Chip Stacking and a Condition on $\widetilde{\Delta^{2m+1}}$

Michael Dotzel

August 9, 2017

1 Introduction

In their paper "Sorting via Chip Firing", Hopkins, McConville, and Propp explore a variant of the standard unlabeled chip-firing process by assigning numerical values to the chips and firing them on $\mathbb{Z}$. They establish that the labeled configuration Δ^{2m} (m>0 an integer) sorts the labeled chips sequentially along $\mathbb{Z}$, and that there are bounds of length $\lfloor n + 1/2 \rfloor$ on the location of any chip (k) for an odd number of chips n. In this paper, we present motivation and proof of a condition on $\widetilde{\Delta^{2m+1}}$ that refines the possible stable permutations of [2m+1] on $\mathbb{Z}$, and some properties of unlabeled configurations that when fired upon yield the unique unlabeled stable configuration $\widetilde{\delta}^m$.

2 Chip Stacking

Suppose we are given a labeled chip configuration on $\mathbb{Z}$ with chips (a), (b) with a<b occupying vertex i. As defined in [1], a **chip firing** at i involving (a) and (b) moves (a) to vertex i-1 and (b) to vertex i+1.

Definition: Given a configuration C with chips (a) at vertex i-1 and (b) at vertex $i + 1$, with $a < b$, a **chip stacking** at vertex i involving (a) and (b) is the operation that moves both (a) and (b) to i. It is easy to see that chip firing and chip stacking are inverse operations of one another.

We will present some other definitions for the corollary and theorem to follow:

Definition: C a configuration with n labeled chips, and suppose there are chips $(a_1), (a_2), \ldots, (a_k), a_1 < a_2 < \cdots < a_k$, and $C(a_1) > C(a_2) > \cdots > C(a_k)$. $\{a_i\}$ is called **k - decreasing**.

Examples of chip stacking

2.1 A Lemma

We now reiterate a lemma found in [1]:

Lemma 1. *(This appears as lemma 12 in [1]) Suppose $\Delta^n \to C$. Then we have $-\lfloor (n + 1 - k)/2 \rfloor \leq C(k) \leq \lfloor k/2 \rfloor$.*

This lemma implies the following

Lemma 2. *Suppose C is a stable configuration on 2m+1 labeled chips. Then it is impossible for there to appear in C a k - decreasing sequence for k $\geq$ m+1.*

(From examples that will hopefully be uploaded soon) We propose the following refinement of lemma 2:

Lemma 3. *Suppose C is a stable configuration on 2m+1 labeled chips. If there exists a 3 - decreasing sequence k < j < i , $C(k) > C(j) > C(i)$ in C, then for any configuration D that can chip fire to C we cannot have i, j, and k all occupy the same vertex of D.*

(In the case of the ternary labeling system Mr. Hyde proposed, where chips 1, . . . , m are given the value -1, m+1 is given the value 0, and the rest are valued 1, although it is certainly very possible to stack 1,0, and -1 together, this makes no distinction between a value 1 chip that's not part of the 3-decreasing sequence and one that is. So our conjecture for a fully labeled set of chips remains plausible.)

Proof. (sketch) First, when we mention stabilization in this proof, we mean with respect to the underlying unlabeled chip configuration $\widetilde{\delta^{2m+1}}$. We will examine the contrapositive of the lemma above, and induct upon the number of chip firings n that must be applied to D in order that it stabilize. For n = 1, we have the chip configuration in which all 2m+1 chips are placed along $\mathbb{Z}$ except chips i, j, and k, which are stacked at $D(i) = 0$. This poses a problem: assuming D stabilizes with the conditions that k < j < i and $C(k) > C(j) > C(i)$, we must apply a chip firing which moves i and k such that they move to the adjacent vertices to the left and right, respectively, and this contradicts the chip firing convention of moving larger chips right. Now assume that for $n \leq M$ chip firings away from a stable configuration with $k < j < i$ and $C(k) > C(j) > C(i)$ $(M > 1)$, i, j, and k cannot be in the same column. Now consider $D*$ a configuration that is $M + 1$ chip firings away from stabilizing (in the unlabeled sense), with i, j, and k occupying vertex $D(i)$, k < j < i and $C(k) > C(j) > C(i)$ in $C = \widetilde{D*}$, should it exist. This means that there is a series of chip firing moves $f_1, f_2, \ldots, f_{M+1}$ that yield a labeled stabilization with the properties described above. However, if our first chip firing move involved none of i, j, or k, then after chip firing we would be left with a configuration M firings away from stabilization with i, j, and k all occupying vertex $D(i)$, which means such a configuration cannot stabilize with the prescribed conditions. So we will assume that our first firing move involves some of i, j, or k. Now consider the second chip firing f_2; if it does not involve i, j, or k, then we can apply a chip stacking operation to stack the i, j, and/or k that were fired through f_1, which would yield a configuration involving i, j, k occupying $D(i)$ which is M firings from a stabilization assumed to involve $k < j < i$ and $C(k) > C(j) > C(i)$; by induction this is a contradiction. So we assume f_2 also involves some of i, j, and k. Continuing in this fashion, we obtain a sequence of chip firings $f_1, f_2, \ldots, f_s$, each of which involves either moving some of i, j, k or firing a vertex such that the inverted subsequence of chip firings (i.e. the set of chip stackings) that move i, j, and k back to $D(i)$ is of size s. Eventually i firing left and/or k firing right will encounter chips p_1 , p_2 such that i > p_1 or k < p_2 upon reaching f_s (this will of course happen if i, k are fired right, left respectively. If i and k did not fire with a chip smaller/greater than it, then i/k would be to the right/left of j, respectively). Denote D^{**} by the configuration obtained by applying $f_1, f_2, \ldots, f_s$ to D^*, and the positions of i, j, and k in D^* are $D^*(i), D^*(j), D^*(k)$. In order for D to stabilize, D^{**} must stabilize, and so there are further chip firings $f_{s+1}, \ldots, f_q$ such that f_q is the first chip firing at $D^{**}(i) = D^*(i)$ involving i and p_1 (or likewise with k and p_2). Upon applying f_q, we have that i/k move to vertex $D^*(i) + 1$ / $D^*(k) - 1$. But we note that we simply need to apply chip stackings which invert the chip firings $f_1, f_2, \ldots, f_{s-1}, f_{s+1}, \ldots, f_{q-1}$ to D^{**} in reverse order to obtain a configuration D which is M-1 chip firings away from stabilization, and in which i,j, and k occupy the vertex $D(i)$, which is a contradiction under the induction hypothesis. □

2.2 Unlabeled chip stacking

Starting from the unlabeled chip spike δ^m, we obtain a unique stable configuration $\widetilde{\delta^m}$. It is then a natural question to study those unlabeled configurations which, when chip firings are applied, yield $\widetilde{\delta^m}$ - in other words, what are the different "maximally unstable" configurations one could obtain via chip stacking from $\widetilde{\delta^m}$?

Definition 2.2.1: A chip configuration C is **maximally unstable** if no chip stacking can be applied to any vertex of C.

It is easy to see that δ^m is an example of a maximally unstable configuration.

Here are a few properties regarding such unstable configurations:

Proposition 1. *Suppose we are given $\widetilde{\delta^m}$; then in order for δ^m to be achieved through a series of chip stackings, our first two moves cannot both involve chip stackings at vertices i and j such that $\mid i - j \mid \geq 3$.*

Proof. Starting from $\widetilde{\delta^m}$, our first chip stacking at vertex i will pinch chips from vertices $i \pm 1$ to stack on vertex i. If our next chip firing is at vertex $i \pm 3$, then vertices $i \pm 1$, $i \pm 2$ will be devoid of chips. It is impossible for any chip stacking process to reconcile the intervals on either side of this gap with each other, since the chip stacking operation only traverses gaps of length 1. In general, if two chip stackings occur at vertices i, j, with $\mid i - j \mid > 3$, then chip-occupied vertices are divided by the gaps created by these stackings. Any series of chip stackings must eventually move the gaps created at vertex $i \pm 1$ and $j \mp 1$ towards one another so as to reconcile the disjoint intervals of chips created by these gaps (if the destabilization δ^m is to be achieved, as the chips must combine into one contiguous interval with at least one chip on each of its vertices). But after finitely many chip stacking operations we arrive at a configuration in which two gaps are adjacent to one another, which is impossible to traverse by a chip stacking operation. □

Definition 2.2.2: A chip stacking operation at vertex i (where there is a nonzero number of chips occupying i) of configuration D is called **selfish** if applying it leaves vertex $i+1$ and $i-1$ devoid of chips, i.e. vertices $i \pm 1$ have only 1 chip occupying them. (include example(s) of selfish stacking)

Definition 2.2.3: A chip stacking operation at vertex i is called **selfless** if vertex $i+1$ is occupied by ≥ 2 chips while vertex $i-1$ is occupied by 1 chip, or vice versa. The former selfless stacking is called **right selfless** and the latter **left selfless**. (include example(s) of selfless stacking)

Note that proposition 1 can be restated as follows: Suppose we are given $\widetilde{\delta^m}$; then in order for δ^m to be achieved through a series of chip stackings, our first two moves cannot both be selfish stackings at vertices i and j such that $\mid i - j \mid \geq 3$.

Definition 2.2.4: A chip stacking operation at vertex i is called **continuing** if both vertices $i+1$ and $i-1$ involve chip stacks of height ≥ 2.

We will first focus on an odd number of chips. Considering $\widetilde{\delta^{2m+1}}$, we have determined via proposition 1 that the second move cannot also be a selfish stacking. However, we can observe that as long as there are two disjoint vertex intervals with chips in them as called for by the first selfish stacking, we cannot apply another selfish stacking. For supposing we do, we argue as we do in proposition 1 to show that a gap of length two would appear, eliminating the possibility of the configuration recovering δ^{2m+1}. This means that in our quest to recover δ^{2m+1} from $\widetilde{\delta^{2m+1}}$, we perform the following process:

1. Choose a vertex $i \in [-m+1, m-1]$ to perform the first selfish stacking.

2. Since we cannot apply another selfish stacking, we must perform a selfless stacking at either vertex $i-1$ or $i+1$. Rightness or leftness depends on which side of i the stacking is to be applied: if left of vertex i, then a right selfless stacking can be applied, and vice versa. Through applying a selfless stacking, there will always remain a stack of height 2, so we may continue applying selfless stackings and continuing stackings until the chips converge to a single contiguous interval of chips.

3. Now that we have a contiguous region, we may (depending on the number of chips available) apply a selfish stacking. However, upon doing so, we cannot perform another selfish stacking, as eventually our stacking process would lead us to a gap of length 2. The same rules apply for the left/right selfless stackings after the selfish stacking is applied. In general, our moves may be any of the other types of stackings as defined above, so long as before another contiguous configuration is acheived we will not apply 1) ≥ 2 selfish stackings, 2) a selfish stacking and a selfless stacking, or 3) a left selfless and right selfless stacking (in that order) at vertices i and $j > i$ such that $\mid i - j \mid \geq 3$.

4. Continue applying continuing, selfish, and selfless chip stacking operations in the same vein as step 2 until we arrive at another contiguous interval with chips.

5. Continuing this process, we arrive at δ^{2m+1}: suppose that instead, our chip stacking operations yield a configuration $D \neq \widetilde{\delta^{2m+1}}$ which has one contiguous interval with chips, but is maximally unstable. This implies that D consists of two columns of chips at vertices s and $s \pm 1$. This is because any contiguous region of chips greater than 2 in length can see a chip stacking act upon it, and so is not maximally unstable (Need to show that this is impossible; i.e. show that such a configuration cannot be obtained by $\widetilde{\delta^{2m+1}}$ - we know it can't converge to δ^{2m+1}). Recall the statistic $\phi_\infty(c) := \sum_{i \in \mathbb{Z}}(i \times c(i))$, and proposition 1 in [1] - namely, that $\phi_\infty(c') = \phi_\infty(c)$ for c' a configuration obtained from c via chip firing at a vertex. $\phi_\infty(\widetilde{\delta^{2m+1}}) = 0$. Let c be the configuration with only two adjacent columns of chips, $2m+1$ chips in total. Then $\phi_\infty(c) = s(k) + (s-1)(2m+1-k) = (s-1)(2m+1) + k = 0$ if it were to be obtained from $\widetilde{\delta^{2m+1}}$ by a series of chip stackings. With k positive, and $(s-1)(2m+1)$ positive, this implies $s = 1$ and $k = 0$. So by contradiction we have that the contiguous intervals as determined by the chip stacking converge up to δ^{2m+1}, since our method gives us contiguous intervals of decreasing size. (Still need to explain: method described gives a way of finding an unstabilizing path to δ^{2m+1} relying on stackings by means of reducing the size of contiguous intervals)

This process for recovering δ^{2m+1} leads to the following

Lemma 4. *Given $\widetilde{\delta^{2m+1}}$, and let $\boldsymbol{C}$ be the set of configurations that destabilize to δ^{2m+1}. Then any element of $\boldsymbol{C}$ can be obtained by an appropriate series of continuing, selfish and selfless stackings (appropriate to the rules explained above in the process for recovering δ^{2m+1}). In other words, all possible allowable sequences of these stackings by the rules above determine all possible configurations that stack to δ^{2m+1}.*

3 Further questions

Some initial aspects of δ^{2m+1} recoverability that we hope to continue exploring are 1) the claim above and 2) the number of different configurations that chip stack to δ^{2m+1}. We also want to continue working on the label restrictions of stabilizations of Δ^{2m+1} so as to shed more light on conjecture 26 in [1].

4 References:

Sam Hopkins, Thomas McConville, and James Propp. *Sorting via Chip-Firing.* 2016.