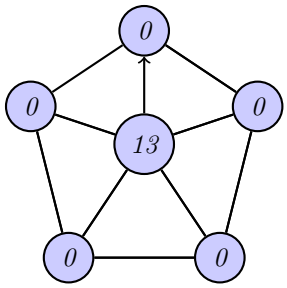
Theorem 3.15. *Cramer's rule replaces the ith column by the $E - B$ to form L_i and than calculates the ith entry of the vector x by :*

$$x_i = \frac{\det(L_i)}{\det(L')}$$

for $Ly = 0$ the i'th column is a zero column

$$y_i = \frac{\det(L_i)}{\det(L')}$$

Example 3.16. *Using our Eriksson graph with $n = 6$ again and the initial position with 13 chips on node 1:*



With maple we find the following solution of the form $x + \lambda y$:

$$\begin{bmatrix} 1 \\ -\frac{187}{66} \\ -\frac{154}{66} \\ -\frac{143}{66} \\ -\frac{143}{66} \\ -\frac{154}{66} \end{bmatrix} + \lambda \begin{bmatrix} \frac{3}{4} \\ \frac{11}{8} \\ 1 \\ \frac{7}{8} \\ \frac{7}{8} \\ 1 \end{bmatrix}$$

If we make y integer by multiplying it by 8 we get:

$$8y = \begin{bmatrix} 6 & 11 & 8 & 7 & 7 & 8 \end{bmatrix}^T$$

now we can easily see that if we choose $\lambda = 9\frac{1}{3}$, we find an integer solution:

$$x + 9\frac{1}{2}y = \begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T$$

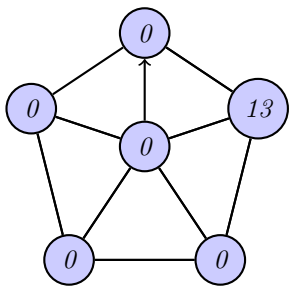
Since this is not a great number of steps we can check if can fire each node as many times as x prescribes. We use the java applet (see appendix) to check this. We see that we can fire each node as many times as x prescribes and that we terminate at the right final position. We can also check our x solution with Erikssons formula for the total number of steps:

$$s_k = \left(\frac{2n}{\sqrt{5}} + \frac{1+\sqrt{5}}{2}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{2k} + \left(\frac{-2n}{\sqrt{5}} - \frac{2}{1+\sqrt{5}}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{-2k} - 2k$$

we used the final position: 22122, so $k = 2$. We get:

$$s_3 = \left(\frac{12}{\sqrt{5}} + \frac{1+\sqrt{5}}{2}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{4} + \left(\frac{-12}{\sqrt{5}} - \frac{2}{1+\sqrt{5}}\right) \cdot \left(\frac{1+\sqrt{5}}{2}\right)^{-4} - 4 = 43$$

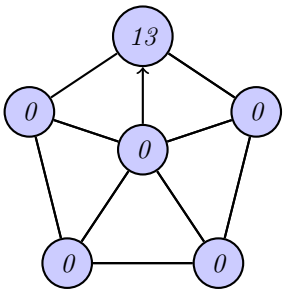
and also $7 + 10 + 7 + 6 + 6 + 7 = 43$
Now we use the same method for a different initial position:



With Cramer's rule we obtain:

$$\begin{bmatrix} 1 \\ \frac{3}{2} \\ -\frac{24}{66} \\ -\frac{39}{66} \\ 1 \\ -\frac{288}{66} \end{bmatrix} + \lambda \begin{bmatrix} \frac{3}{4} \\ \frac{4}{8} \\ 1 \\ \frac{7}{8} \\ \frac{7}{8} \\ 1 \end{bmatrix}$$

We see that $-\frac{39}{66}$ *and* 1 *in the* x *vector have the same slope in the* y *vector. We can write this as two linear lines:* $-\frac{39}{66} + \frac{7}{8}\lambda$ *and* $1 + \frac{7}{8}\lambda$*. Whatever value we choose for* λ*, both linear lines will never be on an integer point at the same time. So the initial position with 13 chips on node 6 gives an infinite game.*
We also consider the initial position where we put 13 chips on node 2:



With Cramer's rule we obtain:

$$\begin{bmatrix} 0 \\ 8 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix} + \lambda \begin{bmatrix} \frac{3}{4} \\ \frac{4}{8} \\ 1 \\ \frac{7}{8} \\ \frac{7}{8} \\ 1 \end{bmatrix}$$

We see that if we choose $\lambda = 0$*, we find a integer solution. This game is also very short, so we can check if the game really ends. Here we also find the same solution as we found with Cramer's rule.*

3.5 Hermite normal form

It is not straight forward to decide whether an integer solution exists using the previous method. We were lucky to find arguments such as: two lines will never be on an integer point at the same time, but we can wonder if we will easily find such an argument with every general graph. Another way is to determine the Hermite form of the Laplacian [11]. The Hermite form of a matrix is an analogue of the reduced echelon form of a integer matrix. It's goal is to preserve the integral solutions. A Hermite form can be row-style or column-style. Here we will use the column-style.

Definition 3.17. *We say that a matrix with integer entries is in column Hermite normal form if:*

1. *All columns with only zero entries are at the right of the matrix.*
2. *The pivot, the first nonzero entry from the top, is always on the diagonal*
3. *All entries in a row to the left of a pivot are nonnegative and strictly smaller than the pivot*
4. *All entries in a row to the right of a leading entry are zeroes (implied by the first two criteria)*

The elementary operations we use to get a integer matrix in column Hermite normal form are:

1. We can exchange columns
2. We can add an integral multiple of one column to another
3. we can multiply a column by -1

For the calculation of an integer solution of the equation $Lx = B - E$ we use some properties of the Hermite matrix:

Theorem 3.18 ([11])**.** *Each rational matrix of full row rank can be brought into Hermite normal form by a series of elementary column operations.*

Theorem 3.19 ([11])**.** *Every rational matrix of full row rank has a unique Hermite normal form.*

Definition 3.20. *We say that a matrix A is nonsingular if there exists a inverse matrix A^{-1} such that $AA^{-1} = I$.*

Theorem 3.21 ([11])**.** *For each rational matrix A of full row rank there is a unimodular matrix U such that AU is het Hermite normal for of A. If A is nonsingular, U is unique.*

Now we can ask ourselves the following question: does the system $Ax = b$ have an integer solution? We can answer this question by using the Hermite normal form.

Theorem 3.22. *Given a rational matrix A with full row rank and a rational vector b, we can find a integer solutionx for $Ax = b$ if and only if there is an integer solution y of $Hy = b$. In that case, $x = Uy$ is an integer solution to $Ax = b$.*

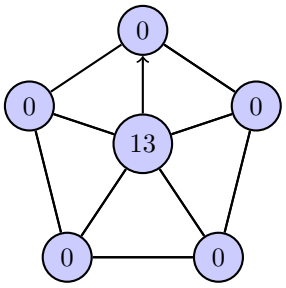
Chapter 4

Results

We use Theorem 12 to calculate an integer solution for the Eriksson graph that we used in our previous example.

4.1 Initial position with 13 chips

We have the following initial position:



We want our matrix L to have full row rank. This means that there does not exist a row which can be formed by a linear combination of other rows. L has one free variable, so we have to delete the last row. This process does not change the set of solutions of $Lx = b$ with $b = E - B$. By deleting the last row of L we form the matrix A. With Maple we find the following Hermite normal form H for matrix A:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix}$$

then we set up the following equation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 13 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \end{bmatrix}$$

We set $y_6 = 1$:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix}$$

With maple we can also find our unimodular matrix U:

$$\begin{bmatrix} -22 & -21 & -5 & -9 & -37 & 6 \\ -39 & -38 & -9 & -16 & -66 & 11 \\ -29 & -28 & -7 & -12 & -49 & 8 \\ -26 & -25 & -6 & -11 & -44 & 7 \\ -27 & -26 & -6 & -11 & -46 & 7 \\ -27 & -26 & -6 & -11 & -46 & 8 \end{bmatrix}$$

U captures every operation done to create H out of L. So We have to set the solution y that we found with our above equation back to the solution for $Lx = b$. We solve the following equation: $A(Uy) = b$ Where $Uy = x$.

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \left(\begin{bmatrix} -22 & -21 & -5 & -9 & -37 & 6 \\ -39 & -38 & -9 & -16 & -66 & 11 \\ -29 & -28 & -7 & -12 & -49 & 8 \\ -26 & -25 & -6 & -11 & -44 & 7 \\ -27 & -26 & -6 & -11 & -46 & 7 \\ -27 & -26 & -6 & -11 & -46 & 8 \end{bmatrix} \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix} \right) =$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \begin{bmatrix} 7 \\ 10 \\ 7 \\ 6 \\ 6 \\ 7 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

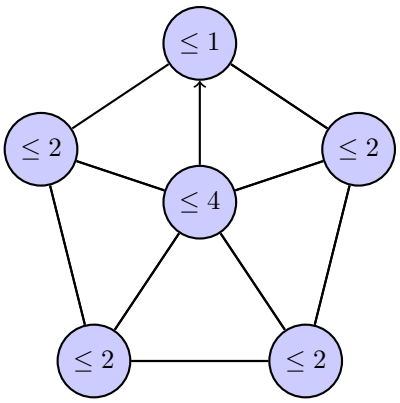
So

$$E = \begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T$$

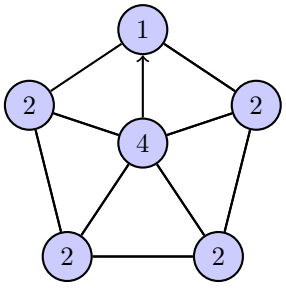
Is our integer solution, this is the same solution as we saw before for the equation $Lx = E - B$ with the specific initial position.

4.2 Initial position with 12 chips

We now analyze the case in which we have only 12 chips on our graph. We still have the same upper limits for a final position:



We have 6 possible nodes where we can remove one chip compared with the final position in the case of 13 chips:



So the possible final positions are:

$$\begin{bmatrix} 3 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

We will check if each final position can be reached. We assume that our initial position is as follows:

$$E = \begin{bmatrix} 12 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

1.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 12 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \end{bmatrix}$$

We set $y_6 = 1$:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix}$$

The first final position gives us the same $E - B$ vector as the final position in the case of 13 chips. Therefore we find the same integer solution:

$$E = \begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T$$

2.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 2 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 12 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 4 \end{bmatrix}$$

We set $y_6 = 1$:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \left(\begin{bmatrix} -22 & -21 & -5 & -9 & -37 & 6 \\ -39 & -38 & -9 & -16 & -66 & 11 \\ -29 & -28 & -7 & -12 & -49 & 8 \\ -26 & -25 & -6 & -11 & -44 & 7 \\ -27 & -26 & -6 & -11 & -46 & 7 \\ -27 & -26 & -6 & -11 & -46 & 8 \end{bmatrix} \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix} \right) =$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \begin{bmatrix} 6 \\ 9 \\ 6 \\ 5 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

This is the right final position, so the integer solution to the problem is:

$$E = \begin{bmatrix} 6 & 9 & 6 & 5 & 5 & 6 \end{bmatrix}^T$$

3.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 12 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -8 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix} \rightarrow \text{ no integer solution}$$

So this final position cannot be reached.

We repeat our calculation with the rest of the final positions and we find that non of them gives an integer solution. So if we use the initial position of 12 chips on node 1 we have the following integer integer solutions:

$\begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 6 \end{bmatrix}^T$ matching with final position $E = \begin{bmatrix} 3 & 1 & 2 & 2 & 2 & 2 \end{bmatrix}^T$ and
$\begin{bmatrix} 6 & 9 & 6 & 5 & 5 & 6 \end{bmatrix}^T$ matching with final position $E = \begin{bmatrix} 4 & 0 & 2 & 2 & 2 & 2 \end{bmatrix}^T$

We see that the first integer solution is strictly greater than the second integer solution:

$\begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T > \begin{bmatrix} 6 & 9 & 6 & 5 & 5 & 6 \end{bmatrix}^T$

If we have word $\alpha \in L$ and word $\beta \in L$, with L a left-hereditary permutable locally free language, and if $[\alpha] < [\beta]$, then we can expand α with the nodes who are fired more in β than in α. So We can expand our solution $\begin{bmatrix} 6 & 9 & 6 & 5 & 5 & 6 \end{bmatrix}^T$ to $\begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T$ But $\begin{bmatrix} 6 & 9 & 6 & 5 & 5 & 6 \end{bmatrix}^T$ already gave us a final position, which means that we cannot fire any more nodes, we can therefore never reach the final position $\begin{bmatrix} 3 & 1 & 2 & 2 & 2 & 2 \end{bmatrix}^T$ which matches with integer solution $\begin{bmatrix} 7 & 10 & 7 & 6 & 6 & 7 \end{bmatrix}^T$ Our conclusion here is that if we have a initial position with 12 chips on node 1, we can only end with the following final position: $\begin{bmatrix} 4 & 0 & 2 & 2 & 2 & 2 \end{bmatrix}^T$

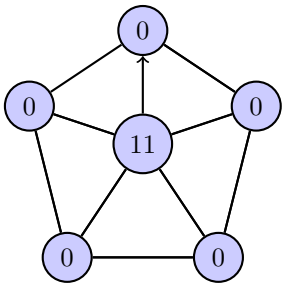
We can see from our Hermite form of L and the first equation towards finding a integer solution for $Lx = E - B$:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \text{ final position } \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

that there is a small chance that we will find a integer solution y. We can chose y_1, y_2, y_3 and y_4 exactly the same as the final position. It follows that $b_5 = 6b_1 + 6b_2 + b_3 + 2b_4 + 11y_5$. So in order to find a integer solution, the final position must meet this specific equation.

4.3 Initial position with 11 chips

We can now ask ourselves if there will be more possible final positions than one that can be reached during the game. We will check this with an example which gives us a lot more possible final positions: we leave out one more chip in our initial position



We have 6 possible nodes where we can leave out one chip compared to the upper bounds. After we chose one node we have again 6 nodes to leave out one more chip. Except for the case in which we chose remove one more chip from node 2. Node 2 has a upper bound of 1, so we can leave out 1 chip here, but not two or more. We first chose 1 node out of 6 nodes, then again we chose 1 node out of 6 nodes except for the case where we chose node 2 the first time, than we chose 1 node out of 5 nodes. So we have $6 \cdot 6 = 36$ possible choices, but we have to disregard the order. We chose 2 chips so we have to divide our possibilities by 2: $\frac{36}{2} = 18$.Then however, we also assumed that the possible final positions where we chose the same node in the second round as in the first round, were counted twice in 36 which is not the case. There are 6 possibilities where we choose 2 times the same node, so we have to add 3: $18+3 = 21$ and we have to remove the possibility to chose node 2 two times, so we have to subtract 1: $21 - 1 = 20$. We end with 20 possible final positions:

$$\begin{bmatrix} 2 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 0 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \\ 2 \\ 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \\ 2 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 1 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \\ 2 \\ 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \\ 2 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 0 \\ 2 \\ 2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 0 \\ 2 \\ 2 \\ 2 \end{bmatrix},$$

$$\begin{bmatrix} 4 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 1 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 0 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 1 \\ 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 4 \\ 1 \\ 2 \\ 2 \\ 2 \\ 0 \end{bmatrix}$$

We check the Hermite equation for every final position, here we present each final position for which we found an integer solution:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}$$

1.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 11 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \end{bmatrix}$$

Again we take $y_6 = 1$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -9 \\ 1 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix}$$

and again we find the integer solution

$$\begin{bmatrix} 7 \\ 10 \\ 7 \\ 6 \\ 6 \\ 7 \end{bmatrix}$$

2.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 2 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 11 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 4 \end{bmatrix}$$

Again we take $y_6 = 1$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -8 \\ 0 \\ 2 \\ 2 \\ 4 \\ 1 \end{bmatrix}$$

and again we find the integer solution:

$$\begin{bmatrix} 6 \\ 9 \\ 6 \\ 5 \\ 5 \\ 6 \end{bmatrix}$$

3.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix} - \begin{bmatrix} 11 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix}$$

Again we take $y_6 = 1$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \left(\begin{bmatrix} -22 & -21 & -5 & -9 & -37 & 6 \\ -39 & -38 & -9 & -16 & -66 & 11 \\ -29 & -28 & -7 & -12 & -49 & 8 \\ -26 & -25 & -6 & -11 & -44 & 7 \\ -27 & -26 & -6 & -11 & -46 & 7 \\ -27 & -26 & -6 & -11 & -46 & 8 \end{bmatrix} \begin{bmatrix} -7 \\ 1 \\ 1 \\ 2 \\ 3 \\ 1 \end{bmatrix} \right) =$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \\ 5 \\ 4 \\ 4 \\ 5 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}$$

We find integer solution:

$$\begin{bmatrix} 5 \\ 7 \\ 5 \\ 4 \\ 4 \\ 5 \end{bmatrix}$$

4.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 2 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 11 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 6 & 1 & 2 & 11 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 2 \\ 1 \\ 1 \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 2 \\ 1 \\ 3 \\ 1 \end{bmatrix}$$

Again we take $y_6 = 1$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 2 \\ 1 \\ 3 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \left(\begin{bmatrix} -22 & -21 & -5 & -9 & -37 & 6 \\ -39 & -38 & -9 & -16 & -66 & 11 \\ -29 & -28 & -7 & -12 & -49 & 8 \\ -26 & -25 & -6 & -11 & -44 & 7 \\ -27 & -26 & -6 & -11 & -46 & 7 \\ -27 & -26 & -6 & -11 & -46 & 8 \end{bmatrix} \begin{bmatrix} -7 \\ 1 \\ 2 \\ 1 \\ 3 \\ 1 \end{bmatrix} \right) =$$

$$\begin{bmatrix} -5 & 0 & 1 & 1 & 1 & 1 \\ 1 & -2 & 1 & 0 & 0 & 1 \\ 1 & 1 & -3 & 1 & 0 & 0 \\ 1 & 0 & 1 & -3 & 1 & 0 \\ 1 & 0 & 0 & 1 & -3 & 1 \\ 1 & 1 & 0 & 0 & 1 & -3 \end{bmatrix} \begin{bmatrix} 9 \\ 14 \\ 10 \\ 9 \\ 9 \\ 10 \end{bmatrix} = \begin{bmatrix} -7 \\ 1 \\ 2 \\ 1 \\ 1 \\ 2 \end{bmatrix}$$

We find integer solution:

$$\begin{bmatrix} 9 \\ 14 \\ 10 \\ 9 \\ 9 \\ 10 \end{bmatrix}$$

So we found integer solutions:

$$\begin{bmatrix} 7 \\ 10 \\ 7 \\ 6 \\ 6 \\ 7 \end{bmatrix}, \begin{bmatrix} 6 \\ 9 \\ 6 \\ 5 \\ 5 \\ 6 \end{bmatrix}, \begin{bmatrix} 5 \\ 7 \\ 5 \\ 4 \\ 4 \\ 5 \end{bmatrix}, \begin{bmatrix} 9 \\ 14 \\ 10 \\ 9 \\ 9 \\ 10 \end{bmatrix}$$

Now we check if we cannot subtract the solution of $Ly = 0$:

$$\begin{bmatrix} 6 & 11 & 8 & 7 & 7 & 8 \end{bmatrix}^T$$

We can subtract y from our fourth solution:

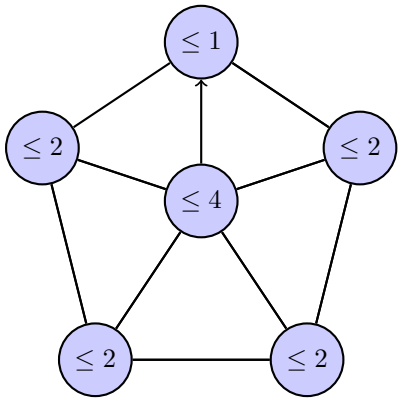
$$\begin{bmatrix} 9 \\ 14 \\ 10 \\ 9 \\ 9 \\ 10 \end{bmatrix} - \begin{bmatrix} 6 \\ 11 \\ 8 \\ 7 \\ 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \\ 2 \\ 2 \\ 2 \\ 8 \end{bmatrix}$$

We see that this is the smallest solution and therefore the only solution that gives a final position. So the 20 possible final positions, gave only one final position that can actually be reached:

$$\begin{bmatrix} 4 & 1 & 2 & 1 & 1 & 2 \end{bmatrix}^T$$

4.4 Initial position with 14 chips

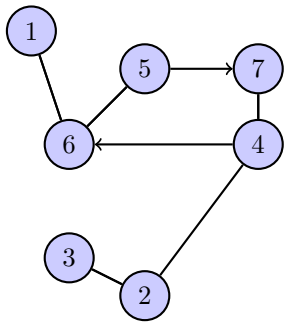
If we begin with a initial position with 14 chips, we can easily obtain that the game will never end. We have the upper bounds on each node for a final game with 13 as the total number of chips on the graph:



If we would add one chip, there will always be a node that has a number of chips which is higher than this upper bounds, this implies that the game will be infinite in this case.

4.5 Another example graph

For the Erkisson graph with $n = 6$ and 11 chips on the center node we found 4 solutions which formed a nice chain. We know that there must always be a smallest integer solution but can we find a graph example where we get two or more solutions y and z, which are not the smallest solutions, but for which: $y \not\leq z$ and $y \not\geq z$. We chose an example for which the Hermite matrix has a very nice form:

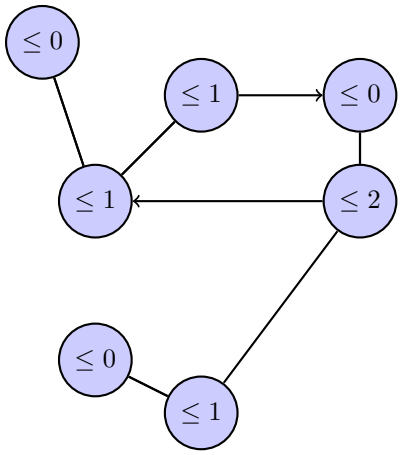


$$L = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -3 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -2 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & -2 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & -1 \end{bmatrix}$$

$$Ly = 0 \rightarrow y = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We place 3 chips on node 1, our upper bounds are as follows:



We have the following final positions:

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

We get the following corresponding integer solutions:

$$\begin{bmatrix} 6 \\ 0 \\ 0 \\ 0 \\ 1 \\ 3 \\ 1 \end{bmatrix}, \begin{bmatrix} 7 \\ 0 \\ 0 \\ 0 \\ 2 \\ 4 \\ 2 \end{bmatrix}, \begin{bmatrix} 6 \\ 1 \\ 0 \\ 2 \\ 1 \\ 4 \\ 9 \end{bmatrix}, \begin{bmatrix} 8 \\ 0 \\ 0 \\ 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}, \begin{bmatrix} 7 \\ 1 \\ 0 \\ 3 \\ 2 \\ 5 \\ 11 \end{bmatrix}, \begin{bmatrix} 10 \\ 0 \\ 0 \\ 1 \\ 3 \\ 7 \\ 4 \end{bmatrix}, \begin{bmatrix} 9 \\ 1 \\ 0 \\ 3 \\ 3 \\ 7 \\ 12 \end{bmatrix}$$

We see that we get an integer solution for every final position, this because the Hermite matrix is very simple. We also see that for instance

$$\begin{bmatrix} 7 & 1 & 0 & 3 & 2 & 5 & 11 \end{bmatrix}^T \not\leq \begin{bmatrix} 10 & 0 & 0 & 1 & 3 & 7 & 4 \end{bmatrix}^T$$

And

$$\begin{bmatrix} 7 & 1 & 0 & 3 & 2 & 5 & 11 \end{bmatrix}^T \not\geq \begin{bmatrix} 10 & 0 & 0 & 1 & 3 & 7 & 4 \end{bmatrix}^T$$

4.6 Eriksson graph for each number of nodes

I made a maple program(see appendix B) which calculates the integer solution for our specific Erkisson graph with an initial position of $3n-5$ chips on the center node. The program gives the

integer solution for each n with the hope of finding a pattern in the solutions. For $n = 1, 2, ..., 20$ we find:

$$\begin{bmatrix} 3 \\ 3 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 7 \\ 10 \\ 7 \\ 6 \\ 6 \\ 7 \end{bmatrix}, \begin{bmatrix} 17 \\ 28 \\ 20 \\ 17 \\ 16 \\ 16 \\ 17 \\ 20 \end{bmatrix}, \begin{bmatrix} 43 \\ 75 \\ 54 \\ 46 \\ 43 \\ 42 \\ 42 \\ 43 \\ 46 \\ 54 \end{bmatrix}, \begin{bmatrix} 111 \\ 198 \\ 143 \\ 122 \\ 114 \\ 111 \\ 110 \\ 110 \\ 111 \\ 114 \\ 122 \\ 143 \end{bmatrix}, \begin{bmatrix} 289 \\ 520 \\ 376 \\ 321 \\ 300 \\ 292 \\ 289 \\ 288 \\ 288 \\ 289 \\ 292 \\ 300 \\ 321 \\ 376 \end{bmatrix}, \begin{bmatrix} 755 \\ 1363 \\ 986 \\ 842 \\ 787 \\ 766 \\ 758 \\ 755 \\ 754 \\ 754 \\ 755 \\ 758 \\ 766 \\ 787 \\ 842 \\ 986 \end{bmatrix}, \begin{bmatrix} 1975 \\ 3570 \\ 2583 \\ 2206 \\ 2062 \\ 2007 \\ 1986 \\ 1978 \\ 1975 \\ 1974 \\ 1974 \\ 1975 \\ 1978 \\ 1986 \\ 2007 \\ 2062 \\ 2206 \\ 2583 \end{bmatrix}, \begin{bmatrix} 5169 \\ 9348 \\ 6764 \\ 5777 \\ 5400 \\ 5256 \\ 5201 \\ 5180 \\ 5172 \\ 5169 \\ 5168 \\ 5168 \\ 5169 \\ 5172 \\ 5180 \\ 5201 \\ 5256 \\ 5400 \\ 5777 \\ 6764 \end{bmatrix}$$

The top node and the center node are fired different times but we do see symmetry on the circuit nodes. We see some interesting pattern of two equal numbers in entry $n/2+2$ and $n/2+3$ followed up by adding 1 for the following entry, 3 for the one after, and we go on with adding, we get the sequence: $1, 3, 8, 21, 55, 144, 377, 987...$ we can conclude this paper with the question if we can give a formula for this sequence and a reason for this behavior? Is this behavior traseable from our graph? And can we use this to calculate the integer solutions faster?

Chapter 5

Conclusion

We found several ways to calculate the finiteness of the chip firing game, given an initial position and a final position. With only the initial position one can just play the game and see where it goes. When we also use the possible final position for the graph we can set up the laplacian and use crammer's rule or we use the Hermite normal form. The difficulty of the last two described methods is that we have to know all the final positions and implement all these positions in the equation to find the final position for our given initial position. In further research about the chip firing game one can investigate if there is a more efficient way or one can build a algorithm were al the possible final positions are calculated and plugged in the Hermite equation.
Another method would be to extend the Java applet. One can build in another function which plays the game for you. You do not have to click on the nodes to fire them, Java will do this for you until the is finished. The best way to do this is to combine it with an algorithm which calculates the final position reachable from your chosen initial position and which calculates the number of steps needed to finish the game. This would be the most efficient way.

another interesting further investigation would be to find a reason for or meaning of the pattern in the solutions of the Eriksson graph for each n.

I worked with great pleasure on this research. I enjoyed working with my supervisor on a challenging project which goes way beyond most peoples interest with it's far-fetched applications and use for society. Although reality often seemed far away, I managed to get more applied by making an applet for the game. I was happy to use my first years subject 'Inleiding programmeren' for a great goal after all. I regret to stop the research with so many ways for more investigation, but I'm happy to see that mathematics is in some way also a infinite game, which will always give us an intellectual challenge and which will always let us continue to play.

Bibliography

[1] Spencer, J. (1976). Balancing Games *Journal of combinatorial theory*, Series B 23, 68-74

[2] Doer, B. (2000). Vector Balancing Games with Aging. *Mathematisches Seminar 2*

[3] Anderson, R. Lovász , L. Shor, P. Spencer, J. Tardos, E. & Winograd, S. (1989) Disks, balls, and walls: analysis of a combinatorial game. *Amer. Math. Monthly, 96(6)* 481-493.

[4] Biggs, N. (1999) Chip firing and the critical group of a graph. *J. Algebraic Combin, 9(1))* 25-45.

[5] Björner, A. & Lovász, L. (1992). Chip-Firing Games on Directed Graphs. *Journal of Algebraic Combinatorics, 1*, 305-328.

[6] Merino, C. (2001) The Chip Firing Game and Matroid Complexes. *Discrete Mathematics and Theoretical Computer Science Proceedings*, 245-256

[7] Björner, A. Lovász, L. & Shor, P. Chip-firing games on graphs. (1991) *European Journal of Combinatorics, 12.4*, 283-291

[8] Eriksson, K. (1991). No polynomial bound for the chip firing gem on directed graphs. *Proceedings of the American Mathematical Society, 112* (4).

[9] Minc, H. (1988) Nonnegative matrices, *J.Wiley & Sons*

[10] Kannan, R. & Bachem A. (1997). Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *Society for Industrial and Applied Mathematics, 8* (4)

[11] Schrijver, A. (1986) Theory of linear and integer programming. John Wiley & Sons.

Appendix A

Java code

The game works as follows, you can click on the screen to place a node. When you have at least placed two nodes you can click in one node and then click in a other node which gives you an edge between this two nodes. When you press the right mouse button you can place one chip on a node by clicking in the node, if click two times in a node you get two chips in the node ect. You can continue with placing chips on node unitl you press the right mouse button again. When you pressed the right mouse button for the second time you can fire nodes.

Java class Punt

```
package bep;

import java.awt.Graphics;
import java.util.ArrayList;
import java.util.List;

import javax.swing.JComponent;
import javax.swing.JTextArea;

public class Punt extends JComponent{
   int x;
   int y;
   int c;
   List<Punt> connecties;

   public Punt(int x, int y) {
      super();
      this.x = x;
      this.y = y;
      this.c = 0;
      this.connecties = new ArrayList<Punt>();
   }

   public void setC(int c) {
      this.c = c;
   }

   public int getC() {
      return c;
   }

   public int getX() {
```

```
      return x;
   }

   public int getY() {
      return y;
   }

   public List<Punt> getConnecties() {
      return connecties;
   }

   public void addConnectie(Punt connectie) {
      connecties.add(connectie);
   }
}
```

Java class rondje

```
package bep;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.RenderingHints;
import java.awt.event.MouseEvent;
import java.awt.geom.AffineTransform;
import java.awt.geom.Ellipse2D;
import java.util.ArrayList;
import java.util.List;

import javax.swing.JFrame;
import javax.swing.JPanel;

@SuppressWarnings("serial")
public class Rondje extends JPanel {
   List<Punt> cirkels = new ArrayList<Punt>();
   Punt selected1 = null;
   Punt selected2 = null;
   int Status = 1;

   public Rondje() {
      addMouseListener(new java.awt.event.MouseAdapter() {
          public void mouseClicked(java.awt.event.MouseEvent evt) {
            if(evt.getButton() == MouseEvent.BUTTON3) {
               Status = Status + 1;
            }

            if(Status == 1) {
               int x = evt.getX();
               int y = evt.getY();
               for(Punt p : cirkels) {
                  if(x > p.x - 15 && x < p.x + 15 && y > p.y -15 && y < p.y +15){
                     if(selected1 == null) {
                        selected1 = p;
                     } else {
                        selected2 = p;
```

```
                }
                break;
            }
        }
        System.out.println(evt.getX());
        if(selected1 == null) {
            tekenCirkel(x, y);
        } else {
            if(selected2 != null) {
                if(selected1.connecties.contains(selected2)) {
                    selected1.connecties.remove(selected2);
                } else {
                    selected1.addConnectie(selected2);
                    if(!selected2.connecties.contains(selected1)) {
                        selected2.addConnectie(selected1);
                    }
                }
                selected1 = null;
                selected2 = null;
            }
        }
    } else if(Status == 2) {
        int x = evt.getX();
        int y = evt.getY();
        for(Punt p : cirkels) {
            if(x > p.x - 15 && x < p.x + 15 && y > p.y -15 && y < p.y +15){
                p.c = p.c +1;
                break;
            }
        }
    } else if (Status == 3){
        int x = evt.getX();
        int y = evt.getY();
        for(Punt p : cirkels) {
            if(x > p.x - 15 && x < p.x + 15 && y > p.y -15 && y < p.y +15){
                if(p.c >= p.connecties.size()) {
                    p.c = p.c - p.connecties.size();
                    for(Punt connectie : p.connecties) {
                        connectie.c = connectie.c + 1;
                    }
                }
                break;
            }
        }
    }

    repaint();

  }
 });
}

public void tekenCirkel(int x, int y) {
 cirkels.add(new Punt(x, y));
 this.repaint();
}
```

```
   @Override
   public void paint(Graphics g) {
      super.paint(g);
      Graphics2D g2d = (Graphics2D) g;
      g2d.setColor(Color.RED);

      System.out.println("paint");
      for(Punt p : cirkels) {
            g2d.drawOval(p.x -15, p.y -15, 30, 30);
            g2d.drawString(p.c + "", p.x, p.y);
            System.out.println("cirkel" + p.x);
            for(Punt other : p.connecties) {
               drawArrow(g2d, p.x, p.y, other.x, other.y);
            }
      }
   }

   void drawArrow(Graphics g1, int x1, int y1, int x2, int y2) {
        Graphics2D g = (Graphics2D) g1.create();

        final int ARR_SIZE = 4;

        double dx = x2 - x1, dy = y2 - y1;
        double angle = Math.atan2(dy, dx);
        int len = (int) Math.sqrt(dx*dx + dy*dy) - 20;
        AffineTransform at = AffineTransform.getTranslateInstance(x1, y1);
        at.concatenate(AffineTransform.getRotateInstance(angle));
        g.transform(at);

        // Draw horizontal arrow starting in (0, 0)
        g.drawLine(0, 0, len, 0);
        g.fillPolygon(new int[] {len, len-ARR_SIZE, len-ARR_SIZE, len},
                      new int[] {0, -ARR_SIZE, ARR_SIZE, 0}, 4);
    }
}
```

Appendix B

Maple code

```
for i from 4 by 2 to 30 do
K :=Matrix(i,i);
K(1..i) := 1;
K(1,1):= -1*i+1;
K(2,2) := -2;
K(3,2) := 1;
K(i,2):= 1;
K(2,i) :=1;
J := Vector[column](i,2);
J(1):=i-2-(3*i-5);
J(2) := 1;
J(3..i) :=2;
 for j from 3 to i do
 K(j,j):= -3;
 K(j-1,j) := 1;
 K(j,j-1) :=1;
 K(1,j) :=1;
 end do:
 l := Vector[column](i,0);
 s:=Vector[column](i,1);
 nulvec := LinearSolve(K,l, free='s');
 subs(s=Vector[column](i,1), nulvec);
 Noemer:= map(denom, nulvec);
 c:= ilcm(op(convert(Noemer,list)));
 nulvec :=c*nulvec;
 K := DeleteRow(K,i);
 H,U := HermiteForm(Transpose(K), output = ['H', 'U']);
 P:=Transpose(H);
 P:=DeleteColumn(P,i);
 Q:=Transpose(U);
 J:=DeleteRow(J,i);
 LinearSolve(P,J);
 k:=Vector[column](i);
 k(1..i-1) := LinearSolve(P,J);
 k(i) := 1;
 intopl := Multiply(Q,k);
 for b from 1 to i do
 while intopl(b) <= 0
 do intopl:= intopl+nulvec;
 end do:
```

```
end do:
mm:=infinity;
for b from 1 to i do
mm := min(mm,floor((intopl(b))/(nulvec(b))));
end do:
intopl := intopl - mm*nulvec;
print(intopl);
end do:
```