REU 2005 · Discrete Mathematics · Lecture 10

Instructor: László Babai
Scribes: T. Schedler, N. Raghunathan

July 11, 2005. Last updated July 13, 2005

10 Algorithmic problems

10.1 The Four main Problem

Problem 10.1 (Sandpile Prediction). *Given* $\mathbf{x}$*, compute the stable configuration* $\sigma(\mathbf{x})$ *obtained from* $\mathbf{x}$ *by repeated toppling.*

Problem 10.2 (Monoid Operation). *Given stable configurations* $\mathbf{x}, \mathbf{y}$*, compute* $\mathbf{x} \oplus \mathbf{y} := \sigma(\mathbf{x} + \mathbf{y})$*.*

Recall the definition of recurrence

Definition 10.3. A **stable** configuration is **recurrent** if it is accessible from every configuration: $\mathbf{x} \in M$ is recurrent if $(\forall \mathbf{y} \in M)(\exists \mathbf{z} \in M)(\mathbf{x} = \mathbf{y} \oplus \mathbf{z})$ (This is equivalent to saying that $\mathbf{x}$ belongs to the sandpile group $\triangleleft_{\min} M$).

Exercise 10.4. *The* ***saturated configuration*** $MAX \in M$*, is defined by* $(\forall i \in V_0)(MAX(i) = \deg^+(i) - 1)$*. Show that* MAX *is recurrent.*

Exercise 10.5. $\mathbf{x}$ *is recurrent iff* $\mathbf{x}$ *is accessible from* MAX*.*

Problem 10.6 (Recognizing Recurrence). *Given a stable configuration, decide whether or not it is recurrent.*

Problem 10.7 (Identity). *Compute the identity configuration. By identity we mean the identity of the sandpile group (the unique recurrent idempotent).*

We have the following redutions between the problems

$$\text{SP PRODUCTION} \succ \text{MONOID OPERATION} \succ \text{ID},$$

where $A \succ B$ denotes that problem B can be reduced to problem A.

Consider the following algorithm to compute the identity:

$$X := \mathrm{MAX} \oplus \mathrm{MAX}$$
$$Y := \mathrm{MAX} - X;$$
$$Z := Y \oplus \mathrm{MAX}.$$

Exercise 10.8. *Prove that Z is the identity of the sandpile group.(*Hint:*(1) Z is recurrent. (yes: it is reachable from MAX). (2) $Z \oplus MAX = MAX$.*

Exercise 10.9. *Prove that a configuration X is recurrent iff $X \oplus ID = X$*

Corollary 10.10. *RECURRENCE $\prec$ MONOID OPERATION.*

Recall that M^{-} is the semigroup generated by M {identity of M}.

Exercise 10.11. *If the ambient space is strongly connected on V_0 then the Rees quotient is nilpotent. (*Hint: *This is equivalent to Exercise 10.14)*

Research Question 10.12 (Nilpotence Class). *Study/estimate the nilpotence class of M^{-}/G for various classes of ambient spaces*

Consider the following process: add a grain at a vertex of your choice. Repeat

Question 10.13. *Does the process necessarily lead to a recurrent configuration?*

Exercise 10.14. *If the digraph V_0 is* ***strongly connected*** *(i.e. $\exists$ a directed path from each vertex to each other vertex), then every sequence of adding grains eventually reaches a recurrent configuration.*

Exercise 10.15. *Show that Exercise 10.14 becomes false if we drop the condition of strong connectivity.*

10.2 Length of an Avalanche

Theorem 10.16 (Gábor Tardos). *If the ambient space is* ***undirected****, then the length of an avalanche is $\leq n \cdot d \cdot N$, where n is the number of nodes (including the sink) and d is the diameter of ambient space (with sink), and N is the initial number of chips.*

Since the diameter $d \leq n-1$, we get a polynomial bound on the length of the avalanche.

Corollary 10.17. *The length of the avalanche is $< n^2 \cdot N$.*

Example 10.18. *Complexity of computing ID on the $m \times m$-grid: we have $n = m^2+1 \approx m^2$, $d \approx m$, and $N \leq 6m^2$. So the complexity is $\leq 12m^5$.*

Exercise 10.19 (Long avalanche in digraphs). *Construct a digraph (without multiple edges) as the ambient space such that an avalanche triggered by one extra grain (from a stable configuration) takes exponentially long (in the number of vertices).*

We now prove prove Tardos' theorem using the lemma below.

Lemma 10.20. *Let $s(i,t)$ be the number of times vertex i has been fired by time t (thus, we have $\sum_{i \in V_0} s(i,t) = t$.) We have, $(\forall i,j,t)$ (if $i \sim j$ then $|s(i,t) - s(j,t)| \leq N$). Here $i \sim j$ means i and j are adjacent.*

Proof of Theorem 10.16 using Lemma 10.20. The sink doesn't fire at all. By Lemma 10.20, all the vertices adjacent to the sink fire at most N times in total (since this will work for any time). By repeated application of Lemma 10.20, it follows that a vertex at a distance k from the sink fires at most kN times in total. Hence every vertex fires at most dN times and the total is at most $n \cdot dN$ firings. □

Proof of Lemma 10.20. Let i and j be adjacent and $s(i,t) < s(j,t)$. We need to prove that $s(j,t) \leq s(i,t) + N$. So let $B = \{\ell \in V \mid s(\ell,t) > s(i,t)\}$ and $A = \{\ell \in V \mid s(\ell,t) \leq s(i,t)\}$. Since all of the vertices of B fired more than any of the vertices of A, the number of chips in A has increased between time 0 and time t. The total increase is

$$I := \sum_{b \in B, a \in A, a \sim b} s(b,t) - s(a,t). \tag{10.1}$$

Note that $I \leq N$. Note that each term on the right hand side of Equation 10.1 is ≥ 0 and therefore$N \geq I \geq s(j,t) - s(i,t)$, as claimed. □

10.3 Open questions

Following are some open cases in sandpile prediction.

1. Thin undirected graph (= small edge multiplicities), tall piles. The heights are k-digit integers (given in binary). Compute the sandpile prediction in time polynomial in n, k.
2. Thick undirected graph: heights and edge multiplicities are given as k-digit integers. Compute the sandpile prediction in time polynomial in n, k.
3. Thin Directed graphs, short piles. Compute sandpile prediction in time polynomial in n or prove this is NP-hard.

Some open cases in digraph recurrence:

1. Is it in NP (verifiable in polynomial time)?
2. Is it in coNP (Is transience (non-recurrence) verifiable in polynomial time)?
3. Is it NP-hard (at least as hard as an NP-complete problem)?

10.4 Burning Algorithm

Starting from a stable configuration $\mathbf{x}$, consider the following process: Fire Sink. Stabilize. This process is called the burning process.

Exercise 10.21. *Prove that starting from any stable configuration* $\mathbf{x}$*, each vertex will fire at most once in the burning process.*

Exercise 10.22 (Dhar's characterization of recurrence for undirected graphs). *Show that a configuration* $\mathbf{x}$ *is* ***not*** *recurrent iff* $\exists$ *a subset* A *of* V_0 *such that*

$$(\forall i \in A)(\mathbf{x}(i) < \deg_A(i)).$$

*(*Hint: *A configuration is recurrent iff each vertex fires exactly once in the burning process)*

This gives a polynomial time algorithm to recognize recurrence if the ambient space is undirected.