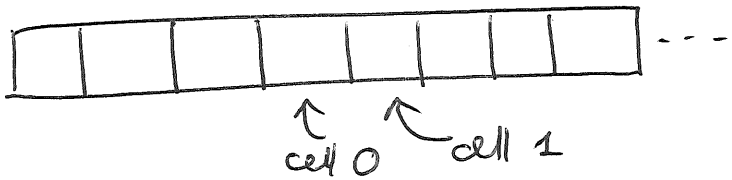
# 7.3 Chip-firing (aka sandpiles)

We start with the simplest situation.

**Def.** Let $V = \mathbb{Z}$.

Regard $V$ as an infinite row: 


A *lhe config.* is a map $f: V \to \mathbb{N}$ such that only finitely many $v \in V$ satisfy $f(v) \neq 0$.

We think of a lhe config. $f$ as putting $f(v)$ chips (or grains of sand, or ants) into cell $v$ of the infinite row.

If $f$ is a lhe config., then $|f| := \sum_{v \in V} f(v)$ is the # of all chips in $f$.

Play the following *game*: Start with any lhe config. In a *move*, you take two chips out of a cell (assuming that this cell has $\geq 2$ chips), and move them to its 2 neighboring cells (1 to the left & 1 to the right).

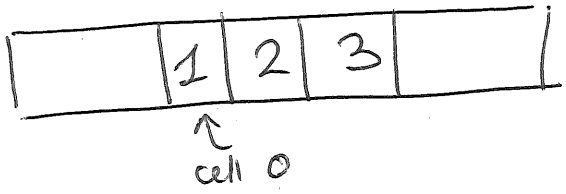
Rigorously: In a move, you pick a $v \in V$ such that $f(v) \geq 2$ (where $f$ is the current line config.), and replace $f(v-1), f(v), f(v+1)$ by $f(v-1)+1, f(v)-2, f(v+1)+1$.

This is called firing $v$.

We will write a line config. $f$ as follows:

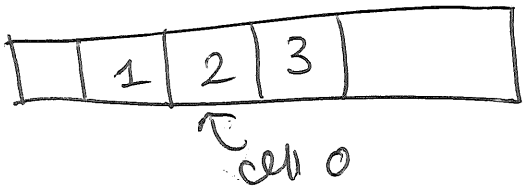
$$f(-a) \quad f(-a+1) \quad \cdots \quad \underline{f(0)} \quad f(1) \quad \cdots \quad f(b-1) \quad f(b)$$

where $-a$ is the leftmost occupied cell, and $b$ is the rightmost occupied cell. We underline $f(0)$ since we want to distinguish



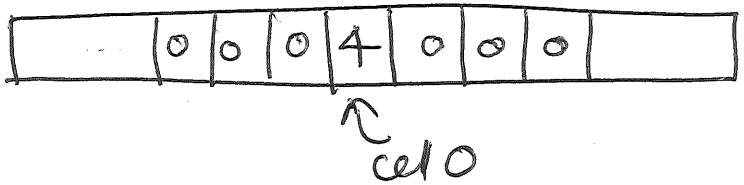

$(\rightsquigarrow \underline{1}23)$

from




$(\rightsquigarrow 1\underline{2}3)$.

Examples: (a) start with the configuration $\underline{4}$.

This means ... 

...

The game can be played as follows:

$$\underline{4} \xrightarrow[\text{fire } 0]{} 1\,\underline{2}\,1 \xrightarrow[\text{fire } 0]{} 2\,\underline{0}\,2$$

$$\xrightarrow[\text{fire } -1]{} 1\,0\,\underline{1}\,2 \xrightarrow[\text{fire } 1]{} 1\,0\,\underline{2}\,0\,1$$

$$\xrightarrow[\text{fire } 0]{} 1\,1\,\underline{0}\,1\,1.$$

No more moves are possible.

(b) Start with the configuration $\underline{6}$.

$$\underline{6} \xrightarrow[\text{fire } 0]{} 1\,\underline{4}\,1 \xrightarrow[\text{fire } 0]{} 2\,\underline{2}\,2 \xrightarrow[\text{fire } -1]{} 1\,0\,\underline{3}\,2$$

$$\xrightarrow[\text{fire } 0]{} 1\,1\,\underline{1}\,3 \xrightarrow[\text{fire } 1]{} 1\,1\,\underline{2}\,1\,1 \xrightarrow[\text{fire } 0]{} 1\,2\,\underline{0}\,2\,1$$

$\xrightarrow[\text{fire } -1]{}$ 2 0 1 2 1 $\xrightarrow[\text{fire } -2]{}$ 1 0 1 1 2 1

$\xrightarrow[\text{fire } 1]{}$ 1 0 1 2 0 2 $\xrightarrow[\text{fire } 0]{}$ 1 0 2 0 1 2

$\xrightarrow[\text{fire } 2]{}$ 1 0 2 0 2 0 1 $\xrightarrow[\text{fire } -1]{}$ 1 1 0 1 2 0 1

$\xrightarrow[\text{fire } 1]{}$ 1 1 0 2 0 1 1 $\xrightarrow[\text{fire } 0]{}$ 1 1 1 0 1 1 1 .

**Prop. 7.5.** Let $f$ be the ~~starting~~ line config. at the start of the game. Let $h = |f| =$ # of chips in $f$.

(a) There will be exactly $h$ chips after each move.

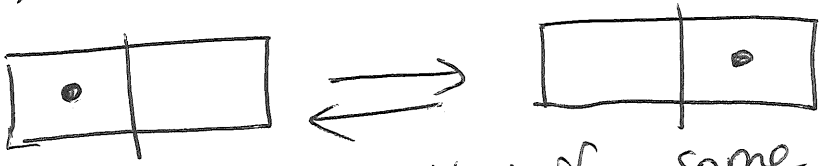
(b) No chips ever leave the interval $[-2h-a, 2h+b)$, where $-a$ is the leftmost occupied cell in $f$, and $b$ is the rightmost occupied cell in $f$.

~~Proof: (a)~~

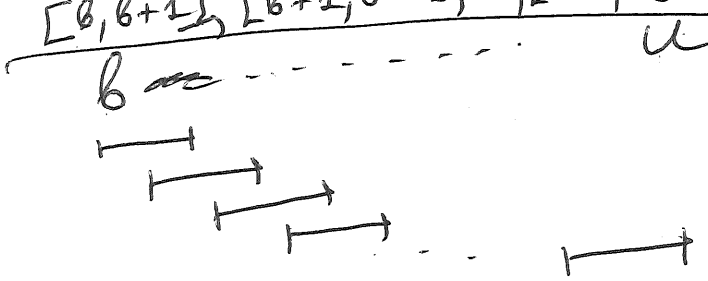
Proof. (a) Each move preserves the total # of chips.

(b) We say that an interval $[q, q+1)$ of size 1 is alive in a config. $f$ if $f(q) + f(q+1) > 0$.

If an interval $[q, q+1)$ is alive at some point in the game, then it must stay alive forever.



Thus, it is easy to see that if some cell $u \geq b$ has 2 chip at some point in the game, then all the $u-b$ intervals $[b, b+1], [b+1, b+2], \ldots, [u-1, u]$ must be alive:




So there must be $\geq \frac{u-b}{2}$ chips.

Thus, $h \geq \frac{u-b}{2}$, so $u - b \leq 2h$, so $u \leq 2h+b$. So we have shown that no chip ever moves to a cell $> 2h+b$.

Similarly, no chip ever moves to a cell $< -2h-a$.

$\Rightarrow$ (b) holds. $\square$

**Prop. 7.6.** Let $f$ be the line config. at the start of the game. Let $h = |f|$. Let $a, b$ be as in Prop. 7.5(b). Then, the game will end (i.e., no more moves will be possible) after $\leq \binom{5h+a+b+1}{h}$ steps.

**Proof.** Prop. 7.5 yields that each config. obtained during the game must have exactly $h$ chips and must be concentrated in the interval $[-2h-a, 2h+b]$.

But there are exactly $\binom{5h+a+b+1}{h}$ such configurations (by the U→L part of the 12fold way).

Thus, after $\binom{5h+a+b+1}{h}$ steps, the game will either have ended, or have run into a cycle (i.e., some config. will have appeared twice in the game). But the latter is impossible, because each move increases the entropy of the configuration. (The entropy of a config. $f$ is defined as

$$\sum_{v \in V} v^2 \cdot f(v).$$

When I fire $w$, this increases by $-2w^2 + (w+1)^2 + (w-1)^2 = 2$.) □

Def. The result of the game is the line config. obtained when no more moves are possible (i.e., when the game has ended). Note that this config. has no 2 chips in the same cell.

**Prop. 7.7.** The result of the game does not depend on the choice of moves.

**1st proof.** Remember MT3 exe 2.

Let $f$ be the config. at the start of the game.

Let $V = \{$configs obtainable from $f$ by a sequence of moves$\}$,

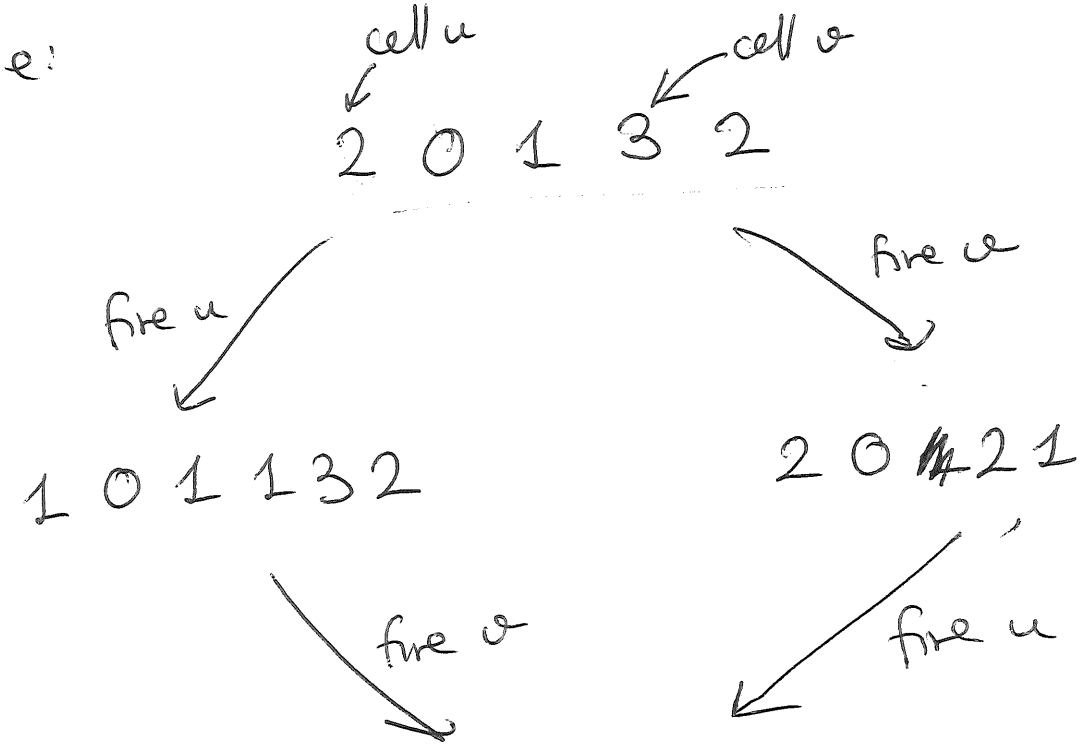
Then $V$ is finite (by Prop. 7.5).

Define a digraph $(V, A)$, where there is an arc $g \to h$ whenever $h$ can be obtained from $g$ by a move.

This digraph is acyclic (as proven in the proof of Prop. 7.6).

Moreover, it satisfies the no-watershed condition from MT3 exe 2: If some configuration $g$ allows firing two different cells $u \in V$ and $v \in V$, then the resulting configs. can be turned into one & the same config. by further firings.

Example:




Thus, MT3 exe 2 yields that the digraph $(V, A)$ has a unique sink $r$ with $f \xrightarrow{*} r$.
In other words, there is a unique config. obtainable from $f$ by playing the game such that no more moves are possible. ☐

2nd proof. [see the 5707 reference I sent out today.]

**Prop. 7.8.** Let $n \in \mathbb{N}$. If we start the ## game with the config. $(2n)$, then the result is

$$\underbrace{1\,1\cdots 1}_{n \text{ ones}}\ \underline{0}\ \underbrace{1\,1\cdots 1}_{n \text{ ones}},$$

and the game will use exactly $\frac{n(n+1)(2n+1)}{6}$ many moves.

**Proof.** It suffices to check that $\forall\, m \in \mathbb{N}$ and $k \geq 2$, we can get from the config.

$$\underbrace{1\,1\cdots 1}_{m \text{ ones}}\ \underline{k}\ \underbrace{1\,1\cdots 1}_{m \text{ ones}}$$

to

$$\underbrace{1\,1\cdots 1}_{m+1 \text{ ones}}\ \underline{(k-2)}\ \underbrace{1\,1\cdots 1}_{m+1 \text{ ones}}$$

by a sequence of $\#(m+1)\#(m+2)$ moves.

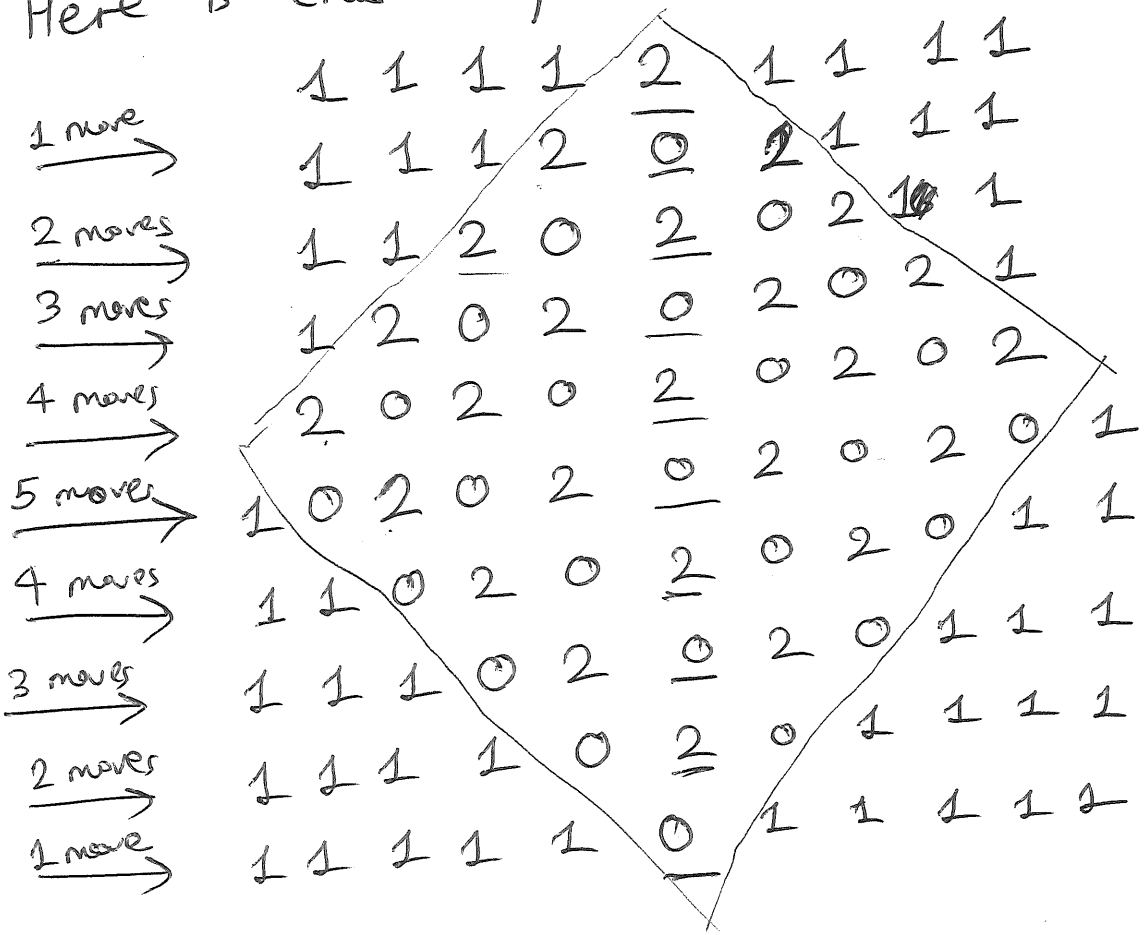
To prove this, it suffices to check that we can set from the config.

$\underbrace{1\,1\,\dots\,1}_{m\text{ ones}}\,2\,\underbrace{1\,1\,\dots\,1}_{m\text{ ones}}$

to $\underbrace{1\,1\,\dots\,1}_{m+1\text{ ones}}\,0\,\underbrace{1\,1\,\dots\,1}_{m+1\text{ ones}}$

by a sequence of $(m+1)(m+2)$ moves.

Here is this sequence: (for $m=4$):




(# moves
$= 1+2+3+4+5$
$+4+3+2+1$).

□

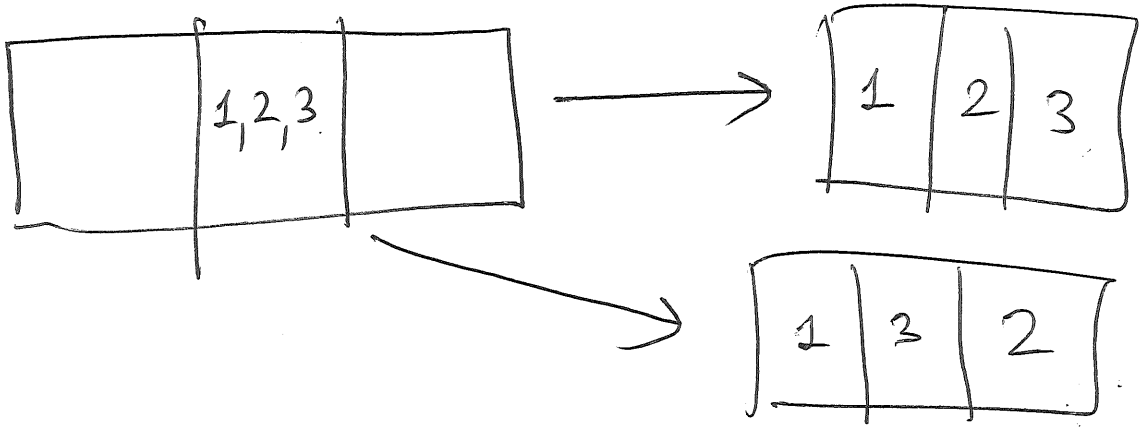
**Prop. 7.9.** Let $n \in \mathbb{N}$. If we start the game with the config. $(2n+1)$, then the result is

$$\underbrace{1\,1\,\dots\,1}_{n \text{ ones}}\,1\,\underbrace{1\,1\,\dots\,1}_{n \text{ ones}}.$$

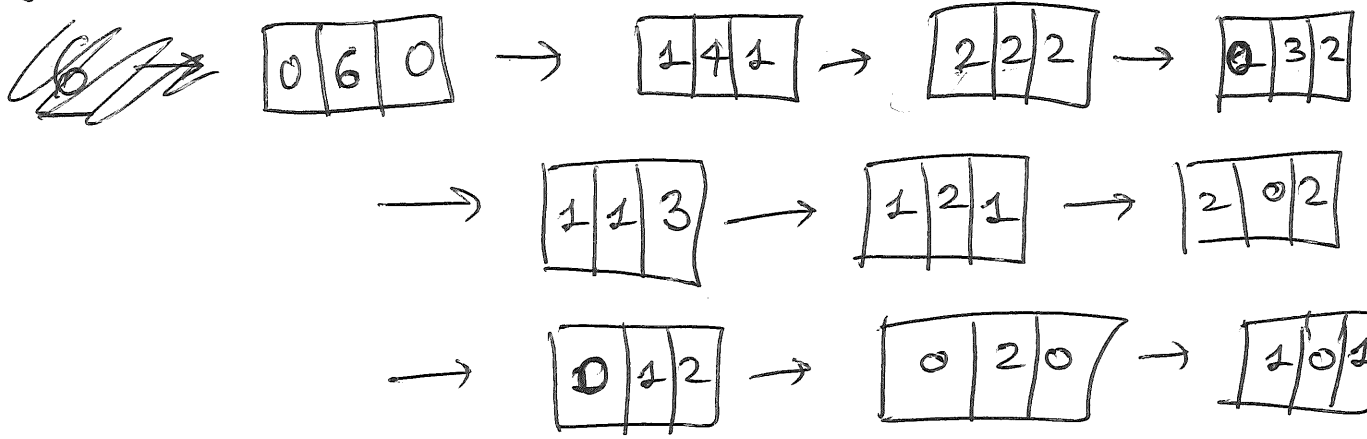
**Proof.** Follows from Prop. 7.8. □

Variants of the game:

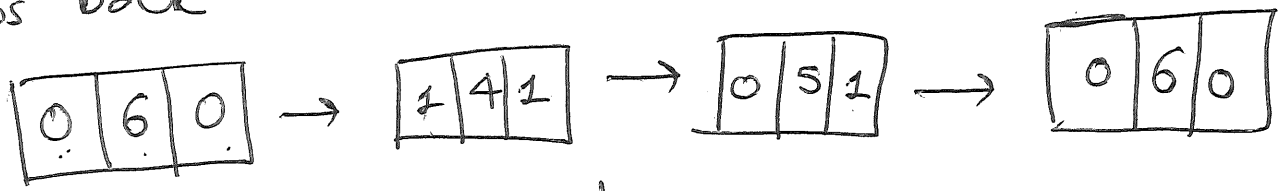
- labelled chips (Hopkins, McCanille, Propp: "Sorting via chip-firing") ♯:
  the chips are $1, 2, \dots, n$; when a cell is fired, two of its chips are taken, the smaller moving left & the larger moving right



- bounded line: instead of $V = \mathbb{Z}$, take $V = [-N, N]$, and any chip that "falls off the interval" is lost.




- same, but the borders of the interval "bounce" the chips back:




No termination!

We can play a similar game on each directed graph.

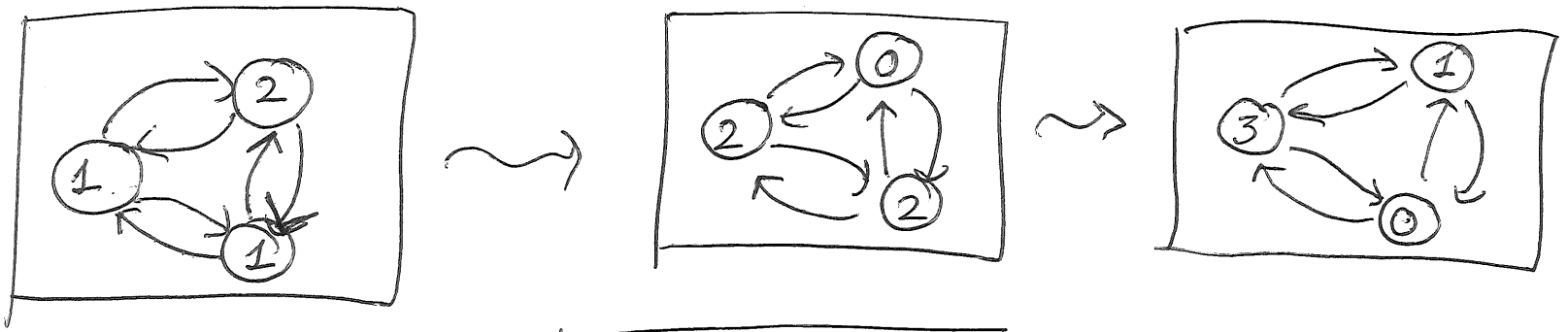
Let $D = (V, A, \phi)$ be a multidigraph.

A <u>config.</u> means a map $f: V \to \mathbb{N}$.

We visualize it as putting $f(v)$ chips on each vertex $v$.

<u>Firing $v$</u> means taking outdeg $v$ chips away from $v$, and sending one of them along each arc with source $v$.

<u>Example:</u>




No termination!