LECTURE 2 Wed 9/5

Last time: 3 games with chips on graphs

Chip Firing Game (A.k.a abelian sandpile model)
[Bak-Tang-Weisenfeld] ← 3 physicists. [Bjorner-Lavosz-Shur].

First introduced by physicists as a mathematical model for processes like avalanches.

THE GAME

no multiple edges or loops

$G=(V,E)$ simple, connected, finite graph

$V=[n]$; chip configuration $c=(c_1,\dots,c_n)\in \mathbb{Z}^n_{\geq 0}$

A vertex v is stable if $c_v < \deg_G(v)$.
A configuration is stable if all vertices are stable

Start with an initial configuration c_{innit}

STEP: If v is not stable, then move one chip from v to each neighbor of v.
This is called firing vertex v. (physisists call it toppling).

Lemma 1: For fixed initial configuration c_{innit}, the result will always be the same no matter the order you fire the vertices.
i.e. either the game never stops, or it stops at some final stable configuration c_{stable}
(The confluence property)

in quotes b/c later we will used sink for something else

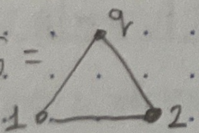
A MODIFICATION: We have a special vertex q called "the sink".
All chips that go into the "sink" disappear.

Lemma 2: For a given c_{init}, the chip firing game with a sink alway stops at the same c_{stop}

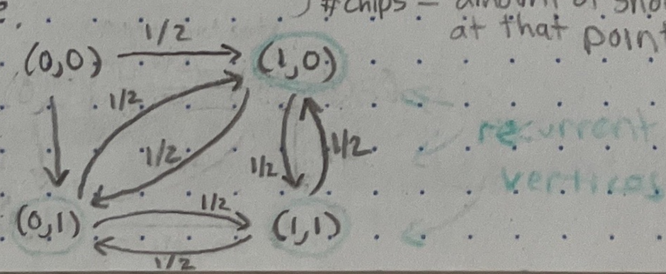
Def: Given an initial configuration c_{init}, stabilization is the process of playing the chip firing game to go from $c_{init} \rightsquigarrow c_{stable}$

Random Process (Markov chains)
States: All stable configurations.
Moves: Randomly pick a vertex $v\neq q$.
Add one chip to v and stabalize.

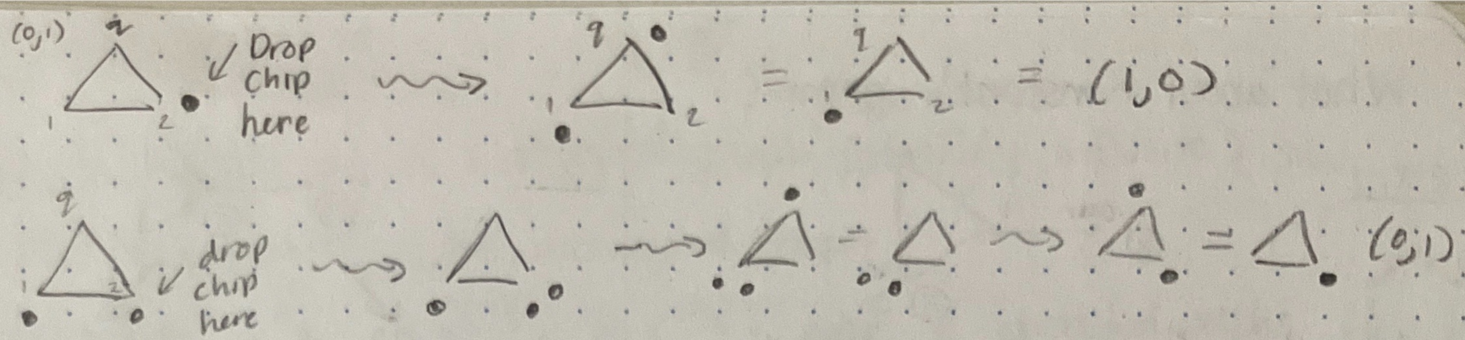
This process is a model for avalanches! #chips = amount of snow at that point.

Ex. $G=$ 

Markov chain on stable configurations:



E.x.



Def: A stable conf. c is called <u>recurrent</u> if there is a non-zero prob. to come to c from any other state (i.e. a directed path from any vertex to c)

Thrm 1: The steady state distribution is the uniform distribution on all recurrent conf.

Thrm 2: # recurrent configurations = # spanning trees of G.

Proof of Confluence (Lemma 1):

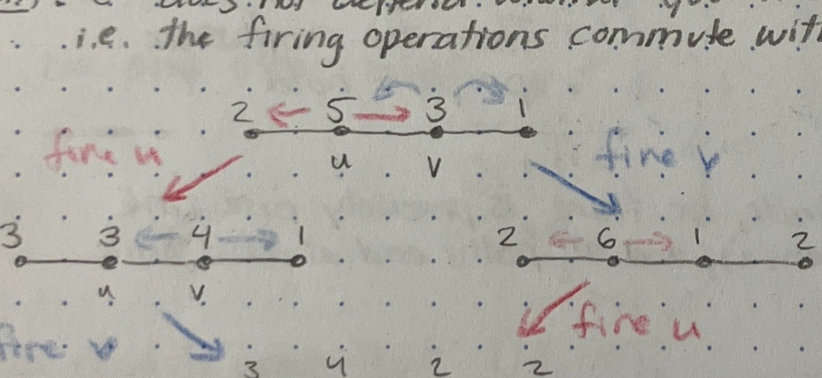
"Diamond Property": Suppose that configuration c has two or more unstable vertices u & v.

$$\begin{tikzcd} & C \arrow[dl, "\text{fire } u"'] \arrow[dr, "\text{fire } v"] & \\ C' \arrow[dr, "\text{fire } v"'] & & C' \arrow[dl, "\text{fire } u"] \\ & C''' & \end{tikzcd}$$

After firing u, other unstable nodes remain unstable
⇒ After firing u we can fire v and vice versa

Claim: C''' does not depend whether you fire u or v first
i.e. the firing operations commute with each other

E.x.



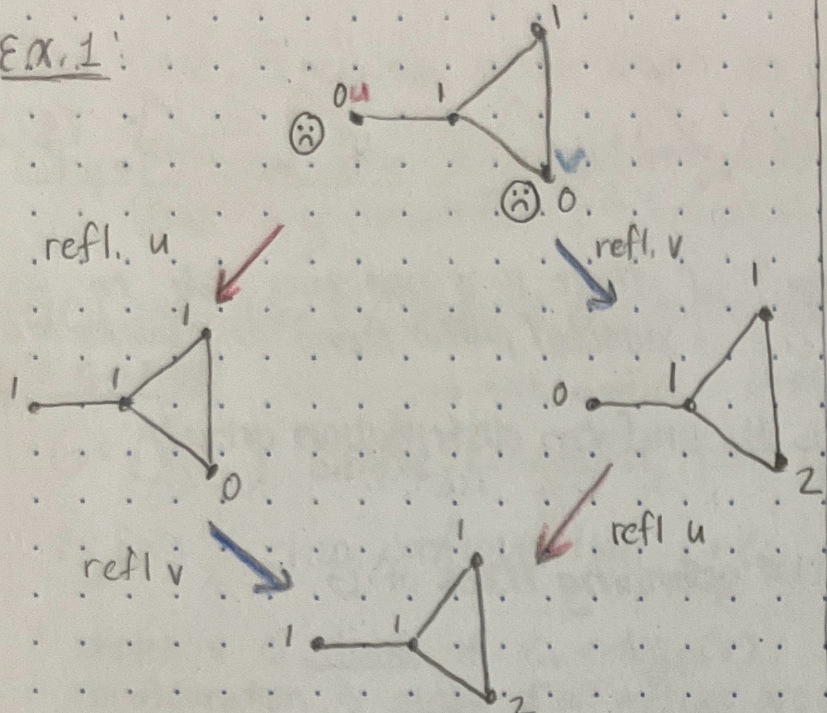
Claim: Diamond property implies confluence.

Proof Sketch: By induction, assume confluence is already showed for middle height (and below) configurations in diamond.
Since they have the same bottom point, must go to same stable conf.
⇒ Top config. in diamond must go to same conf. too

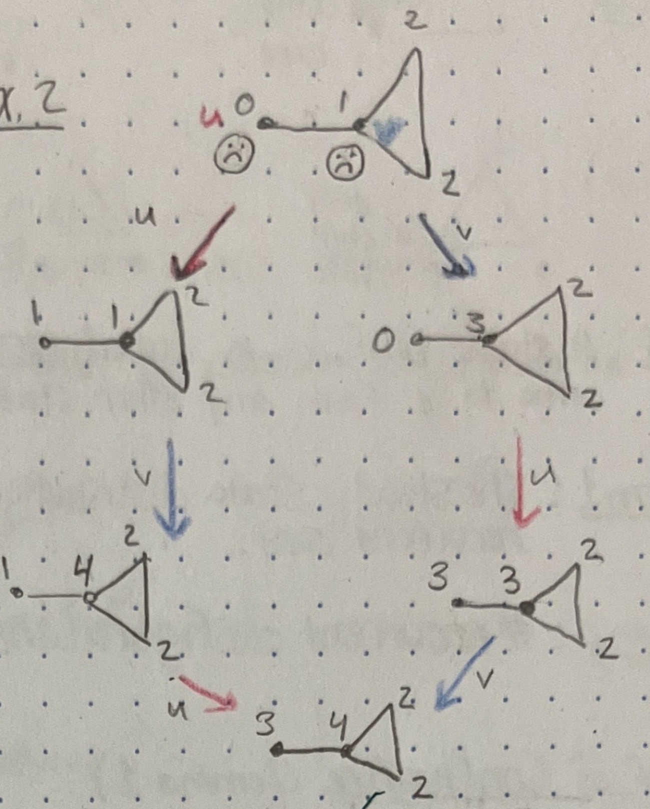
Note: Basically same strategy works to prove confluence of sponsor game and excited sponsor game.

What about Konstant's game?

Ex. 1



Ex. 2

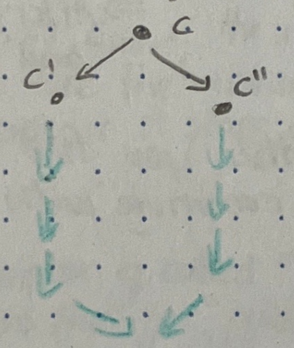


Exercise: We will always get a diamond or hexigon depending on whether vertices are adjacent or non-adjacent.

Roman Lemma Suppose Γ is a (possibly infinited) directed graph whose underlying non-directed graph is connected, and s.t.

b/c all roads lead to Rome

for any fork



$\exists$ coverging directed paths of the same length

Then the confluence propert hold.

i.e. every directed walk is infinite, or there is precisely one sink (now in the graph theory sense), and all paths end at this sink